Documentation des Web Services de Geoconcept Web

GEOCONCEPT SAS

Version : 2022-1

Date : 24/03/22

Cette documentation concerne la version 2022 de Geoconcept Web.

Introduction

Geoconcept Web est fourni avec des outils à destination des intégrateurs qui permettent :

- d’ajouter de nouveaux Widgets dans le Designer;
- d’utiliser la carte dans des interfaces autonomes avec l’API Javascript ;
- d’exploiter les nombreux web services (SOAP/REST) fournis avec la solution;
- ainsi que des exemples d’intégration.

Le Kit de développement des widgets

Cette section explique comment créer des widgets pour les intégrer dans Geoconcept Web. Ces nouveaux widgets pourront alors profiter des avantages fournis par le mécanisme mis en place dans la solution.

Introduction

Existant

Le projet devait dans un premier temps être ISO fonctionnel & ISO compatible avec geoweb-easy v1. Pour cela des procédures de migration automatique ont été développées pour assurer l'installation de geoweb-easy v2 avec une base de données geoweb-easy v1 existante. Ces procédures de migration seront expliquées dans la partie server.

Choix technologiques

Contrairement à geoweb-easy v1, geoweb-easy v2 embarquera côté client un moteur full JavaScript. Si nous reprenons les cinq parties du projet :

1. La partie widgets sera basée sur la dernière version de YUI : 3.5.1
2. La partie Java est basée sur Java 5 et utilisera les bibliothèques Sax pour la gestion des widgets
3. La partie base de données utilise la version courante de Hibernate utilisée en standard dans Geoweb c'est à dire Hibernate 3.2
4. La partie Dispatcher d'URL est basée sur Spring MVC 2.5
5. La partie Web Services REST est basée sur Apache CXF, et utilise notamment le gestionnaire de plugins de Web Services de Geoweb.

La bibliothèque Jackson 1.8.5 permet de faire le mapping client – server et le projet embarquera cityportal-jar pour la gestion des POI

Les widgets

Introduction

Le widget est un composant d'interface graphique (n'importe quel élément de type HTML/JavaScript) et est aussi un outils permettant de récupérer des informations via le server, ou une source de données interne/externe.

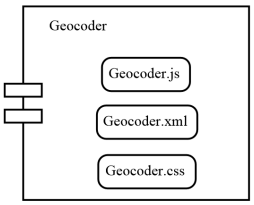
Le but des widgets est donc de pouvoir développer ses propres composants clients sans avoir besoin de modifier le code de Easy geoweb. Il doit être interprété par le client de façon générique et autonome.

Ce chapitre présente en détails la composition, les dépendances et la configuration des widgets.

Architecture

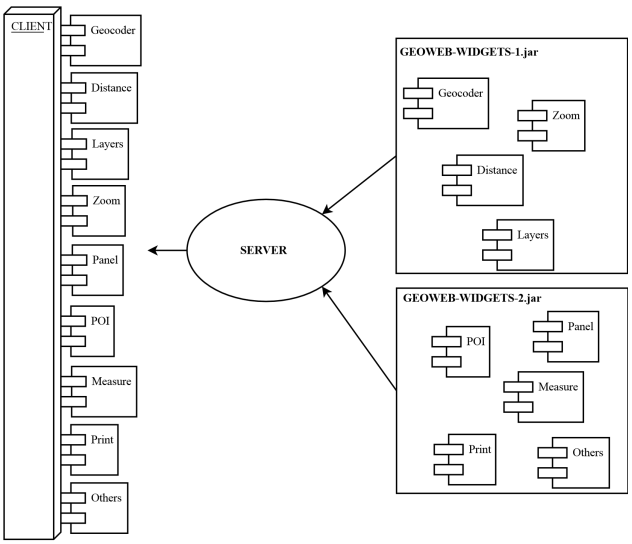
Chaque widget est indépendant de l'autre. Il est composé d'un fichier de description XML, d'un fichier source JavaScript et d'un fichier de style CSS :

Architecture d’un widget



Le client est composé d’un gestionnaire de widgets au niveau JavaScript et est étoffé d’un ensemble de widgets.

Architecture globale des widgets



Le principe est de décrire le widget via un fichier de configuration XML qui pointera vers le fichier source correspondant. Lors du chargement du widget par le moteur de widgets (server), celui-ci parsera le fichier XML pour en extraire les informations nécessaires. Il mettra ces informations sous forme de JSON qu’il renverra au client. Ensuite le client appellera la combo pour charger chaque fichier JS/CSS à l’aide des informations contenues dans le JSON.

Fichier XML

Description générale

Le fichier XML est un fichier de description du widget. Il est parsé par le moteur de widget pour en extraire les informations qui lui seront utiles et les renvoyer au client sous forme JSON. Ce fichier doit respecter les mêmes règles pour tous les widgets :

- `id` : l’id du widget doit être unique
- `icon` : le chemin relatif (par rapport au jar) de l’icône du widget
- `js` : le fichier source correspondant
- `module` : le module YUI du widget, il doit être unique et identique à celui du fichier source

- **`display`** : les widgets peuvent hériter d'autres widgets ou de classes YUI que nous n'avons pas besoin forcément d'afficher, cet attribut permet de spécifier si l'on veut afficher le widget dans le builder/portail ou non
- **`groups`** : les groupes définissent les utilisateurs qui auront accès au widget dans le cas d'un portail privé. Il est possible de définir les groupes directement dans l'outil de conception
- **`lang`** : le fichier de ressources permet l'internationalisation du widget. Le chemin fait référence à un fichier properties. Dans l'exemple ci-dessous, le fichier recherché est par défaut resources/widgets.properties. Un widget peut supporter plusieurs langues par exemple resources/widgets_fr.properties ou resources/widgets_en.properties. La langue sera affiché en fonction de la langue du navigateur.
- **`category`** : la catégorie permet de déterminer l'emplacement du widget dans la bibliothèque du Composer. Si ce champ n'est pas renseigné, la catégorie sera définie en fonction du répertoire où sera stocké le widget. La liste des catégories est disponible dans le fichier EasyGeowebCategories.xml. A partir de ce fichier, il est possible de rajouter de nouvelles catégories (voir la liste des catégories).
- **`position`** : Le champ position permet de définir l'emplacement du widget dans la bibliothèque de widget. Si cette valeur n'est pas renseignée, le widget sera placé à la fin.

```
<widget id="geocoder"
        category="navigation"
        position="1"
    icon="images/geocoder.png"
    js="view/GeocoderWidget.js"
    module="geoconcept-widget-geocoder"
    display="true"
        groups="Standard"
        lang="resources/widgets">
</widget>
```

Il est possible d'ajouter des propriétés pour configurer le widget dans le Composer dans le fichier XML. En fonction de ce qui est renseigné, les propriétés sont automatiquement générées dans la fenêtre de configuration des widgets du Composer.

Le label de la propriété affichée est structuré ainsi, "widget." + id du widget + "." + id de la propriété. La valeur à ajouter dans le fichier properties est par exemple "widget.geocoder.mode=Mode d'affichage".

Le champ *params* permet de déterminer une valeur par défaut à la propriété :

```
<widget id="geocoder"
                category="navigation"
                position="1"
                icon="resources/images/Geocode_32.png"
                js="navigation/GeocoderWidget.js"
                module="geoconcept-widget-geocoder"
                display="true"
                groups="Standard"
                lang="resources/widgets">
        <properties>
                <property id="mode" type="radiobutton-2">
                        <params value="0"/>
                </property>
        </properties>
</widget>
```

Liste des propriétés

Voici la liste des propriétés disponibles pour les widgets :

- **`checkbox`** : manipule les valeurs true ou false.

```
<property id="checkboxProperty" type="checkbox">
        <params value="true"/>
</property>
```

- **`text`** : manipule des valeurs textuelles.

```
<property id="textProperty" type="text">
        <params value="default text"/>
</property>
```

- **`slider`** : manipule des valeurs numériques. Les valeurs maximales sont déterminées par min et max. La valeur sélectionnée correspond à value.

```
<property id="sliderProperty" type="slider">
        <params min="1" max="12" value="6"/>
</property>
```

- **`colorpicker`** : manipule des valeurs hexadécimales.

```
<property id="colorpickerProperty" type="colorpicker">
        <params value="#ffffff"/>
</property>
```

- **`combobox`** : manipule des valeurs textuelles. La liste des valeurs disponibles se base sur les `getComboboxOptions` et `setComboboxOptions` (voir l'API des widgets).

```
<property id="comboboxProperty" type="combobox">
        <params value="comboboxValueName"/>
</property>
```

- **`datagrid`** : manipule des valeurs textuelles. Les valeurs par défaut sont séparées par des ;. La liste des valeurs disponibles se base sur les `getDgOptions` et `setDgOptions` (voir l'API des widgets).

```
<property id="datagridProperty" type="datagrid">
        <params value="datagridValueName"/>
</property>
```

- **`radiobutton-2`** : manipule des valeurs numérique. Les valeurs possibles sont 0 ou 1. Pour afficher un label à chaque radiobouton, il est nécessaire d'ajouter 2 clés dans le fichier properties en les terminant par ".left" et ".right"

```
<property id="radiobuttonProperty" type="radiobutton-2">
        <params value="0"/>
</property>
```

- **`checkbox-2`** : manipule des valeurs numérique. Fonctionne à l'identique de la propriété "radiobutton-2" sauf qu'il est possible de sélectionner les 2 valeurs de la propriété.

```
<property id="checkbox2Property" type="checkbox-2">
```

```
        <params value="0"/>
</property>
```

Chacune de ces propriétés peut ensuite être utilisées dans le widget en utilisant la méthode `getPropertyValue` (voir l'API des widgets).

Liste des catégories

Un ensemble de catégories sont définies dans un fichier de configuration du projet *`EasyGeowebCategories.xml`*. Chaque widget est dépendant d'une de ces catégories. Pour ce faire le widget doit être placé dans un répertoire comportant le nom d'une catégorie. Exemple: notre widget geocode est placé dans *`navigation/Distance.js`*, ainsi il appartient à la catégorie navigation. Le widget distance est placé dans *`measure/Distance.js`* donc il appartient à la catégorie measure.

Il est possible d'ajouter de nouvelles catégories dans *`EasyGeowebCategories.xml`*. Pour cela il suffit d'ajouter une balise *entry*. *key* permet de définir le positionnement dans l'affichage de l'accordéon.

```
<properties>
        <entry key="0">layout</entry>
        <entry key="1">general</entry>
        <entry key="2">navigation</entry>
        <entry key="3">view</entry>
        <entry key="4">measure</entry>
        <entry key="5">annotation</entry>
        <entry key="6">data</entry>
        <entry key="7">modification</entry>
</properties>
```

Le nom de la catégorie permet de gérer son internationalisation. Par exemple, pour une nouvelle catégorie nommée *`graphic`*, il faudra ajouter la clé suivante dans *`easy.properties`*

```
accordion.category.graphic = graphique
```

Fichier JavaScript

WidgetBase

Le fichier JavaScript est le fichier source du widget. Celui-ci sera chargé par la combo pour être injecté dans le client. Voici le squelette d'un fichier source :

```
YUI.add("module-name", function(Y) {

        function WidgetClassName(config) {
        WidgetClassName.superclass.constructor.apply(this, arguments);
        }

        WidgetClassName.NAME = "WidgetClassName";

        Y.extend(WidgetClassName, Y.GCUI.WidgetBase,  {
            initializer: function(config) {
            },

            destructor: function() {
            },
```

```
            getWidget: function() {
                // return html description widget
            },

            configureAction: function() {
                WidgetClassName.superclass.configureAction.apply(this);

                // some actions
            }

        });
        Y.namespace("GCUI");
        Y.GCUI.WidgetClassName = WidgetClassName;

}, "1.0.0", {requires: ["project-widgetbase"] });
```

La fonction `getWidget()` sera appelée dans un premier temps par le WidgetManager (client) pour afficher le widget. Ainsi cette méthode renvoie la représentation du widget. Elle peut contenir tout ce que le développeur désire afficher. La seule contrainte est de rester dans une balise HTML `<div></div>`. En effet le widget ne sera affiché que s'il est contenu dans cette balise.

`WidgetBase` permet d'afficher des TreeView, des listbox, des combobox etc.

La fonction `configureAction()` sera appelée par le WidgetManager pour configurer le widget une fois celui-ci inséré dans le DOM. `WidgetBase` propose un série de fonction de base formant l'API des widgets. Ainsi tous les widgets ont accès à la carte, au détail du projet mais aussi à un panel que l'on peut utiliser pour y afficher diverses informations.

Tous les widgets héritant de WidgetBase possèdent des propriétés par défaut : Affichage de l'étiquette, label de l'étiquette et affichage d'un séparateur d'étiquette.

Cette partie la section intitulée « Widgets par l'exemple » ainsi que le tutorial présente des cas concrets d'utilisation des méthodes du `WidgetBase`.

WidgetButton

`WidgetButton` fonctionne de la même façon que `WidgetBase`, il suffit d'hériter de cette classe et de mettre le « requires **`project-widgetbutton`** ».

Il existe 3 types de boutons à définir dans `initializer()` :

- **`BUTTON`** : bouton de base, l'action s'execute au moment du clic souris
- **`TOOL`** : suite au clic souris, le bouton s'active et reste activé. Pour le désactiver, il faut cliquer sur un autre bouton de type TOOL.
- **`TOGGLE`** : un clic souris active le bouton, un second clic le désactive.

Voici le squelette d'un `WidgetButton`

```
YUI.add("geoconcept-widget-button", function(Y) {

        function SchemaButton(config) {
                SchemaButton.superclass.constructor.apply(this, arguments);
        }
```

```
        SchemaButton.NAME = "schemaButton";

        Y.extend(SchemaButton, Y.GCUI.WidgetButton,  {
                initializer: function(config) {
                        SchemaButton.superclass.initializer.apply(this);
                        this.setType(Y.GCUI.WidgetButton.TYPE_TOOL);
                },

                destructor: function() {
                        SchemaButton.superclass.destructor.apply(this);
                },

                configureAction: function() {
                        SchemaButton.superclass.configureAction.apply(this);
                },

                // Button type action
                trigger:function() {
                        SchemaButton.superclass.trigger.apply();
                        // some actions
            },

                // Toggle and tool type action
                activate:function() {
                        SchemaButton.superclass.activate.apply();
                        // some actions
            },

            // Toggle and tool type action
            deactivate:function() {
                SchemaButton.superclass.deactivate.apply();
            }

        });
        Y.namespace("GCUI");
        Y.GCUI.SchemaButton = SchemaButton;

}, "1.0.0", {requires: ["project-widgetbutton"] });
```

Tous les widgets héritant de WidgetButton possèdent des propriétés par défaut : Affichage de l'étiquette, label de l'étiquette et widget par défaut s'il s'agit d'un bouton de type toggle ou tool.

API

Méthode

```
getJson() : {JSON} Retourne la description JSON du widget.
```

```
getJsonProject() : {JSON} Retourne la description JSON du projet.
```

```
getJsonLayers() : {JSON} Retourne la description JSON de la liste des couches définie pour le
 gestionnaire de couches
```

```
getWidgetName() : {String} Retourne le nom internationalisé du widget.
```

```
getWidgetDescription() : {String} Retourne la description internationalisé du widget.
```

```
getEGWApi() : {Y.GCUI.EGWRestAPI} Retourne l'objet EGWRestAPI afin d'avoir accès aux méthodes du service
 Rest de Easy-geoweb.
```

```
getParameters() : {JSON} Retourne les paramètres par défaut de l'application.
```

```
getMap() : {GCUI.Map} Retourne l'objet carte.
```

```
getNode() : {Node object} Retourne le widget sous format javascript.
```

```
getContextUrl() : {String} Retourne le contexte de l'url par exemple "geoweb".
```

```
getWidgetPath() : {String} Retourne l'url pour accéder aux fichiers présent dans le JAR par exemple
 "geoweb/widget?".
```

```
getHost() : {String} Retourne le host de l'url par exemple "http://localhost:8080/".
```

```
getImageSrc() : {String} Retourne l'url pour accéder à l'image définie dans icon du fichier XML.
```

```
isPortal() : {Boolean} Retourne true si l'application est démarrée dans le portail, false si c'est dans
 l'outil de conception.
```

```
getWrapper() : {String} Retourne le wrapper, utilisé pour désactiver le widget dans l'outil de
 conception.
```

```
getPropertyValue(id) : {String} Retourne la valeur de la propriété définie dans le fichier XML en
 fonction de son identifiant (id).
```

```
updatePropertyValue(id,value) : Met à jour la valeur (value) de la propriété définie dans le fichier XML
 en fonction de son identifiant (id).
```

```
getPropertiesPanel() : {Node object} Retourne l'objet panneau de propriété sous l'accordéon dans le but
 d'ajouter de nouvelles propriétés.
```

```
emptyPropertiesPanel() : Vide le panneau de propriété.
```

```
addDynamicProperty(order, propertyId, label, type, defaultValue, defaultMin, defaultMax) : Permet de
 créer une propriété sans
le renseigné dans le fichier XML du widget. order comme position d'insertion parmi les autres
 propriétés. propertyId comme
identifiant. label d'affichage. type, à choisir parmi les types disponibles (par exemple "text").
 defaultValue, defaultMin
et defaultMax pour définir les valeurs par défaut, minimale (optionnel) et maximale (optionnel).
```

```
movePropertyToOrder(property, order) : Déplacer une propriété. property comme identifiant de la
 propriété et order en numéro de
placement parmi les autres propriétés.
```

```
getComboboxOptions(id) : List{String} Retourne la liste des valeurs disponibles pour la propriété
 "combobox"
en fonction de l'id de la propriété.
```

```
setComboboxOptions(id, data) : Enregistre en mémoire la liste des valeurs pour la propriété "combobox"
 avec en paramètre l'id de la
propriété et data en liste de string.
```

```
getDgOptions(name) : List{String} Retourne la liste des valeurs disponibles pour la propriété "datagrid"
    en fonction de l'id de la propriété.
```

```
setDgOptions(name, data) : Enregistre en mémoire la liste des valeurs pour la propriété "datagrid" avec
 en paramètre l'id de la
propriété et data en liste de string.
```

```
_(key) : {String} Converti une clé en chaine de caractère internationalisable.
```

Méthode vide à enrichir pour utilisation

reinitView() : Méthode permettant de réinitialiser l'action du widget.

```
getWidget() : {String} Méthode permettant de définir la représentation du widget.
```

```
checkboxPropertyAction(propertyId, value) : Action appelée après l'enregistrement d'une propriété de
 type checkbox. PropertyId
comme identifiant de la propriété enregistré. Value comme valeur enregistré.
```

```
textPropertyAction(propertyId, value) : Action appelée après l'enregistrement d'une propriété de type
 text. PropertyId
comme identifiant de la propriété enregistré. Value comme valeur enregistré.
```

```
hiddenPropertyAction(propertyId, value) : Action appelée après l'enregistrement d'une propriété
 utilisant des input de type
hidden (datagrid, combobox...). PropertyId comme identifiant de la propriété enregistré. Value comme
 valeur enregistré.
```

```
radioPropertyAction(propertyId, value) : Action appelée après l'enregistrement d'une propriété de type
 radiobutton. PropertyId
comme identifiant de la propriété enregistré. Value comme valeur enregistré.
```

L'ensemble des méthodes de WidgetBase sont réutilisables dans WidgetButton.

Constante

WidgetButton.NAME_PANEL : Nom du panneau

```
WidgetButton.TYPE_TOGGLE : Bouton de type Toggle (activable et désactivable par l'utilisateur).
```

```
WidgetButton.TYPE_BUTTON : Bouton de type Bouton (action executé au moment du clic).
```

```
WidgetButton.TYPE_TOOL : Bouton de type Tool (un seul bouton de type Tool peut être est activé, si
 l'utilisateur active un autre bouton de type Tool, tous les autres sont désactivés).
```

Méthode

```
isActive() : {Boolean} Retourne true si le bouton est activé, false si non.
```

```
getPanel() : {Panel node} Retourne l'objet panel lié au widget. Chaque widget possède son propre panel
 par défaut.
```

```
getWidget() : {Widget node} Retourne l'objet représentant le widget sous forme de bouton avec une icone
 défini dans le fichier XML.
```

```
reinitView() : Réinitialise le widget en désactivant le bouton.
```

```
createPanel(title) : Créé le panel par défaut avec le titre (title) en paramètre.
```

```
showPanel() : Affiche le panel.
```

```
hidePanel() : Masque le panel.
```

```
hidePopup() : Masque les popups de type _GCUI.Control.Popup_.
```

Méthode vide à enrichir pour utilisation

activate() : Méthode appelée suite à l'activation d'un bouton de type Tool ou Toggle.

```
deactivate() : Méthode appelée suite à l'désactivation d'un bouton de type Tool ou Toggle.
```

```
trigger() : Méthode appelée suite au clic sur un bouton de type Button.
```

```
onClosePanel() : Méthode appelée suite à la fermeture du panel.
```

Widgets par l'exemple

Voici dans le détail certains widgets pour étudier leur implémentation.

Hand

Ce widget permet de déplacer la carte. Il est de type bouton donc héritera de la classe WidgetButton.

```
YUI.add("geoconcept-widget-hand", function(Y) {

        /* Hand class constructor */
        function Hand(config) {
                Hand.superclass.constructor.apply(this, arguments);
        }

        /*
         * Required NAME static field, to identify the Widget class and
         * used as an event prefix, to generate class names etc. (set to the
         * class name in camel case).
         */
        Hand.NAME = "hand";

        Y.extend(Hand, Y.GCUI.WidgetButton,  {
                initializer: function(config) {
                        Hand.superclass.initializer.apply(this);
                        this.setType(Y.GCUI.WidgetButton.TYPE_TOOL);
                },
```

```
                destructor: function() {
                        Hand.superclass.destructor.apply(this);
                },

                configureAction: function() {
                        Hand.superclass.configureAction.apply(this);
                },

                activate:function() {
                        Hand.superclass.activate.apply();
                        var map = this.getMap();

                        DynMapSetMouseMode(map,DynMapMoveSelectionMode());
            },

            deactivate:function() {
                Hand.superclass.deactivate.apply();
            }

        });
        Y.namespace("GCUI");
        Y.GCUI.Hand = Hand;

}, "1.0.0", {requires: ["project-widgetbutton"] });
```

Ce widget reprend l'exemple standard décrit plus haut. La méthode `activate()` permet au widget d'appeler la méthode de l'API Javascript [http://api.geoconcept.net/htc/index.html] pour activer le mode de déplacement de la carte.

Voici son fichier de description XML :

```
<widget id="hand"
                icon="resources/images/hand.png"
                js="navigation/Hand.js"
                module="geoconcept-widget-hand"
                display="true"
                lang="resources/widgets">
</widget>
```

Le nom du module correspond à celui définit dans le fichier source. De plus on peut remarquer que le widget se trouvera dans la catégorie position de l'accordion.

MeasureArea

Ce widget permet de mesurer des surfaces.

```
YUI.add("geoconcept-widget-measure-area", function(Y) {

        [...]

        Y.extend(MeasureArea, Y.GCUI.MeasureBase,  {

                [...]

            /**
             * Manage widget action
             */
                activate:function() {
```

```
                        MeasureArea.superclass.activate.apply(this,[this.getCallbacks()]);
                        this.createPanelMeasure();
                        this.showPanel();
                },

                deactivate:function() {
                        MeasureArea.superclass.deactivate.apply(this);
                },

                /**
                 * Manage default panel
                 */
                // Create panel for area information
                createPanelMeasure:function() {
                var panelNode = this.createPanel(this._("widget.measureArea.popup.title"));
                        var id = this.getPanelId();

                var areaInfo = Y.Node.create("<div id='" + id + "' class='areaMeasure'></div>");
                areaInfo.append("<div class='areaSegmentTitle'>" +
 this._("widget.measureArea.segment.measure") + "</div>");
                areaInfo.append("<div id='" + MeasureArea.ID_PANEL_SEGMENT_MEASURE + id + "'
 class='areaSegmentValue'>0.000 " + this._("widget.measureArea.segment.kilometer") + "<sup>2</sup></div>");

                        panelNode.appendChild(areaInfo);
            }
        });
        Y.namespace("GCUI");
        Y.GCUI.MeasureArea = MeasureArea;

}, "1.0.0", {requires: ["geoconcept-widget-measure-base"] });
```

Ce widget étend Y.GCUI.MeasureBase utilisant le Control OpenLayers de mesure, *OpenLayers.Control.Measure* Ce widget utilise le panel pour afficher son résultat.

Layers

Le widget layers n'est pas un bouton mais affiche un TreeView sur un div donné.

```
YUI.add("geoconcept-widget-layers", function(Y) {

        [...]

        Y.extend(Layers, Y.GCUI.LayersBase, {

                [...]

                configureAction : function() {
                        Layers.superclass.configureAction.apply(this);

                        if (!this.isExistingLayers) {
                                var yuiNode = Y.one("#"+this.getId());
                                if (!this.isPortal()) {
                                        yuiNode.setContent(this.getWrapper());
                }
                                var map = this.getMap();
                                var wLayers = this;
                                var options = {
                                        div : document.getElementById(this.getId())
                                };
                                var ls = new GCUI.Control.LayerSwitcher(options);
                                map.addControl(ls);
```

```
                                var json = this.initJson();
                                ls.initFromJson(json);

                                yuiNode.addClass("addedWidget");

                                if (this.isPortal()) {
                        var layers = map.layers;
                                        for (var i = 0; i < layers.length; i++) {
                                                var layer = layers[i];
                                                layer.events.on({
                                            "visibilitychanged" : wLayers.layerChangeVisibility,
                                            scope : wLayers
                                        });
                                                }
                                                this.showLegends(layers);
                        }
                                }
                        },

                        [...]

                });
                Y.namespace("GCUI");
                Y.GCUI.Layers = Layers;

}, "1.0.0", {requires : ["geoconcept-widget-layers-base"] });
```

Ce widget étend Y.GCUI.LayersBase qui étend lui même Y.GCUI.WidgetBase. Le TreeView est généré par GCUI.Control.LayerSwitcher. Il affiche la liste des couches à afficher dans un JSON.

Contact

Ce widget a la particularité d'utiliser des propriétés, ces propriétés sont définies dans son fichier de description XML :

```
<widget id="contact"
                icon="resources/images/contact.png"
                js="general/Contact.js"
                module="geoconcept-widget-contact"
                display="true"
                lang="resources/widgets">
        <properties>
                <property id="mailContact" type="text"/>
        </properties>
</widget>
```

Depuis le client il sera donc possible de déterminer l'adresse mail dans un champ texte.

Le fichier source prend en compte ces propriétés qui lui sont injectées en JSON par le client. En effet, le moteur de widgets va lire le fichier XML du widget, injecter le JSON correspondant au client qui créera une instance du widget et le client se chargera d'instancier le widget avec le descriptif JSON. Ce descriptif JSON est *setté* à l'instance. A chaque changement d'une des valeurs des propriétés, le JSON est mis à jour. Lors de la sauvegarde du builder, le fichier JSON de chaque instance est récupéré pour être ensuite sauvegardé par le builder. (cf partie client)

Voici le code source du widget :

```
YUI.add("geoconcept-widget-contact", function(Y) {

        [...]

        Contact.MAIL_CONTACT_PROPERTY = "mailContact"

        [...]

        Y.extend(Contact, Y.GCUI.WidgetButton,  {

                [...]

                configureAction: function() {
                        Contact.superclass.configureAction.apply(this);
                        var defaultMail = this.getPropertyValue(Contact.MAIL_CONTACT_PROPERTY);
                        if (defaultMail == undefined || defaultMail == "") {
                                this.updatePropertyValue(Contact.MAIL_CONTACT_PROPERTY,
 this.getParameters().contact);
                        }
                },

                trigger: function() {
                        window.location = "mailto:" + this.getPropertyValue(Contact.MAIL_CONTACT_PROPERTY);
                }

        });
        Y.namespace("GCUI");
        Y.GCUI.Contact = Contact;

}, "1.0.0", {requires: ["project-widgetbutton"] });
```

La partie importante ici est la gestion des properties. En parsant le fichier XML, le moteur de widgets va appeler la `addProperties()` lors de la sélection de l'instance dans l'outil de conception si le widget possède des propriétés. `updatePropertyValue()` est utilisée ici pour mettre à jour la valeur de la propriété dans le JSON. Il sera ensuite sauvegardé dans la description du projet puis stocké dans la base de données.

Tutorial

Introduction

Avant tout propos, la lecture des documentations de YUI [http://yuilibrary.com/] et OpenLayers [http://openlayers.org/] est fortement recommandée pour réaliser des widgets. De nombreux exemples sont à disposition YUI exemples [http://yuilibrary.com/yui/docs/guides/] et OpenLayers exemples [http://openlayers.org/dev/examples/]. De plus, la communauté autour de ces 2 librairies est très active et de nombreuses réponses se trouvent sur les forums.

Ce tutorial présente plusieurs exemples pour créer son propre widget. Avant de commencer, il convient de s'assurer qu'Eclipse est installé sur votre poste ainsi que le module Maven. Il vous permettra de générer un fichier JAR qui contiendra vos widgets.

Easy geoweb accepte plusieurs fichiers JAR avec des widgets mais chaque widget doit posséder un identifiant unique.

Un widget est un module qui sera chargé dynamiquement dans le portail et qui se compose de 3 fichiers :

- `fichier XML` pour décrire les ressources nécessaires (javascript, properties, image) et pour déterminer l'identification du module.
- `fichier JS` pour configurer les actions et les interactions du widget avec le portail.
- `fichier CSS` pour paramétrer le style du widget.

L'ensemble des sources nécessaires à la réalisation de ce tutorial est fourni avec la documentation.

Projet Maven

Création du projet

Dans Eclipse, créer un nouveau projet Maven.

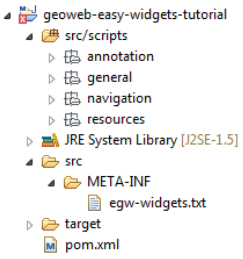
Choisir Create a simple project (skyp archetype selection). Pour ce tutorial, la description de l'artifact est :

- `Group Id` : com.geoconcept.geoweb.product
- `Artifact Id` : geoweb-easy-widgets-tutorial
- `Version` : 0.0.1-SNAPSHOT
- `Packaging` : jar

Une fois le nouveau projet créé, supprimer la classe principale java et le répertoire *test* et *main* dans *src*. Dans le répertoire *src*, créer un répertoire *scripts* avec Source Folder (clic droit sur le projet, New → Source Folder).

Ensuite créer un répertoire *META-INF* dans *src*. Il contiendra le fichier *egw-widgets.txt*. A l'intérieur de ce fichier, il est nécessaire d'inscrire une chaine de caractère, par exemple "easy-geoweb widgets module". Ce fichier permet d'identifier et de filtrer les JAR pour conserver uniquement ceux qui possède des widgets.

L'organisation finale est la suivante :



Créer 4 nouveaux répertoires dans *src/scripts* :

- general : contiendra les fichiers des nouveaux widgets. Ce répertoire sera visible dans l'accordéon du Composer. Il existe plusieurs catégories/répertoires disponibles pour le Composer (Annotation/*annotation*, Informations sur les objets/*data*, Général/*general*, Mise en page/*layout*, Mesures/*measure*, Modifications des données/*modification*, Navigation/*navigation*, Vue/*view*). En plus de general, ajouter les répertoires *annotation* et *navigation* pour réaliser les exemples.
- resources : contiendra les fichiers de ressources (images, localisation,…).

Dans resources créer :

- images : contiendra toutes les images utilisées
- exampleLang.properties : fichier contenant toutes les chaines de localisation par défaut. Pour ajouter de nouvelles langues aux widgets, il faut dupliquer ce fichier en ajoutant au nom du fichier la localisation souhaitée. Par exemple, pour une localisation française, le fichier devra se nommer exampleLang_fr.properties.

Attention, un JAR de widget doit contenir uniquement des fichiers liés aux widgets.

Configuration du POM

Dans le fichier pom.xml, ajouter les éléments suivants afin de construire le projet au format JAR :

```
<build>
	<plugins>
		<plugin>
			<groupId>org.apache.maven.plugins</groupId>
			<artifactId>maven-jar-plugin</artifactId>
			<configuration>
				<classesDirectory>src</classesDirectory>
				<outputDirectory>D:/widgets</outputDirectory>
				<excludes>
					<exclude>**/pom.xml</exclude>
				</excludes>
				<archive>
					<addMavenDescriptor>false</addMavenDescriptor>
				</archive>
			</configuration>
		</plugin>
	</plugins>
</build>
```

classesDirectory est le chemin d'accès aux fichiers des widgets.

outputDirectory est le chemin dans lequel sera déposé le fichier JAR généré.

Widget afficher une légende

Objectif

Ce widget permet d'afficher la légende de la carte. Il s'agit d'un widget de type bouton. La légende correspond à une image affichée dans un panneau.

Le widget est nommé ExampleLegend.

Initialisation

Créer les 3 fichiers du widget dans le répertoire *general* :

- ExampleLegend.xml
- ExampleLegend.js
- ExampleLegend.css

ExampleLegend.xml

Cet exemple est le plus simple. Voici la configuration du fichier de description :

```
<widget id="exampleLegend"
        category="standard"
        position="12"
        icon="resources/images/exampleLegend.png"
        js="general/ExampleLegend.js"
        module="geoconcept-widget-example-legend"
        display="true"
        lang="resources/exampleLang">
</widget>
```

- **`id`** : identifiant unique
- **`icon`** : chemin d'accès à la ressource image pour l'afficher comme bouton.
- **`js`** : chemin d'accès au fichier de configuration de l'action du widget.
- **`module`** : nom du module qui sera appelé dans le widget manager pour charger l'instance du widget.
- **`display`** : booléen pour déterminer si le widget sera affiché dans le Composer. Utiliser false pour créer un sous-widget qui proposera des fonctionnalités commune à d'autres widgets.
- **`lang`** : chemin d'accès au fichier de localisation.

Pour plus d'informations, voir la description des fichiers XML dans le chapitre des widgets.

ExampleLegend.js

Le fichier js permet de construire un nouveau module. Le widget créé est un bouton simple, l'action sera executée à chaque fois que l'utilisateur effectuera une pression sur le bouton. La méthode appelée suite à un clic souris est trigger:function(). Cette méthode est définie dans WidgetBase mais il convient ici de la surcharger afin d'afficher l'image de la légende. Voici la méthode finale :

```
trigger:function() {
        var panelNode = this.createPanel(this._("widget.exampleLegend.popup.title"));

        var linkNode = Y.Node.create(this.getImageHtml());
        panelNode.appendChild(linkNode);

        this.showPanel();
},
```

Les méthodes _(key), createPanel(name) et showPanel() appartiennent à l'API de base d'un widget. Chaque widget possède une fenêtre (panel) qui est liée avec un identifiant unique. Il est possible de créer d'autres fenêtres mais il faudra les définir directement de le widget. La clé de localisation doit être ajoutée au fichier properties du jar.

La partie HTML de l'intérieur de la fenêtre est défini par la méthode getImageHtml(). Un noeud est ensuite créé puis ajouté à la fenêtre :

```
getImageHtml:function() {
        var imageUrl = "/" + this.getContextUrl() + "/combo?resources/images/sample_legende.png";
        var html = "<img src='" + imageUrl + "' class='imageLegend'>";
```

```
        return html;
}
```

Dans cette méthode, la balise HTML est créée. La méthode getContextUrl() appartient à l'API de base d'un widget. L'image est stockée dans les ressources du widget. Une classe est associée afin de définir la position de l'image dans la fenêtre.

A présent, voici la description complète de ce widget :

```
YUI.add("geoconcept-widget-example-legend", function(Y) {

        /* ExampleLegend class constructor */
        function ExampleLegend(config) {
                ExampleLegend.superclass.constructor.apply(this, arguments);
        }

        /*
         * Required NAME static field, to identify the Widget class and
         * used as an event prefix, to generate class names etc. (set to the
         * class name in camel case).
         */
        ExampleLegend.NAME = "exampleLegend";

        Y.extend(ExampleLegend, Y.GCUI.WidgetButton,  {
                initializer: function(config) {
                        ExampleLegend.superclass.initializer.apply(this);
                },

                destructor: function() {
                        ExampleLegend.superclass.destructor.apply(this);
                },

                configureAction: function() {
                        ExampleLegend.superclass.configureAction.apply(this);
                },

                /*
                 * Action after click on button
                 */
                trigger:function() {
                        var panelNode = this.createPanel(this._("widget.exampleLegend.popup.title"));

                        var linkNode = Y.Node.create(this.getImageHtml());
                        panelNode.appendChild(linkNode);

                        this.showPanel();
                },

                /*
                 * Get HTML description for image legend
                 * return string html
                 */
                getImageHtml:function() {
                        var imageUrl = "/" + this.getContextUrl() + "/combo?resources/images/
sample_legende.png";
                        var html = "<img src='" + imageUrl + "' class='imageLegend'>";
                        return html;
                }
        });
        Y.namespace("GCUI");
        Y.GCUI.ExampleLegend = ExampleLegend;
```

```
}, "1.0.0", {requires: ["project-widgetbutton"] });
```

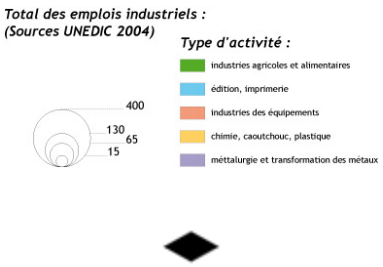
ExampleLegend.css

Dans cet exemple, une classe a été ajoutée pour définir les marges autour de la légende pour améliorer son positionnement dans la fenêtre.

```
.imageLegend {
        margin: 5px;
}
```

Ressources

Voici les images nécessaires à la réalisation de ce widget.



Placer ces images dans le répertoire *src/scripts/resources/images*.

Par défaut, un bouton possède 2 clés de localisation, *label* et *description*. *label* est utilisé dans l'accordéon du Composer et *description* est affiché lorsque la souris survole le bouton.

Attention les clés doivent obligatoirement commencer par widget.(id du widget).(fin de la clé). Pour ce widget, la clé du label sera widget.exampleLegend.label

Voici les clés de base à placer dans le fichier properties :

```
widget.exampleLegend.label=Exemple de l\u00E9gende
widget.exampleLegend.description=Afficher la l\u00E9gende principale
```

Voici la clé pour afficher un titre dans la fenêtre :

```
widget.exampleLegend.popup.title=L\u00E9gende
```

Widget dessiner un point

Objectif

Ce widget permet de dessiner un point sur la carte. Il s'agit d'un widget de type bouton poussoir, en cliquant dessus, l'utilisateur active la fonctionnalité. Une propriété dans le Composer permettra de déterminer l'image stockée en base de données à afficher après un clic sur la carte.

Le widget est nommé ExampleDrawPOI.

Initialisation

Créer les 3 fichiers du widget dans le répertoire *annotation* :

- ExampleDrawPOI.xml
- ExampleDrawPOI.js
- ExampleDrawPOI.css

ExampleDrawPOI.xml

Cet exemple permet de rajouter une propriété au widget. Voici la configuration du fichier de description :

```
<widget id="exampleDrawPOI"
                category="standard"
                position="13"
                icon="resources/images/exampleDrawPOI.png"
                js="annotation/ExampleDrawPOI.js"
                module="geoconcept-widget-example-draw-poi"
                display="true"
                lang="resources/exampleLang">
        <properties>
                <property id="defaultImage" type="combobox">
                        <params value="poi_blue_small"/>
                </property>
        </properties>
</widget>
```

- **`properties`** : permet d'ajouter des propriétés au widget.
- **`property`** : détail d'une propriété composé d'un id unique et d'un type.
- **`params`** : paramètre par défaut pour la propriété correspondante. Cette balise n'est pas obligatoire dans le cas où il n'y a pas de valeur par défaut.

Pour plus d'informations, voir la description des fichiers XML dans le chapitre des widgets.

ExampleDrawPOI.js

Le widget créé est un bouton de type TOOL, lorsque l'utilisateur clique sur le bouton, il reste activé.
Un seul bouton de type TOOL peut être activé à la fois, tous les autres boutons de ce type sont alors désactivés. Les deux méthodes utilisés sont activate() et deactivate(). A l'activation, le widget appelle une fonction pour activer l'écouteur d'un clic sur la carte. Pour la désactivation, l'écouteur sera supprimé. Voici les 2 méthodes de base du widget :

```
activate:function() {
        ExampleDrawPOI.superclass.activate.apply(this);
        this.createPoiListener();
},

deactivate:function() {
        ExampleDrawPOI.superclass.deactivate.apply(this);
        this.removeCreatePoiListener();
},
```

La création d'un écouteur suite à un clic souris sur la carte est basée sur l'API Javascript [http://api.geoconcept.net/htc/index.html]

```
createPoiListener:function() {
        var wExampleDrawPOI = this;

        function clickListener(dynMap){}

        clickListener.prototype.onSelect = function(x,y,xpx,ypx){
                wExampleDrawPOI.createObject(x,y);
        };

        var listener = new clickListener();
        DynMapAddMouseSelectionEventListener(this.getMap(), "clickOnMap", listener)
},
```

La méthode s'appuie sur la fonction DynMapAddMouseSelectionEventListener dans laquelle il est possible de lui passer un écouteur (*listener*) en paramètre. Un clic souris active la fonction createObject(x,y) du widget. Les coordonnées x et y permettront de positionner l'objet sur la carte.

L'ajout du point se fait dans la couche objet de l'API Javascript. L'image affichée pour représenter le point est définie à partir d'une url.

```
var imageUrl = this.getHost() + this.getContextUrl() + "/Image/showImage.do?name=" +

 this.getPropertyValue(ExampleDrawPOI.IMAGE_PROPERTY_NAME);
```

Les méthodes getHost(), getContextUrl() et getPropertyValue(name) appartiennent à l'API de base d'un widget. ExampleDrawPOI.IMAGE_PROPERTY_NAME est une constante définie dans le module de la façon suivante :

```
ExampleDrawPOI.IMAGE_PROPERTY_NAME = "defaultImage";
```

getPropertyValue(name) est la méthode permettant de rechercher la valeur d'une propriété. Chaque widget peut bénéficier de propriétés qu'il faut définir dans son fichier de description. Tous les widgets possèdent par défaut une propriété sur le choix des groupes ayant accès à ce widget dans le cas d'un portail privé. Les widgets de type bouton possèdent en plus une propriété pour définir si le widget est activé par défaut. Les autres propriétés sont créées automatiquement en fonction du fichier de description mais si le type de propriété n'existe pas, il est alors possible d'utiliser la méthode `addProperties(groups)`.

Pour récupérer les images de la base de données, il faut faire appel à la méthode getImages de l'API REST. Le résultat est retourné dans la méthode onSuccess() ou onError(), en cas de succès ou d'échec de la recherche.

```
this.getEGWApi().getImages({
        onSuccess:function(response) {
        },
        onError:function(response) {
        }
});
```

A la suite de la récupération des images, il faut associer le résultat à la liste déroulante de la propriété en utilisant la méthode `setComboboxOptions(propertyId, data)`

```
this.setComboboxOptions(ExampleDrawPOI.IMAGE_PROPERTY_NAME, images);
```

A présent, voici la description complète de ce widget :

```
YUI.add("geoconcept-widget-example-draw-poi", function(Y) {

        /* ExampleDrawPOI class constructor */
        function ExampleDrawPOI(config) {
                ExampleDrawPOI.superclass.constructor.apply(this, arguments);
        }

        /*
         * Required NAME static field, to identify the Widget class and
         * used as an event prefix, to generate class names etc. (set to the
         * class name in camel case).
         */
        ExampleDrawPOI.NAME = "exampleDrawPOI";
        ExampleDrawPOI.IMAGE_PROPERTY_NAME = "defaultImage";

        Y.extend(ExampleDrawPOI, Y.GCUI.WidgetButton,  {
                initializer: function(config) {
                        ExampleDrawPOI.superclass.initializer.apply(this);
                        this.setType(Y.GCUI.WidgetButton.TYPE_TOOL);
                        this.getImages();
                },

                destructor: function() {
                        ExampleDrawPOI.superclass.destructor.apply(this);
                },

                configureAction: function() {
                        ExampleDrawPOI.superclass.configureAction.apply(this);
                },

                activate:function() {
                        ExampleDrawPOI.superclass.activate.apply(this);
                        this.createPoiListener();
            },

            deactivate:function() {
                ExampleDrawPOI.superclass.deactivate.apply(this);
                this.removeCreatePoiListener();
            },

            createPoiListener:function() {
                        var wExampleDrawPOI = this;

                        function clickListener(dynMap){}

                        clickListener.prototype.onSelect = function(x,y,xpx,ypx){
                                wExampleDrawPOI.createObject(x,y);
                        };

                        var listener = new clickListener();
                        DynMapAddMouseSelectionEventListener(this.getMap(), "clickOnMap", listener)
                },

                // Remove create poi listener
                removeCreatePoiListener:function() {
                        DynMapRemoveListener(this.getMap(),"click","clickOnMap");
                },
```

```
            // add points from list of citysite
                createObject:function(x,y) {
                        var map = this.getMap();
                        var imageUrl = this.getHost() + this.getContextUrl() + "/Image/showImage.do?name=" +

  this.getPropertyValue(ExampleDrawPOI.IMAGE_PROPERTY_NAME);
                        var poi = {
                        mapx : x,
                        mapy : y,
                        imgsrc : imageUrl
                    };
                    map.objectLayer.addObject(poi);
                        map.objectLayer.refresh();
                },

                /**
                 * Method: Get images on geoweb server. Call egw api rest method
                 */
                getImages : function() {
                        var wExampleDrawPOI = this;

                        this.getEGWApi().getImages({
                                onSuccess : function(response) {
                                        // For images properties, we add the images of geoweb
                                        // database
                                        var result = response.result;
                                        var images = [""];

                                        for (i = 0; i < result.length; i++) {
                                                var image = result[i];
                                                images.push(image.name);
                                        }


  wExampleDrawPOI.setComboboxOptions(ExampleDrawPOI.IMAGE_PROPERTY_NAME, images);
                                },
                                onError : function(response) {
                                        alert("Poi getImages error " + response.status + ", "
                                                        + response.message);
                                }
                        });
                }
        });
        Y.namespace("GCUI");
        Y.GCUI.ExampleDrawPOI = ExampleDrawPOI;

}, "1.0.0", {requires: ["project-widgetbutton"] });
```

ExampleDrawPOI.css

Dans cet exemple, nous n'utiliserons pas de style par défaut.

Ressources

Voici l'image nécessaire à la réalisation de ce widget.

Voici les clés de base à placer dans le fichier properties :

```
widget.exampleDrawPOI.label=Exemple de points
widget.exampleDrawPOI.description=Cr\u00E9er un point sur la carte
```

Pour l'intitulé de la propriété du choix de l'image, il est nécessaire d'ajouter la clé suivante :

```
widget.exampleDrawPOI.defaultImage=Images :
```

Pour plus d'information sur les ressources, se référer au widget ExampleLegend.

Widget afficher les coordonnées X et Y

Objectif

Ce widget permet d'afficher les coordonnées X et Y suite à un clic sur la carte.

Le widget est nommé ExampleXY.

Initialisation

Créer les 3 fichiers du widget dans le répertoire *navigation* :

- ExampleXY.xml
- ExampleXY.js
- ExampleXY.css

ExampleXY.xml

Voici la configuration du fichier de description :

```
<widget id="exampleXY"
        category="standard"
        position="14"
        icon="resources/images/exampleXY.png"
        js="navigation/ExampleXY.js"
        module="geoconcept-widget-example-xy"
        display="true"
        lang="resources/exampleLang">
</widget>
```

Pour plus d'informations, voir la description des fichiers XML dans le chapitre des widgets.

ExampleXY.js

Le widget créé est de type basique. Il ne possède aucune forme, c'est au développeur de réaliser son interface. L'ihm du widget est à définir dans la méthode getWidget(). Pour l'affichage dans l'outil de conception, le widget doit être désactivé. Pour cela, 2 méthodes sont disponibles dans l'API de base d'un widget, isPortal() et getWrapper(). La 1ère détermine si l'application est affichée dans le portail ou l'outil de conception. La seconde permet de récupérer le masque que l'on pourra superpositionner pour désactiver le widget.

```
getWidget: function() {
        var htmlContent = "<div class='xyBG'><div class='xyField'><label>"+this._("widget.exampleXY.x") +
        "</label></br><input type='text' id='x' size='15' disabled='disabled'/></div>" +
    "<div class='xyField'><label>" + this._("widget.exampleXY.y") +
```

```
    "</label></br><input type='text' id='y' size='15' disabled='disabled'/></div>" +
    "<div class='xyButton egw-btn'><button id='searchXY'>" + this._("widget.exampleXY.button") + "</
button></div>";

        if (!this.isPortal()) {
                htmlContent = htmlContent + this.getWrapper();
        }
        htmlContent += "</div>";
        return ("<div id='"+this.getId()+"'>"+htmlContent+"</div>");
},
```

Comme pour l'exemple ExampleDrawPOI, ce widget utilise un écouteur lors du clic souris. L'activation se fera lors du clic sur le bouton *searchXY*. L'évènement positionné dans la fonction *configureAction()* activera l'écouteur. La fonction *configureAction()* est appelée suite à la construction du widget dans l'ihm.

```
configureAction: function() {
        ExampleXY.superclass.configureAction.apply(this);
        if (this.isPortal()) {
                Y.one("#searchXY").on("click", this.clickOnMapListener, this);
        }
},
```

Pour désactiver l'écouteur, la fonction surchargée est *reinitView()*. Cette fonction peut être appelée par l'évènement Y.fire("reinitView") (utilisé notamment par le widget Erase fourni par défaut).

```
reinitView: function() {
        this.removeClickOnMapListener();
},
```

Suite à l'activation de ce widget, chaque clic sur la carte permettra d'afficher les coordonnées x et y de la souris.

```
setXY: function(x,y) {
        var map = this.getMap();
        Y.one("#x").set("value", x*map.precision);
        Y.one("#y").set("value", y*map.precision);
}
```

A présent, voici la description complète de ce widget :

```
YUI.add("geoconcept-widget-example-xy", function(Y) {

        /* ExampleXY class constructor */
        function ExampleXY(config) {
                ExampleXY.superclass.constructor.apply(this, arguments);
        }

        /*
         * Required NAME static field, to identify the Widget class and
         * used as an event prefix, to generate class names etc. (set to the
         * class name in camel case).
         */
        ExampleXY.NAME = "exampleXY";

        Y.extend(ExampleXY, Y.GCUI.WidgetBase,  {
                initializer: function(config) {
                        ExampleXY.superclass.initializer.apply(this);
                },
```

```
                destructor: function() {
                        ExampleXY.superclass.destructor.apply(this);
                },

                configureAction: function() {
                        ExampleXY.superclass.configureAction.apply(this);
                        if (this.isPortal()) {
                                Y.one("#searchXY").on("click", this.clickOnMapListener, this);
                        }
                },

                reinitView: function() {
                        this.removeClickOnMapListener();
                },

                getWidget: function() {
                        var htmlContent = "<div class='xyBG'><div
  class='xyField'><label>"+this._("widget.exampleXY.x") +
                        "</label></br><input type='text' id='x' size='15' disabled='disabled'/></div>" +
                    "<div class='xyField'><label>" + this._("widget.exampleXY.y") +
                    "</label></br><input type='text' id='y' size='15' disabled='disabled'/></div>" +
                    "<div class='xyButton egw-btn'><button id='searchXY'>" +
  this._("widget.exampleXY.button") + "</button></div>";

                        if (!this.isPortal()) {
                                htmlContent = htmlContent + this.getWrapper();
                        }
                        htmlContent += "</div>";
                        return ("<div id='"+this.getId()+"'>"+htmlContent+"</div>");
                },

                clickOnMapListener: function() {
                        var wExampleXY = this;

                        function clickListener(){}

                        clickListener.prototype.onSelect = function(x,y,xpx,ypx){
                                wExampleXY.setXY(x,y);
                        };

                        var listener = new clickListener();
                        DynMapAddMouseSelectionEventListener(this.getMap(), "getXY", listener)
                },

                // Remove create poi listener
                removeClickOnMapListener: function() {
                        DynMapRemoveListener(this.getMap(),"click","getXY");
                },

                setXY: function(x,y) {
                        var map = this.getMap();
                        Y.one("#x").set("value", x*map.precision);
                        Y.one("#y").set("value", y*map.precision);
                }
        });
        Y.namespace("GCUI");
        Y.GCUI.ExampleXY = ExampleXY;

  }, "1.0.0", {requires: ["project-widgetbase"] });
```

ExampleXY.css

Dans cet exemple, le widget n'étant pas un bouton, le fichier css doit être enrichit pour personnaliser l'outil. Ci-dessous se trouve une des possibilités d'affichage du widget.

```
.xyField {
        padding: 0 10px;
        float: left;
}

.xyButton {
        float: left;
        padding: 13px 0px 0px;
}

.xyBG {
        width: 100%;
        height: 40px;
}

.xyField label{
        font-weight: bold;
}
```

Ressources

Voici l'image nécessaire à la réalisation de ce widget.

Voici les clés de base à placer dans le fichier properties :

```
widget.exampleXY.label=Exemple de coordonn\u00E9es
widget.exampleXY.description=R\u00E9cup\u00E9rer les coordonn\u00E9es \u00E0 partir d\'un clic
```

Pour les autres textes du widget, il est nécessaire d'ajouter les clés suivantes :

```
widget.exampleXY.button=Activer
widget.exampleXY.x=X
widget.exampleXY.y=Y
```

Pour plus d'information sur les ressources, se référer au widget ExampleLegend.

API Javascript

L'API Javascript fournit une bibliothèque complète pour interagir avec la carte dans une interface web.

La documentation [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-js-intro.html] décrit les fonctions Javascript disponibles pour construire votre application web et l'enrichir de fonctionnalités.

Web Services

Les différents types de web services qui sont fournis par Geoconcept Web sont explicités dans les parties suivantes, en prenant comme entrées les différents types d'architecture réalisables. Pour chaque type de service web, ce manuel décrit quelles sont les fonctionnalités offertes, comment les paramétrer et les utiliser, à l'aide d'exemples de requêtes et de réponses ou encore de FAQ.

WMS et WMTS

En plus des composants paramétrables et utilisables à travers Geoconcept Web, comme décrit dans les paragraphes précédents, Geoconcept Web permet de diffuser des cartes Geoconcept en respectant les normes de l'Open Geospatial Consortium (OGC). Ainsi, deux types de Web Services permettent de diffuser une carte Geoconcept :

- Web Map Service (WMS) : il s'agit d'un standard OGC de service web qui permet de produire dynamiquement des cartes à partir de données géoréférencées ; le serveur est alors sollicité à chaque requête d'un poste client. La carte Geoconcept peut alors être partagée sur un serveur et visible à travers des postes clients lisant le format WMS, comme le SIG Geoconcept,
- Web Map Tiled Service (WMTS) : il s'agit d'un standard décrivant la manière de diffuser des données cartographiques sous forme de tuiles prédéfinies. Le WMTS est un complément au WMS. Par rapport au WMS, le WMTS a pour avantage principal d'offrir de meilleures performances dans la diffusion de données cartographiques, les tuiles étant stockées en cache lors de la première génération. En contrepartie, le WMTS n'est pas recommandé lorsque les données sont mises à jour fréquemment : l'invalidation du cache devenant alors nécessaire pour que les données soient à jour.

Pré-requis

Le module Administration de l'application web offre la fonctionnalité de publier les onglets de visibilité des cartes Geoconcept au format WMS ou WMTS, en respectant les normes de l'OGC.

Il est donc nécessaire d'avoir déployé l'application web en suivant les étapes décrites dans le guide d'installation de Geoconcept Web.

Création des couches au format WMS ou WMTS

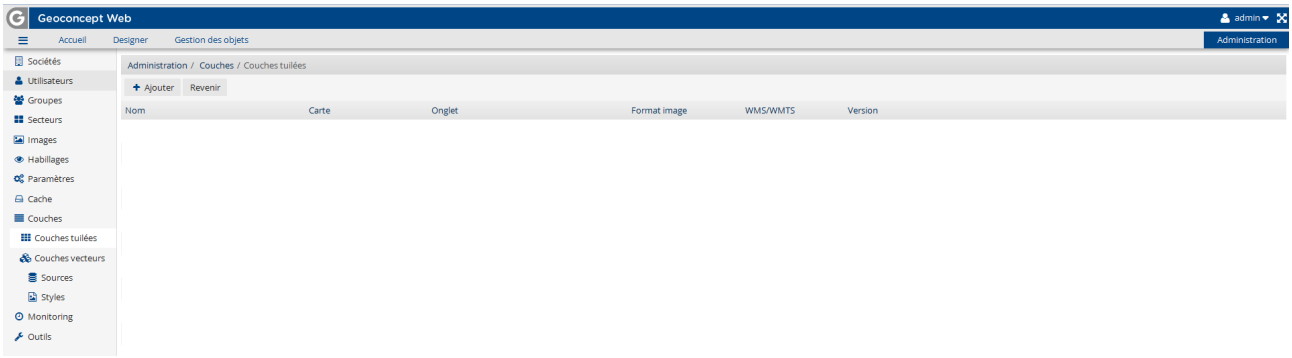
Cette fonctionnalité dispose d'une interface pour créer les couches qui seront diffusées au format WMS ou WMTS.

> Ne pas oublier d'activer les services WMS ou WMTS (se reporter au paragraphe dans Installation de Geoconcept Web / Paramètres / Activation des WMS et WMTS de Geoconcept Web.

La publication d'un onglet de visibilité au format WMS ou WMTS se déroule de la façon suivante :

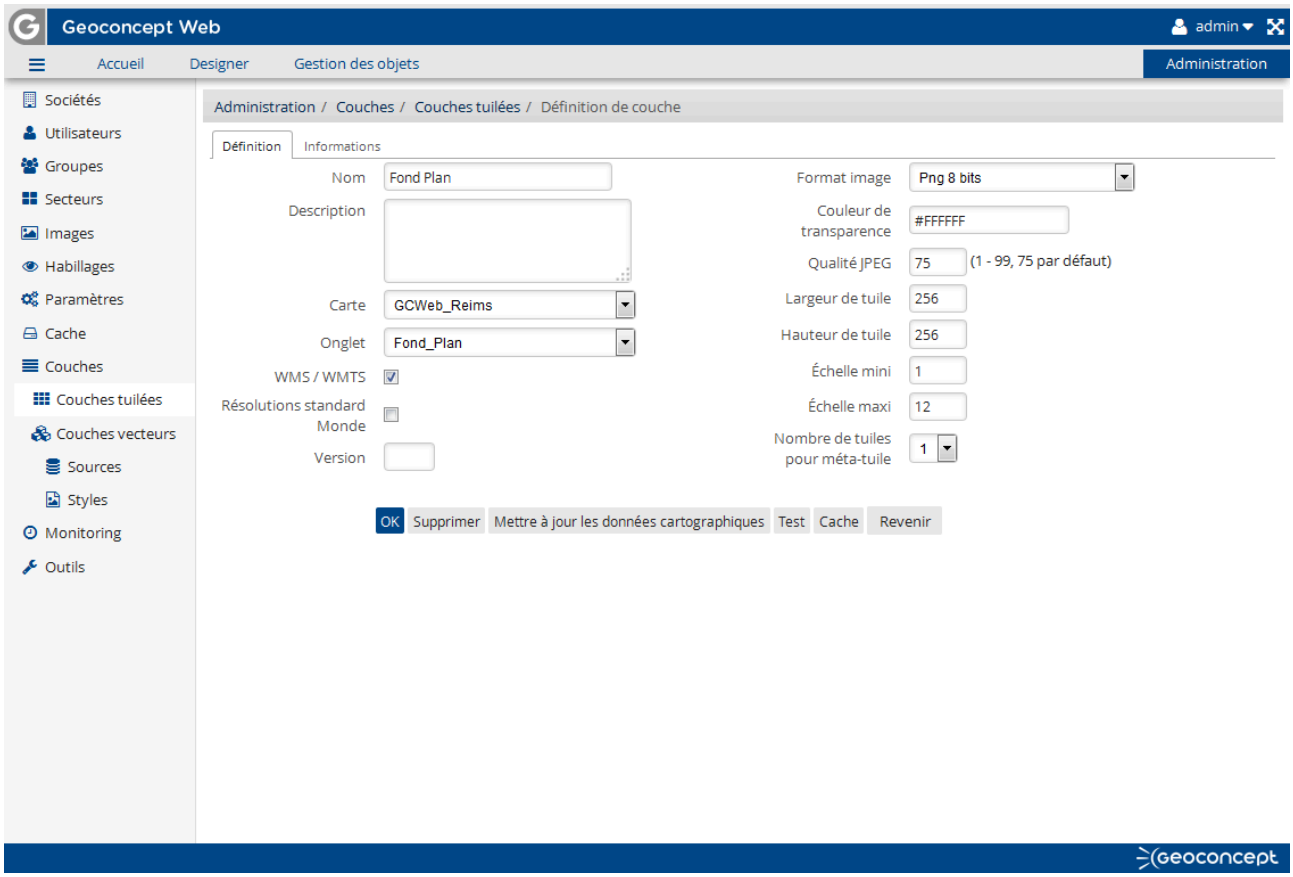
- Dans l'onglet Administration puis dans le menu Couches / Couches tuilées, choisir Ajouter :

Ajouter une couche



- Dans l'interface de saisie, renseigner les champs suivants :
  - Nom : renseigner le nom qui sera visible par les utilisateurs lorsqu'ils interrogeront le service WMS ou WMTS depuis une application tierce (par exemple le SIG Geoconcept),
  - la case « WMS/WMTS » doit être cochée,
  - Carte : choisir la carte à publier parmi celles proposées dans la liste déroulante,
  - Onglet : choisir l'onglet de visibilité de la carte à publier,
  - Choisir le format image souhaité :
    - Le format PNG est conseillé comme format d'image car permet une meilleure résolution des images générées,
    - Le format PNG with transparency permet de transformer la « couleur blanche » d'une carte Geoconcept en transparent dans l'image générée. Ce format est utile pour afficher des couches superposées entre elles.

Saisir les éléments relatifs à l'onglet de visibilité de Geoconcept



> Par précaution, ne pas mettre de caractères spéciaux ni d'espace dans le nom de la couche, le nom de la carte et le nom de l'onglet de visibilité.

- Une fois ces paramètres renseignés, cliquer sur OK .

Résultats

Une couche configurée dans le portail est accessible au format WMS ou WMTS avec la même configuration. Le format demandé dépend du client : s'il souhaite du WMS, il le spécifie dans sa requête avec le terme WMS. La façon de procéder est la même avec le WMTS.

> Chaque couche publiée est disponible dans trois systèmes de projection :
> - la projection d'origine de la carte Geoconcept
> - en latitude / longitude WGS84 (epsg:4326)
> - en Mercator sur sphère (epsg:3857)

WMS - Web Map Service

GetCapabilities :

http://<server>/<webapp>/wms?request=GetCapabilities&service=WMS

Exemple : http://gcweb.geoconcept.com/gws/wms?request=GetCapabilities&service=WMS

L'onglet de visibilité de votre carte Geoconcept est publié au format WMS. Il est accessible via un outil de visualisation tiers capable de lire le format WMS par la requête suivante :

http://<server>/<webapp>/wms

Exemple de requête WMS de demande d'image : http://gcweb.geoconcept.com/gws/wms?request=GetMap&VERSION=1.3.0&CRS=epsg:27572&BBOX=551533.765952,2313284.041120,566370.515430,2321%2fpng&WIDTH=800&HEIGHT=600

WMTS - Web Map Tile Service

L'application supporte les protocoles KVP et REST pour le WMTS. La norme OGC WMTS 1.0.0 est supporté (http://www.opengeospatial.org/standards/wmts).

GetCapabilities :

- KVP encoding : http://<server>/<webapp>/wmts?request=GetCapabilities&version=1.0.0&service=WMTS

Exemple : http://gcweb.geoconcept.com/gws/wmts?request=GetCapabilities&version=1.0.0&service=WMTS

- REST encoding : http://<server>/<webapp>/wmts/1.0.0/WmtsCapabilities.xml

Exemple : http://gcweb.geoconcept.com/gws/wmts/1.0.0/WmtsCapabilities.xml

http://<server>/<webapp>/gws/wmts?request=GetCapabilities&service=WMTS

GetTile :

- KVP encoding : http://<server>/<webapp>/wmts?SERVICE=WMTS&REQUEST=GetTile&VERSION=1.0.0&LAYER=<layerIdentifier>&STYLE=<StyleIdentifier>&TILE
png

Exemple : http://gcweb.geoconcept.com/gws/wmts?SERVICE=WMTS&REQUEST=GetTile&VERSION=1.0.0&LAYER=France&STYLE=Server2&TILEMATRIXSET=epsg%3A27572&TILEMATRIX=0&TILEROW=0&TILECOL=0&FORMAT=image%2Fpng

- REST encoding : la ResourceURL d'un layer est décrite dans le getCapabilities : http://<server>/<webapp>/wmts/<layerIdentifier>/<styleIdentifier>/<TileMatrixSet>/<TileMatrix>/<TileRow>/<TileCol>.png

Exemple : http://gcweb.geoconcept.com/gws/wmts/France/Server2/epsg:27572/0/0/0.png

- L'onglet de visibilité de votre carte Geoconcept est publié au format WMTS. Il est accessible via un outil de visualisation tiers capable de lire le format WMTS par la requête suivante :

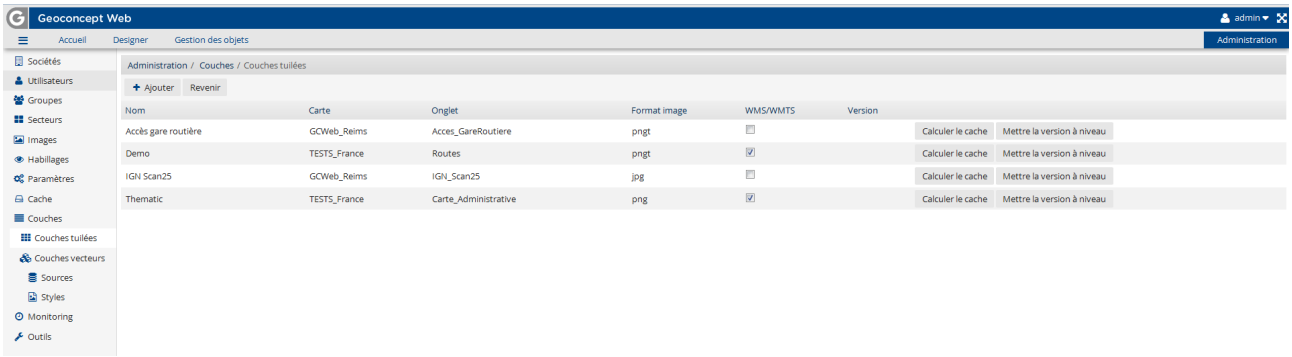
http://<server>/<webapp>/wmts

Exemple de résultat WMS

L'exemple suivant a été configuré pour afficher dans la solution SIG Geoconcept monoposte (client pour lire les couches WMS) des couches configurées dans le serveur WMS présenté précédemment.

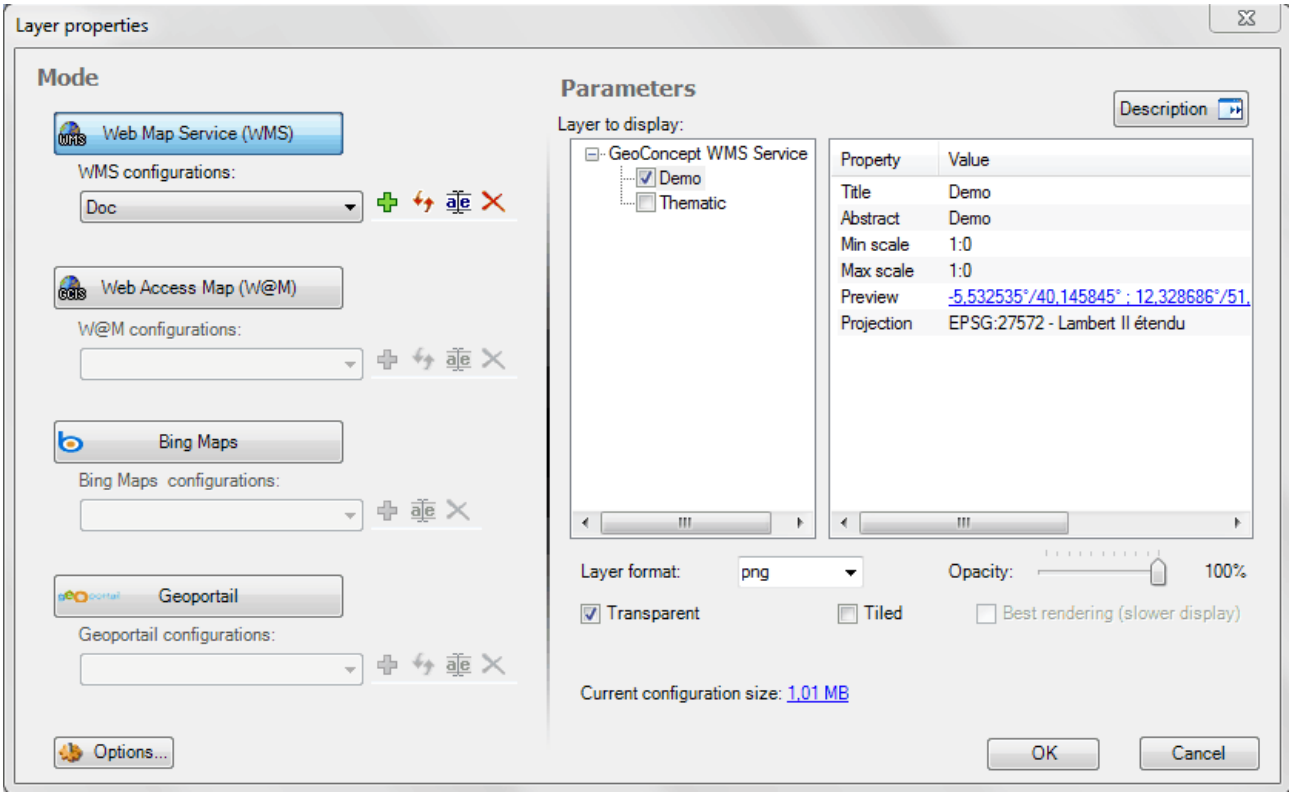
Les onglets de visibilité Demo et Thematic ont été configurées comme couches WMS ou WMTS.

Configuration de couches WMS / WMTS



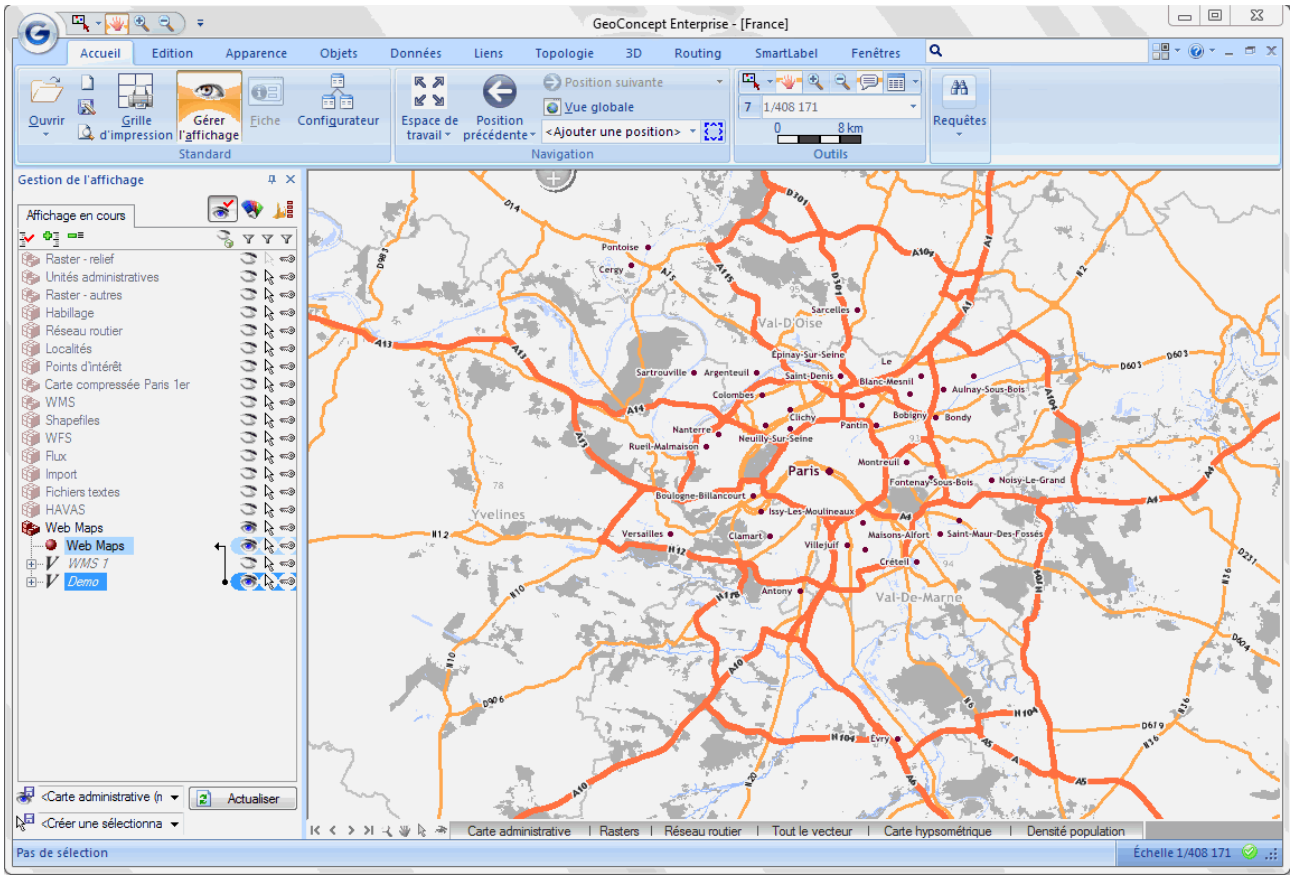
Lors de l'appel au serveur depuis Geoconcept, la boîte de dialogue permet d'afficher les couches qui ont été configurées :

Requête des couches WMS via Geoconcept



La couche Demo est alors affichée via les Webmaps dans Geoconcept.

Affichage des couches WMS dans Geoconcept



Liste les couches d’un projet

Cette page n’est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

Ce web service retourne la liste de toutes les couches (layers) d’un projet/portail.

Disponibilité

Ce web service est disponible en permanence avec Geoconcept Web.

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
name	Nom du projet	non	

En sortie

paramètre	type	min/max	description
type	string	1/1	Types de couches - raster : couches tuilées - vector : couches vecteurs - groupLayer : couches hybrides - group : groupes de couches
		name	string
1/1	Nom de la couche	label	string
1/1	Label de la couche affiché dans le gestionnaire de couche	depth	string
1/1	0 = niveau de base, 1 = les couches d’un groupe	internalName	string
1/1	non-utilisé	rights	string
1/1	non-utilisé	disabled	string
1/1	non-utilisé	metadataUrl	string
1/1	Url de métadonnées	legendUrl	string
1/1	Url de légende	format	string
1/1	format d’image (PNG, JPG, PNG24)	transparent	string
1/1	Tranparence de l’image PNG (true/false)	visibility	string
1/1	1 = visible , O = invisible dans le gestionnaire de couche	opacity	string
1/1	Opacité de la couche (0 - 100)	background	string
1/1	Couche principale (true/false)	singleTiledLayer	string
1/1	couche dynamique (true/false)	expanded	string
1/1	Groupe développé (true/false)	layerId	string
1/1	Identifiant de la couche	legend	string
1/1	Afficher la légende (true/false)	isDefaultLayer	string
1/1	Couche principale (true/false)	isWebmap	string
1/1	Couche Webmap (true/false)	tabname	string

paramètre	type	min/max	description
1/1	Onglet de la carte	map	string

REST (GET)

Requête

Requête JSON

```
http://<server>/<webapp>/api/easy/project/layers.json?name=Loire-Atlantique
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON


```
{
  "result": {
    "layers": [
      {
        "type": "vector",
        "name": "samplePoints",
        "label": "Letterbox",
        "depth": 0,
        "internalName": null,
        "rights": "",
        "disabled": false,
        "metadataUrl": null,
        "legendUrl": null,
        "visibility": 0,
        "opacity": 100,
        "layerId": "122",
        "legend": true
      },
      {
        "type": "layer",
        "name": "ADMINISTRATIVE",
        "label": "Administrative",
        "depth": 0,
        "internalName": null,
        "rights": "",
        "disabled": false,
        "metadataUrl": null,
        "legendUrl": null,
        "format": "pngt",
        "transparent": true,
        "visibility": 0,
        "opacity": 100,
        "background": false,
        "singleTiledLayer": false,
        "legend": true,
        "isDefaultLayer": false,
        "isWebmap": false,
        "tabname": "ADMINISTRATIVE",
        "map": "Loire-Atlantique"
      },
      {
```

```
          "type": "layer",
          "name": "BASEMAP",
          "label": "BASEMAP",
          "depth": 0,
          "internalName": null,
          "rights": "",
          "disabled": false,
          "metadataUrl": null,
          "legendUrl": null,
          "format": "pngt",
          "transparent": true,
          "visibility": 1,
          "opacity": 100,
          "background": true,
          "singleTiledLayer": false,
          "legend": false,
          "isDefaultLayer": true,
          "isWebmap": false,
          "tabname": "STANDARD",
          "map": "Loire-Atlantique"
        },
        {
          "type": "layer",
          "name": "COMPLETE",
          "label": "COMPLETE",
          "depth": 0,
          "internalName": null,
          "rights": "",
          "disabled": false,
          "metadataUrl": null,
          "legendUrl": null,
          "format": "png",
          "transparent": false,
          "visibility": 0,
          "opacity": 100,
          "background": false,
          "singleTiledLayer": false,
          "legend": true,
          "isDefaultLayer": false,
          "isWebmap": false,
          "tabname": "COMPLETE",
          "map": "Loire-Atlantique"
        }
      ]
    },
    "message": "Layers in project Loire-Atlantique",
    "status": "OK"
  }
```


Liste des couches

Principe

Ce web service retourne la liste de toutes les couches (layers) déclarées dans Geoconcept Web.

Disponibilité

Ce web service est disponible avec Geoconcept Web en passant le paramètre *services.activate.getAllLayers* à true (cf. section paramètres avancés).

V1

Paramètres / propriétés

En entrée

Pas de paramètre en entrée

En sortie

paramètre	type	min/max	description
id	long	1/1	Identifiant de la couche.
name	string	1/1	Nom de la couche.
kind	string	1/1	Types de couches - RASTER : pour les couches tuilées - VECTOR : pour les couches vecteurs - HYBRID : pour les couches hybrides

REST (GET)

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/layers/getAll
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
  "layers": [
    {
      "id": "123",
      "name": "ADMINISTRATIVE",
      "kind": "RASTER"
    },
    {
      "id": "124",
      "name": "BASEMAP",
      "kind": "RASTER"
    },
    {
      "id": "128",
      "name": "ADMINISTRATIVE MAP",
      "kind": "RASTER"
    },
    {
      "id": "130",
      "name": "EMPTY",
      "kind": "RASTER"
    },
    {
      "id": "133",
```

```
      "name": "COMPLETE",
      "kind": "RASTER"
    },
    {
      "id": "122",
      "name": "samplePoints",
      "kind": "VECTOR"
    }
  ]
}
```

Géométries des objets

Principe

Ce web service retourne la géométrie des objets, au format geojson, filtrés par couche et optionnellement par bounding box. Disponible uniquement pour les couches de type vecteurs (cf. Le web service de liste des couches).

Disponibilité

Ce web service est disponible avec Geoconcept Web en passant le paramètre *services.activate.getFeatures* à true (cf. section paramètres avancés).

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
layerId	Identifiant de la couche vecteur	non	
bbox	La bounding box de filtre, elle doit être dans même projection que les objets.	oui	

En sortie

paramètre	type	min/max	description
message	string	1/1	"Get geojson features"
status	string	1/1	"OK" ou "ERROR"
result	array	0/illimité	Résultat

Résultat (result)

paramètre	type	min/max	description
id	string	1/1	null
geojson	string	1/1	geojson
pagination	string	1/1	null
sort	string	1/1	null
distinctValues	string	1/1	null
fields	array	0/illimité	Description des champs utilisés dans le geojson

Description des champs utilisés dans le geojson (fields)

paramètre	type	min/max	description
name	string	1/1	Libellé du champ de la base de données.
alias	string	1/1	Alias du champ.
type	string	1/1	Typage de champ.

REST (POST)

Requête

Requête

```
http://<server>/<webapp>/api/lbs/layers/getFeatures
```

Data (JSON)

```
{
   "layerId":"136",
   "bbox":"-262000,5982000,-258000,5988000"
}
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message":"Get geojson features",
   "status":"OK",
   "result":{
      "id":null,
      "geojson":"{[geojson]}",
      "pagination":null,
      "sort":null,
      "distinctValues":null,
      "fields":[
         {
            "name":"co_insee_com",
            "alias":"city_code",
            "type":"Integer"
         },
         {
            "name":"lb_com",
            "alias":"city",
            "type":"String"
         },
         {
            "name":"lb_voie_ext",
            "alias":"street",
            "type":"String"
         },
         {
            "name":"va_no_voie",
            "alias":"number",
```

```
            "type":"Integer"
         }
      ]
   }
}
```

Geojson

```
{
   "type":"FeatureCollection",
   "features":[
      {
         "type":"Feature",
         "geometry":{
            "type":"Point",
            "coordinates":[
               -258650.09,
               5982645.79
            ]
         },
         "properties":{
            "co_insee_com":"44132",
            "lb_com":"PORNICHET",
            "lb_voie_ext":"AVENUE DE BONNE SOURCE",
            "va_no_voie":"150"
         },
         "id":"gw_sample_point.A5U8T4"
      },
      [...]
      }
   ]
}
```

Retours possibles

Cas d'une absence de layer

```
{"message":null,"status":"ERROR","result":
{"id":null,"geojson":null,"pagination":null,"sort":null,"distinctValues":null,"fields":null}}
```

cas d'une mauvaise Bounding box

```
{"message":"Error to get feature source : gw_sample_point","status":"ERROR","result":
{"id":null,"geojson":null,"pagination":null,"sort":null,"distinctValues":null,"fields":null}}
```

Service de transformation de coordonnées

Cette page n'est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

Ce web service permet de transformer des couples de coordonnées d'une projection à une autre. Par exemple, de Lambert 2 étendue vers WGS 84.

Disponibilité

Ce web service est disponible en permanence avec Geoconcept Web.

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
inSrs	projection en entrée (code EPSG comme epsg:4326 ou wgs84)	non	
outSrs	projection en sortie (code EPSG comme epsg:4326 ou wgs84)	non	
point (ou points en JSON / JSON-P)	Coordonnées des points en SOAP, il est nécessaire de remplir les coordonnées dans chacune des balises dédiées <x></x> et <y></y> en REST, Les points sont séparés par le caractère « , ». Les points sont caractèrisés par un couple de coordonnées XY séparés par une « ; ».	non	

En sortie

Points transformés (srsTransformResult)

paramètre	type	min/max	description
point (ou points en JSON / JSON-P)	geographicPoint (ou array en JSON / JSON-P)	0/illimité	points tranformés

Points(geographicPoint)

paramètre	type	min/max	description
x	double	1/1	première coordonnée ou longitude
y	double	1/1	deuxième coordonnée ou latitude

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/srsTransformService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:srsTransform>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <inSrs>epsg:27572</inSrs>
            <!--Optional:-->
```

```
            <outSrs>epsg:4326</outSrs>
            <!--Zero or more repetitions:-->
            <point>
               <x>246087</x>
               <y>2262265</y>
            </point>
             <point>
               <x>255081</x>
               <y>2262679</y>
            </point>
            <point>
               <x>298976</x>
               <y>2254643</y>
            </point>
         </request>
      </sch:srsTransform>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:srsTransformResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <srsTransformResult>
            <status>OK</status>
            <point>
               <x>-2.343140132718342</x>
               <y>47.26525078744117</y>
            </point>
            <point>
               <x>-2.2248034057039128</x>
               <y>47.27372258177934</y>
            </point>
            <point>
               <x>-1.6400114587002539</x>
               <y>47.22297983102492</y>
            </point>
         </srsTransformResult>
      </ns2:srsTransformResponse>
   </soap:Body>
</soap:Envelope>
```

REST (POST)

Requête

Requête

```
http://<server>/<webapp>/api/lbs/srsTransform.xml
```

Data (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<srsTransformRequest>
  <inSrs>epsg:27572</inSrs>
  <outSrs>epsg:4326</outSrs>
  <point>
    <x>246087</x>
```

```
    <y>2262265</y>
  </point>
  <point>
    <x>255081</x>
    <y>2262679</y>
   </point>
   <point>
    <x>298976</x>
    <y>2254643</y>
    </point>
</srsTransformRequest>
```

Réponse

La réponse est toujours encodée en UTF-8.

Format XML

```
<srsTransformResult>
   <status>OK</status>
   <point>
      <x>-2.343140132718342</x>
      <y>47.26525078744117</y>
   </point>
   <point>
      <x>-2.2248034057039128</x>
      <y>47.27372258177934</y>
   </point>
   <point>
      <x>-1.6400114587002539</x>
      <y>47.22297983102492</y>
   </point>
</srsTransformResult>
```

REST (GET)

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/srsTransform.json?
&inSrs=epsg:27572&outSrs=epsg:4326&points=246087,2262265;255081,2262679;298976,2254643
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/srsTransform.json?
&inSrs=epsg:27572&outSrs=epsg:4326&points=246087,2262265;255081,2262679;298976,2254643&callback=MyCallBack
```

Requête XML

```
http://<server>/<webapp>/api/lbs/srsTransform.xml?
&inSrs=epsg:27572&outSrs=epsg:4326&points=246087,2262265;255081,2262679;298976,2254643
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message": null,
   "status": "OK",
   "points":    [
            {
         "x": -2.343140132718342,
         "y": 47.26525078744117
      },
            {
         "x": -2.2248034057039128,
         "y": 47.27372258177934
      },
            {
         "x": -1.6400114587002539,
         "y": 47.22297983102492
      }
   ]
}
```

Format JSON-P

```
myCallback(
    {
       "message": null,
       "status": "OK",
       "points":    [
               {
            "x": -2.343140132718342,
            "y": 47.26525078744117
         },
               {
            "x": -2.2248034057039128,
            "y": 47.27372258177934
         },
               {
            "x": -1.6400114587002539,
            "y": 47.22297983102492
         }
       ]
    }
);
```

Format XML

```
<srsTransformResult>
   <status>OK</status>
   <point>
      <x>-2.343140132718342</x>
      <y>47.26525078744117</y>
   </point>
   <point>
      <x>-2.2248034057039128</x>
      <y>47.27372258177934</y>
   </point>
   <point>
      <x>-1.6400114587002539</x>
      <y>47.22297983102492</y>
   </point>
</srsTransformResult>
```

Retours possibles

Cas d'un itinéraire trouvé (srsTransformResult/status est OK)

```
<srsTransformResult>
    <status>OK</status>
    <point>
       <x>-2.343140132718342</x>
       <y>47.26525078744117</y>
    </point>
    <point>
       <x>-2.2248034057039128</x>
       <y>47.27372258177934</y>
    </point>
    <point>
       <x>-1.6400114587002539</x>
       <y>47.22297983102492</y>
    </point>
</srsTransformResult>
```

Cas de la projection de départ vide

```
<soap:Fault>
    <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
    <faultstring>inSrs can't be null or empty</faultstring>
</soap:Fault>
```

Cas de la projection de destination vide

```
<soap:Fault>
    <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
    <faultstring>outSrs can't be null or empty</faultstring>
</soap:Fault>
```

Cas d'une projection incorrecte

```
<soap:Fault>
    <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
    <faultstring>inSrs or outSrs do not exist</faultstring>
</soap:Fault>
```

Cas de coordonnée d'un point vide

```
<soap:Fault>
    <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
    <faultstring>Unmarshalling Error:</faultstring>
</soap:Fault>
```

Cas d'un mauvais typage sur les coordonnées des points

```
<soap:Fault>
    <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
    <faultstring>Unmarshalling Error: 488degree
488degree
488degree
488degre</faultstring>
</soap:Fault>
```

Complétion automatique

Principe

Ce service de complétion automatique, ou autocomplétion, a pour but de faire gagner du temps à l'internaute qui saisit un localisant. Il propose des localisants probables au fur et à mesure qu'un internaute tape les lettres d'une adresse ou d'un lieu-dit, ou les efface, dans le champ de saisie du localisant.

exemple : l'internaute rentre « avenue des cham », le service renvoie « Avenue des Champs-Élysées, Paris », ainsi que d'autres localisants commençant par la même chaîne.

Ce service prend en entrée une chaîne de caractère, en option un identifiant de territoire, et un identifiant de type d'entité recherchée. A partir d'index optimisés, il est capable de fournir instantanément un extrait des localisants (adresses) commençant par cette chaîne, dans ce territoire, et de ce type.

Les localisants sont classés dans les fichiers de référence par un poids permettant de présenter d'abord les plus probables en prennant en compte la taille de la ville.

Il est nécessaire de copier des fichiers de référence et de configurer le service grâce à l'interface Administration de Geoconcept Web, voir le chapitre Installation de Geoconcept Web / Paramètres / Paramétrages de UGC / Web service d'autocomplétion.

Ce web service ne fait pas d'opération de géocodage. Le web service de géocodage peut être utilisé avec les données reçues du service d'autocomplétion en prenant l'adresse trouvée par le module d'autocomplétion comme paramètre d'entrée du geocoder.

Pour une optimisation des performances, nous préconisons de construire le géocodage du résultat de l'autocomplétion avec des champs séparés, et non à partir du champ fulltext (les deux sont fournis dans le résultat de l'autocomplétion).

Nous préconisons une architecture 64 bits d'UGC Server car l'autocomplétion nécessite des ressources importantes.

Pour des raisons de performance, (le service devant être appelé à chaque frappe de l'internaute), il sera publié sous forme de service JSON. En option, il pourra également être appelé sous forme de service JSON-P.

Disponibilité

Ce web service est disponible en permanence avec Geoconcept Web avec les fichiers de référence.

Changement de version

Les versions précédente du web service sont conservées dans Geoconcept Web pour assurer la compatibilité avec les développements antérieurs. Il est recommandé d'utiliser la version la plus récente.

Changements avec la v2

- Le paramètre "zipcode" est renommé "postCode"

V2

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
text	Localisant à saisir	oui	
type	Permet de choisir la source de fichiers d'autocomplétion en indiquant l'alias de cette dernière : si une seule source d'autocompletion est existante, la valeur doit rester vide s'il existe plusieurs sources pour l'autocomplétion, il faut utiliser les alias de ces dernières, préalablement définies cf. Geoconcept Web / Administration / Paramètres `geocoder.autocomplete.alias.<alias_name>.datasource` la valeur de cette clef doit indiquer le nom du répertoire contenant l'ensemble des fichiers nécessaires pour l'autocompletion	oui	
terr	Filtre sur le territoire où l'on veut rechercher les localisants en question. exemple, si l'on souhaite que la requête s'applique à la commune, il est nécessaire de renseigner le code postal : 92220 si l'on souhaite que la recherche s'applique à plusieurs communes ou départements, il faut séparer les territoires par un « ; » : 75;78;92 si la valeur reste vide, la recherche s'appliquera sur l'ensemble du territoire couvert par les fichiers d'autocomplétion et les résultats seront triés dans l'ordre alphabétique croissant des rues.	oui	
maximumResponses	nombre maximum de résultats d'adresses dans la réponse. si la valeur est laissée vide, le paramètre définit dans l'administration s'appliquera	oui	cf. Geoconcept Web / Administration / Paramètres `geocoder.autocomplete`

En sortie

propriété	type	min/max	description
city	string	0/1	ville trouvée
classification	integer	1/1	code indiquant dans quelle catégorie se trouve la ville retournée par le webservice. exemple, la catégorisation des tables HERE va de 1 (villes les plus importantes) à 8 (village, hameaux)
country	string	0/1	pays sur deux lettres (code ISO 3166-1 ou ccTLD) par exemple "fr" lorsque le résultat est trouvé en France
fulltext	string	0/1	rue trouvée avec une nomenclature spécifique incluant le code postal et le nom de la ville
street	string	1/1	rue trouvée

propriété	type	min/max	description
postCode	string	0/1	code postal trouvé

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/autoCompleteService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:autocompleteV2>
         <!--Optional:-->
         <text>rue noire</text>
         <!--Optional:-->
         <type></type>
         <!--Optional:-->
         <terr>44</terr>
         <!--Optional:-->
         <maximumResponses>3</maximumResponses>
      </sch:autocompleteV2>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:autocompleteV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <AutoCompleteResults>
            <status>OK</status>
            <result>
               <city>NANTES</city>
               <classification>2</classification>
               <country>FR</country>
               <fulltext>RUE NOIRE, 44000 NANTES</fulltext>
               <postCode>44000</postCode>
               <street>RUE NOIRE</street>
            </result>
            <result>
               <city>BLAIN</city>
               <classification>5</classification>
               <country>FR</country>
               <fulltext>RUE NOIRE, 44130 BLAIN</fulltext>
               <postCode>44130</postCode>
               <street>RUE NOIRE</street>
            </result>
            <result>
               <city>NOTRE-DAME-DES-LANDES</city>
               <classification>7</classification>
               <country>FR</country>
               <fulltext>RUE NOIRE, 44130 NOTRE-DAME-DES-LANDES</fulltext>
               <postCode>44130</postCode>
               <street>RUE NOIRE</street>
            </result>
         </AutoCompleteResults>
```

```
      </ns2:autocompleteV2Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/autocomplete.json?text=rue%20noire&terr=44&maximumResponses=3
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/autocomplete.json?text=rue
%20noire&terr=44&maximumResponses=3&callback=myFonction
```

Requête XML

```
http://<server>/<webapp>/api/lbs/autocomplete.xml?text=rue%20noire&terr=44&maximumResponses=3
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message":null,
   "status":"OK",
   "results":[
      {
         "fulltext":"RUE NOIRE, 44000 NANTES",
         "country":"FR",
         "classification":2,
         "street":"RUE NOIRE",
         "city":"NANTES",
         "zipcode":"44000"
      },
      {
         "fulltext":"RUE NOIRE, 44130 BLAIN",
         "country":"FR",
         "classification":5,
         "street":"RUE NOIRE",
         "city":"BLAIN",
         "zipcode":"44130"
      },
      {
         "fulltext":"RUE NOIRE, 44130 NOTRE-DAME-DES-LANDES",
         "country":"FR",
         "classification":7,
         "street":"RUE NOIRE",
         "city":"NOTRE-DAME-DES-LANDES",
         "zipcode":"44130"
      }
   ]
}
```

Format JSON-P

```
myFonction(
    {
   "message":null,
   "status":"OK",
   "results":[
      {
         "fulltext":"RUE NOIRE, 44000 NANTES",
         "country":"FR",
         "classification":2,
         "street":"RUE NOIRE",
         "city":"NANTES",
         "zipcode":"44000"
      },
      {
         "fulltext":"RUE NOIRE, 44130 BLAIN",
         "country":"FR",
         "classification":5,
         "street":"RUE NOIRE",
         "city":"BLAIN",
         "zipcode":"44130"
      },
      {
         "fulltext":"RUE NOIRE, 44130 NOTRE-DAME-DES-LANDES",
         "country":"FR",
         "classification":7,
         "street":"RUE NOIRE",
         "city":"NOTRE-DAME-DES-LANDES",
         "zipcode":"44130"
      }
   ]
}
);
```

Format XML

```
<?xml version="1.0" encoding="UTF-8"?>
<autoCompleteResults>
   <status>OK</status>
   <result>
      <city>NANTES</city>
      <classification>2</classification>
      <country>FR</country>
      <fulltext>RUE NOIRE, 44000 NANTES</fulltext>
      <street>RUE NOIRE</street>
      <zipcode>44000</zipcode>
   </result>
   <result>
      <city>BLAIN</city>
      <classification>5</classification>
      <country>FR</country>
      <fulltext>RUE NOIRE, 44130 BLAIN</fulltext>
      <street>RUE NOIRE</street>
      <zipcode>44130</zipcode>
   </result>
   <result>
      <city>NOTRE-DAME-DES-LANDES</city>
      <classification>7</classification>
      <country>FR</country>
      <fulltext>RUE NOIRE, 44130 NOTRE-DAME-DES-LANDES</fulltext>
      <street>RUE NOIRE</street>
```

```
      <zipcode>44130</zipcode>
   </result>
</autoCompleteResults>
```

Retours possibles

Cas d'addresses trouvées (autocompleteResponse/AutoCompleteResults/status est OK)

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:autocompleteV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <AutoCompleteResults>
            <status>OK</status>
            <result>
               <city>NANTES</city>
               <classification>2</classification>
               <country>FR</country>
               <fulltext>RUE NOIRE, 44000 NANTES</fulltext>
               <postCode>44000</postCode>
               <street>RUE NOIRE</street>
            </result>
            <result>
               <city>BLAIN</city>
               <classification>5</classification>
               <country>FR</country>
               <fulltext>RUE NOIRE, 44130 BLAIN</fulltext>
               <postCode>44130</postCode>
               <street>RUE NOIRE</street>
            </result>
            <result>
               <city>NOTRE-DAME-DES-LANDES</city>
               <classification>7</classification>
               <country>FR</country>
               <fulltext>RUE NOIRE, 44130 NOTRE-DAME-DES-LANDES</fulltext>
               <postCode>44130</postCode>
               <street>RUE NOIRE</street>
            </result>
         </AutoCompleteResults>
      </ns2:autocompleteV2Response>
   </soap:Body>
</soap:Envelope>
```

Cas d'une adresse qui n'existe pas (autocompleteResponse/AutoCompleteResults/status est ERROR)

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:autocompleteV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <AutoCompleteResults>
            <status>ERROR</status>
         </AutoCompleteResults>
      </ns2:autocompleteV2Response>
   </soap:Body>
</soap:Envelope>
```

Cas d'une requête ayant une erreur XML ou ne respectant pas le WSDL ⇒ erreur avec faultstring qui contient la description

```
<soap:Fault>
        <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
```

```
        <faultstring>Message part {http://geoconcept.com/gc/schemas}geocodeABCD was not recognized.  (Does
 it exist in service WSDL?)</faultstring>
</soap:Fault>
```

Cas d'une requête avec un territoire inexistant (autocompleteResponse/AutoCompleteResults/status est ERROR)

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:autocompleteResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <AutoCompleteResults>
            <status>ERROR</status>
         </AutoCompleteResults>
      </ns2:autocompleteResponse>
   </soap:Body>
</soap:Envelope>
```

Cas d'une requête un nombre maximum de réponse qui est mal renseigné. Par exemple, mettre un nombre à virgule ou tout autre caractère. C'est la valeur renseignée dans l'Adminitration qui prend alors le dessus cf. Geoconcept Web / Administration / Paramètres **`geocoder.maxCandidates`**

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:autocompleteResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <AutoCompleteResults>
            <status>OK</status>
            <result>
               <city>PARIS</city>
               <classification>1</classification>
               <country>FR</country>
               <fulltext>AVENUE DU GÉNÉRAL BALFOURIER, 75016 PARIS</fulltext>
               <kind/>
               <street>AVENUE DU GÉNÉRAL BALFOURIER</street>
               <x>0.0</x>
               <y>0.0</y>
               <postCode>75016</postCode>
            </result>
         </AutoCompleteResults>
      </ns2:autocompleteResponse>
   </soap:Body>
</soap:Envelope>
```

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
text	Localisant à saisir	oui	
type	Permet de choisir la source de fichiers d'autocomplétion en indiquant l'alias de cette dernière : si une seule source d'autocompletion est existante, la valeur doit rester vide s'il existe plusieurs sources pour l'autocomplétion, il faut utiliser les alias de ces dernières, préalablement définies cf. Geoconcept Web / Administration / Paramètres **`geocoder.autocomplete.alias.<alias_name>.datasource`**	oui	

paramètre	description	optionnel	défaut
	la valeur de cette clef doit indiquer le nom du répertoire contenant l'ensemble des fichiers nécessaires pour l'autocompletion		
terr	Filtre sur le territoire où l'on veut rechercher les localisants en question. exemple, si l'on souhaite que la requête s'applique à la commune, il est nécessaire de renseigner le code postal : 92220 si l'on souhaite que la recherche s'applique à plusieurs communes ou départements, il faut séparer les territoires par un « ; » : 75;78;92 si la valeur reste vide, la recherche s'appliquera sur l'ensemble du territoire couvert par les fichiers d'autocomplétion et les résultats seront triés dans l'ordre alphabétique croissant des rues.	oui	
maximumResponses	nombre maximum de résultats d'adresses dans la réponse. si la valeur est laissée vide, le paramètre définit dans l'administration s'appliquera	oui	cf. Geoconcept Web / Administration / Paramètres **geocoder.maxCandidate**

En sortie

propriété	type	min/max	description
city	string	0/1	ville trouvée
classification	integer	1/1	code indiquant dans quelle catégorie se trouve la ville retournée par le webservice. exemple, la catégorisation des tables HERE va de 1 (villes les plus importantes) à 8 (village, hameaux)
country	string	0/1	pays sur deux lettres (code ISO 3166-1 ou ccTLD) par exemple "fr" lorsque le résultat est trouvé en France
fulltext	string	0/1	rue trouvée avec une nomenclature spécifique incluant le code postal et le nom de la ville
street	string	1/1	rue trouvée
zipcode	string	0/1	code postal trouvé

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/autoCompleteService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:autocomplete>
         <!--Optional:-->
         <text>Avenue Général</text>
         <!--Optional:-->
         <type></type>
         <!--Optional:-->
         <terr>75</terr>
```

```
         <!--Optional:-->
         <maximumResponses>3</maximumResponses>
         <!--Optional:-->
         <repeat>?</repeat>
      </sch:autocomplete>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:autocompleteResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <AutoCompleteResults>
            <status>OK</status>
            <result>
               <city>PARIS</city>
               <classification>1</classification>
               <country>FR</country>
               <fulltext>AVENUE DU GÉNÉRAL BALFOURIER, 75016 PARIS</fulltext>
               <kind/>
               <street>AVENUE DU GÉNÉRAL BALFOURIER</street>
               <x>0.0</x>
               <y>0.0</y>
               <zipcode>75016</zipcode>
            </result>
            <result>
               <city>PARIS</city>
               <classification>1</classification>
               <country>FR</country>
               <fulltext>AVENUE DU GÉNÉRAL CLAVERY, 75016 PARIS</fulltext>
               <kind/>
               <street>AVENUE DU GÉNÉRAL CLAVERY</street>
               <x>0.0</x>
               <y>0.0</y>
               <zipcode>75016</zipcode>
            </result>
            <result>
               <city>PARIS</city>
               <classification>1</classification>
               <country>FR</country>
               <fulltext>AVENUE DU GÉNÉRAL DÉTRIE, 75007 PARIS</fulltext>
               <kind/>
               <street>AVENUE DU GÉNÉRAL DÉTRIE</street>
               <x>0.0</x>
               <y>0.0</y>
               <zipcode>75007</zipcode>
            </result>
         </AutoCompleteResults>
      </ns2:autocompleteResponse>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/autocomplete.json?text=rue%20general&terr=75&maximumResponses=3
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/autocomplete.json?text=rue
%20general&terr=75&maximumResponses=3&callback=myFonction
```

Requête XML

```
http://<server>/<webapp>/api/lbs/autocomplete.xml?text=rue%20general&terr=75&maximumResponses=3
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON


```
{
   "message": null,
   "status": "OK",
   "results":    [
            {
         "fulltext": "RUE DU GÉNÉRAL ANSELIN, 75016 PARIS",
         "country": "FR",
         "classification": 1,
         "street": "RUE DU GÉNÉRAL ANSELIN",
         "city": "PARIS",
         "zipcode": "75016",
         "x": 0,
         "y": 0,
         "kind": ""
      },
            {
         "fulltext": "RUE DU GÉNÉRAL ANSELIN, 75116 PARIS",
         "country": "FR",
         "classification": 1,
         "street": "RUE DU GÉNÉRAL ANSELIN",
         "city": "PARIS",
         "zipcode": "75116",
         "x": 0,
         "y": 0,
         "kind": ""
      },
            {
         "fulltext": "RUE DU GÉNÉRAL APPERT, 75116 PARIS",
         "country": "FR",
         "classification": 1,
         "street": "RUE DU GÉNÉRAL APPERT",
         "city": "PARIS",
         "zipcode": "75116",
         "x": 0,
         "y": 0,
         "kind": ""
      }
   ]
}
```


Format JSON-P


```
myFonction({
        "message":null,"status":"OK",
```

```
"results":[
                {
                        "fulltext":"RUE DU GÉNÉRAL ANSELIN, 75016
 PARIS","country":"FR","classification":1,"street":"RUE DU GÉNÉRAL
 ANSELIN","city":"PARIS","zipcode":"75016","x":0.0,"y":0.0,"kind":""
                },
                {
                        "fulltext":"RUE DU GÉNÉRAL ANSELIN, 75116
 PARIS","country":"FR","classification":1,"street":"RUE DU GÉNÉRAL
 ANSELIN","city":"PARIS","zipcode":"75116","x":0.0,"y":0.0,"kind":""
                },
                {
                        "fulltext":"RUE DU GÉNÉRAL APPERT, 75116
 PARIS","country":"FR","classification":1,"street":"RUE DU GÉNÉRAL
 APPERT","city":"PARIS","zipcode":"75116","x":0.0,"y":0.0,"kind":""}
]});
```

Format XML

```
<autoCompleteResults>
   <status>OK</status>
   <result>
      <city>PARIS</city>
      <classification>1</classification>
      <country>FR</country>
      <fulltext>RUE DU GÉNÉRAL ANSELIN, 75016 PARIS</fulltext>
      <kind/>
      <street>RUE DU GÉNÉRAL ANSELIN</street>
      <x>0.0</x>
      <y>0.0</y>
      <zipcode>75016</zipcode>
   </result>
   <result>
      <city>PARIS</city>
      <classification>1</classification>
      <country>FR</country>
      <fulltext>RUE DU GÉNÉRAL ANSELIN, 75116 PARIS</fulltext>
      <kind/>
      <street>RUE DU GÉNÉRAL ANSELIN</street>
      <x>0.0</x>
      <y>0.0</y>
      <zipcode>75116</zipcode>
   </result>
   <result>
      <city>PARIS</city>
      <classification>1</classification>
      <country>FR</country>
      <fulltext>RUE DU GÉNÉRAL APPERT, 75116 PARIS</fulltext>
      <kind/>
      <street>RUE DU GÉNÉRAL APPERT</street>
      <x>0.0</x>
      <y>0.0</y>
      <zipcode>75116</zipcode>
   </result>
</autoCompleteResults>
```

Retours possibles

Cas d’addresses trouvées (autocompleteResponse/AutoCompleteResults/status est OK)

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
```

```
   <soap:Body>
      <ns2:autocompleteResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <AutoCompleteResults>
            <status>OK</status>
            <result>
               <city>PARIS</city>
               <classification>1</classification>
               <country>FR</country>
               <fulltext>AVENUE DU GÉNÉRAL BALFOURIER, 75016 PARIS</fulltext>
               <kind/>
               <street>AVENUE DU GÉNÉRAL BALFOURIER</street>
               <x>0.0</x>
               <y>0.0</y>
               <zipcode>75016</zipcode>
            </result>
            <result>
               <city>PARIS</city>
               <classification>1</classification>
               <country>FR</country>
               <fulltext>AVENUE DU GÉNÉRAL CLAVERY, 75016 PARIS</fulltext>
               <kind/>
               <street>AVENUE DU GÉNÉRAL CLAVERY</street>
               <x>0.0</x>
               <y>0.0</y>
               <zipcode>75016</zipcode>
            </result>
            <result>
               <city>PARIS</city>
               <classification>1</classification>
               <country>FR</country>
               <fulltext>AVENUE DU GÉNÉRAL DÉTRIE, 75007 PARIS</fulltext>
               <kind/>
               <street>AVENUE DU GÉNÉRAL DÉTRIE</street>
               <x>0.0</x>
               <y>0.0</y>
               <zipcode>75007</zipcode>
            </result>
         </AutoCompleteResults>
      </ns2:autocompleteResponse>
   </soap:Body>
</soap:Envelope>
```

Cas d'une adresse qui n'existe pas (autocompleteResponse/AutoCompleteResults/status est ERROR)

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:autocompleteResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <AutoCompleteResults>
            <status>ERROR</status>
         </AutoCompleteResults>
      </ns2:autocompleteResponse>
   </soap:Body>
</soap:Envelope>
```

Cas d'une requête ayant une erreur XML ou ne respectant pas le WSDL ⇒ erreur avec faultstring qui contient la description

```
<soap:Fault>
        <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
        <faultstring>Message part {http://geoconcept.com/gc/schemas}geocodeABCD was not recognized.  (Does
 it exist in service WSDL?)</faultstring>
```

```
</soap:Fault>
```

Cas d'une requête avec un territoire inexistant (autocompleteResponse/AutoCompleteResults/status est ERROR)

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:autocompleteResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <AutoCompleteResults>
            <status>ERROR</status>
         </AutoCompleteResults>
      </ns2:autocompleteResponse>
   </soap:Body>
</soap:Envelope>
```

Cas d'une requête un nombre maximum de réponse qui est mal renseigné. Par exemple, mettre un nombre à virgule ou tout autre caractère. C'est la valeur renseignée dans l'Adminitration qui prend alors le dessus cf. Geoconcept Web / Administration / Paramètres **`geocoder.maxCandidates`**

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:autocompleteResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <AutoCompleteResults>
            <status>OK</status>
            <result>
               <city>PARIS</city>
               <classification>1</classification>
               <country>FR</country>
               <fulltext>AVENUE DU GÉNÉRAL BALFOURIER, 75016 PARIS</fulltext>
               <kind/>
               <street>AVENUE DU GÉNÉRAL BALFOURIER</street>
               <x>0.0</x>
               <y>0.0</y>
               <zipcode>75016</zipcode>
            </result>
         </AutoCompleteResults>
      </ns2:autocompleteResponse>
   </soap:Body>
</soap:Envelope>
```

Géocodage

Cette page n'est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

La requête comprend une adresse en entrée, le service retourne une ou plusieurs réponses possibles (s'il y a ambiguïté), en incluant l'adresse reconnue, la position, le score de géocodage, le type de géocodage.

Ce service de géocodage interroge la table de référence paramétrée grâce à l'interface Administration de Geoconcept Web.

Disponibilité

Ce web service est disponible en permanence avec Geoconcept Web et une table de référence.

Changement de version

Les versions précédente du web service sont conservées dans Geoconcept Web pour assurer la compatibilité avec les développements antérieurs. Il est recommandé d'utiliser la version la plus récente.

Changements avec la v2

- Suppression du paramètre "projection", remplacé par "srs"
- Suppression du paramètre "geocodeScore", remplacé par "score"
- Suppression des paramètres "projection", "srs" et "maxResponses" dans le bloc "initialAddress"
- Le paramètre "postalCode" est renommé "postCode"

V2

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
countryCode	pays sur 2 ou 3 lettres (code ISO 3166-1) par exemple "fr" ou "fra", ce paramètre peut être utilisé pour distinguer plusieurs référentiels sur un même territoire (IGN, HERE, …)	oui	cf. Geoconcept Web / Administration / Paramètres `geocoder.datasource`
postCode	code postal	oui *	
addressLine	adresse comprenant numéro, indice de répétition, type de voie et libellé de voie.	oui *	
city	ville	oui *	
region	État, Comté, …	oui	
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	Sans projection, le résultat est en projection native de l'index de géocodage, habituellement wgs84
maxResponses	nombre maximum de résultats d'adresses dans la réponse	oui	cf. Geoconcept

paramètre	description	optionnel	défaut
			Web / Administration / Paramètres **geocoder.maxCandidates**
streetMinScore	la valeur de ce paramètre varie, comme pour le score, entre 0 et 100. Le streetMinScore a pour effet de filtrer les candidats possibles de la manière suivante : Si la note du candidat est égale ou supérieure au score, alors le candidat est retenu comme tel, Si la note du candidat est inférieure au score et qu'aucun candidat n'a déjà été retenu, alors le géocodage est proposé à la commune, Si la note du candidat est inférieure au score et qu'un autre candidat a déjà été retenu, alors le candidat ne sera présenté que dans la limite du **geocoder.maxCandidates** La valeur de streetMinScore doit toujours être supérieure à **filterMinScore** , si c'est pas le cas le geocodage force la valeur de **filterMinScore** pour streetMinScore. La valeur définit pour streetMinScore supplante celle définit dans le paramètre **geocoder.geocoder.streetMinScore** . Les paramètres **geocoder.maxCandidates** , **geocoder.filterMinScore** et **geocoder.geocoder.streetMinScore** sont détaillés dans Geoconcept Web / Administration / Paramètres	oui	

(*) Au moins l'un des trois paramètres postCode, addressLine et city doit être renseigné.

En sortie

paramètre	type	min/max	description
geocodedAddress (ou geocodedAddresses en JSON / JSON-P)	geocodedAddress (ou array en JSON / JSON-P)	0/illimité	Adresses géocodées
initialAddress	geocodeInitialAddress	0/1	Adresse initiale

Adresses géocodées (geocodedAddress)

paramètre	type	min/max	description
addressLine	string	0/1	rue trouvée et le cas échéant le numéro
city	string	0/1	ville trouvée
region	string	0/1	État, Comté, … trouvée, varie en fonction des pays, peut également être vide
countryCode	string	1/1	cf. description du paramètre en entrée
postCode	string	0/1	code postal trouvé
secondaryZone	string	0/1	zone qui dépend de l'index de géocodage, habituellement en France il s'agit du code IRIS
score	double	1/1	note du géocodage de 0 à 100, avec 100 pour une correspondance parfaite
geocodeType	int	1/1	type de géocodage : - ville = 1 - rue = 2 - rue amélioré = 3 - numéro de rue = 4 - non géocodé = 0

paramètre	type	min/max	description
x	double	1/1	coordonnées X
y	double	1/1	coordonnées Y
place (ou places en JSON / JSON-P)	string (ou array en JSON / JSON-P)	0/illimité	liste des attributs. Valeur des attributs de l'adresse trouvée, en relation avec *placeTypes* (par exemple ["751010206","930005Y001XCHE"]). Dépend du référentiel utilisé.
placeType (ou placeTypes en JSON / JSON-P)	string (ou array en JSON / JSON-P)	0/illimité	liste des types d'attributs (par exemple ["IRIS","FANTOIR"]) .Dépend de la table de référence utilisée.
streetNumber	string	0/1	numéro de rue
streetWayType	string	0/1	type de rue (avenue, street, etc)
streetWayName	string	0/1	nom de la rue
streetWay	string	0/1	nom complet de la rue

Adresse initiale (initialAddress)

paramètre	type	min/max	description
addressLine	string	0/1	rue trouvée et le cas échéant le numéro
city	string	0/1	ville trouvée
region	string	0/1	État, Comté, … trouvée, varie en fonction des pays, peut également être vide
countryCode	string	1/1	cf. description du paramètre en entrée
postCode	string	0/1	code postal trouvé

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/geocodeService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:geocodeV2>
         <!--Optional:-->
         <countryCode>FR</countryCode>
         <!--Optional:-->
         <postCode>44000</postCode>
         <!--Optional:-->
         <addressLine>45 Rue Noire</addressLine>
         <!--Optional:-->
         <city>nantes</city>
         <!--Optional:-->
         <region></region>
         <!--Optional:-->
         <srs>epsg:4326</srs>
         <maxResponses>2</maxResponses>
```

```
         <streetMinScore></streetMinScore>
      </sch:geocodeV2>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:geocodeV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <GeocodeResult>
            <status>OK</status>
            <geocodedAddress>
               <addressLine>45 RUE NOIRE</addressLine>
               <city>Nantes</city>
               <countryCode>FR</countryCode>
               <postCode>44109</postCode>
               <srs>epsg:4326</srs>
               <secondaryZone/>
               <score>94.95</score>
               <geocodeType>4</geocodeType>
               <x>-1.563901</x>
               <y>47.224987</y>
               <places>
                  <place/>
               </places>
               <placeTypes>
                  <placeType>Attribut</placeType>
               </placeTypes>
               <streetNumber>45</streetNumber>
               <streetWayType>RUE</streetWayType>
               <streetWayName>NOIRE</streetWayName>
               <streetWay>RUE NOIRE</streetWay>
            </geocodedAddress>
            <initialAddress>
               <addressLine>45 Rue Noire</addressLine>
               <city>nantes</city>
               <region/>
               <countryCode>FR</countryCode>
               <postCode>44000</postCode>
            </initialAddress>
         </GeocodeResult>
      </ns2:geocodeV2Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/geocode/v2.json?addressLine=45%20Rue
%20Noire&postCode=44000&city=nantes&countryCode=fr&srs=epsg:4326
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/geocode/v2.json?addressLine=45%20Rue
%20Noire&postCode=44000&city=nantes&countryCode=fr&srs=epsg:4326&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/geocode/v2.xml?addressLine=45%20Rue
%20Noire&postCode=44000&city=nantes&countryCode=fr&srs=epsg:4326
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON


```
{
   "message":null,
   "status":"OK",
   "geocodedAddresses":[
      {
         "addressLine":"45 RUE NOIRE",
         "city":"Nantes",
         "countryCode":"FR",
         "postCode":"44109",
         "srs":"epsg:4326",
         "secondaryZone":"",
         "score":94.95,
         "geocodeType":4,
         "x":-1.563901,
         "y":47.224987,
         "places":[
            ""
         ],
         "placeTypes":[
            "Attribut"
         ],
         "streetNumber":"45",
         "streetWayType":"RUE",
         "streetWayName":"NOIRE",
         "streetWay":"RUE NOIRE"
      }
   ],
   "initialAddress":{
      "addressLine":"45 Rue Noire",
      "city":"nantes",
      "countryCode":"fr",
      "postCode":"44000"
   }
}
```


Format JSON-P


```
MyCallBack({
   "message":null,
   "status":"OK",
   "geocodedAddresses":[
      {
         "addressLine":"45 RUE NOIRE",
         "city":"Nantes",
         "countryCode":"FR",
         "postCode":"44109",
         "srs":"epsg:4326",
         "secondaryZone":"",
         "score":94.95,
```

```
         "geocodeType":4,
         "x":-1.563901,
         "y":47.224987,
         "places":[
            ""
         ],
         "placeTypes":[
            "Attribut"
         ],
         "streetNumber":"45",
         "streetWayType":"RUE",
         "streetWayName":"NOIRE",
         "streetWay":"RUE NOIRE"
      }
   ],
   "initialAddress":{
      "addressLine":"45 Rue Noire",
      "city":"nantes",
      "countryCode":"fr",
      "postCode":"44000"
   }
});
```


Format XML


```
<?xml version="1.0" encoding="UTF-8"?>
<geocodeResultV2>
   <status>OK</status>
   <geocodedAddress>
      <addressLine>45 RUE NOIRE</addressLine>
      <city>Nantes</city>
      <countryCode>FR</countryCode>
      <postCode>44109</postCode>
      <srs>epsg:4326</srs>
      <secondaryZone />
      <score>94.95</score>
      <geocodeType>4</geocodeType>
      <x>-1.563901</x>
      <y>47.224987</y>
      <places>
         <place />
      </places>
      <placeTypes>
         <placeType>Attribut</placeType>
      </placeTypes>
      <streetNumber>45</streetNumber>
      <streetWayType>RUE</streetWayType>
      <streetWayName>NOIRE</streetWayName>
      <streetWay>RUE NOIRE</streetWay>
   </geocodedAddress>
   <initialAddress>
      <addressLine>45 Rue Noire</addressLine>
      <city>nantes</city>
      <countryCode>fr</countryCode>
      <postCode>44000</postCode>
   </initialAddress>
</geocodeResultV2>
```


API JavaScript

Inclure la librairie JavaScript.

```
var geocoder = new GCUI.Control.GeoCode();
geocoder.geocode({url : 'http://<server>/<webapp>/api/lbs/geocode/v2.json', city : 'Nantes', postCode :
 '44000', addressLine : '45 Rue Noire', countryCode : 'fr', srs : 'wgs84',
callback : function(result) {...} }) ;
```

La variable **`result`** est au format JSON décrit ci-dessus. La fonction **`callback`** passée en paramètre est appelée à la fin du géocodage.

Retours possibles

Cas d'une adresse trouvée (geocodeResponse/GeocodeResult/status est OK)

```
<ns2:geocodeV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
    <?xml version="1.0" encoding="UTF-8"?>
    <geocodeResultV2>
       <status>OK</status>
       <geocodedAddress>
          <addressLine>45 RUE NOIRE</addressLine>
          <city>Nantes</city>
          <countryCode>FR</countryCode>
          <postCode>44109</postCode>
          <srs>epsg:4326</srs>
          <secondaryZone />
          <score>94.95</score>
          <geocodeType>4</geocodeType>
          <x>-1.563901</x>
          <y>47.224987</y>
          <places>
             <place />
          </places>
          <placeTypes>
             <placeType>Attribut</placeType>
          </placeTypes>
          <streetNumber>45</streetNumber>
          <streetWayType>RUE</streetWayType>
          <streetWayName>NOIRE</streetWayName>
          <streetWay>RUE NOIRE</streetWay>
       </geocodedAddress>
       <initialAddress>
          <addressLine>45 Rue Noire</addressLine>
          <city>nantes</city>
          <countryCode>fr</countryCode>
          <postCode>44000</postCode>
       </initialAddress>
    </geocodeResultV2>
</ns2:geocodeV2Response>
```

Cas d'une adresse non trouvée (geocodeResponse/GeocodeResult/status est OK et pas de geocodedAddress)

```
<ns2:geocodeV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
        <GeocodeResult>
                <status>OK</status>
                <initialAddress>
                        <city>#hdkvnjsdvn</city>
                </initialAddress>
        </GeocodeResult>
</ns2:geocodeV2Response>
```

Cas d'une requête ayant une erreur XML ou ne respectant pas le WSDL ⇒ erreur avec faultstring qui contient la description

```
<soap:Fault>
        <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
        <faultstring>Message part {http://geoconcept.com/gc/schemas}geocodeABCD was not recognized.  (Does
 it exist in service WSDL?)</faultstring>
</soap:Fault>
```

Cas d'une requête avec un système de reprojection inexistant ⇒ erreur avec faultstring qui contient la description

```
<soap:Fault>
        <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
        <faultstring>Geocode failed Failed to process geocoding task Unsupported coordinate system
 'epsg:432666666'</faultstring>
</soap:Fault>
```

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
countryCode	pays sur deux lettres (code ISO 3166-1 ou ccTLD) par exemple "fr", ce paramètre peut être utilisé pour distinguer plusieurs référentiels sur un même territoire (IGN, HERE, …)	oui	cf. Geoconcept Web / Administration / Paramètres **`geocoder.datasource`**
postalCode	code postal	oui *	
addressLine	adresse comprenant numéro, indice de répétition, type de voie et libellé de voie.	oui *	
city	ville	oui *	
region	État, Comté, …	oui	
projection	déprécié, remplacé par srs	oui	
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	Sans projection, le résultat est en projection native de l'index de géocodage, habituellement wgs84
maxResponses	nombre maximum de résultats d'adresses dans la réponse	oui	cf. Geoconcept Web / Administration /

paramètre	description	optionnel	défaut
			Paramètres `geocoder.maxCandidate`
streetMinScore	la valeur de ce paramètre varie, comme pour le score, entre 0 et 100. Le streetMinScore a pour effet de filtrer les candidats possibles de la manière suivante : Si la note du candidat est égale ou supérieure au score, alors le candidat est retenu comme tel, Si la note du candidat est inférieure au score et qu'aucun candidat n'a déjà été retenu, alors le géocodage est proposé à la commune, Si la note du candidat est inférieure au score et qu'un autre candidat a déjà été retenu, alors le candidat ne sera présenté que dans la limite du `geocoder.maxCandidates` La valeur de streetMinScore doit toujours être supérieure à `filterMinScore` , si c'est pas le cas le geocodage force la valeur de `filterMinScore` pour streetMinScore. La valeur définit pour streetMinScore supplante celle définit dans le paramètre `geocoder.geocoder.streetMinScore` . Les paramètres `geocoder.maxCandidates` , `geocoder.filterMinScore` et `geocoder.geocoder.streetMinScore` sont détaillés dans Geoconcept Web / Administration / Paramètres	oui	

(*) Au moins l'un des trois paramètres postalCode, addressLine et city doit être renseigné.

En sortie

paramètre	type	min/max	description
geocodedAddress (ou geocodedAddresses en JSON / JSON-P)	geocodedAddress (ou array en JSON / JSON-P)	0/illimité	Adresses géocodées
initialAddress	geocodeInitialAddress	0/1	Adresse initiale

Adresses géocodées (geocodedAddress)

paramètre	type	min/max	description
secondaryZone	string	0/1	zone qui dépend de l'index de géocodage, habituellement en France il s'agit du code IRIS
score	double	1/1	note du géocodage de 0 à 100, avec 100 pour une correspondance parfaite
geocodeScore	double	1/1	Déprécié, , remplacé par score : note du géocodage de 0 à 20, avec 20 pour une correspondance parfaite
geocodeType	int	1/1	type de géocodage : - ville = 1 - rue = 2 - rue amélioré = 3 - numéro de rue = 4 - non géocodé = 0
x	double	1/1	coordonnées X
y	double	1/1	coordonnées Y
place (ou places en JSON / JSON-P)	string (ou array en JSON / JSON-P)	0/illimité	liste des attributs. Valeur des attributs de l'adresse trouvée, en relation avec *placeTypes* (par exemple ["751010206","930005Y001XCHE"]). Dépend du référentiel utilisé.

paramètre	type	min/max	description
placeType (ou placeTypes en JSON / JSON-P)	string (ou array en JSON / JSON-P)	0/illimité	liste des types d'attributs (par exemple ["IRIS","FANTOIR"]) .Dépend de la table de référence utilisée.
streetNumber	string	0/1	numéro de rue
streetWayType	string	0/1	type de rue (avenue, street, etc)
streetWayName	string	0/1	nom de la rue
streetWay	string	0/1	nom complet de la rue

Adresse initiale (initialAddress)

paramètre	type	min/max	description
addressLine	string	0/1	rue trouvée et le cas échéant le numéro
city	string	0/1	ville trouvée
region	string	0/1	État, Comté, … trouvée, varie en fonction des pays, peut également être vide
countryCode	string	1/1	cf. description du paramètre en entrée
postalCode	string	0/1	code postal trouvé
projection	string	1/1	Déprécié, remplacé par srs
srs	string	1/1	cf. description du paramètre en entrée
maxResponses	string	0/1	cf. description du paramètre en entrée

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/geocodeService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:geocode>
         <!--Optional:-->
         <countryCode></countryCode>
         <!--Optional:-->
         <postalCode>75013</postalCode>
         <!--Optional:-->
         <addressLine>25, rue de toldiac</addressLine>
         <!--Optional:-->
         <city>paris</city>
         <!--Optional:-->
         <region></region>
         <!--Optional:-->
         <projection></projection>
         <!--Optional:-->
         <srs></srs>
         <maxResponses>2</maxResponses>
         <streetMinScore></streetMinScore>
      </sch:geocode>
```

```
    </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:geocodeResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <GeocodeResult>
            <status>OK</status>
            <geocodedAddress>
               <addressLine>25 RUE DE TOLBIAC</addressLine>
               <city>PARIS</city>
               <countryCode>FR</countryCode>
               <postalCode>75013</postalCode>
               <projection>epsg:27572</projection>
               <srs>epsg:27572</srs>
               <secondaryZone>751135014</secondaryZone>
               <geocodeScore>19.8101</geocodeScore>
               <score>99.05</score>
               <geocodeType>4</geocodeType>
               <x>602722.1475334</x>
               <y>2425598.4510822</y>
               <places>
                  <place>751135014</place>
               </places>
               <placeTypes>
                  <placeType>IRIS</placeType>
               </placeTypes>
               <streetNumber>25</streetNumber>
               <streetWayType>RUE</streetWayType>
               <streetWayName>DE TOLBIAC</streetWayName>
               <streetWay>RUE DE TOLBIAC</streetWay>
            </geocodedAddress>
            <geocodedAddress>
               <addressLine>PONT DE TOLBIAC</addressLine>
               <city>PARIS</city>
               <countryCode>FR</countryCode>
               <postalCode>75013</postalCode>
               <projection>epsg:27572</projection>
               <srs>epsg:27572</srs>
               <secondaryZone>751135099</secondaryZone>
               <geocodeScore>19.0505</geocodeScore>
               <score>95.25</score>
               <geocodeType>2</geocodeType>
               <x>603221.0049634</x>
               <y>2426021.5068995</y>
               <places>
                  <place>751135099</place>
               </places>
               <placeTypes>
                  <placeType>IRIS</placeType>
               </placeTypes>
               <streetNumber/>
               <streetWayType>PONT</streetWayType>
               <streetWayName>DE TOLBIAC</streetWayName>
               <streetWay>PONT DE TOLBIAC</streetWay>
            </geocodedAddress>
            <initialAddress>
               <addressLine>25, rue de toldiac</addressLine>
               <city>paris</city>
               <region/>
               <countryCode/>
```

```
               <postalCode>75013</postalCode>
               <projection/>
               <srs/>
            </initialAddress>
         </GeocodeResult>
      </ns2:geocodeResponse>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/geocode.json?countryCode=fr&addressLine=25%20rue%20de
%20toldiac&postalCode=75013&city=Paris&srs=epsg:4326&maxResponses=2
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/geocode.json?countryCode=fr&addressLine=25%20rue%20de
%20toldiac&postalCode=75013&city=Paris&srs=epsg:4326&maxResponses=2&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/geocode.xml?countryCode=fr&addressLine=25%20rue%20de
%20toldiac&postalCode=75013&city=Paris&maxResponses=2
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
        "message":null,"status":"OK",
        "geocodedAddresses":    [
                {
                        "addressLine":"25 RUE DE
 TOLBIAC","city":"PARIS","countryCode":"FR","postalCode":"75013",

 "projection":"epsg:4326","srs":"epsg:4326","maxResponses":null,"secondaryZone":"751135014",
                        "geocodeScore":19.8101,"score":99.05,"geocodeType":4,"x":2.373562,"y":48.828757,
                        "places":["751135014"],"placeTypes":["IRIS"],
                        "streetNumber":"25","streetWayType":"RUE","streetWayName":"DE
 TOLBIAC","streetWay":"RUE DE TOLBIAC"
                },
                {
                        "addressLine":"PONT DE
 TOLBIAC","city":"PARIS","countryCode":"FR","postalCode":"75013",

 "projection":"epsg:4326","srs":"epsg:4326","maxResponses":null,"secondaryZone":"751135099",
                        "geocodeScore":19.0505,"score":95.25,"geocodeType":2,"x":2.380356,"y":48.832557,
                        "places":["751135099"],"placeTypes":["IRIS"],
                        "streetNumber":"","streetWayType":"PONT","streetWayName":"DE
 TOLBIAC","streetWay":"PONT DE TOLBIAC"
                }
```

```
        ],
        "initialAddress":               {
                "addressLine":"25 rue de toldiac","city":"Paris","countryCode":"fr","postalCode":"75013",
                "projection":"epsg:4326","srs":"epsg:4326"
        }
}
```


Format JSON-P


```
myCallback({
        "message":null,"status":"OK",
        "geocodedAddresses":    [
                {
                        "addressLine":"25 RUE DE
 TOLBIAC","city":"PARIS","countryCode":"FR","postalCode":"75013",

 "projection":"epsg:4326","srs":"epsg:4326","maxResponses":null,"secondaryZone":"751135014",
                        "geocodeScore":19.8101,"score":99.05,"geocodeType":4,"x":2.373562,"y":48.828757,
                        "places":["751135014"],"placeTypes":["IRIS"],
                        "streetNumber":"25","streetWayType":"RUE","streetWayName":"DE
 TOLBIAC","streetWay":"RUE DE TOLBIAC"
                },
                {
                        "addressLine":"PONT DE
 TOLBIAC","city":"PARIS","countryCode":"FR","postalCode":"75013",

 "projection":"epsg:4326","srs":"epsg:4326","maxResponses":null,"secondaryZone":"751135099",
                        "geocodeScore":19.0505,"score":95.25,"geocodeType":2,"x":2.380356,"y":48.832557,
                        "places":["751135099"],"placeTypes":["IRIS"],
                        "streetNumber":"","streetWayType":"PONT","streetWayName":"DE
 TOLBIAC","streetWay":"PONT DE TOLBIAC"
                }
        ]
        "initialAddress":       {
                "addressLine":"25 rue de toldiac","city":"Paris","countryCode":"fr","postalCode":"75013",
                "projection":"epsg:4326","srs":"epsg:4326"
        }
});
```


Format XML


```
<geocodeResult>
        <status>OK</status>
        <geocodedAddress>
                <addressLine>25 RUE DE TOLBIAC</addressLine>
                <city>PARIS</city>
                <countryCode>FR</countryCode>
                <postalCode>75013</postalCode>
                <projection>epsg:27572</projection>
                <srs>epsg:27572</srs>
                <secondaryZone>751135014</secondaryZone>
                <geocodeScore>19.8101</geocodeScore>
                <score>99.05</score>
                <geocodeType>4</geocodeType>
                <x>602722.1475334</x>
                <y>2425598.4510822</y>
                <places>
                        <place>751135014</place>
                </places>
                <placeTypes>
                        <placeType>IRIS</placeType>
                </placeTypes>
```

```
                <streetNumber>25</streetNumber>
                <streetWayType>RUE</streetWayType>
                <streetWayName>DE TOLBIAC</streetWayName>
                <streetWay>RUE DE TOLBIAC</streetWay>
        </geocodedAddress>
        <geocodedAddress>
                <addressLine>PONT DE TOLBIAC</addressLine>
                <city>PARIS</city>
                <countryCode>FR</countryCode>
                <postalCode>75013</postalCode>
                <projection>epsg:27572</projection>
                <srs>epsg:27572</srs>
                <secondaryZone>751135099</secondaryZone>
                <geocodeScore>19.0505</geocodeScore>
                <score>95.25</score>
                <geocodeType>2</geocodeType>
                <x>603221.0049634</x>
                <y>2426021.5068995</y>
                <places>
                        <place>751135099</place>
                </places>
                <placeTypes>
                        <placeType>IRIS</placeType>
                </placeTypes>
                <streetNumber/>
                <streetWayType>PONT</streetWayType>
                <streetWayName>DE TOLBIAC</streetWayName>
                <streetWay>PONT DE TOLBIAC</streetWay>
        </geocodedAddress>
        <initialAddress>
                <addressLine>25 rue de toldiac</addressLine>
                <city>Paris</city>
                <countryCode>fr</countryCode>
                <postalCode>75013</postalCode>
        </initialAddress>
</geocodeResult>
```

API JavaScript

Inclure la librairie JavaScript.

```
var geocoder = new GCUI.Control.GeoCode();
geocoder.geocode({url : 'http://<server>/<webapp>/api/lbs/geocode.json', city : 'Paris', postalCode :
 '75013', addressLine : '25 rue de Toldiac', countryCode : 'fr', srs : 'wgs84',
callback : function(result) {...} }) ;
```

La variable **`result`** est au format JSON décrit ci-dessus. La fonction **`callback`** passée en paramètre est appelée à la fin du géocodage.

Retours possibles

Cas d'une adresse trouvée (geocodeResponse/GeocodeResult/status est OK)

```
<ns2:geocodeResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
        <GeocodeResult>
                <status>OK</status>
                <geocodedAddress>
                        <addressLine/>
                        <city>BAGNEUX</city>
                        <countryCode>FR</countryCode>
```

```
                        <postalCode>92220</postalCode>
                        <projection>epsg:4326</projection>
                        <srs>epsg:4326</srs>
                        <secondaryZone>920070110</secondaryZone>
                        <geocodeScore>20.0</geocodeScore>
                        <score>100.0</score>
                        <geocodeType>1</geocodeType>
                        <x>2.30811</x>
                        <y>48.79692</y>
                        <place>920070110</place>
                        <placeType>IRIS</placeType>
                        <streetNumber/>
                        <streetWayType/>
                        <streetWayName/>
                        <streetWay/>
                </geocodedAddress>
                <initialAddress>
                        <city>Bagneux</city>
                        <postalCode>92</postalCode>
                        <projection>epsg:4326</projection>
                </initialAddress>
        </GeocodeResult>
</ns2:geocodeResponse>
```

Cas d'une adresse non trouvée (geocodeResponse/GeocodeResult/status est OK et pas de geocodedAddress)

```
<ns2:geocodeResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
        <GeocodeResult>
                <status>OK</status>
                <initialAddress>
                        <city>#hdkvnjsdvn</city>
                </initialAddress>
        </GeocodeResult>
</ns2:geocodeResponse>
```

Cas d'une requête ayant une erreur XML ou ne respectant pas le WSDL ⇒ erreur avec faultstring qui contient la description

```
<soap:Fault>
        <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
        <faultstring>Message part {http://geoconcept.com/gc/schemas}geocodeABCD was not recognized.  (Does
 it exist in service WSDL?)</faultstring>
</soap:Fault>
```

Cas d'une requête avec un système de reprojection inexistant ⇒ erreur avec faultstring qui contient la description

```
<soap:Fault>
        <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
        <faultstring>Geocode failed Failed to process geocoding task Unsupported coordinate system
 'epsg:432666666'</faultstring>
</soap:Fault>
```

FAQ

1. Comment interpréter la note obtenu dans le *score* ?

Une correspondance stricte (équivalent à une note de 100) équivaut à n'admettre aucune tolérance sur un fichier qui ne doit comporter aucune erreur. En choisissant cette note, on n'accepte que les propositions dont la correspondance est parfaite entre la table de référence et l'adresse recherchée. Plus on est permissif (plus on baisse la note) et plus l'utilisateur accepte des lettres approchantes entre les adresses du fichier à géocoder et la table de référence. Plus on a de chance que le logiciel affecte des coordonnées fausses à certaines adresses du fichier par mauvaise interprétation. Il est recommandé de ne retenir que les candidats ayant une note supérieure ou égale à 90. Cette note est un bon compromis dans la mesure où une ou deux approximations entre les adresses à géocoder et la table de référence sont tolérées. Il est vivement recommandé de ne pas conserver les candidats ayant une note inférieure à 75 : elle correspond a au moins 4 fautes d'orthographe ou de syntaxe dans l'adresse. Différents critères interviennent dans le calcul de cette note :

- le nombre de lettres équivalentes entre les deux chaînes ;
- le nombre de mots ;
- la longueur de chaque mot constituant la chaîne.

Dans le cas des correspondances d'adresses, seuls les noms de voie interviennent dans la recherche de correspondance orthographique. Le type de voie intervient comme facteur supplémentaire de la note uniquement dans le cas où le type de voie recherché et celui proposé sont identiques.

2. Combien de caractères sont retournés dans la balise *postCode* ?

   Cela dépend des caractéristiques des codes postaux de chaque pays. Par ailleurs, la chaîne retournée pourra être tronquée aux N premiers caractères si l'adresse cherchée n'est pas assez précise.

3. Quelles sont les règles sur les coordonnées X-Y (format, plage de coordonnées) ?

   Ce sont des nombres sous forme décimale, avec le "." pour séparateur de la partie décimale. Dans le cas de coordonnées géographiques en degrés, le retour doit être compris entre -180 et 180 (X=longitude) et -90 et 90 (Y=latitude), avec 6 chiffres après la virgule. Dans le cas de cordonnées projetées, le résultat est en mètres, avec deux chiffres après la virgule.

4. Ce service est-il capable de trouver une adresse approchante ?

   Si la ville n'existe pas, il retournera une liste vide.

5. Comment sont gérés les *Cedex* ?

   La grammaire, décrite dans la documentation UGC.PDF, applique un filtre afin de ne pas tenir compte du Cedex et des codes qui le suivent lors de la recherche des adresses.

6. Peut-on avoir en entrée tous les éléments de l'adresse dans un seul champ ?

   Oui, l'élément *addressLine* peut être utilisé pour réaliser un géocodage fulltext via un champ unique comportant la concaténation des champs : numéro de rue, type de voie, nom de la voie, code postal et nom de la ville. Les adresses rentrées via l'élément *addressLine* sont analysées en deux phases :

- Le mécanisme de recherche par motif permet d'abord de retrouver la ou les villes présentes dans la chaîne. La recherche par motif retrouve très rapidement un libellé correspondant à un dictionnaire, ici un dictionnaire des villes. Ce libellé peut être mal orthographié, avec les erreurs habituelles dans les saisies sur internet : doublon de lettre, oubli de lettre, remplacement d'une lettre par une autre donnant un son quasi équivalent (C a la place d'un K, etc…), partie du nom pour une ville avec un nom composé. Par ce biais on obtient un ensemble de villes candidates en un temps très court.
- Ensuite, les couples ville trouvée + chaine initiale sans la partie « ville » venant d'être reconnue sont données au géocodeur pour réaliser le géocodage habituel.

Remarque : quand un nom de ville est également présent dans le nom d'une rue, par exemple « 12 rue d'Aix Antibes » nous aurons potentiellement 2 candidats : rue d'Aix à Antibes et rue d'Antibes à Aix. Le candidat optimal est déterminé par sa position relative dans la chaîne.

7. Peut-on utiliser simultanément plusieurs référentiels ?

   Oui, il existe deux méthodes : La première consiste à s'appuyer sur un paramétrage via la « définition du géocodage » Cf. Administration / Outils / Définition du géocodage. La seconde, plus technique, se déroule en plusieurs étapes :

   - Stopper le service Tomcat,
   - Déposer les fichiers du référentiel de géocodage (8 fichiers portant des extensions .ugc.xxi.) dans le répertoire approprié. Exemple : Ici, deux tables sont déposées dans le répertoire suivant : C:\[Home]\Geoconcept\UGC\JEE\ugc\reftables, dans le cadre d'un paramétrage par défaut.
   - Editer le fichier « Service.xml » associé au module UGC JEE. Ce dernier se trouve ici : C:\[Home]\Geoconcept\UGC\JEE\ugc\conf.
   - Dans l'éditeur de texte de votre choix, copier/coller autant de fois que vous avez de tables de référence le bloc de texte encadré par les balises **<default-datasource>** (dans notre exemple, il y a 2 sources différentes, à copier/coller 2 fois par conséquent),
   - Pour chacun des blocs copié/collé, renommer la balise **<default-datasource>** en **<datasource>**,
   - La balise **<file />** doit indiquer le nom de l'une des deux tables de référence. La balse **<name />** permet de donner un alias à cette source, exemple ici : « HERE ».

Edition du premier bloc associé à la première table de référence

```
<datasource>
  <file>france_dom_M20.ugc.mdi</file>
  <name>HERE</name>
  <title />
  <abstract />
  <online-resource />
  <zone-meaning />
  <uniqueId-meaning />
  <secondaryZone-meaning />
  <roadId-meaning />
  <country />
  <version />
  <min-processors>0</min-processors>
  <max-processors>1</max-processors>
```

```
    <cache-enabled>true</cache-enabled>
    <cache-max-size>65536</cache-max-size>
    <city-scoring-method>levenshtein</city-scoring-method>
    <street-scoring-method>levenshtein</street-scoring-method>
    <min-streets>0</min-streets>
    <default-hierachy-search-depth>-1</default-hierachy-search-depth>
    <city-name-prefix-index-size>0</city-name-prefix-index-size>
    <strategy>
      <on-no-exact-city-match score-all-candidates="true" />
      <on-no-exact-street-match score-all-candidates="true" />
      <city-search-scoring max-error-threshold="85" />
      <street-search-scoring max-error-threshold="85" />
      <search hirerachy-depth="-1" />
      <!-- <force-search-two-keys /> -->
      <!-- <filter-zone filters="&gt;##" /> -->
    </strategy>
    <country />
    <coordinateSystem />
    <bounding-box />
    <max-candidates>10</max-candidates>
    <find-type>street number</find-type>
    <tolerate-find-type>7</tolerate-find-type>
    <max-meter-error>50</max-meter-error>
    <discrepancy>0.0</discrepancy>
    <discrepancy-along-street>0.0</discrepancy-along-street>
    <favor-city-match-element>favor city name</favor-city-match-element>
    <zone-match-digits>auto</zone-match-digits>
    <score-threshold>0.0</score-threshold>
    <score-threshold-to-tolerate-street-geocoding-type>0.0</score-threshold-to-tolerate-street-geocoding-
type>
    <score-threshold-to-accept-city>0.88</score-threshold-to-accept-city>
  </datasource>
```

- Laisser les autres paramètres par défaut,
- Répéter les étapes de la copie du bloc **<datasource>** et l'édition des balises **<file/>** et **<name/>** pour paramétrer la seconde source de données.

Edition du second bloc associé à l’autre table de référence qui sera exploitée

```
  <datasource>
    <file>BD_ADRESSE_france_M20.ugc.mdi</file>
    <name>IGN</name>
    <title />
    <abstract />
    <online-resource />
    <zone-meaning />
    <uniqueId-meaning />
    <secondaryZone-meaning />
    <roadId-meaning />
    <country />
    <version />
    <min-processors>0</min-processors>
    <max-processors>1</max-processors>
    <cache-enabled>true</cache-enabled>
    <cache-max-size>65536</cache-max-size>
    <city-scoring-method>levenshtein</city-scoring-method>
    <street-scoring-method>levenshtein</street-scoring-method>
    <min-streets>0</min-streets>
    <default-hierachy-search-depth>-1</default-hierachy-search-depth>
    <city-name-prefix-index-size>0</city-name-prefix-index-size>
    <strategy>
```

```
        <on-no-exact-city-match score-all-candidates="true" />
        <on-no-exact-street-match score-all-candidates="true" />
        <city-search-scoring max-error-threshold="85" />
        <street-search-scoring max-error-threshold="85" />
        <search hirerachy-depth="-1" />
        <!-- <force-search-two-keys /> -->
        <!-- <filter-zone filters="&gt;##" /> -->
      </strategy>
      <country />
      <coordinateSystem />
      <bounding-box />
      <max-candidates>10</max-candidates>
      <find-type>street number</find-type>
      <tolerate-find-type>7</tolerate-find-type>
      <max-meter-error>50</max-meter-error>
      <discrepancy>0.0</discrepancy>
      <discrepancy-along-street>0.0</discrepancy-along-street>
      <favor-city-match-element>favor city name</favor-city-match-element>
      <zone-match-digits>auto</zone-match-digits>
      <score-threshold>0.0</score-threshold>
      <score-threshold-to-tolerate-street-geocoding-type>0.0</score-threshold-to-tolerate-street-geocoding-
type>
      <score-threshold-to-accept-city>0.88</score-threshold-to-accept-city>
    </datasource>
```

- Sauvegarder le fichier « Service.xml » et relancer le service Tomcat,
- Se connecter au portail Geoconcept Web et aller dans le menu `Administration ‣ Paramètres` ,
- Ajouter une nouvelle clef de paramétrage via le bouton `Ajouter` , indiquer les informations suivantes : Nom = geocoder.countryDatasource et Valeur = %country% , puis valider en cliquant sur `OK` .
- Désormais, lorsque l'on fait appel au WebService de Géocodage, l'utilisateur devra appeler une des deux ressources dans sa requête en indiquant dans le paramètre **<countryCode>** l'alias qui a été défini auparavant via la balise **<name>**. si aucun alias n'est passé, la source définie dans le paramètre geocoder.datasource sera utilisée par défaut.

Géocodage batch

Cette page n'est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

La requête comprend une adresse ou plusieurs adresses en entrée, le service retourne, pour chaque item, une ou plusieurs réponses possibles (s'il y a ambiguïté), en incluant l'adresse reconnue, la position, le score de géocodage, le type de géocodage.

Ce service de géocodage interroge la table de référence paramétrée grâce à l'interface Administration de Geoconcept Web.

Disponibilité

Ce web service est disponible en permanence avec Geoconcept Web et une table de référence.

Changement de version

Les versions précédente du web service sont conservées dans Geoconcept Web pour assurer la compatibilité avec les développements antérieurs. Il est recommandé d'utiliser la version la plus récente.

Changements avec la v2

- Suppression du paramètre "projection", remplacé par "srs"
- Suppression du paramètre "geocodeScore", remplacé par "score"
- Suppression des paramètres "projection", "srs" et "maxResponses" dans le bloc "initialAddress"
- Le paramètre "postalCode" est renommé "postCode"

V2

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
address (geocodeInitialAddress)	Les adresses à géocoder	non	
streetMinScore	la valeur de ce paramètre varie, comme pour le score, entre 0 et 100 (Cf. description complète dans le web service du géocodage)	oui	
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	Sans projection, le résultat est en projection native de l'index de géocodage, habituellement wgs84 .
maxResponses	nombre maximum de résultats d'adresses dans la réponse	oui	

Adresses (geocodeInitialAddress)

paramètre	description	optionnel	défaut
addressLine	adresse comprenant numéro, indice de répétition, type de voie et libellé de voie.	oui *	
city	ville	oui *	
region	État, Comté, …	oui	
countryCode	pays sur 2 ou 3 lettres (code ISO 3166-1) par exemple "fr" ou "fra",	oui	
postCode	code postal	oui *	

(*) Au moins l'un des trois paramètres postCode, addressLine et city doit être renseigné.

En sortie

paramètre	type	min/max	description
geocodedAddress (ou geocodedAddresses en JSON / JSON-P)	geocodedAddress (ou array en JSON / JSON-P)	0/illimité	Adresses géocodées
initialAddress	geocodeInitialAddress	0/1	Adresse initiale

Adresses géocodées (geocodedAddress)

paramètre	type	min/max	description
addressLine	string	0/1	rue trouvée et le cas échéant le numéro
city	string	0/1	ville trouvée
region	string	0/1	État, Comté, … trouvée, varie en fonction des pays, peut également être vide
countryCode	string	1/1	cf. description du paramètre en entrée
postCode	string	0/1	code postal trouvé
secondaryZone	string	0/1	zone qui dépend de l'index de géocodage, habituellement en France il s'agit du code IRIS
score	double	1/1	note du géocodage de 0 à 100, avec 100 pour une correspondance parfaite
geocodeType	int	1/1	type de géocodage : - ville = 1 - rue = 2 - rue amélioré = 3 - numéro de rue = 4 - non géocodé = 0
x	double	1/1	coordonnées X
y	double	1/1	coordonnées Y
place (ou places en JSON / JSON-P)	string (ou array en JSON / JSON-P)	0/illimité	liste des attributs. Valeur des attributs de l'adresse trouvée, en relation avec *placeTypes* (par exemple ["751010206","930005Y001XCHE"]). Varie en fonction du référentiel utilisé.
placeType (ou placeTypes en JSON / JSON-P)	string (ou array en JSON / JSON-P)	0/illimité	liste des types d'attributs (par exemple ["IRIS","FANTOIR"]). Varie en fonction du référentiel utilisé.
streetNumber	string	0/1	numéro de rue
streetWayType	string	0/1	type de rue (avenue, street, etc)
streetWayName	string	0/1	nom de la rue
streetWay	string	0/1	nom complet de la rue

Adresse initiale (initialAddress)

paramètre	type	min/max	description
addressLine	string	0/1	rue trouvée et le cas échéant le numéro
city	string	0/1	ville trouvée
region	string	0/1	État, Comté, … trouvée, varie en fonction des pays, peut également être vide
countryCode	string	1/1	cf. description du paramètre en entrée

paramètre	type	min/max	description
postCode	string	0/1	code postal trouvé

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/geocodeService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:batchGeocodeV2>
         <!--Optional:-->
         <addresses>
            <!--Zero or more repetitions:-->
            <address>
               <!--Optional:-->
               <addressLine>23, rue de la gare</addressLine>
               <!--Optional:-->
               <city>Saint-Herblain</city>
               <!--Optional:-->
               <region></region>
               <!--Optional:-->
               <countryCode>FR</countryCode>
               <!--Optional:-->
               <postCode>44800</postCode>
            </address>
            <address>
               <!--Optional:-->
               <addressLine>32 Route de Pornic</addressLine>
               <!--Optional:-->
               <city>Bouguenais</city>
               <!--Optional:-->
               <region></region>
               <!--Optional:-->
               <countryCode>FR</countryCode>
               <!--Optional:-->
               <postCode>44340</postCode>
            </address>
            <address>
               <!--Optional:-->
               <addressLine>5, Avenue Victor Hugo</addressLine>
               <!--Optional:-->
               <city>Sainte-Luce-sur-Loire</city>
               <!--Optional:-->
               <region></region>
               <!--Optional:-->
               <countryCode>FR</countryCode>
               <!--Optional:-->
               <postCode>44980</postCode>
            </address>
            <streetMinScore></streetMinScore>
            <!--Optional:-->
            <srs></srs>
            <maxResponses>2</maxResponses>
         </addresses>
      </sch:batchGeocodeV2>
```

```
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:batchGeocodeV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <BatchGeocodeResult>
            <status>OK</status>
            <result>
               <status>OK</status>
               <geocodedAddress>
                  <addressLine>23 RUE DE LA GARE</addressLine>
                  <city>SAINT-HERBLAIN</city>
                  <countryCode>FR</countryCode>
                  <postCode>44800</postCode>
                  <srs>epsg:4326</srs>
                  <secondaryZone>441620701</secondaryZone>
                  <score>100.0</score>
                  <geocodeType>4</geocodeType>
                  <x>-1.656851</x>
                  <y>47.21028</y>
                  <places>
                     <place>441620701</place>
                     <place>441620701_010</place>
                     <place>4401621080</place>
                     <place>44162</place>
                  </places>
                  <placeTypes>
                     <placeType>IRIS</placeType>
                     <placeType>ILOT</placeType>
                     <placeType>FANTOIR</placeType>
                     <placeType>INSEE</placeType>
                  </placeTypes>
                  <streetNumber>23</streetNumber>
                  <streetWayType>RUE</streetWayType>
                  <streetWayName>DE LA GARE</streetWayName>
                  <streetWay>RUE DE LA GARE</streetWay>
               </geocodedAddress>
               <initialAddress>
                  <addressLine>23, rue de la gare</addressLine>
                  <city>Saint-Herblain</city>
                  <region/>
                  <countryCode>FR</countryCode>
                  <postCode>44800</postCode>
               </initialAddress>
            </result>
            <result>
               <status>OK</status>
               <geocodedAddress>
                  <addressLine>ROUTE DE PORNIC</addressLine>
                  <city>BOUGUENAIS</city>
                  <countryCode>FR</countryCode>
                  <postCode>44340</postCode>
                  <srs>epsg:4326</srs>
                  <secondaryZone>440200106</secondaryZone>
                  <score>100.0</score>
                  <geocodeType>2</geocodeType>
                  <x>-1.581521</x>
                  <y>47.189556</y>
                  <places>
                     <place>440200106</place>
```

```
                  <place>440200106_038</place>
                  <place>4400202800</place>
                  <place>44020</place>
               </places>
               <placeTypes>
                  <placeType>IRIS</placeType>
                  <placeType>ILOT</placeType>
                  <placeType>FANTOIR</placeType>
                  <placeType>INSEE</placeType>
               </placeTypes>
               <streetNumber/>
               <streetWayType>ROUTE</streetWayType>
               <streetWayName>DE PORNIC</streetWayName>
               <streetWay>ROUTE DE PORNIC</streetWay>
            </geocodedAddress>
            <initialAddress>
               <addressLine>32 Route de Pornic</addressLine>
               <city>Bouguenais</city>
               <region/>
               <countryCode>FR</countryCode>
               <postCode>44340</postCode>
            </initialAddress>
         </result>
         <result>
            <status>OK</status>
            <geocodedAddress>
               <addressLine/>
               <city>SAINTE-LUCE-SUR-LOIRE</city>
               <countryCode>FR</countryCode>
               <postCode>44980</postCode>
               <srs>epsg:4326</srs>
               <secondaryZone>44172</secondaryZone>
               <score>100.0</score>
               <geocodeType>1</geocodeType>
               <x>-1.48669</x>
               <y>47.24961</y>
               <places>
                  <place>44172</place>
               </places>
               <placeTypes>
                  <placeType>IRIS</placeType>
               </placeTypes>
               <streetNumber/>
               <streetWayType/>
               <streetWayName/>
               <streetWay/>
            </geocodedAddress>
            <initialAddress>
               <addressLine>5, Avenue Victor Hugo</addressLine>
               <city>Sainte-Luce-sur-Loire</city>
               <region/>
               <countryCode>FR</countryCode>
               <postCode>44980</postCode>
            </initialAddress>
         </result>
      </BatchGeocodeResult>
   </ns2:batchGeocodeV2Response>
</soap:Body>
</soap:Envelope>
```

REST (POST)

Requête

Requête

```
http://<server>/<webapp>/api/lbs/geocode/batch/v2.xml
```

Data (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<batchGeocodeRequestV2>
  <address>
    <addressLine>23, rue de la gare</addressLine>
    <city>Saint-Herblain</city>
    <region></region>
    <countryCode>FR</countryCode>
    <postCode>44800</postCode>
  </address>
  <address>
    <addressLine>32 Route de Pornic</addressLine>
    <city>Bouguenais</city>
    <region></region>
    <countryCode>FR</countryCode>
    <postCode>44340</postCode>
  </address>
    <address>
    <addressLine>5, Avenue Victor Hugo</addressLine>
    <city>Sainte-Luce-sur-Loire</city>
    <region></region>
    <countryCode>FR</countryCode>
    <postCode>44980</postCode>
  </address>
  <streetMinScore></streetMinScore>
  <srs>epsg:4326</srs>
  <maxResponses>2</maxResponses>
</batchGeocodeRequestV2>
```

Réponse

La réponse est toujours encodée en UTF-8.

Format XML

```
<batchGeocodeResultV2>
   <status>OK</status>
   <result>
      <status>OK</status>
      <geocodedAddress>
         <addressLine>23 RUE DE LA GARE</addressLine>
         <city>SAINT-HERBLAIN</city>
         <countryCode>FR</countryCode>
         <postCode>44800</postCode>
         <srs>epsg:4326</srs>
         <secondaryZone>441620701</secondaryZone>
         <score>100.0</score>
         <geocodeType>4</geocodeType>
         <x>-1.656851</x>
         <y>47.21028</y>
         <places>
```

```
            <place>441620701</place>
            <place>441620701_010</place>
            <place>4401621080</place>
            <place>44162</place>
         </places>
         <placeTypes>
            <placeType>IRIS</placeType>
            <placeType>ILOT</placeType>
            <placeType>FANTOIR</placeType>
            <placeType>INSEE</placeType>
         </placeTypes>
         <streetNumber>23</streetNumber>
         <streetWayType>RUE</streetWayType>
         <streetWayName>DE LA GARE</streetWayName>
         <streetWay>RUE DE LA GARE</streetWay>
      </geocodedAddress>
      <initialAddress>
         <addressLine>23, rue de la gare</addressLine>
         <city>Saint-Herblain</city>
         <region/>
         <countryCode>FR</countryCode>
         <postCode>44800</postCode>
      </initialAddress>
   </result>
   <result>
      <status>OK</status>
      <geocodedAddress>
         <addressLine>ROUTE DE PORNIC</addressLine>
         <city>BOUGUENAIS</city>
         <countryCode>FR</countryCode>
         <postCode>44340</postCode>
         <srs>epsg:4326</srs>
         <secondaryZone>440200106</secondaryZone>
         <score>100.0</score>
         <geocodeType>2</geocodeType>
         <x>-1.581521</x>
         <y>47.189556</y>
         <places>
            <place>440200106</place>
            <place>440200106_038</place>
            <place>4400202800</place>
            <place>44020</place>
         </places>
         <placeTypes>
            <placeType>IRIS</placeType>
            <placeType>ILOT</placeType>
            <placeType>FANTOIR</placeType>
            <placeType>INSEE</placeType>
         </placeTypes>
         <streetNumber/>
         <streetWayType>ROUTE</streetWayType>
         <streetWayName>DE PORNIC</streetWayName>
         <streetWay>ROUTE DE PORNIC</streetWay>
      </geocodedAddress>
      <initialAddress>
         <addressLine>32 Route de Pornic</addressLine>
         <city>Bouguenais</city>
         <region/>
         <countryCode>FR</countryCode>
         <postCode>44340</postCode>
      </initialAddress>
   </result>
   <result>
```

```
      <status>OK</status>
      <geocodedAddress>
         <addressLine/>
         <city>SAINTE-LUCE-SUR-LOIRE</city>
         <countryCode>FR</countryCode>
         <postCode>44980</postCode>
         <srs>epsg:4326</srs>
         <secondaryZone>44172</secondaryZone>
         <score>100.0</score>
         <geocodeType>1</geocodeType>
         <x>-1.48669</x>
         <y>47.24961</y>
         <places>
            <place>44172</place>
         </places>
         <placeTypes>
            <placeType>IRIS</placeType>
         </placeTypes>
         <streetNumber/>
         <streetWayType/>
         <streetWayName/>
         <streetWay/>
      </geocodedAddress>
      <initialAddress>
         <addressLine>5, Avenue Victor Hugo</addressLine>
         <city>Sainte-Luce-sur-Loire</city>
         <region/>
         <countryCode>FR</countryCode>
         <postCode>44980</postCode>
      </initialAddress>
   </result>
</batchGeocodeResultV2>
```

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/geocode/batch/v2.json
```

JSON

```
{
  "addresses" :
  [
  {
    "addressLine" : "23, rue de la gare",
    "city" : "Saint-Herblain",
    "region" : "",
    "countryCode" : "FR",
    "postCode" : "44800"
  },
  {
    "addressLine" : "32 Route de Pornic",
    "city" : "Bouguenais",
    "region" : "",
    "countryCode" : "FR",
    "postCode" : "44340"
  },
  {
    "addressLine" : "5, Avenue Victor Hugo",
    "city" : "Sainte-Luce-sur-Loire",
```

```
    "region" : "",
    "countryCode" : "FR",
    "postCode" : "44980"
  }
 ],
  "streetMinScore" : 80 ,
  "srs" : "epsg:4326",
  "maxResponses" : 2
}
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message": null,
   "status": "OK",
   "results":    [
            {
         "message": null,
         "status": "OK",
         "geocodedAddresses": [         {
            "addressLine": "23 RUE DE LA GARE",
            "city": "SAINT-HERBLAIN",
            "countryCode": "FR",
            "postCode": "44800",
            "srs": "epsg:4326",
            "secondaryZone": "",
            "score": 100,
            "geocodeType": 4,
            "x": -1.65801,
            "y": 47.20903,
            "places": [""],
            "placeTypes": ["IRIS"],
            "streetNumber": "23",
            "streetWayType": "RUE",
            "streetWayName": "DE LA GARE",
            "streetWay": "RUE DE LA GARE"
         }],
         "initialAddress":          {
            "addressLine": "23, rue de la gare",
            "city": "Saint-Herblain",
            "region": "",
            "countryCode": "FR",
            "postCode": "44800"
         }
      },
            {
         "message": null,
         "status": "OK",
         "geocodedAddresses": [         {
            "addressLine": "32 ROUTE DE PORNIC",
            "city": "BOUGUENAIS",
            "countryCode": "FR",
            "postCode": "44340",
            "srs": "epsg:4326",
            "secondaryZone": "",
            "score": 100,
            "geocodeType": 3,
            "x": -1.58862,
```

```
            "y": 47.18768,
            "places": [""],
            "placeTypes": ["IRIS"],
            "streetNumber": "32",
            "streetWayType": "ROUTE",
            "streetWayName": "DE PORNIC",
            "streetWay": "ROUTE DE PORNIC"
         }],
         "initialAddress":          {
            "addressLine": "32 Route de Pornic",
            "city": "Bouguenais",
            "region": "",
            "countryCode": "FR",
            "postCode": "44340"
         }
      },
            {
         "message": null,
         "status": "OK",
         "geocodedAddresses": [         {
            "addressLine": "",
            "city": "SAINTE-LUCE-SUR-LOIRE",
            "countryCode": "FR",
            "postCode": "44980",
            "srs": "epsg:4326",
            "secondaryZone": "",
            "score": 100,
            "geocodeType": 1,
            "x": -1.48669,
            "y": 47.24961,
            "places": [""],
            "placeTypes": ["IRIS"],
            "streetNumber": "",
            "streetWayType": "",
            "streetWayName": "",
            "streetWay": ""
         }],
         "initialAddress":          {
            "addressLine": "5, Avenue Victor Hugo",
            "city": "Sainte-Luce-sur-Loire",
            "region": "",
            "countryCode": "FR",
            "postCode": "44980"
         }
      }
   ]
}
```


Retours possibles

Cas d’une adresse trouvée (batchGeocodeResponse/batchGeocodeResult/status est OK)


```
<ns2:batchGeocodeV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
 <BatchGeocodeResult>
    <status>OK</status>
    <result>
       <status>OK</status>
       <geocodedAddress>
          <addressLine>23 RUE DE LA GARE</addressLine>
          <city>SAINT-HERBLAIN</city>
          <countryCode>FR</countryCode>
          <postCode>44800</postCode>
```

```
            <srs>epsg:4326</srs>
            <secondaryZone>441620701</secondaryZone>
            <score>100.0</score>
            <geocodeType>4</geocodeType>
            <x>-1.656851</x>
            <y>47.21028</y>
            <places>
               <place>441620701</place>
               <place>441620701_010</place>
               <place>4401621080</place>
               <place>44162</place>
            </places>
            <placeTypes>
               <placeType>IRIS</placeType>
               <placeType>ILOT</placeType>
               <placeType>FANTOIR</placeType>
               <placeType>INSEE</placeType>
            </placeTypes>
            <streetNumber>23</streetNumber>
            <streetWayType>RUE</streetWayType>
            <streetWayName>DE LA GARE</streetWayName>
            <streetWay>RUE DE LA GARE</streetWay>
         </geocodedAddress>
         <initialAddress>
            <addressLine>23, rue de la gare</addressLine>
            <city>Saint-Herblain</city>
            <region/>
            <countryCode>FR</countryCode>
            <postCode>44800</postCode>
         </initialAddress>
      </result>
      <result>
          [...]
      </result>
   </BatchGeocodeResult>
</ns2:batchGeocodeV2Response>
```

Cas d'une adresse non trouvée (batchGeocodeResponse/BatchGeocodeResult/status est OK et pas de geocodedAddress)

```
<BatchGeocodeResult>
<status>OK</status>
<result>
   <status>OK</status>
   <initialAddress>
      <addressLine/>
      <city>#hdkvnjsdvn</city>
      <region/>
      <countryCode/>
      <postCode/>
   </initialAddress>
</result>
</BatchGeocodeResult>
```

Cas d'une requête ayant une erreur XML ou ne respectant pas le WSDL ⇒ erreur avec faultstring qui contient la description

```
<soap:Fault>
 <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
 <faultstring>Message part {http://geoconcept.com/gc/schemas}batchGeocodeV2fff was not recognized.  (Does it
 exist in service WSDL?)</faultstring>
```

```
</soap:Fault>
```

Cas d'une requête avec un système de reprojection inexistant ⇒ erreur avec faultstring qui contient la description

```
<soap:Fault>
        <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
        <faultstring>Geocode failed Failed to process geocoding task Unsupported coordinate system
 'epsg:432666666'</faultstring>
</soap:Fault>
```

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
address (geocodeInitialAddress)	Les adresses à géocoder	non	
streetMinScore	la valeur de ce paramètre varie, comme pour le score, entre 0 et 100 (Cf. description complète dans le web service du géocodage)	oui	
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	Sans projection, le résultat est en projection native de l'index de géocodage, habituellement wgs84 .
maxResponses	nombre maximum de résultats d'adresses dans la réponse	oui	

Adresses (geocodeInitialAddress)

paramètre	description	optionnel	défaut
addressLine	adresse comprenant numéro, indice de répétition, type de voie et libellé de voie.	oui *	
city	ville	oui *	
region	État, Comté, …	oui	
countryCode	pays sur deux lettres (code ISO 3166-1 ou ccTLD) par exemple "fr",	oui	
postalCode	code postal	oui *	
projection	déprécié	oui	
srs	déprécié	oui	
maxResponses	déprécié	oui	

(*) Au moins l'un des trois paramètres postalCode, addressLine et city doit être renseigné.

En sortie

paramètre	type	min/max	description
geocodedAddress (ou geocodedAddresses en JSON / JSON-P)	geocodedAddress (ou array en JSON / JSON-P)	0/illimité	Adresses géocodées
initialAddress	geocodeInitialAddress	0/1	Adresse initiale

Adresses géocodées (geocodedAddress)

paramètre	type	min/max	description
secondaryZone	string	0/1	zone qui dépend de l'index de géocodage, habituellement en France il s'agit du code IRIS
score	double	1/1	note du géocodage de 0 à 100, avec 100 pour une correspondance parfaite
geocodeType	int	1/1	type de géocodage : - ville = 1 - rue = 2 - rue amélioré = 3 - numéro de rue = 4 - non géocodé = 0
x	double	1/1	coordonnées X
y	double	1/1	coordonnées Y
place (ou places en JSON / JSON-P)	string (ou array en JSON / JSON-P)	0/illimité	liste des attributs. Valeur des attributs de l'adresse trouvée, en relation avec *placeTypes* (par exemple ["751010206","930005Y001XCHE"]). Dépend du référentiel utilisé.
placeType (ou placeTypes en JSON / JSON-P)	string (ou array en JSON / JSON-P)	0/illimité	liste des types d'attributs (par exemple ["IRIS","FANTOIR"]) .Dépend de la table de référence utilisée.
streetNumber	string	0/1	numéro de rue
streetWayType	string	0/1	type de rue (avenue, street, etc)
streetWayName	string	0/1	nom de la rue
streetWay	string	0/1	nom complet de la rue

Adresse initiale (initialAddress)

paramètre	type	min/max	description
addressLine	string	0/1	rue trouvée et le cas échéant le numéro
city	string	0/1	ville trouvée
region	string	0/1	État, Comté, … trouvée, varie en fonction des pays, peut également être vide
countryCode	string	1/1	cf. description du paramètre en entrée
postalCode	string	0/1	code postal trouvé

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/geocodeService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:batchGeocode>
         <!--Optional:-->
         <addresses>
            <!--Zero or more repetitions:-->
            <address>
               <!--Optional:-->
               <addressLine>23, rue de la gare</addressLine>
               <!--Optional:-->
               <city>Saint-Herblain</city>
               <!--Optional:-->
               <region></region>
               <!--Optional:-->
               <countryCode>FR</countryCode>
               <!--Optional:-->
               <postalCode>44800</postalCode>
               <!--Optional:-->
               <projection></projection>
               <!--Optional:-->
               <srs>epsg:4326</srs>
               <!--Optional:-->
               <maxResponses></maxResponses>
            </address>
            <address>
               <!--Optional:-->
               <addressLine>32 Route de Pornic</addressLine>
               <!--Optional:-->
               <city>Bouguenais</city>
               <!--Optional:-->
               <region></region>
               <!--Optional:-->
               <countryCode>FR</countryCode>
               <!--Optional:-->
               <postalCode>44340</postalCode>
               <!--Optional:-->
               <projection></projection>
               <!--Optional:-->
               <srs>epsg:4326</srs>
               <!--Optional:-->
               <maxResponses></maxResponses>
            </address>
            <address>
               <!--Optional:-->
               <addressLine>5, Avenue Victor Hugo</addressLine>
               <!--Optional:-->
               <city>Sainte-Luce-sur-Loire</city>
               <!--Optional:-->
               <region></region>
               <!--Optional:-->
               <countryCode>FR</countryCode>
               <!--Optional:-->
               <postalCode>44980</postalCode>
               <!--Optional:-->
               <projection></projection>
               <!--Optional:-->
               <srs>epsg:4326</srs>
               <!--Optional:-->
               <maxResponses></maxResponses>
```

```
            </address>
            <streetMinScore></streetMinScore>
            <!--Optional:-->
            <srs>epsg:4326</srs>
            <maxResponses>2</maxResponses>
         </addresses>
      </sch:batchGeocode>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:batchGeocodeResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <BatchGeocodeResult>
            <status>OK</status>
            <result>
               <status>OK</status>
               <geocodedAddress>
                  <addressLine>23 RUE DE LA GARE</addressLine>
                  <city>SAINT-HERBLAIN</city>
                  <countryCode>FR</countryCode>
                  <postalCode>44800</postalCode>
                  <projection>epsg:4326</projection>
                  <srs>epsg:4326</srs>
                  <secondaryZone>441620701</secondaryZone>
                  <geocodeScore>20.0</geocodeScore>
                  <score>100.0</score>
                  <geocodeType>4</geocodeType>
                  <x>-1.6568714</x>
                  <y>47.2102641</y>
                  <places>
                     <place>441620701</place>
                  </places>
                  <placeTypes>
                     <placeType>IRIS</placeType>
                  </placeTypes>
                  <streetNumber>23</streetNumber>
                  <streetWayType>RUE</streetWayType>
                  <streetWayName>DE LA GARE</streetWayName>
                  <streetWay>RUE DE LA GARE</streetWay>
               </geocodedAddress>
               <initialAddress>
                  <addressLine>23, rue de la gare</addressLine>
                  <city>Saint-Herblain</city>
                  <region/>
                  <countryCode>FR</countryCode>
                  <postalCode>44800</postalCode>
                  <projection/>
                  <srs>epsg:4326</srs>
                  <maxResponses/>
               </initialAddress>
            </result>
            <result>
               <status>OK</status>
               <geocodedAddress>
                  <addressLine>ROUTE DE PORNIC</addressLine>
                  <city>BOUGUENAIS</city>
                  <countryCode>FR</countryCode>
                  <postalCode>44340</postalCode>
                  <projection>epsg:4326</projection>
                  <srs>epsg:4326</srs>
```

```
                  <secondaryZone>440200106</secondaryZone>
                  <geocodeScore>20.0</geocodeScore>
                  <score>100.0</score>
                  <geocodeType>2</geocodeType>
                  <x>-1.5799805</x>
                  <y>47.1901969</y>
                  <places>
                     <place>440200106</place>
                  </places>
                  <placeTypes>
                     <placeType>IRIS</placeType>
                  </placeTypes>
                  <streetNumber/>
                  <streetWayType>ROUTE</streetWayType>
                  <streetWayName>DE PORNIC</streetWayName>
                  <streetWay>ROUTE DE PORNIC</streetWay>
               </geocodedAddress>
               <initialAddress>
                  <addressLine>32 Route de Pornic</addressLine>
                  <city>Bouguenais</city>
                  <region/>
                  <countryCode>FR</countryCode>
                  <postalCode>44340</postalCode>
                  <projection/>
                  <srs>epsg:4326</srs>
                  <maxResponses/>
               </initialAddress>
            </result>
            <result>
               <status>OK</status>
               <geocodedAddress>
                  <addressLine/>
                  <city>SAINTE-LUCE-SUR-LOIRE</city>
                  <countryCode>FR</countryCode>
                  <postalCode>44980</postalCode>
                  <projection>epsg:4326</projection>
                  <srs>epsg:4326</srs>
                  <secondaryZone>441720105</secondaryZone>
                  <geocodeScore>20.0</geocodeScore>
                  <score>100.0</score>
                  <geocodeType>1</geocodeType>
                  <x>-1.4787216</x>
                  <y>47.2565577</y>
                  <places>
                     <place>441720105</place>
                  </places>
                  <placeTypes>
                     <placeType>IRIS</placeType>
                  </placeTypes>
                  <streetNumber/>
                  <streetWayType/>
                  <streetWayName/>
                  <streetWay/>
               </geocodedAddress>
               <initialAddress>
                  <addressLine>5, Avenue Victor Hugo</addressLine>
                  <city>Sainte-Luce-sur-Loire</city>
                  <region/>
                  <countryCode>FR</countryCode>
                  <postalCode>44980</postalCode>
                  <projection/>
                  <srs>epsg:4326</srs>
                  <maxResponses/>
```

```
               </initialAddress>
            </result>
         </BatchGeocodeResult>
      </ns2:batchGeocodeResponse>
   </soap:Body>
</soap:Envelope>
```

REST (POST)

Requête

Requête

```
http://<server>/<webapp>/api/lbs/geocode/batch/v1.xml
```

Data (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<batchGeocodeRequest>
  <address>
    <addressLine>23, rue de la gare</addressLine>
    <city>Saint-Herblain</city>
    <region></region>
    <countryCode>FR</countryCode>
    <postalCode>44800</postalCode>
    <projection></projection>
  </address>
  <address>
    <addressLine>32 Route de Pornic</addressLine>
    <city>Bouguenais</city>
    <region></region>
    <countryCode>FR</countryCode>
    <postalCode>44340</postalCode>
    <projection></projection>
  </address>
    <address>
    <addressLine>5, Avenue Victor Hugo</addressLine>
    <city>Sainte-Luce-sur-Loire</city>
    <region></region>
    <countryCode>FR</countryCode>
    <postalCode>44980</postalCode>
    <projection></projection>
  </address>
  <streetMinScore></streetMinScore>
  <srs>epsg:4326</srs>
  <maxResponses>2</maxResponses>
</batchGeocodeRequest>
```

Réponse

La réponse est toujours encodée en UTF-8.

Format XML

```
<batchGeocodeResult>
   <status>OK</status>
   <result>
      <status>OK</status>
      <geocodedAddress>
         <addressLine>23 RUE DE LA GARE</addressLine>
```

```
            <city>SAINT-HERBLAIN</city>
            <countryCode>FR</countryCode>
            <postalCode>44800</postalCode>
            <projection>epsg:4326</projection>
            <srs>epsg:4326</srs>
            <secondaryZone>441620701</secondaryZone>
            <geocodeScore>20.0</geocodeScore>
            <score>100.0</score>
            <geocodeType>4</geocodeType>
            <x>-1.6568714</x>
            <y>47.2102641</y>
            <places>
               <place>441620701</place>
            </places>
            <placeTypes>
               <placeType>IRIS</placeType>
            </placeTypes>
            <streetNumber>23</streetNumber>
            <streetWayType>RUE</streetWayType>
            <streetWayName>DE LA GARE</streetWayName>
            <streetWay>RUE DE LA GARE</streetWay>
         </geocodedAddress>
         <initialAddress>
            <addressLine>23, rue de la gare</addressLine>
            <city>Saint-Herblain</city>
            <region/>
            <countryCode>FR</countryCode>
            <postalCode>44800</postalCode>
            <projection/>
         </initialAddress>
      </result>
      <result>
         <status>OK</status>
         <geocodedAddress>
            <addressLine>ROUTE DE PORNIC</addressLine>
            <city>BOUGUENAIS</city>
            <countryCode>FR</countryCode>
            <postalCode>44340</postalCode>
            <projection>epsg:4326</projection>
            <srs>epsg:4326</srs>
            <secondaryZone>440200106</secondaryZone>
            <geocodeScore>20.0</geocodeScore>
            <score>100.0</score>
            <geocodeType>2</geocodeType>
            <x>-1.5799805</x>
            <y>47.1901969</y>
            <places>
               <place>440200106</place>
            </places>
            <placeTypes>
               <placeType>IRIS</placeType>
            </placeTypes>
            <streetNumber/>
            <streetWayType>ROUTE</streetWayType>
            <streetWayName>DE PORNIC</streetWayName>
            <streetWay>ROUTE DE PORNIC</streetWay>
         </geocodedAddress>
         <initialAddress>
            <addressLine>32 Route de Pornic</addressLine>
            <city>Bouguenais</city>
            <region/>
            <countryCode>FR</countryCode>
            <postalCode>44340</postalCode>
```

```
         <projection/>
      </initialAddress>
   </result>
   <result>
      <status>OK</status>
      <geocodedAddress>
         <addressLine/>
         <city>SAINTE-LUCE-SUR-LOIRE</city>
         <countryCode>FR</countryCode>
         <postalCode>44980</postalCode>
         <projection>epsg:4326</projection>
         <srs>epsg:4326</srs>
         <secondaryZone>441720105</secondaryZone>
         <geocodeScore>20.0</geocodeScore>
         <score>100.0</score>
         <geocodeType>1</geocodeType>
         <x>-1.4787216</x>
         <y>47.2565577</y>
         <places>
            <place>441720105</place>
         </places>
         <placeTypes>
            <placeType>IRIS</placeType>
         </placeTypes>
         <streetNumber/>
         <streetWayType/>
         <streetWayName/>
         <streetWay/>
      </geocodedAddress>
      <initialAddress>
         <addressLine>5, Avenue Victor Hugo</addressLine>
         <city>Sainte-Luce-sur-Loire</city>
         <region/>
         <countryCode>FR</countryCode>
         <postalCode>44980</postalCode>
         <projection/>
      </initialAddress>
   </result>
</batchGeocodeResult>
```

Retours possibles

Cas d'une adresse trouvée (batchGeocodeResponse/batchGeocodeResult/status est OK)

```
<ns2:batchGeocodeResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <BatchGeocodeResult>
            <status>OK</status>
            <result>
               <status>OK</status>
               <geocodedAddress>
                  <addressLine>23 RUE DE LA GARE</addressLine>
                  <city>SAINT-HERBLAIN</city>
                  <countryCode>FR</countryCode>
                  <postalCode>44800</postalCode>
                  <projection>epsg:4326</projection>
                  <srs>epsg:4326</srs>
                  <secondaryZone>441620701</secondaryZone>
                  <geocodeScore>20.0</geocodeScore>
                  <score>100.0</score>
                  <geocodeType>4</geocodeType>
                  <x>-1.6568714</x>
                  <y>47.2102641</y>
```

```
                  <places>
                     <place>441620701</place>
                  </places>
                  <placeTypes>
                     <placeType>IRIS</placeType>
                  </placeTypes>
                  <streetNumber>23</streetNumber>
                  <streetWayType>RUE</streetWayType>
                  <streetWayName>DE LA GARE</streetWayName>
                  <streetWay>RUE DE LA GARE</streetWay>
               </geocodedAddress>
               <initialAddress>
                  <addressLine>23, rue de la gare</addressLine>
                  <city>Saint-Herblain</city>
                  <region/>
                  <countryCode>FR</countryCode>
                  <postalCode>44800</postalCode>
                  <projection/>
                  <srs>epsg:4326</srs>
                  <maxResponses/>
               </initialAddress>
            </result>
            <result>
            ...
            </result>
         </BatchGeocodeResult>
</ns2:batchGeocodeResponse>
```

Cas d'une adresse non trouvée (batchGeocodeResponse/BatchGeocodeResult/status est OK et pas de geocodedAddress)

```
<ns2:batchGeocodeResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
        <BatchGeocodeResult>
                <status>OK</status>
                <initialAddress>
                        <city>#hdkvnjsdvn</city>
                </initialAddress>
        </BatchGeocodeResult>
</ns2:batchGeocodeResponse>
```

Cas d'une requête ayant une erreur XML ou ne respectant pas le WSDL ⇒ erreur avec faultstring qui contient la description

```
<soap:Fault>
        <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
        <faultstring>Message part {http://geoconcept.com/gc/schemas}geocodeABCD was not recognized.  (Does
 it exist in service WSDL?)</faultstring>
</soap:Fault>
```

Cas d'une requête avec un système de reprojection inexistant ⇒ erreur avec faultstring qui contient la description

```
<soap:Fault>
        <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
        <faultstring>Geocode failed Failed to process geocoding task Unsupported coordinate system
 'epsg:432666666'</faultstring>
</soap:Fault>
```

FAQ

Voir web service de géocodage.

1. Comment géocoder depuis la commande cURL ?

   Il est nécessaire d’appeler un fichier d’adresse, ici au format json, avec la ligne de commande suivante :

```
curl -X POST "https://<server>/<webapp>/api/lbs/geocode/batch/v2.json" -H "Content-Type: application/json"
 --data-binary @"adresses.json"
```

```
Fichier : adresses.json
```

```
{
   "addresses":[
      {
         "addressLine":"200 Quai Charles de Gaulle",
         "city":"Lyon",
         "region":"",
         "countryCode":"FR",
         "postCode":"69006"
      },
      {
         "addressLine":"Bruno Kreisky Platz 1",
         "city":"Wien",
         "region":"",
         "countryCode":"AT",
         "postCode":"1220"
      },
      {
         "addressLine":"Route des Morillons 15",
         "city":"Genève",
         "region":"",
         "countryCode":"CH",
         "postCode":"1202"
      }
   ],
   "streetMinScore":80,
   "srs":"epsg:4326",
   "maxResponses":2
}
```

Contraction d’adresse

> Cette page n’est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

La requête prend en entrée une adresse (décomposée en ligne adresse, code postal, commune) et la taille de contraction normalisée souhaitée (32 ou 38).

Le service retourne en sortie une contraction de la partie ligne adresse selon les règles exprimées dans les normes AFNOR correspondantes (norme 32 caractères et norme 38 caractères).

Concernant la fonction de restitution de la ligne adresse à 38 caractères, seule la norme AFNOR NF Z10-011 de janvier 2013 (en 38 caractères) sera applicable.

Concernant la fonction de restitution de la ligne adresse à 32 caractères ;

- En premier lieu la norme AFNOR NF Z10-011 de janvier 2013 (en 38 caractères) sera applicable :
  - vis-à-vis des règles de réduction, mais jusqu'à l'objectif de 32 caractères,
  - vis-à-vis des abréviations de mots, y compris celles qui ne sont pas définies en annexe B de la norme NF Z10-011 d'août 1989 (en 32 caractères).
- En dernier ressort, à l'issue de l'application de la norme AFNOR NF Z10-011 de janvier 2013, la norme NF Z10-011 d'août 1989 (en 32 caractères) sera applicable vis-à-vis des abréviations de mots, mais uniquement si la longueur maximale demandée est de 32 et si la longueur résultante demeure supérieure à 32 caractères. Dans ce cas, elle se limitera à l'application des abréviations de l'annexe B de la norme NF Z10-011 d'août 1989 (en 32 caractères) pour lesquelles la norme AFNOR NF Z10-011 de janvier 2013 dans ses différentes annexes ne propose pas d'abréviation.

Ce service interroge des fichiers d'abréviations, reprenant en standard les annexes des normes AFNOR citées.

Disponibilité

Ce web service est disponible en permanence avec Geoconcept Web.

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
maxLength	maxLength définit, sous forme de nombre entier, la taille de référence à atteindre pour la contraction de la ligne adresse. Les valeurs acceptées sont exclusivement 32 ou 38.	oui	38
addressLine	Ligne d'adresse recevant, soit la VOIE sous la forme numéro, indice de répétition, type de voie et libellé de voie, soit le LIEU-DIT. La justesse de la contraction d'adresse en sortie du WS ne pourra être assurée que sous réserve d'une fourniture d'un paramètre addressLine résultant d'une adresse validée à l'aide d'un référentiel Adresses.	non	
city	Commune correspondant à la ligne adresse renseignée. Aucun traitement ne sera effectué sur ce paramètre.	oui	
postCode	Code postal correspondant à la ligne adresse renseignée. Aucun traitement ne sera effectué sur ce paramètre.	oui	

En sortie

paramètre	type	min/max	description
id	long	1/1	Incrément automatique d’identifiant de réponse.
status	string	0/1	Indique la réussite ou l’échec du processus de contraction « OK » ou « ERROR ».
statusDetails	string	0/1	Vide, en cas de réussite du processus de contraction. Avec le message d’erreur correspondant, en cas d’échec. "Invalid maxLength parameter. It needs to be either 32 or 38". "The address contraction can’t reach the expected maxLength". "The supplied addressline was above 100 characters".
maxLength	int	1/1	Reprend la valeur fournie en entrée « 32 » ou « 38 ».
address	array (normalizedAddress)	0/1	Adresse normalisée.
normedAddressLineLength	int	1/1	Longueur de la ligne adresse obtenue, après les opérations de contraction effectuées par le service.

Adresse normalisée (normalizedAddress)

paramètre	type	min/max	description
normedAddressLine	string	0/1	Résultat final de la contraction de la ligne adresse effectuée par le service.
city	string	0/1	Retourne la commune fournie en entrée, sans modification, (vide sinon).
postCode	string	0/1	Retourne le code postal fourni en entrée, sans modification, (vide sinon).

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/addressNormalizerService?wsdl

Requête

L’exemple de requête prend en compte la ligne adresse suivante : ``123 Bis, Boulevard du MARÉCHAL Jean de-Lattre-de-Tassigny Inférieur".

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
  <soapenv:Header/>
  <soapenv:Body>
     <sch:normalize>
        <maxLength>38</maxLength>
        <addressLine>123 Bis, Boulevard du MARÉCHAL Jean de-Lattre-de-Tassigny Inférieur</addressLine>
        <!--Optional:-->
        <city></city>
        <!--Optional:-->
        <postCode></postCode>
     </sch:normalize>
  </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
     <ns2:normalizeResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
        <NormalizerResponse>
           <id>2</id>
           <status>OK</status>
           <statusDetails/>
           <maxLength>38</maxLength>
           <address>
              <normedAddressLine>123 B BD MAL J LATTRE TASSIGNY INF</normedAddressLine>
              <city/>
              <postCode/>
           </address>
           <normedAddressLineLength>34</normedAddressLineLength>
        </NormalizerResponse>
     </ns2:normalizeResponse>
  </soap:Body>
</soap:Envelope>
```

REST

Requête

L'exemple de requête prend en compte la ligne adresse suivante : ``123 Bis, Boulevard du MARÉCHAL Jean de-Lattre-de-Tassigny Inférieur".

Requête JSON

```
http://<server>/<webapp>/api/lbs/normalizer.json?addressLine=123%20Bis,%20Boulevard%20du%20MAR%C3%89CHAL
%20Jean%20de-Lattre-de-Tassigny%20Inf%C3%A9rieur
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/normalizer.json?addressLine=123%20Bis,%20Boulevard%20du%20MAR%C3%89CHAL
%20Jean%20de-Lattre-de-Tassigny%20Inf%C3%A9rieur&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/normalizer.xml?addressLine=123%20Bis,%20Boulevard%20du%20MAR%C3%89CHAL
%20Jean%20de-Lattre-de-Tassigny%20Inf%C3%A9rieur
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "id": 3,
   "status": "OK",
   "statusDetails": "",
   "maxLength": 38,
   "address":    {
      "normedAddressLine": "123 B BD MAL J LATTRE TASSIGNY INF",
      "city": "",
      "postCode": ""
   },
```

```
   "normedAddressLineLength": 34
}
```

Format JSON-P

```
myCallback({
           "id": 3,
           "status": "OK",
           "statusDetails": "",
           "maxLength": 38,
           "address":    {
                  "normedAddressLine": "123 B BD MAL J LATTRE TASSIGNY INF",
                  "city": "",
                  "postCode": ""
           },
           "normedAddressLineLength": 34
});
```

Format XML

```
<normalizerResponse>
   <id>4</id>
   <status>OK</status>
   <statusDetails/>
   <maxLength>38</maxLength>
   <address>
      <normedAddressLine>123 B BD MAL J LATTRE TASSIGNY INF</normedAddressLine>
      <city/>
      <postCode/>
   </address>
   <normedAddressLineLength>34</normedAddressLineLength>
</normalizerResponse>
```

Retours possibles

Cas d’une adresse normalisée (NormalizerResponse/NormalizerResponse/status est OK)

```
<ns2:normalizeResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
 <NormalizerResponse>
        <id>2</id>
        <status>OK</status>
        <statusDetails/>
        <maxLength>38</maxLength>
        <address>
           <normedAddressLine>123 B BD MAL J LATTRE TASSIGNY INF</normedAddressLine>
           <city>Paris</city>
           <postCode/>
        </address>
        <normedAddressLineLength>34</normedAddressLineLength>
 </NormalizerResponse>
</ns2:normalizeResponse>
```

Cas d’un positionnement de la taille de réduction à une valeur autre que « 32 » ou « 38 » (NormalizerResponse/NormalizerResponse/status est ERROR)

```
<ns2:normalizeResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
 <NormalizerResponse>
        <id>5</id>
```

```
        <status>ERROR</status>
        <statusDetails>Invalid maxLength parameter. It needs to be either 32 or 38</statusDetails>
        <maxLength>0</maxLength>
        <address>
           <normedAddressLine/>
           <city/>
           <postCode/>
        </address>
        <normedAddressLineLength>0</normedAddressLineLength>
 </NormalizerResponse>
</ns2:normalizeResponse>
```

Cas d'une adresse de plus de 100 caractères (NormalizerResponse/NormalizerResponse/status est ERROR)

```
<ns2:normalizeResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
 <NormalizerResponse>
        <id>6</id>
        <status>ERROR</status>
        <statusDetails>The supplied addressline was above 100 characters</statusDetails>
        <maxLength>0</maxLength>
        <address>
           <normedAddressLine/>
           <city/>
           <postCode/>
        </address>
        <normedAddressLineLength>0</normedAddressLineLength>
 </NormalizerResponse>
</ns2:normalizeResponse>
```

Cas d'un échec de la contraction de la ligne adresse à hauteur de la taille attendue (NormalizerResponse/NormalizerResponse/status est ERROR)

```
<ns2:normalizeResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
 <NormalizerResponse>
        <id>7</id>
        <status>ERROR</status>
        <statusDetails>The address contraction can't reach the expected maxLength</statusDetails>
        <maxLength>0</maxLength>
        <address>
           <normedAddressLine>123 B BD A F ZQ S E MAL J L TASSIGNY INF</normedAddressLine>
           <city>Paris</city>
           <postCode/>
        </address>
        <normedAddressLineLength>40</normedAddressLineLength>
 </NormalizerResponse>
</ns2:normalizeResponse>
```

FAQ

1. Est-il possible de modifier, d'ajouter ou de supprimer les chaines utilisées pour les abréviations ?

   Oui, ces informations sont éditables. Dans le dossier « <TOMCAT_HOME> »\webapps \geoconcept-web\WEB-INF\lib\, ouvrir le fichier geoweb-ws-(version).jar, suivre le chemin com \geoconcept\lbs\webservices à l'intérieur se trouve le fichier Substitutions.xml. Ce fichier contient le vocabulaire utilisé pour la normalisation (primary way types, secondary way types, way type extensions, articles, indicateur de répétitions, prénoms et autres abréviations. Ainsi qu'une liste, à la fin, pour définir les mots utilisés seulement pour la norme AFNOR de 32 caractères.

Géocodage inverse

Cette page n'est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

La requête comprend un couple de coordonnées en entrée, le service retourne une ou plusieurs réponse(s) possible(s) (en fonction du nombre maximum de réponses souhaitées), en incluant l'adresse, le code postal, la ville, le pays ainsi que la distance entre le résultat et le point de départ (coordonnées indiquées en entrée).

Ce service de géocodage inverse interroge le graphe préalablement installé dans le répertoire dédié.

Disponibilité

Ce web service est disponible en permanence avec Geoconcept Web et avec un graphe.

Changement de version

Les versions précédente du web service sont conservées dans Geoconcept Web pour assurer la compatibilité avec les développements antérieurs. Il est recommandé d'utiliser la version la plus récente.

Changements avec la v3

- Ajout du paramètre "coordinates" dans la réponse

Changements avec la v2

- Le paramètre "postalCode" est renommé "postCode"
- Le paramètre "distanceToLocation" est renommé "distanceMetersToLocation"
- Ajout du paramètre "srs" dans la réponse

V2

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
locations	Coordonnées du point de départ pour le géocodage inverse en SOAP, il est nécessaire de remplir les coordonnées dans chacune des balises dédiées <x></x> et <y></y> en REST, Les points sont séparés par le caractère « ; ». Les points sont caractèrisés par un couple de coordonnées XY séparés par une « , ».	non	
graphName	indiquer le nom du graphe à utiliser pour faire fonctionner le WebService	oui	
maxDistance	distance maximale de recherche pour trouver les rues avoisinantes au point de départ du géocodage inverse	non	1000

paramètre	description	optionnel	défaut
	si le paramètre est passé, il est conseillé de mettre une valeur significativement supérieure à 1 (en mètres).		
maxCandidates	nombre maximum de résultats d'adresses dans la réponse si le paramètre est passé, la valeur minimum à indiquer est 1.	oui	1
maxDistanceBetweenCandidates	Distance maximale (en mètres) entre chaque candidat. si le paramètre est passé, il est conseillé de mettre une valeur significativement supérieure à 1 (en mètres).	non	100

En sortie

Rappel des coordonnées du départ de la recherche (reverseGeocodingV2Response)

propriété	type	min/max	description
location	string	0/1	coordonnées XY du point de départ pour le géocodage inverse
srs	string	0/1	projection (code EPSG comme epsg:4326 ou wgs84)
address / addresses		0/illimité	liste des adresses (candidats) retournées

Candidats retournés (Address)

propriété	type	min/max	description
addressLine	string	0/1	rue trouvée
postCode	string	0/1	code postal trouvé
city	string	0/1	ville en relation avec le candidat retourné
country	string	0/1	pays en lien avec le candidat retourné
distanceMetersToLocation	double	1/1	Distance (en mètres) entre le candidat retourné et le point de départ
coordinates	string	1/1	Coordonnées de l'adresse

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/reverseGeocodingService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:reverseGeocodingV3>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <srs>epsg:4326</srs>
            <!--Zero or more repetitions:-->
            <locations>
               <x>-1.565769</x>
               <y>47.226236</y>
            </locations>
            <!--Optional:-->
```

```
            <graphName></graphName>
            <!--Optional:-->
            <maxDistance>1000</maxDistance>
            <!--Optional:-->
            <maxCandidates>2</maxCandidates>
            <!--Optional:-->
            <maxDistanceBetweenCandidates>100</maxDistanceBetweenCandidates>
         </request>
      </sch:reverseGeocodingV2>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:reverseGeocodingV3Response xmlns:ns2="http://geoconcept.com/gc/schemas">
        <reverseGeocodingResultsV3>
           <status>OK</status>
           <reverseGeocodingResult>
              <location>-1.565769;47.226236</location>
              <srs>epsg:4326</srs>
              <addresses>
                 <address>
                    <addressLine>26 RUE DU MAINE</addressLine>
                    <postCode>44000</postCode>
                    <city>NANTES</city>
                    <country>France</country>
                    <distanceMetersToLocation>1.2300580212448586</distanceMetersToLocation>
                    <coordinates>-1.565757,47.226229</coordinates>
                 </address>
                 <address>
                    <addressLine>2 RUE JOYAU</addressLine>
                    <postCode>44000</postCode>
                    <city>NANTES</city>
                    <country>France</country>
                    <distanceMetersToLocation>27.310568595900534</distanceMetersToLocation>
                    <coordinates>-1.566068,47.226099</coordinates>
                 </address>
              </addresses>
           </reverseGeocodingResult>
        </reverseGeocodingResultsV3>
      </ns2:reverseGeocodingV2Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/reverseGeocoding/v3.json?
locations=-1.565769,47.226236&maxDistance=1000&maxCandidates=2&maxDistanceBetweenCandidates=100
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/reverseGeocoding/v3.json?
locations=-1.565769,47.226236&maxDistance=1000&maxCandidates=2&maxDistanceBetweenCandidates=100&fonction=myFonction
```

Requête XML

```
http://<server>/<webapp>/api/lbs/reverseGeocoding/v3.xml?
locations=-1.565769,47.226236&maxDistance=1000&maxCandidates=2&maxDistanceBetweenCandidates=100
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON


```
{
   "message": null,
   "status": "OK",
   "reverseGeocodingResults": [
      {
         "location": "-1.565769;47.226236",
         "srs": null,
         "addresses": [
            {
               "addressLine": "26 RUE DU MAINE",
               "postCode": "44000",
               "city": "NANTES",
               "country": "France",
               "distanceMetersToLocation": 1.2300581,
               "coordinates": "-1.565757,47.226229"
            },
            {
               "addressLine": "2 RUE JOYAU",
               "postCode": "44000",
               "city": "NANTES",
               "country": "France",
               "distanceMetersToLocation": 27.310568,
               "coordinates": "-1.566068,47.226099"
            }
         ]
      }
   ]
}
```


Format JSON-P


```
myCallback(
    {
       "message": null,
       "status": "OK",
       "reverseGeocodingResults": [
          {
             "location": "-1.565769;47.226236",
             "srs": null,
             "addresses": [
                {
                   "addressLine": "26 RUE DU MAINE",
                   "postCode": "44000",
                   "city": "NANTES",
                   "country": "France",
                   "distanceMetersToLocation": 1.2300581,
                   "coordinates": "-1.565757,47.226229"
                },
                {
                   "addressLine": "2 RUE JOYAU",
```

```
                    "postCode": "44000",
                    "city": "NANTES",
                    "country": "France",
                    "distanceMetersToLocation": 27.310568,
                    "coordinates": "-1.566068,47.226099"
                }
            ]
        }
    ]
}
)
```

Format XML

```
<reverseGeocodingResultsV3>
   <status>OK</status>
   <reverseGeocodingResult>
      <location>-1.565769;47.226236</location>
      <addresses>
         <address>
            <addressLine>26 RUE DU MAINE</addressLine>
            <postCode>44000</postCode>
            <city>NANTES</city>
            <country>France</country>
            <distanceMetersToLocation>1.2300580212448586</distanceMetersToLocation>
            <coordinates>-1.565757,47.226229</coordinates>
         </address>
         <address>
            <addressLine>2 RUE JOYAU</addressLine>
            <postCode>44000</postCode>
            <city>NANTES</city>
            <country>France</country>
            <distanceMetersToLocation>27.310568595900534</distanceMetersToLocation>
            <coordinates>-1.566068,47.226099</coordinates>
         </address>
      </addresses>
   </reverseGeocodingResult>
</reverseGeocodingResultsV3>
```

Retours possibles

Exemple d’un géocodage inverse correct (reverseGeocodingResult/status est OK)

```
<reverseGeocodingResultsV3>
   <status>OK</status>
   <reverseGeocodingResult>
      <location>-1.565769;47.226236</location>
      <addresses>
         <address>
            <addressLine>26 RUE DU MAINE</addressLine>
            <postCode>44000</postCode>
            <city>NANTES</city>
            <country>France</country>
            <distanceMetersToLocation>1.2300580212448586</distanceMetersToLocation>
            <coordinates>-1.565757,47.226229</coordinates>
         </address>
         <address>
            <addressLine>2 RUE JOYAU</addressLine>
            <postCode>44000</postCode>
            <city>NANTES</city>
            <country>France</country>
```

```
            <distanceMetersToLocation>27.310568595900534</distanceMetersToLocation>
            <coordinates>-1.566068,47.226099</coordinates>
         </address>
      </addresses>
   </reverseGeocodingResult>
</reverseGeocodingResultsV3>
```

Cas d'un géocodage inverse où le maxDistance ne respecte pas une valeur > à 1 (on préconise une valeur minimum de 100). (reverseGeocodingResult/status est OK)

```
<?xml version="1.0" encoding="UTF-8"?>
<serviceResult>
   <message>ServiceException: Error in reverse geocoding computation Error in smartrouting Failed to execute
 ReverseGeocode com.geoconcept.smartrouting.SmartRoutingNativeException: failed to connect { -1.565769,
 47.226236, 0.000000 } failed to connect { -1.565769, 47.226236, 0.000000 }</message>
   <status>ERROR</status>
</serviceResult>
```

Cas d'une requête où l'un des paramètres (<locations>,…) n'est pas renseigné. (reverseGeocodingResult/status est ERROR)

```
<reverseGeocodingResultsV3>
    <message>locations must be not null</message>
    <status>ERROR</status>
</reverseGeocodingResultsV3>
```

Cas d'une requête qui ne s'accroche pas au graphe (reverseGeocodingResult/status est ERROR)

```
<serviceResult>
    <message>ServiceException: Error in reverse geocoding computation Error in smartrouting Failed
 to execute ReverseGeocode com.geoconcept.smartrouting.SmartRoutingNativeException: failed to connect
 { -1.565769, 147.226236, 0.000000 } failed to connect { -1.565769, 147.226236, 0.000000 }
    </message>
    <status>ERROR</status>
</serviceResult>
```

V2

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
locations	Coordonnées du point de départ pour le géocodage inverse en SOAP, il est nécessaire de remplir les coordonnées dans chacune des balises dédiées <x></x> et <y></y> en REST, Les points sont séparés par le caractère « ; ». Les points sont caractèrisés par un couple de coordonnées XY séparés par une « , ».	non	
graphName	indiquer le nom du graphe à utiliser pour faire fonctionner le WebService	oui	
maxDistance	distance maximale de recherche pour trouver les rues avoisinantes au point de départ du géocodage inverse si le paramètre est passé, il est conseillé de mettre une valeur significativement supérieure à 1 (en mètres).	non	1000

paramètre	description	optionnel	défaut
maxCandidates	nombre maximum de résultats d'adresses dans la réponse si le paramètre est passé, la valeur minimum à indiquer est 1.	oui	1
maxDistanceBetweenCandidates	Distance maximale (en mètres) entre chaque candidat. si le paramètre est passé, il est conseillé de mettre une valeur significativement supérieure à 1 (en mètres).	non	100

En sortie

Rappel des coordonnées du départ de la recherche (reverseGeocodingV2Response)

propriété	type	min/max	description
location	string	0/1	coordonnées XY du point de départ pour le géocodage inverse
srs	string	0/1	projection (code EPSG comme epsg:4326 ou wgs84)
address / addresses		0/illimité	liste des adresses (candidats) retournées

Candidats retournés (Address)

propriété	type	min/max	description
addressLine	string	0/1	rue trouvée
postCode	string	0/1	code postal trouvé
city	string	0/1	ville en relation avec le candidat retourné
country	string	0/1	pays en lien avec le candidat retourné
distanceMetersToLocation	double	1/1	Distance (en mètres) entre le candidat retourné et le point de départ

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/reverseGeocodingService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:reverseGeocodingV2>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <srs>epsg:4326</srs>
            <!--Zero or more repetitions:-->
            <locations>
               <x>-1.565769</x>
               <y>47.226236</y>
            </locations>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <maxDistance>1000</maxDistance>
            <!--Optional:-->
            <maxCandidates>2</maxCandidates>
            <!--Optional:-->
```

```
            <maxDistanceBetweenCandidates>100</maxDistanceBetweenCandidates>
         </request>
      </sch:reverseGeocodingV2>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:reverseGeocodingV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <ReverseGeocodingResults>
            <status>OK</status>
            <reverseGeocodingResult>
               <location>-1,565769;47,226236</location>
               <srs>epsg:4326</srs>
               <addresses>
                  <address>
                     <addressLine>24 RUE DU MAINE</addressLine>
                     <postCode>44000</postCode>
                     <city>NANTES</city>
                     <country>France</country>
                     <distanceMetersToLocation>1.2509365114145279</distanceMetersToLocation>
                  </address>
                  <address>
                     <addressLine>2 RUE JOYAU</addressLine>
                     <postCode>44000</postCode>
                     <city>NANTES</city>
                     <country>France</country>
                     <distanceMetersToLocation>27.312077690953934</distanceMetersToLocation>
                  </address>
               </addresses>
            </reverseGeocodingResult>
         </ReverseGeocodingResults>
      </ns2:reverseGeocodingV2Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/reverseGeocoding/v2.json?
locations=-1.565769,47.226236&maxDistance=1000&maxCandidates=2&maxDistanceBetweenCandidates=100
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/reverseGeocoding/v2.json?
locations=-1.565769,47.226236&maxDistance=1000&maxCandidates=2&maxDistanceBetweenCandidates=100&fonction=myFonction
```

Requête XML

```
http://<server>/<webapp>/api/lbs/reverseGeocoding/v2.xml?
locations=-1.565769,47.226236&maxDistance=1000&maxCandidates=2&maxDistanceBetweenCandidates=100
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON


```
{
  "message": null,
  "status": "OK",
  "reverseGeocodingResults": [
    {
      "location": "-1.565769;47.226236",
      "addresses": [
        {
          "addressLine": "26 RUE DU MAINE",
          "postalCode": "44000",
          "city": "NANTES",
          "country": "France",
          "distanceToLocation": 1.2370980024960025
        },
        {
          "addressLine": "2 RUE JOYAU",
          "postalCode": "44000",
          "city": "NANTES",
          "country": "France",
          "distanceToLocation": 27.302664051548852
        }
      ]
    }
  ]
}
```


Format JSON-P


```
myCallback(
    {
      "message": null,
      "status": "OK",
      "reverseGeocodingResults": [
        {
          "location": "-1.565769;47.226236",
          "addresses": [
            {
              "addressLine": "26 RUE DU MAINE",
              "postalCode": "44000",
              "city": "NANTES",
              "country": "France",
              "distanceToLocation": 1.2370980024960025
            },
            {
              "addressLine": "2 RUE JOYAU",
              "postalCode": "44000",
              "city": "NANTES",
              "country": "France",
              "distanceToLocation": 27.302664051548852
            }
          ]
        }
      ]
    }
)
```


Format XML

```
<reverseGeocodingResultsV2>
   <status>OK</status>
   <reverseGeocodingResult>
      <location>-1.565769;47.226236</location>
      <srs>epsg:4326</srs>
      <addresses>
         <address>
            <addressLine>26 RUE DU MAINE</addressLine>
            <postCode>44000</postCode>
            <city>NANTES</city>
            <country>France</country>
            <distanceMetersToLocation>1.2370980024960025</distanceMetersToLocation>
         </address>
         <address>
            <addressLine>2 RUE JOYAU</addressLine>
            <postCode>44000</postCode>
            <city>NANTES</city>
            <country>France</country>
            <distanceMetersToLocation>27.302664051548852</distanceMetersToLocation>
         </address>
      </addresses>
   </reverseGeocodingResult>
</reverseGeocodingResultsV2>
```

Retours possibles

Exemple d'un géocodage inverse correct (reverseGeocodingResult/status est OK)

```
<reverseGeocodingResultsV2>
   <status>OK</status>
   <reverseGeocodingResult>
      <location>-1.565769;47.226236</location>
      <srs>epsg:4326</srs>
      <addresses>
         <address>
            <addressLine>26 RUE DU MAINE</addressLine>
            <postCode>44000</postCode>
            <city>NANTES</city>
            <country>France</country>
            <distanceMetersToLocation>1.2370980024960025</distanceMetersToLocation>
         </address>
         <address>
            <addressLine>2 RUE JOYAU</addressLine>
            <postCode>44000</postCode>
            <city>NANTES</city>
            <country>France</country>
            <distanceMetersToLocation>27.302664051548852</distanceMetersToLocation>
         </address>
      </addresses>
   </reverseGeocodingResult>
</reverseGeocodingResultsV2>
```

Cas d'un géocodage inverse où le maxDistance ne respecte pas une valeur > à 1 (on préconise une valeur minimum de 100). (reverseGeocodingResult/status est OK)

```
<?xml version="1.0" encoding="UTF-8"?>
<serviceResult>
   <message>ServiceException: Error in reverse geocoding computation Error in smartrouting Failed to execute
 ReverseGeocode com.geoconcept.smartrouting.SmartRoutingNativeException: failed to connect { -1.565769,
 47.226236, 0.000000 } failed to connect { -1.565769, 47.226236, 0.000000 }</message>
```

```
   <status>ERROR</status>
</serviceResult>
```

Cas d'une requête où l'un des paramètres (<locations>,…) n'est pas renseigné. (reverseGeocodingResult/status est ERROR)

```
<reverseGeocodingResultsV2>
    <message>locations must be not null</message>
    <status>ERROR</status>
</reverseGeocodingResultsV2>
```

Cas d'une requête qui ne s'accroche pas au graphe à partir de la version 2.0.9 de SmartRouting Server (reverseGeocodingResult/status est ERROR)

```
<serviceResult>
    <message>ServiceException: Error in reverse geocoding computation Error in smartrouting Failed
 to execute ReverseGeocode com.geoconcept.smartrouting.SmartRoutingNativeException: failed to connect
 { -1.565769, 147.226236, 0.000000 } failed to connect { -1.565769, 147.226236, 0.000000 }
    </message>
    <status>ERROR</status>
</serviceResult>
```

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
locations	Coordonnées du point de départ pour le géocodage inverse en SOAP, il est nécessaire de remplir les coordonnées dans chacune des balises dédiées <x></x> et <y></y> en REST, Les points sont séparés par le caractère « ; ». Les points sont caractèrisés par un couple de coordonnées XY séparés par une « , ».	non	
graphName	indiquer le nom du graphe à utiliser pour faire fonctionner le WebService	oui	
maxDistance	distance maximale de recherche pour trouver les rues avoisinantes au point de départ du géocodage inverse si le paramètre est passé, il est conseillé de mettre une valeur significativement supérieure à 1 (en mètres).	non	
maxCandidates	nombre maximum de résultats d'adresses dans la réponse si le paramètre est passé, la valeur minimum à indiquer est 1.	oui	
maxDistanceBetweenCandidates	Distance maximale (en mètres) entre chaque candidat. si le paramètre est passé, il est conseillé de mettre une valeur significativement supérieure à 1 (en mètres).	non	

En sortie

Rappel des coordonnées du départ de la recherche (location)

propriété	type	min/max	description
location	string	0/1	coordonnées XY du point de départ pour le géocodage inverse

propriété	type	min/max	description
address / addresses		0/illimité	liste des adresses (candidats) retournées

Candidats retournés (Address)

propriété	type	min/max	description
addressLine	string	0/1	rue trouvée
postalCode	string	0/1	code postal trouvé
city	string	0/1	ville en relation avec le candidat retourné
country	string	0/1	pays en lien avec le candidat retourné
distanceToLocation	integer	1/1	Distance (en mètres) entre le candidat retourné et le point de départ

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/reverseGeocodingService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:reverseGeocoding>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <srs></srs>
            <!--Zero or more repetitions:-->
            <locations>
               <x>2.324382</x>
               <y>48.803305</y>
            </locations>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <maxDistance>1000</maxDistance>
            <!--Optional:-->
            <maxCandidates>10</maxCandidates>
            <!--Optional:-->
            <maxDistanceBetweenCandidates>100</maxDistanceBetweenCandidates>
         </request>
      </sch:reverseGeocoding>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:reverseGeocodingResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <ReverseGeocodingResults>
            <status>OK</status>
            <reverseGeocodingResult>
               <location>2.324382;48.803305</location>
```

```
               <addresses>
                  <address>
                     <addressLine>AVENUE ARISTIDE BRIAND</addressLine>
                     <postalCode>92220</postalCode>
                     <city>BAGNEUX</city>
                     <country>FRANCE</country>
                     <distanceToLocation>18.661267300047538</distanceToLocation>
                  </address>
                  <address>
                     <addressLine>RUE DU MIDI</addressLine>
                     <postalCode>94110</postalCode>
                     <city>ARCUEIL</city>
                     <country>FRANCE</country>
                     <distanceToLocation>29.691380904228758</distanceToLocation>
                  </address>
               </addresses>
            </reverseGeocodingResult>
         </ReverseGeocodingResults>
      </ns2:reverseGeocodingResponse>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/reverseGeocoding.json?
locations=2.324382,48.803305&maxDistance=1000&maxCandidates=2&maxDistanceBetweenCandidates=100
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/reverseGeocoding.json?
locations=2.324382,48.803305&maxDistance=1000&maxCandidates=2&maxDistanceBetweenCandidates=100&fonction=myFonction
```

Requête XML

```
http://<server>/<webapp>/api/lbs/reverseGeocoding.xml?
locations=2.324382,48.803305&maxDistance=1000&maxCandidates=2&maxDistanceBetweenCandidates=100
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message": null,
   "status": "OK",
   "reverseGeocodingResults": [   {
      "location": "2.324382;48.803305",
      "addresses":       [
                  {
            "addressLine": "AVENUE ARISTIDE BRIAND",
            "postalCode": "92220",
            "city": "BAGNEUX",
            "country": "FRANCE",
```

```
            "distanceToLocation": 18.661267300047538
         },
                  {
            "addressLine": "RUE DU MIDI",
            "postalCode": "94110",
            "city": "ARCUEIL",
            "country": "FRANCE",
            "distanceToLocation": 29.691380904228758
         }
      ]
   }]
}
```


Format JSON-P


```
myCallback({
        "message":null,"status":"OK",
        "reverseGeocodingResults":[
                {
                        "location":"2.324382;48.803305",
                                "addresses":[
                                        {
                                                "addressLine":"AVENUE ARISTIDE BRIAND",
                                                "postalCode":"92220",
                                                "city":"BAGNEUX",
                                                "country":"FRANCE",
                                                "distanceToLocation":18.661267300047538
                                        },
                                        {
                                                "addressLine":"RUE DU MIDI",
                                                "postalCode":"94110",
                                                "city":"ARCUEIL",
                                                "country":"FRANCE",
                                                "distanceToLocation":29.691380904228758
                                        }
]}]}
```


Format XML


```
<reverseGeocodingResults>
   <status>OK</status>
   <reverseGeocodingResult>
      <location>2.324382;48.803305</location>
      <addresses>
         <address>
            <addressLine>AVENUE ARISTIDE BRIAND</addressLine>
            <postalCode>92220</postalCode>
            <city>BAGNEUX</city>
            <country>FRANCE</country>
            <distanceToLocation>18.661267300047538</distanceToLocation>
         </address>
         <address>
            <addressLine>RUE DU MIDI</addressLine>
            <postalCode>94110</postalCode>
            <city>ARCUEIL</city>
            <country>FRANCE</country>
            <distanceToLocation>29.691380904228758</distanceToLocation>
         </address>
      </addresses>
   </reverseGeocodingResult>
</reverseGeocodingResults>
```

Retours possibles

Exemple d'un géocodage inverse correct (reverseGeocodingResult/status est OK)

```
<reverseGeocodingResults>
   <status>OK</status>
   <reverseGeocodingResult>
      <location>2.324382;48.803305</location>
      <addresses>
         <address>
            <addressLine>AVENUE ARISTIDE BRIAND</addressLine>
            <postalCode>92220</postalCode>
            <city>BAGNEUX</city>
            <country>FRANCE</country>
            <distanceToLocation>18.661267300047538</distanceToLocation>
         </address>
         <address>
            <addressLine>RUE DU MIDI</addressLine>
            <postalCode>94110</postalCode>
            <city>ARCUEIL</city>
            <country>FRANCE</country>
            <distanceToLocation>29.691380904228758</distanceToLocation>
         </address>
      </addresses>
   </reverseGeocodingResult>
</reverseGeocodingResults>
```

Cas d'un géocodage inverse où le maxDistanceBetweenCandidates ne respecte pas une valeur > à 1 (on préconise une valeur minimum de 100). (reverseGeocodingResult/status est OK)

```
<reverseGeocodingResults>
   <status>OK</status>
   <reverseGeocodingResult>
      <location>2.324382;48.803305</location>
      <addresses>
         <address>
            <addressLine>AVENUE ARISTIDE BRIAND</addressLine>
            <postalCode>92220</postalCode>
            <city>BAGNEUX</city>
            <country>FRANCE</country>
            <distanceToLocation>18.661267300047538</distanceToLocation>
         </address>
      </addresses>
   </reverseGeocodingResult>
</reverseGeocodingResults>
```

Cas d'un géocodage inverse où le maxDistance ne respecte pas une valeur > à 1 (on préconise une valeur minimum de 100). (reverseGeocodingResult/status est OK)

```
<reverseGeocodingResults>
   <status>OK</status>
   <reverseGeocodingResult>
      <location>2.324382;48.803305</location>
      <addresses/>
   </reverseGeocodingResult>
</reverseGeocodingResults>
```

Cas d'une requête où le paramètre <locations> n'est pas indiqué. (reverseGeocodingResult/status est ERROR)

```
<reverseGeocodingResults>
   <message>locations must be not null</message>
   <status>ERROR</status>
</reverseGeocodingResults>
```

Cas d'une requête qui ne s'accroche pas au graphe à partir de la version 2.0.9 de SmartRouting Server (serviceResult/status est ERROR)

```
<serviceResult>
    <message>ServiceException: Error in reverse geocoding computation
    Error in smartrouting
    Failed to execute ReverseGeocode
com.geoconcept.smartrouting.SmartRoutingNativeException: failed to connect { 2.165931, 49.218690, 0.000000 }
failed to connect { 2.165931, 49.218690, 0.000000 }
    </message>
    <status>ERROR</status>
</serviceResult>
```

FAQ

1. Comment est interprété la distance maximale ?

   La balise **<maxDistance>** est obligatoire et nécessite d'avoir une valeur supérieure à 1. Attention tout de même, si la distance maximale n'est pas suffisamment importante, le risque de n'avoir aucun candidat retourné sera grand.

2. Quel est le rôle du paramètre maxCandidates ?

   Il permet de définir le nombre de réponses que l'on souhaite obtenir, lorsque l'on est dans un cas où le nombre de candidats potentiels est important. Si aucune valeur n'est indiquée, c'est la valeur iti.maxTargets renseignée dans l'Administration de Geoconcept Web qui prend le relai.

3. Quelle est l'importance du paramètre maxDistanceBetweenCandidates ?

   Si ce paramètre est mal renseigné, le webservice peut ne pas renvoyer de réponse positive. Il est vivement conseillé de mettre une valeur supérieur à 1. Si aucune valeur n'est mentionnée, c'est la valeur iti.candidatesDeltaDistance renseignée dans l'Administration de Geoconcept Web qui prédomine. Attention, si la distance entre les candidats n'est pas assez grande, le risque de n'avoir aucun candidat retourné est grand.

4. Peut-on utiliser des alias à défaut des noms de fichiers .siti pour appeler une datasource ?

   Oui. Un paramétrage spécifique est à réaliser pour permettre l'exploitation d'alias permettant d'utiliser plusieurs sources de graphes. Les étapes à respecter sont les suivantes :

   - Stopper le service Tomcat,
   - Déposer les graphes (.siti et en option les fichiers .siti.MG et .siti.GA) dans le répertoire approprié. Exemple : Ici, deux graphes sont déposées dans le répertoire suivant : [Home]\Geoconcept\SmartRouting\server\SRJEE\smartrouting\graphs, dans le cadre d'un paramétrage par défaut.
   - Editer le fichier « Service.xml » associé au module SmartRouting JEE. Ce dernier se trouve ici : [Home]\Geoconcept\SmartRouting\server\SRJEE\smartrouting\conf.
   - Dans l'éditeur de texte de votre choix, copier/coller autant de fois que vous avez de graphes, le bloc de texte encadré par les balises **<datasource>** présenté ci-dessous (dans notre exemple, il y a 2 sources différentes, à copier/coller 2 fois par conséquent),

- Copier ce bloc <datasource> sous la balise <default-datasource> du bloc initial,
- La balise <file /> doit indiquer le nom de l'une des deux graphes. La balise <name /> permet de donner un alias à cette source, exemple ici : « HEREQ414 ».

Edition du premier bloc associé au premier graphe

```
        <datasource>
        <file>navteq_maps_for_geoconcept_Q414_france_v1.siti</file>
        <name>HEREQ414</name>

        <load-graph-settings>
            <full-load>false</full-load>
            <primary-language>en</primary-language>
        </load-graph-settings>

        <calculate-route-options>
            <vehicle-type/>
            <!-- a vehicle type name present in the vehicle types catalog -->
            <vehicle-type-id/>
            <!-- a vehicle type identifier present in the vehicle types catalog -->
            <route-search-criteria>time</route-search-criteria>
            <!-- distance, time, or an integer matching cost table index (graph dependant) -->
            <max-graph-snap-distance>500</max-graph-snap-distance>
            <!-- in meters -->
            <graph-snap-speed>1.11111111</graph-snap-speed>
            <!-- in m/s -->
            <route-descr-items/>
            <!-- composition (separated by comma) -->
            <reference-level>4</reference-level>
            <reject-flags/>
            <!-- composition (separated by comma) -->
            <speed-profile/>
            <input-coordinate-system/>
            <output-coordinate-system/>
            <use-meta-graph>true</use-meta-graph>
            <use-graph-accelerator>true</use-graph-accelerator>
            <fields/>
            <!-- composition (separated by comma) -->
            <tour-optimization>auto</tour-optimization>
            <language/>
            <!-- default output language -->
        </calculate-route-options>

        <calculate-route-settings>
            <calculate-ecotax>true</calculate-ecotax>
            <!-- calculate ecotax when available  -->
        </calculate-route-settings>

        <!-- if a section named 'search-around-options' (same definition as calculate-route-options) is
 present it defines particular default options for search around -->
        <!-- if a section named 'calculate-route-matrix-options' (same definition as calculate-route-
options) is present it defines particular default options for calculate route matrix -->
        <!-- if a section named 'calculate-tour-options' (same definition as calculate-route-options) is
 present it defines particular default options for calculate tour -->

        <!-- if a section named 'calculate-route-matrix-settings' (same definition as calculate-route-
matrix-settings) is present it defines particular settings for calculate route matrix -->
        <!-- if a section named 'search-around-settings' (same definition as calculate-route-settings) is
 present it defines particular settings for search around -->

        <reverse-geocode-options>
```

```
            <max-distance>1000</max-distance>
            <!-- in meters -->
            <max-distance-between-candidates>2</max-distance-between-candidates>
            <!-- in meters -->
            <max-candidates>10</max-candidates>

            <input-coordinate-system/>
            <output-coordinate-system/>

            <include-street-number>true</include-street-number>
            <accept-emtpy-segments>true</accept-emtpy-segments>

            <language/>
            <places/>

        </reverse-geocode-options>

        <!-- reverse geocoding configuration usually depends on each particular graph -->
        <reverse-geocode-settings>
            <address-line-field>${built-in-field:name}</address-line-field>
            <city-name-field>City name</city-name-field>

            <postal-code-field>Postcode</postal-code-field>

            <country-field>Country</country-field>

            <right-suffix> right</right-suffix>
            <left-suffix> left</left-suffix>

            <number-begin-left-field>Begin left number</number-begin-left-field>
            <number-end-left-field>End left number</number-end-left-field>
            <number-begin-right-field>Begin right number</number-begin-right-field>
            <number-end-right-field>End right number</number-end-right-field>

        </reverse-geocode-settings>

        <calculate-tour-settings>
            <tour-optimization-auto-threshold>10</tour-optimization-auto-threshold>
        </calculate-tour-settings>

        <calculate-concentric-reachable-areas-options>
            <vehicle-type/>
            <!-- a vehicle type name present in the vehicle types catalog -->
            <vehicle-type-id/>
            <!-- a vehicle type identifier present in the vehicle types catalog -->
            <route-search-criteria>time</route-search-criteria>
            <!-- distance, time, or an integer matching cost table index (graph dependant) -->
            <max-graph-snap-distance>500</max-graph-snap-distance>
            <!-- in meters -->
            <graph-snap-speed>1.11111111</graph-snap-speed>
            <!-- in m/s -->
            <route-descr-items/>
            <!-- composition (separated by comma) -->
            <reference-level>4</reference-level>
            <reject-flags/>
            <!-- composition (separated by comma) -->
            <speed-profile/>
            <input-coordinate-system/>
            <output-coordinate-system/>
            <use-meta-graph>true</use-meta-graph>
            <use-graph-accelerator>true</use-graph-accelerator>
```

```
            <default-reachable-area-specification>
                <row-col>2000</row-col>
                <tile-resolution>50</tile-resolution>
                <max-holes>0</max-holes>
                <min-hole-size>0</min-hole-size>
                <extended-smoothing>false</extended-smoothing>
                <accept-non-connectable-links>true</accept-non-connectable-links>
            </default-reachable-area-specification>

        </calculate-concentric-reachable-areas-options>

        <calculate-concentric-reachable-areas-settings>
        </calculate-concentric-reachable-areas-settings>
    </datasource>
```

- laisser les autres paramètres par défaut, sauf si vous décidez de faire un paramétrage spécifique à votre installation.
- Répéter les étapes de la copie du bloc <datasource> et l'édition des balises <file/> et <name/> pour paramétrer la seconde source de données.

Edition du second bloc associé à l'autre table de référence qui sera exploitée

```
        <datasource>
        <file>BD_ADRESSE_France_v4.siti</file>
        <name>IGN</name>

        <load-graph-settings>
            <full-load>false</full-load>
            <primary-language>en</primary-language>
        </load-graph-settings>

        <calculate-route-options>
            <vehicle-type/>
            <!-- a vehicle type name present in the vehicle types catalog -->
            <vehicle-type-id/>
            <!-- a vehicle type identifier present in the vehicle types catalog -->
            <route-search-criteria>time</route-search-criteria>
            <!-- distance, time, or an integer matching cost table index (graph dependant) -->
            <max-graph-snap-distance>500</max-graph-snap-distance>
            <!-- in meters -->
            <graph-snap-speed>1.11111111</graph-snap-speed>
            <!-- in m/s -->
            <route-descr-items/>
            <!-- composition (separated by comma) -->
            <reference-level>4</reference-level>
            <reject-flags/>
            <!-- composition (separated by comma) -->
            <speed-profile/>
            <input-coordinate-system/>
            <output-coordinate-system/>
            <use-meta-graph>true</use-meta-graph>
            <use-graph-accelerator>true</use-graph-accelerator>
            <fields/>
            <!-- composition (separated by comma) -->
            <tour-optimization>auto</tour-optimization>
            <language/>
            <!-- default output language -->
        </calculate-route-options>

        <calculate-route-settings>
            <calculate-ecotax>true</calculate-ecotax>
```

```
            <!-- calculate ecotax when available  -->
        </calculate-route-settings>

        <!-- if a section named 'search-around-options' (same definition as calculate-route-options) is
 present it defines particular default options for search around -->
        <!-- if a section named 'calculate-route-matrix-options' (same definition as calculate-route-
options) is present it defines particular default options for calculate route matrix -->
        <!-- if a section named 'calculate-tour-options' (same definition as calculate-route-options) is
 present it defines particular default options for calculate tour -->

        <!-- if a section named 'calculate-route-matrix-settings' (same definition as calculate-route-
matrix-settings) is present it defines particular settings for calculate route matrix -->
        <!-- if a section named 'search-around-settings' (same definition as calculate-route-settings) is
 present it defines particular settings for search around -->

        <reverse-geocode-options>

            <max-distance>1000</max-distance>
            <!-- in meters -->
            <max-distance-between-candidates>2</max-distance-between-candidates>
            <!-- in meters -->
            <max-candidates>10</max-candidates>

            <input-coordinate-system/>
            <output-coordinate-system/>

            <include-street-number>true</include-street-number>
            <accept-emtpy-segments>true</accept-emtpy-segments>

            <language/>
            <places/>

        </reverse-geocode-options>

        <!-- reverse geocoding configuration usually depends on each particular graph -->
        <reverse-geocode-settings>
            <address-line-field>${built-in-field:name}</address-line-field>
            <city-name-field>City name</city-name-field>

            <postal-code-field>Postcode</postal-code-field>

            <country-field>Country</country-field>

            <right-suffix> right</right-suffix>
            <left-suffix> left</left-suffix>

            <number-begin-left-field>Begin left number</number-begin-left-field>
            <number-end-left-field>End left number</number-end-left-field>
            <number-begin-right-field>Begin right number</number-begin-right-field>
            <number-end-right-field>End right number</number-end-right-field>

        </reverse-geocode-settings>

        <calculate-tour-settings>
            <tour-optimization-auto-threshold>10</tour-optimization-auto-threshold>
        </calculate-tour-settings>

        <calculate-concentric-reachable-areas-options>
            <vehicle-type/>
            <!-- a vehicle type name present in the vehicle types catalog -->
            <vehicle-type-id/>
            <!-- a vehicle type identifier present in the vehicle types catalog -->
            <route-search-criteria>time</route-search-criteria>
```

```
            <!-- distance, time, or an integer matching cost table index (graph dependant) -->
            <max-graph-snap-distance>500</max-graph-snap-distance>
            <!-- in meters -->
            <graph-snap-speed>1.11111111</graph-snap-speed>
            <!-- in m/s -->
            <route-descr-items/>
            <!-- composition (separated by comma) -->
            <reference-level>4</reference-level>
            <reject-flags/>
            <!-- composition (separated by comma) -->
            <speed-profile/>
            <input-coordinate-system/>
            <output-coordinate-system/>
            <use-meta-graph>true</use-meta-graph>
            <use-graph-accelerator>true</use-graph-accelerator>

            <default-reachable-area-specification>
                <row-col>2000</row-col>
                <tile-resolution>50</tile-resolution>
                <max-holes>0</max-holes>
                <min-hole-size>0</min-hole-size>
                <extended-smoothing>false</extended-smoothing>
                <accept-non-connectable-links>true</accept-non-connectable-links>
            </default-reachable-area-specification>

        </calculate-concentric-reachable-areas-options>

        <calculate-concentric-reachable-areas-settings>
        </calculate-concentric-reachable-areas-settings>
    </datasource>
```

- Sauvegarder le fichier « Service.xml » et relancer le service Tomcat,
- Désormais, lorsque l’on fait appel au WebService, l’utilisateur devra appeler une des deux ressources dans sa requête en indiquant dans le paramètre **<GraphName>** l’alias qui a été défini auparavant via la balise **<name>**. si aucun alias n’est passé, la source définie dans le paramètre **`iti.graphname`** sera utilisée par défaut.

Recherche d’objet

Cette page n’est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

Ce web service permet, en passant une coordonnée, de récupérer les informations contenues dans les champs des objets d’une carte Geoconcept. Les paramètres du nom de la couche, de la structure de la carte et du ou des champ(s) à interroger sont à renseigner lors de l’appel au web service.

Disponibilité

Ce web service est disponible en permanence avec Geoconcept Web et une carte Geoconcept.

Changement de version

Les versions précédente du web service sont conservées dans Geoconcept Web pour assurer la compatibilité avec les développements antérieurs. Il est recommandé d'utiliser la version la plus récente.

Changements avec la V2

- Le nom du web service a changé de "findObjectsUnder" à "findObjectV2"
- Le Web service n'est plus disponible en REST GET
- Remplacement de "mapName" par "layerName"
- Ajout de plusieurs paramètres de recherche "findObjectsTarget"
- Ajout des paramètres "targetID", "maxDistance" et "maxCandidates"
- Ajout des paramètres en sortie "distance" et "wktGeometry"
- Suppression du paramètre "topology"

V2

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
srs	projection (code EPSG comme epsg:4326 ou wgs84)	non	
geometries	Coordonnées à chercher (série de triplets Id,X,Y séparées par le caractère ";")	non	
layerName	Nom de la couche à utiliser	non	
targets/target (findObjectsTarget)	Propriétés des objets cherchés	non	

Cibles (findObjectsTarget)

paramètre	description	optionnel	défaut
targetID	Identifiant	oui	
className	Nom du Type	non	
subclassName	Nom du Sous-type	non	
fields	Nom des champs séparés par le caractère ";"	non	
maxDistance	Rayon de recherche maximal (en mètre)	oui	
maxCandidates	Nombre maximum de candidats à retourner	oui	

En sortie

paramètre	type	min/max	description
id	string	0/1	Identifiant des objets

paramètre	type	min/max	description
targets/target (findTargetResult)	array (objectInfo)	0/illimité	Objets

Cibles Infos (findTargetResult)

paramètre	type	min/max	description
targetID	string	0/1	Identifiant
objects/object (objectInfoV2)	string	0/illimité	Description des objets

Objets Infos (objectInfoV2)

paramètre	type	min/max	description
distance	integer	1/1	Distance en mètre entre le point d’orgine et l’objet trouvé -1 si la distance calculée n’est pas diponible
wktGeometry	string	0/1	Géométrie de l’objet

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/findObjectsService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:findObjectV2>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <srs>epsg:4326</srs>
            <geometries>1,-1.80280,47.15524</geometries>
            <layerName>Fond de plan</layerName>
            <!--Optional:-->
            <targets>
               <!--1 or more repetitions:-->
               <target>
                  <!--Optional:-->
                  <targetId>1</targetId>
                  <className>Unité administrative</className>
                  <!--Optional:-->
                  <subClassName>Commune</subClassName>
                  <fields>Name;Code gouvernement</fields>
                  <!--Optional:-->
                  <maxDistance>1000</maxDistance>
                  <!--Optional:-->
                  <maxCandidates>5</maxCandidates>
               </target>
            </targets>
         </request>
      </sch:findObjectV2>
   </soapenv:Body>
```

```
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:findObjectV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <FindObjectResult>
            <status>OK</status>
            <results>
               <id>1</id>
               <targets>
                  <target>
                     <targetId>1</targetId>
                     <objects>
                        <object>
                           <fields>
                              <field>Port-Saint-Père</field>
                              <field>44133</field>
                           </fields>
                           <distance>0</distance>
                        </object>
                        <object>
                           <fields>
                              <field>Rouans</field>
                              <field>44145</field>
                           </fields>
                           <distance>447</distance>
                        </object>
                     </objects>
                  </target>
               </targets>
            </results>
         </FindObjectResult>
      </ns2:findObjectV2Response>
   </soap:Body>
</soap:Envelope>
```

REST (POST)

Requête

Requête

```
http://<server>/<webapp>/api/lbs/find/findObject/v2.xml
```

Data (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<findObjectRequestV2>
  <srs>epsg:4326</srs>
  <geometries>1,-1.80280,47.15524</geometries>
  <layerName>Fond de plan</layerName>
  <targets>
    <target>
      <targetId>1</targetId>
      <className>Unité administrative</className>
      <subClassName>Commune</subClassName>
      <fields>Name;Code gouvernement</fields>
```

```
      <maxDistance>0</maxDistance>
      <maxCandidates>1</maxCandidates>
    </target>
  </targets>
</findObjectRequestV2>
```

Réponse

La réponse est toujours encodée en UTF-8.

Format XML

```
<findObjectResultsV2>
   <status>OK</status>
   <results>
      <id>1</id>
      <targets>
         <target>
            <targetId>1</targetId>
            <objects>
               <object>
                  <fields>
                     <field>Port-Saint-Père</field>
                     <field>44133</field>
                  </fields>
                  <distance>0</distance>
               </object>
            </objects>
         </target>
      </targets>
   </results>
</findObjectResultsV2>
```

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/find/findObject/v2.json
```

JSON

```
{
  "srs" : "epsg:4326",
  "geometries" : "1,-1.80280,47.15524",
  "layerName" : "Fond de plan",
  "targets" : [ {
  "className" : "Unité administrative",
  "subClassName" : "Commune",
  "fields" : "Name;Code gouvernement"
  }]
}
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message": null,
   "status": "OK",
   "results": [   {
      "id": "1",
      "targets": [{"objects": [      {
         "distance": 0,
         "fields":          [
            "Port-Saint-Père",
            "44133"
         ]
      }]}]
   }]
}
```

Retours possibles

Cas d'une réponse trouvée (findObjectsResults/status est OK)

```
<findObjectResultsV2>
   <status>OK</status>
   <results>
      <id>1</id>
      <targets>
         <target>
            <targetId>1</targetId>
            <objects>
               <object>
                  <fields>
                     <field>Port-Saint-Père</field>
                     <field>44133</field>
                  </fields>
                  <distance>0</distance>
               </object>
            </objects>
         </target>
      </targets>
   </results>
</findObjectResultsV2>
```

Absence d'objet sur la position recherché (findObjectsResultsV2/status est OK)

```
<findObjectsResultsV2>
   <status>OK</status>
   <results>
      <id>1</id>
      <objects/>
   </results>
</findObjectsResultsV2>
```

Mauvaise projection / srs (serviceResult/status est ERROR)

```
<serviceResult>
    <message>
        CoordinateTransformEngineException: Coordinate transformation failed
    </message>
    <status>ERROR</status>
</serviceResult>
```

Cas, absence d'un argument pour ***geometries*** (serviceResult/status est ERROR)

```
<serviceResult>
   <message>IllegalArgumentException: Geometry has too few fields : String[][{1,-1.80280}]</message>
   <status>ERROR</status>
</serviceResult>
```

Cas de la couche non trouvée (serviceResult/status est ERROR)

```
<serviceResult>
   <message>WebServiceException: Layer name 'Fond de lan' does not exist</message>
   <status>ERROR</status>
</serviceResult>
```

Cas du Type non trouvé (serviceResult/status est ERROR)

```
<serviceResult>
   <message>WebServiceException: Error in finding objects
failed to execute text request
failed to execute gcis request (text response)
failed to execute gcis request
exception occured while servicing request
Failed to service request
com.geoconcept.gc.GcException: native returned exception (code=1)
native returned exception
[FindObject-88304] unknown type 'Unité administrtive'</message>
   <status>ERROR</status>
</serviceResult>
```

Cas du Sous-type non trouvé (serviceResult/status est ERROR)

```
<serviceResult>
   <message>WebServiceException: Error in finding objects
failed to execute text request
failed to execute gcis request (text response)
failed to execute gcis request
exception occured while servicing request
Failed to service request
com.geoconcept.gc.GcException: native returned exception (code=1)
native returned exception
[FindObject-88305] unknown subtype 'Commun'</message>
   <status>ERROR</status>
</serviceResult>
```

Cas de champ non trouvé (serviceResult/status est ERROR)

```
<serviceResult>
   <message>WebServiceException: Error in finding objects
failed to execute text request
failed to execute gcis request (text response)
failed to execute gcis request
exception occured while servicing request
Failed to service request
com.geoconcept.gc.GcException: native returned exception (code=1)
native returned exception
[FindObject-88300] unknown field 'Code governement'</message>
   <status>ERROR</status>
</serviceResult>
```

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
geometries	Coordonnées à chercher (série de triplets Id,X,Y séparées par le caractère ";")	non	
topology	Relation topologique entre le **`geometries`** et les objets à chercher	oui	
mapName	Nom de la carte	non	
className	Nom du Type	non	
subclassName	Nom du Sous-type	non	
fields	Nom des champs séparés par le caractère ";"	non	

En sortie

Objets (findObjectsResult)

paramètre	type	min/max	description
id	string	0/1	Identifiant des objets
objects	array (objectInfo)	0/illimité	Objets

Objets Infos (objectInfo)

paramètre	type	min/max	description
fields	string	0/illimité	Attributs des champs

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/findObjectsService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:findObjectsUnder>
         <!--Optional:-->
         <srs>epsg:4326</srs>
         <!--Optional:-->
         <geometries>1,-1.80280,47.15524</geometries>
         <!--Optional:-->
         <topology></topology>
         <!--Optional:-->
         <mapName>Loire.gcm</mapName>
         <!--Optional:-->
         <className>Unité administrative</className>
         <!--Optional:-->
```

```
         <subclassName>Commune</subclassName>
         <!--Optional:-->
         <fields>Nom;Population</fields>
      </sch:findObjectsUnder>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:findObjectsUnderResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <FindObjectsResult>
            <status>OK</status>
            <results>
               <id>1</id>
               <objects>
                  <object>
                     <fields>
                        <field>Port-Saint-Père</field>
                        <field>2724</field>
                     </fields>
                  </object>
               </objects>
            </results>
         </FindObjectsResult>
      </ns2:findObjectsUnderResponse>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/find/under.json?
srs=epsg:4326&geometries=1,-1.80280,47.15524&mapName=Loire.gcm&className=Unit
%C3%A9%20administrative&subclassName=Commune&fields=Nom;Population
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/find/under.xml?
srs=epsg:4326&geometries=1,-1.80280,47.15524&mapName=Loire.gcm&className=Unit
%C3%A9%20administrative&subclassName=Commune&fields=Nom;Population&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/find/under.xml?
srs=epsg:4326&geometries=1,-1.80280,47.15524&mapName=Loire.gcm&className=Unit
%C3%A9%20administrative&subclassName=Commune&fields=Nom;Population
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message":null,
   "status":"OK",
   "results":[
      {
         "id":"1",
         "objects":[
            {
               "fields":[
                  "Port-Saint-Père",
                  "2724"
               ]
            }
         ]
      }
   ]
}
```

Format JSON-P

```
myCallback({
           "message":null,
           "status":"OK",
           "results":[
                  {
                         "id":"1",
                         "objects":[
                                {
                                   "fields":[
                                          "Port-Saint-Père",
                                          "2724"
                                   ]
                                }
                         ]
                  }
           ]
        }
);
```

Format XML

```
<findObjectsResults>
        <status>OK</status>
        <results>
                <id>1</id>
                <objects>
                        <object>
                                <fields>
                                        <field>Port-Saint-Père</field>
                                        <field>2724</field>
                                </fields>
                        </object>
                </objects>
        </results>
</findObjectsResults>
```

Retours possibles

Cas d’une réponse trouvée (findObjectsResults/status est OK)

```
<findObjectsResults>
        <status>OK</status>
        <results>
                <id>1</id>
                <objects>
                        <object>
                                <fields>
                                        <field>Port-Saint-Père</field>
                                        <field>2724</field>
                                </fields>
                        </object>
                </objects>
        </results>
</findObjectsResults>
```

Absence d'objet sur la position recherché (findObjectsResults/status est OK)

```
<findObjectsResults>
   <status>OK</status>
   <results>
      <id>1</id>
      <objects/>
   </results>
</findObjectsResults>
```

Mauvaise projection / srs (serviceResult/status est ERROR)

```
<serviceResult>
    <message>
        CoordinateTransformEngineException: Coordinate transformation failed
    </message>
    <status>ERROR</status>
</serviceResult>
```

Cas, absence d'un argument pour ***geometries*** (serviceResult/status est ERROR)

```
<serviceResult>
   <message>IllegalArgumentException: Candidate has too few fields : [Ljava.lang.String;@5890d398</message>
   <status>ERROR</status>
</serviceResult>
```

Cas de la carte non trouvée (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: getLayerInfo : Error reading map info on Loir.gcm
No metadata for this map has been found</message>
   <status>ERROR</status>
</serviceResult>
```

Cas du Type non trouvé (serviceResult/status est ERROR)

```
<serviceResult>
   <message>WebServiceException: Error in finding objects
Error on ObjClass</message>
   <status>ERROR</status>
</serviceResult>
```

Cas du Sous-type non trouvé (serviceResult/status est ERROR)

```
<serviceResult>
   <message>WebServiceException: Error in finding objects
Error on ObjSubclass</message>
   <status>ERROR</status>
</serviceResult>
```

Cas de champ non trouvé (serviceResult/status est ERROR)

```
<serviceResult>
   <message>WebServiceException: Error in finding objects
Error on ObjField</message>
   <status>ERROR</status>
</serviceResult>
```

FAQ

1. Est-il possible d'utiliser d'autres relations topologiques entre les objets ?

   Non, pour l'instant seule la relation *Intersect* est disponible.

2. Comment traiter les cartes avec des encodages non latin ?

   Dans Administration / Paramètres changer la valeur de "geographics.server.gcis.gcisServerCharset" par "UTF-8".

3. Est-il possible de passer plusieurs *geometries* lors d'un appel au Web Service?

   Oui, il suffit de préciser plus de triplets séparés par le caractère ";"

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:findObjectsUnder>
         <!--Optional:-->
         <srs>epsg:4326</srs>
         <!--Optional:-->
         <geometries>1,6.01501,49.350;2,6.01701,49.390</geometries>
         <!--Optional:-->
         <topology></topology>
         <!--Optional:-->
         <mapName>geowebSmp.gcm</mapName>
         <!--Optional:-->
         <className>Pyr</className>
         <!--Optional:-->
         <subclassName>P2L2</subclassName>
         <!--Optional:-->
         <fields>Nom;X;Y</fields>
      </sch:findObjectsUnder>
   </soapenv:Body>
</soapenv:Envelope>
```

Retourne :

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
        <soap:Body>
                <ns2:findObjectsUnderResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
```

```
                        <FindObjectsResult>
                                <status>OK</status>
                                <results>
                                        <id>1</id>
                                        <objects>
                                                <object>
                                                        <fields>
                                                                <field>FONTOY</field>
                                                                <field>868858</field>
                                                                <field>2489508</field>
                                                        </fields>
                                                </object>
                                        </objects>
                                </results>
                                <results>
                                        <id>2</id>
                                        <objects>
                                                <object>
                                                        <fields>
                                                                <field>HAVANGE</field>
                                                                <field>864342.4</field>
                                                                <field>2494239.52</field>
                                                        </fields>
                                                </object>
                                        </objects>
                                </results>
                        </FindObjectsResult>
                </ns2:findObjectsUnderResponse>
        </soap:Body>
</soap:Envelope>
```

Calcul d’itinéraire

Cette page n’est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

Ce web service calcule un itinéraire entre deux points et renvoie une feuille de route complète. L’ajout d'étapes intermédiaires est une possibilité. Il s’appuie sur le graphe paramétré dont le nom a été spécifié dans l’interface d’administration de Geoconcept Web.

Disponibilité

Ce web service est disponible en permanence avec Geoconcept Web et un graphe.

Changement de version

Les versions précédente du web service sont conservées dans Geoconcept Web pour assurer la compatibilité avec les développements antérieurs. Il est recommandé d’utiliser la version la plus récente.

Changements avec la v6

- Ajout des paramètres "formatItems" et "fields".

Changements avec la v5

- Ajout des paramètres "maxCost", "timeOut" et "computeOptions".
- Ajout dans la snapMethod "nodes" de l'accrochage aux noeuds les plus proches.
- Ajout des formats "compressedgeometry" et "compressedsimplifiedgeometry" avec utilisation de l'algorithme de compression *encodage de polylines* (Encoded Polyline Algorithm Format [https://developers.google.com/maps/documentation/utilities/polylinealgorithm]) qui permet de stocker les coordonnées dans une simple chaîne de caractères ASCII afin de réduire de manière significative la taille globale des données.
- Ajout du format "tollcost", nécessite une licence d'accès à la plateforme HERE.

Changements avec la v4

- Ajout de la notion de noeud, plus rapide, pour s'accrocher aux noeuds du graphe plutôt qu'a des coordonnées géographiques. Ajout des éléments suivants : "originNode", "destinationNode", "waypointNodes", de "node" dans le paramètre format et de "nodes" dans le paramètre snapMethod.
- Ajout dans la réponse des éléments suivants : "originNode", "destinationNode", "waypointNodes" et "carbonFootprint".
- Ajout des paramètres "avoidArea" et "configName".

Changements avec la v3

- Ajout du paramètre "snapMethod"
- Suppression du paramètre "projection", remplacé par "srs"
- Suppression du paramètre "RejectFlags", remplacé par "exclusions"
- Changement du typage des paramètres "distanceMeters" et "durationSeconds" de string/Double à double
- Les paramètres "geometryWkt" et "simplifiedWkt" sont renommés respectivement "wktGeometry" et "wktSimplifiedGeometry"
- Le paramètre "navInstruction" est renommé "navigationInstruction"
- Dans RouteResultV3 ajout du paramètre "srs"
- Le paramètre "elapsedTimeSeconds" est renommé "durationSeconds" et son typage passe de string à double
- Le paramètre "coordinates" est renommé "location"

Changements avec la V2

- Ajout du paramètre "timeLine"
- Suppression des paramètres "map" et "applyMapPrecisionOut"

- Les paramètres "distance" et "duration" sont renommés respectivement "distanceMeters" et "durationSeconds"

V6

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
origin	Coordonnées du point de départ. Les coordonnées longitude et latitude sont séparées par la caractère ,	non	
originNode	Noeud de départ. Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
destination	Coordonnées du point d'arrivée. Les coordonnées longitude et latitude sont séparées par la caractère ,	non	
destinationNode	Noeud d'arrivée. Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
waypoints	Coordonnées des étapes. Chaque couple de coordonnées d'une étape est encadré par la balise <waypoint> Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	
waypointNodes	Noeuds d'étapes. Chaque couple de coordonnées d'une étape est encadré par la balise <waypointNode> Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
format	- standard : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt / géométrie simplifiée au format wkt / liste des segments concaténés (sans géométries) - extended : résultat = résumé de l'itinéraire / bornes / géométrie simplifiée au format wkt / liste des segments non-concaténés (avec géométries) - summary : résultat = résumé de l'itinéraire - geometry : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt - simplifiedgeometry : résultat = résumé de l'itinéraire / bornes / géométrie simplifiée au format wkt - geometries : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt / géométrie simplifiée au format wkt - brief : résultat = résumé de l'itinéraire / Liste des segments avec durée et distance - standardext : résultat = résumé de l'itinéraire / bornes / liste des segments concaténés avec durée et distance et géométrie - compressedgeometry : résultat = résumé de l'itinéraire / bornes / géometrie compressée avec encodage de polylines - compressedsimplifiedgeometry : résultat = résumé de l'itinéraire / bornes / géometrie simplifiée compressée avec encodage de polylines - node : résultat = résumé de l'itinéraire / id des noeuds accrochés - completegeometry : result = résumé de l'itinéraire / bornes / géométrie au format wkt / liste de segments concaténés avec géométrie au format wkt + - compresscompletegeometry : result = résumé de l'itinéraire / bornes /	oui	standard

paramètre	description	optionnel	défaut
	géometrie simplifiée compressée avec encodage de polylines / liste de segments concaténés et géometrie compressée avec encodage de polylines - tollcost : résultat = résumé de l'itinéraire / montant des péages. Le montant est calculé en fonction du type de véhicule définit *configName* ainsi que de la date/heure du départ de l'itinéraire *startDateTime* - custom : retourne la liste des items déclarés dans le paramètre *customFormat*.		
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
tolerance	Distance de tolérance (en mètre) de simplification de la géométrie.	oui	
graphName (déprécié)	Nom du graphe à utiliser Ce paramètre est omis si le paramètre configName est utilisé.	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris. Attention le caractère + peut être mal interprété par les navigateurs, dans ce cas, il faut le remplacer par *%2B*.	oui	
profileId (déprécié)	Identifiant du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileName (déprécié)	Profil du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère *;* (Exemple : Toll, Tunnel, Bridge)	oui	
timeLine	Liste de durées intermédiaires pour calculer des positions le long du trajet. Exprimées en secondes, séparés par le caractère *;*. La position retournée correspond au noeud précédent, sur le trajet, connectable au réseau. Par exemple, sur une autoroute, la position renvoyée est celle la dernière sortie ou aire de service avant la position demandée.	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction - nodes : Accrochage direct aux noeuds fournis par les paramètres originNode, destinationNode et waypointNodes ou, si ces premiers ne sont pas renseignés, aux noeuds les plus proches des paramètres origin, destination et waypoint	oui	standard
avoidArea	Zone de transit interdit au format WKT (POLYGON ou MULTIPOLYGON) dans la projection (paramètre srs) demandée Exemple en wgs84 : `POLYGON ((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689))` - `MULTIPOLYGON (((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689)), ((-1.558639 47.218144, -1.558639 47.218193, [...] -1.558639 47.218144)))` Attention les géométries WKT doivent être fermées.	oui	
configName	Nom de la configuration à utiliser (défini dans Geoconcept Web - Administration / Outils / Définitions des graphes) Il remplace l'usage de graphName, profileId et profileName	oui	
computeOptions	Liste des options pour le calcul, séparées par le caractère *;*	oui	

paramètre	description	optionnel	défaut
	- trafficPatterns : utilise les statistiques routières (il est nécessaire de renseigner le paramètre startDateTime et d'utiliser un graphe intégrant les informations de traffic patterns) - speedPattern (M18) : utilise d'une *speed pattern* tel que définis dans le fichier SmartRoutingVehicles.xml qui se trouve dans le dossier `"*<GEOCONCEPT_WEB_HOME>*"*smartrouting**jee**smartrouting**conf*\ . Usage : "speedPattern:slow-speed" - length (M18) : longueur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "length:950" - width (M18) : largeur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "width:255" - height (M18) : hauteur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "height:360" - weight (M18) : poids maximum autorisé en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weight:18000" - axles (M18) : nombre maximum d'axe autorisé (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "axles:2" - weightPerAxle (M18) : poids maximum autorisé par axe en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weightPerAxle:9000" - fuelType : type de carburant utilisé pour la calcul de l'empreinte carbone du trajet. Les valeurs disponibles sont : "UndefinedFuelType", "NoFuel", "Diesel", "UnleadedFuel", "LGP" et "CustomFuelType". Usage : "fuelType:Diesel" - averageConsumption : consommation moyenne en litre pour 100 kilomètres. Usage : "averageConsumption:6.45" - customAverageCO2UnitEmission : définit l'empreinte carbone en kilograme par litre. Usage : "customAverageCO2UnitEmission:2.724" - snapSpeed : vitesse d'accrochage au graphe en kilomètre par heure. Usage : "snapSpeed:10"		
maxCost	Coût maximum à ne pas dépasser dans le calcul -1 : pas de coût maximum à prendre en compte 0 : prendre la valeur par default défini dans la configuration de SmartRouting Server sinon : valeur en mètres si method=distance ou en secondes si method=time	oui	
timeOut	Time out pour le calcul (en millisecondes)	oui	
formatItems	Liste des items de formats disponibles, si le paramètre *format* est définit sur custom - ROUTE_DISTANCE - ROUTE_DURATION - ROUTE_DISTANCE_FORMATTED - ROUTE_DURATION_FORMATTED - ROUTE_BOUNDS - SRS - ROUTE_CARBON_FOOTPRINT - START_FINISH_DATETIME - SUBROUTE_DISTANCE - SUBROUTE_DURATION - SUBROUTE_DISTANCE_FORMATTED - SUBROUTE_DURATION_FORMATTED	oui	

paramètre	description	optionnel	défaut
	- SEGMENT_DISTANCE - SEGMENT_DURATION - SEGMENT_DISTANCE_FORMATTED - SEGMENT_DURATION_FORMATTED - SEGMENT_NAME - SEGMENT_NAVIGATIONINSTRUCTION - SEGMENT_POINTSLIST - SEGMENT_FIELDSVALUES - SEGMENT_UNCONSOLIDATED - ROUTE_GEOMETRY_WKT - ROUTE_SIMPLIFIEDGEOMETRY_WKT - ROUTE_GEOMETRY_COMPRESSED - ROUTE_SIMPLIFIEDGEOMETRY_COMPRESSED - SUBROUTE_GEOMETRY_WKT - SUBROUTE_GEOMETRY_COMPRESSED - TIMELINE - POINTS_GRAPHNODES - ROUTE_TOLLCOST - COMPUTATION_TIME		
fields	Liste des champs à afficher en resultat de chaque segment (pour les *format* standard et extended) - Iso country code - Country - Postcode left - Postcode right - City name left - City name right - Ramp - Urban - Tunnel - Toll - Bicycles - Automobiles - Frontage - Bridge	oui	

(M18) Disponible à partir de la version M18 des graphes fournis par GEOCONCEPT SAS.

En sortie

Itinéraire (routeResultV6)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l'itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l'itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	double	0/1	Distance totale de l'itinéraire en mètres.
durationSeconds	double	0/1	Durée totale de l'itinéraire en secondes.
bounds	string	0/1	Bornes (BoundingBox) de la géométrie de l'itinéraire.

paramètre	type	min/max	description
wktGeometry	string	0/1	Géométrie de l’itinéraire au format WKT.
wktSimplified	string	0/1	Géométrie simplifiée de l’itinéraire au format WKT.
leg (ou legs en JSON / JSON-P)	subRouteV5 (ou array en JSON / JSON-P)	0/illimité	Liste des portions d’itinéraires.
startDateTime	string	0/1	Date et heure de départ (format : ISO8601 sans code zone : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 pour un départ le 21 janvier 2014, à 9h à Paris
finishDateTime	string	0/1	Date et heure d’arrivée (format : ISO8601 sans code zone : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 pour une arrivée le 21 janvier 2014, à 9h à Paris
timeLineItem	timeLineItemV5 (ou array en JSON / JSON-P)	0/illimité	Liste des positions calculées.
srs	string	0/1	projection passée en entrée (code EPSG comme epsg:4326 ou wgs84)
originNode	string	0/1	Identifiant du noeud de départ (renseigné si format=NODE).
destinationNode	string	0/1	Identifiant du noeud d’arrivée (renseigné si format=NODE).
waypointNodes	array de waypointNodes (string)	0/illimité	Identifiants des noeuds d'étapes (renseigné si format=NODE).
carbonFootprint	double	0/1	Empreinte carbone (émission CO2 en kilogrammes)
tollCost	Array de tollCost	0/illimité	Information sur le coût des péages (renseigné si format=tollCost).
fieldsNames	Array de fields	0/illimité	liste des champs demandés avec le paramètre *field*.

Portion d’itinéraire (subRouteV6)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l’itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l’itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	double	0/1	Distance de la portion de l’itinéraire en mètres.
durationSeconds	double	0/1	Durée de la portion de l’itinéraire en secondes.
step ou (ou steps en JSON / JSON-P)	segmentV5 (ou array en JSON / JSON-P)	0/illimité	Liste des segments composants la portion d’itinéraire.

Segment d’itinéraire (segmentV6)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l’itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l’itinéraire, formaté :

paramètre	type	min/max	description
			- HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	double	0/1	Distance du segment de l'itinéraire en mètres.
durationSeconds	double	0/1	Durée du segment de l'itinéraire en secondes.
navigationInstruction	string	0/1	Code d'instruction de navigation : - F : Tout droit - FR: Tourner légèrement à droite - FL: Tourner légèrement à gauche - R: Tourner à droite - L: Tourner à gauche - BR: Tourner fortement à droite - BL: Tourner fortement à gauche - B: Demi-tour - round_about_entry: S'engager sur le rond-point - round_about_exit: Sortir du rond-point
name	string	0/1	Nom de rue du segment.
point	string	0/illimité	Liste de coordonnées séparées par la caractère ,

Position d'itinéraire (timeLineItemV6)

paramètre	type	min/max	description
durationSeconds	double	0/1	Durée de l'itinéraire en secondes jusquà cette position.
message	string	0/1	Message d'erreur pour cette position.
status	string	0/1	Status de cette position.
distanceMeters	double	0/1	Distance de l'itinéraire en mètres jusquà cette position.
location	string	0/1	Coordonnées de la position.

Coût des péages (tollCost)

paramètre	type	min/max	description
amount	double	0/1	Coût total des péages.
currency	string	0/1	Devise.
costsByCountry	Array de costsByCountry	0/ unlimited	Liste des coût des péages par pays.

Coût des péages par pays (costsByCountry)

paramètre	type	min/max	description
amount	double	0/1	Coût des péages.
country	string	0/1	Code pays (3-lettres ISO codes pays).

SOAP

WSDL

http://<*server*>/<*webapp*>/api/ws/routeService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:routeV6>
         <!--Optional:-->
         <request>
            <origin>
               <x>-1.351448</x>
               <y>47.446923</y>
            </origin>
            <!--Optional:-->
            <originNode></originNode>
            <destination>
               <x>-1.34529</x>
               <y>47.4479931</y>
            </destination>
            <!--Optional:-->
            <destinationNode></destinationNode>
            <!--Optional:-->
            <waypoints>
               <!--Zero or more repetitions:-->
               <waypoint>
                  <x>-1.34981</x>
                  <y>47.44837</y>
               </waypoint>
            </waypoints>
            <!--Optional:-->
            <waypointNodes>
               <!--Zero or more repetitions:-->
               <waypointNode></waypointNode>
            </waypointNodes>
            <!--Optional:-->
            <srs>epsg:4326</srs>
            <!--Optional:-->
            <method>time</method>
            <!--Optional:-->
            <format>STANDARD</format>
            <!--Optional:-->
            <tolerance>0.</tolerance>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <timeLine>
               <!--Zero or more repetitions:-->
               <timeLineItem></timeLineItem>
            </timeLine>
            <!--Optional:-->
            <snapMethod></snapMethod>
            <!--Optional:-->
```

```
            <avoidArea></avoidArea>
            <!--Optional:-->
            <computeOptions></computeOptions>
            <!--Optional:-->
            <timeOut></timeOut>
            <!--Optional:-->
            <configName></configName>
         </request>
      </sch:routeV6>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:routeV6Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <RouteResult>
            <status>OK</status>
            <distance>640 m</distance>
            <duration>0:02:06</duration>
            <distanceMeters>639.97</distanceMeters>
            <durationSeconds>126.88</durationSeconds>
            <bounds>-1.351448,47.446922;-1.345263,47.44848</bounds>
            <wktGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, [...] ,  -1.34529
 47.447993)</wktGeometry>
            <wktSimplifiedGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922,  [...] , -1.34529
 47.447993)</wktSimplifiedGeometry>
            <leg>
               <distance>233 m</distance>
               <duration>0:00:46</duration>
               <distanceMeters>233.1</distanceMeters>
               <durationSeconds>46.28</durationSeconds>
               <step>
                  <distance>139 m</distance>
                  <duration>0:00:29</duration>
                  <distanceMeters>138.59</distanceMeters>
                  <durationSeconds>29.28</durationSeconds>
                  <name>RUE LA FONTAINE BRUNEAU</name>
               </step>
               <step>
                  <distance>91 m</distance>
                  <duration>0:00:16</duration>
                  <distanceMeters>91.33</distanceMeters>
                  <durationSeconds>16.43</durationSeconds>
                  <navigationInstruction>FR</navigationInstruction>
                  <name>RUE DES SAULES</name>
               </step>
               <step>
                  <distance>3 m</distance>
                  <duration>0:00:00</duration>
                  <distanceMeters>3.18</distanceMeters>
                  <durationSeconds>0.57</durationSeconds>
                  <navigationInstruction>round_about_entry</navigationInstruction>
                  <name />
               </step>
            </leg>
            <leg>
               <distance>407 m</distance>
               <duration>0:01:20</duration>
               <distanceMeters>406.87</distanceMeters>
               <durationSeconds>80.6</durationSeconds>
               <step>
```

```
                  <distance>18 m</distance>
                  <duration>0:00:03</duration>
                  <distanceMeters>17.64</distanceMeters>
                  <durationSeconds>3.17</durationSeconds>
                  <name />
               </step>
               <step>
                  <distance>67 m</distance>
                  <duration>0:00:15</duration>
                  <distanceMeters>66.62</distanceMeters>
                  <durationSeconds>15.05</durationSeconds>
                  <navigationInstruction>round_about_exit</navigationInstruction>
                  <name>RUE DES FOURS</name>
               </step>
               <step>
                  <distance>36 m</distance>
                  <duration>0:00:08</duration>
                  <distanceMeters>35.74</distanceMeters>
                  <durationSeconds>8.57</durationSeconds>
                  <navigationInstruction>F</navigationInstruction>
                  <name>PLACE DE L'ÉGLISE</name>
               </step>
               <step>
                  <distance>72 m</distance>
                  <duration>0:00:15</duration>
                  <distanceMeters>72.25</distanceMeters>
                  <durationSeconds>15.24</durationSeconds>
                  <navigationInstruction>F</navigationInstruction>
                  <name>RUE DES PRESSOIRS</name>
               </step>
               <step>
                  <distance>26 m</distance>
                  <duration>0:00:04</duration>
                  <distanceMeters>26.02</distanceMeters>
                  <durationSeconds>4.68</durationSeconds>
                  <navigationInstruction>round_about_entry</navigationInstruction>
                  <name />
               </step>
               <step>
                  <distance>183 m</distance>
                  <duration>0:00:32</duration>
                  <distanceMeters>182.84</distanceMeters>
                  <durationSeconds>32.91</durationSeconds>
                  <navigationInstruction>round_about_exit</navigationInstruction>
                  <name>RUE DU BOURG DRAPÉ</name>
               </step>
               <step>
                  <distance>6 m</distance>
                  <duration>0:00:00</duration>
                  <distanceMeters>5.76</distanceMeters>
                  <durationSeconds>0.98</durationSeconds>
                  <navigationInstruction>FR</navigationInstruction>
                  <name>RUE DE LA CURE</name>
               </step>
            </leg>
            <srs>epsg:4326</srs>
            <carbonFootprint>0.0</carbonFootprint>
         </RouteResult>
      </ns2:routeV6Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/route/v6.json?
origin=-1.351448,47.446923&destination=-1.34529,47.4479931&waypoints=-1.34981,47.44837
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/route/v6.json?
origin=-1.351448,47.446923&destination=-1.34529,47.4479931&waypoints=-1.34981,47.44837&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/route/v6.xml?
origin=-1.351448,47.446923&destination=-1.34529,47.4479931&waypoints=-1.34981,47.44837
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
    "message": null,
    "status": "OK",
    "distance": "640 m",
    "duration": "0:02:09",
    "distanceMeters": 639.95,
    "durationSeconds": 129.48,
    "bounds": "-1.351448,47.446922;-1.345263,47.44848",
    "wktGeometry": "LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])",
    "wktSimplifiedGeometry":"LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699,
 [...])",
    "legs": [
        {
            "distance": "233 m",
            "duration": "0:00:46",
            "distanceMeters": 233.09,
            "durationSeconds": 46.84,
            "steps": [
                {
                    "distance": "139 m",
                    "duration": "0:00:29",
                    "distanceMeters": 138.59,
                    "durationSeconds": 29.83,
                    "navigationInstruction": null,
                    "name": "RUE LA FONTAINE BRUNEAU",
                    "points": []
                },
                {
                    [...]
                }
            ]
        },
        {
            "distance": "407 m",
```

```
            "duration": "0:01:22",
            "distanceMeters": 406.86,
            "durationSeconds": 82.64,
            "steps": [
                {
                    "distance": "18 m",
                    "duration": "0:00:03",
                    "distanceMeters": 17.63,
                    "durationSeconds": 3.18,
                    "navigationInstruction": null,
                    "name": "",
                    "points": []
                },
                {
                   [...]
                }
            ]
        }
    ],
    "startDateTime": null,
    "finishDateTime": null,
    "srs": null,
    "originNode": null,
    "waypointNodes": null,
    "destinationNode": null,
    "carbonFootprint": 0.195038864105688
}
```


Format JSON-P


```
myCallback(
    {
        "message": null,
        "status": "OK",
        "distance": "640 m",
        "duration": "0:02:09",
        "distanceMeters": 639.95,
        "durationSeconds": 129.48,
        "bounds": "-1.351448,47.446922;-1.345263,47.44848",
        "wktGeometry": "LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])",
        "wktSimplifiedGeometry":"LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699,
 [...])",
        "legs": [
            {
                "distance": "233 m",
                "duration": "0:00:46",
                "distanceMeters": 233.09,
                "durationSeconds": 46.84,
                "steps": [
                    {
                        "distance": "139 m",
                        "duration": "0:00:29",
                        "distanceMeters": 138.59,
                        "durationSeconds": 29.83,
                        "navigationInstruction": null,
                        "name": "RUE LA FONTAINE BRUNEAU",
                        "points": []
                    },
                    {
                        [...]
                    }
                ]
            },
```

```
            {
                "distance": "407 m",
                "duration": "0:01:22",
                "distanceMeters": 406.86,
                "durationSeconds": 82.64,
                "steps": [
                    {
                        "distance": "18 m",
                        "duration": "0:00:03",
                        "distanceMeters": 17.63,
                        "durationSeconds": 3.18,
                        "navigationInstruction": null,
                        "name": "",
                        "points": []
                    },
                    {
                       [...]
                    }
                ]
            }
        ],
        "startDateTime": null,
        "finishDateTime": null,
        "srs": null,
        "originNode": null,
        "waypointNodes": null,
        "destinationNode": null,
        "carbonFootprint": 0.195038864105688
    }
);
```

Format XML

```
<?xml version="1.0" encoding="UTF-8"?>
<routeResultV6>
        <status>OK</status>
        <distance>640 m</distance>
        <duration>0:02:09</duration>
        <distanceMeters>639.95</distanceMeters>
        <durationSeconds>129.48</durationSeconds>
        <bounds>-1.351448,47.446922;-1.345263,47.44848</bounds>
        <wktGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
wktGeometry>
        <wktSimplifiedGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699,
 [...])</wktSimplifiedGeometry>
        <leg>
                <distance>233 m</distance>
                <duration>0:00:46</duration>
                <distanceMeters>233.09</distanceMeters>
                <durationSeconds>46.84</durationSeconds>
                <step>
                        <distance>139 m</distance>
                        <duration>0:00:29</duration>
                        <distanceMeters>138.59</distanceMeters>
                        <durationSeconds>29.83</durationSeconds>
                        <name>RUE LA FONTAINE BRUNEAU</name>
                </step>
                <step>
            [...]
                </step>
        </leg>
        <leg>
                <distance>407 m</distance>
```

```
                <duration>0:01:22</duration>
                <distanceMeters>406.86</distanceMeters>
                <durationSeconds>82.64</durationSeconds>
                <step>
                        <distance>18 m</distance>
                        <duration>0:00:03</duration>
                        <distanceMeters>17.63</distanceMeters>
                        <durationSeconds>3.18</durationSeconds>
                        <name />
                </step>
                <step>
                        [...]
                </step>
        </leg>
        <carbonFootprint>0.195038864105688</carbonFootprint>
</routeResultV6>
```

API JavaScript

Inclure la librairie JavaScript.

```
var routeCtrl = new GCUI.Control.Route();
routeCtrl.route({
    url:'http://<server>/<webapp>/api/lbs/route/v5.json',
    tolerance : 100,
    origin : new OpenLayers.LonLat(0.691012, 47.384813),
    destination : new OpenLayers.LonLat(0.691012, 47.384813),
    waypoints : [ new OpenLayers.LonLat(2.344408, 49.898798) ],
    callback : function(result, options) {
        console.log(result);
    }
});
```

La variable **`result`** est au format JSON décrit ci-dessus. La fonction **`callback`** passée en paramètre est appelée à la fin du calcul d'itinéraire.

Retours possibles

Cas d'un itinéraire trouvé (routeResultV6/status est OK)

```
<?xml version="1.0" encoding="UTF-8"?>
<routeResultV6>
        <status>OK</status>
        <distance>640 m</distance>
        <duration>0:02:09</duration>
        <distanceMeters>639.95</distanceMeters>
        <durationSeconds>129.48</durationSeconds>
        <bounds>-1.351448,47.446922;-1.345263,47.44848</bounds>
        <wktGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
wktGeometry>
        <wktSimplifiedGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699,
 [...])</wktSimplifiedGeometry>
        <leg>
                <distance>233 m</distance>
                <duration>0:00:46</duration>
                <distanceMeters>233.09</distanceMeters>
                <durationSeconds>46.84</durationSeconds>
                <step>
                        <distance>139 m</distance>
                        <duration>0:00:29</duration>
```

```
                        <distanceMeters>138.59</distanceMeters>
                        <durationSeconds>29.83</durationSeconds>
                        <name>RUE LA FONTAINE BRUNEAU</name>
                </step>
                <step>
            [...]
                </step>
        </leg>
        <leg>
                <distance>407 m</distance>
                <duration>0:01:22</duration>
                <distanceMeters>406.86</distanceMeters>
                <durationSeconds>82.64</durationSeconds>
                <step>
                        <distance>18 m</distance>
                        <duration>0:00:03</duration>
                        <distanceMeters>17.63</distanceMeters>
                        <durationSeconds>3.18</durationSeconds>
                        <name />
                </step>
                <step>
                        [...]
                </step>
        </leg>
        <carbonFootprint>0.195038864105688</carbonFootprint>
</routeResultV6>
```

Cas d'un oubli de spécification du point de départ ou d'arrivée (routeResultV6/status est ERROR)

```
<routeResultV6>
   <message>Origin and destination must be not null</message>
   <status>ERROR</status>
   <distanceMeters>0.0</distanceMeters>
   <durationSeconds>0.0</durationSeconds>
</routeResultV6>
```

Cas d'un mauvais formatage du point de départ, d'arrivée ou des étapes (routeResultV6/status est ERROR)

```
<routeResultV6>
   <message>Origin, destination and waypoints should be represented by a couple of coordinates</message>
   <status>ERROR</status>
   <distanceMeters>0.0</distanceMeters>
   <durationSeconds>0.0</durationSeconds>
</routeResultV6>
```

Cas d'un mauvais typage (serviceResult/status est ERROR)

```
<serviceResult>
        <message>NumberFormatException: For input string: "AAA"</message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'une erreur d'accrochage au graphe (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in route computation
Error in smartrouting
Failed to execute calculateRoute
```

```
com.geoconcept.smartrouting.SmartRoutingNativeException: failed to connect waypoint { 146.691012, 47.384813,
 0.000000 }
failed to connect waypoint { 146.691012, 47.384813, 0.000000 }</message>
   <status>ERROR</status>
</serviceResult>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
    <message>ServiceException: Error in route computation
Error in smartrouting
datasource is null</message>
    <status>ERROR</status>
</serviceResult>
```

V5

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
origin	Coordonnées du point de départ. Les coordonnées longitude et latitude sont séparées par la caractère ,	non	
originNode	Noeud de départ. Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
destination	Coordonnées du point d'arrivée. Les coordonnées longitude et latitude sont séparées par la caractère ,	non	
destinationNode	Noeud d'arrivée. Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
waypoints	Coordonnées des étapes. Chaque couple de coordonnées d'une étape est encadré par la balise <waypoint> Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	
waypointNodes	Noeuds d'étapes. Chaque couple de coordonnées d'une étape est encadré par la balise <waypointNode> Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
format	- standard : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt / géométrie simplifiée au format wkt / liste des segments concaténés (sans géométries) - extended : résultat = résumé de l'itinéraire / bornes / géométrie simplifiée au format wkt / liste des segments non-concaténés (avec géométries) - summary : résultat = résumé de l'itinéraire - geometry : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt - simplifiedgeometry : résultat = résumé de l'itinéraire / bornes / géométrie simplifiée au format wkt - geometries : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt / géométrie simplifiée au format wkt	oui	standard

paramètre	description	optionnel	défaut
	- brief : résultat = résumé de l'itinéraire / Liste des segments avec durée et distance - standardext : résultat = résumé de l'itinéraire / bornes / liste des segments concaténés avec durée et distance et géométrie - compressedgeometry : résultat = résumé de l'itinéraire / bornes / géometrie compressée avec encodage de polylines - compressedsimplifiedgeometry : résultat = résumé de l'itinéraire / bornes / géometrie simplifiée compressée avec encodage de polylines - node : résultat = résumé de l'itinéraire / id des noeuds accrochés - completegeometry : result = résumé de l'itinéraire / bornes / géométrie au format wkt / liste de segments concaténés avec géométrie au format wkt + - compresscompletegeometry : result = résumé de l'itinéraire / bornes / géometrie simplifiée compressée avec encodage de polylines / liste de segments concaténés et géometrie compressée avec encodage de polylines - tollcost: résultat = résumé de l'itinéraire / montant des péages. Le montant est calculé en fonction du type de véhicule définit *configName* ainsi que de la date/heure du départ de l'itinéraire *startDateTime*.		
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
tolerance	Distance de tolérance (en mètre) de simplification de la géométrie.	oui	
graphName (déprécié)	Nom du graphe à utiliser Ce paramètre est omis si le paramètre configName est utilisé.	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris. Attention le caractère + peut être mal interprété par les navigateurs, dans ce cas, il faut le remplacer par *%2B*.	oui	
profileId (déprécié)	Identifiant du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileName (déprécié)	Profil du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère *;* (Exemple : Toll, Tunnel, Bridge)	oui	
timeLine	Liste de durées intermédiaires pour calculer des positions le long du trajet. Exprimées en secondes, séparés par le caractère *;*. La position retournée correspond au noeud précédent, sur le trajet, connectable au réseau. Par exemple, sur une autoroute, la position renvoyée est celle la dernière sortie ou aire de service avant la position demandée.	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction - nodes : Accrochage direct aux noeuds fournis par les paramètres originNode, destinationNode et waypointNodes ou, si ces premiers ne sont pas renseignés, aux noeuds les plus proches des paramètres origin, destination et waypoint	oui	standard
avoidArea	Zone de transit interdit au format WKT (POLYGON ou MULTIPOLYGON) dans la projection (paramètre srs) demandée Exemple en wgs84 : `POLYGON ((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689))` - `MULTIPOLYGON`	oui	

paramètre	description	optionnel	défaut
	`(((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689)), ((-1.558639 47.218144, -1.558639 47.218193, [...] -1.558639 47.218144)))` Attention les géométries WKT doivent être fermées.		
configName	Nom de la configuration à utiliser (défini dans Geoconcept Web - Administration / Outils / Définitions des graphes) Il remplace l'usage de graphName, profileId et profileName	oui	
computeOptions	Liste des options pour le calcul, séparées par le caractère *;* - trafficPatterns : utilise les statistiques routières (il est nécessaire de renseigner le paramètre startDateTime et d'utiliser un graphe intégrant les informations de traffic patterns) - speedPattern (M18) : utilise d'une *speed pattern* tel que définis dans le fichier SmartRoutingVehicles.xml qui se trouve dans le dossier `"*<GEOCONCEPT_WEB_HOME>*"*smartrouting**jee**smartrouting**conf*\ . Usage : "speedPattern:slow-speed" - length (M18) : longueur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "length:950" - width (M18) : largeur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "width:255" - height (M18) : hauteur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "height:360" - weight (M18) : poids maximum autorisé en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weight:18000" - axles (M18) : nombre maximum d'axe autorisé (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "axles:2" - weightPerAxle (M18) : poids maximum autorisé par axe en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weightPerAxle:9000" - fuelType : type de carburant utilisé pour la calcul de l'empreinte carbone du trajet. Les valeurs disponibles sont : "UndefinedFuelType", "NoFuel", "Diesel", "UnleadedFuel", "LGP" et "CustomFuelType". Usage : "fuelType:Diesel" - averageConsumption : consommation moyenne en litre pour 100 kilomètres. Usage : "averageConsumption:6.45" - customAverageCO2UnitEmission : définit l'empreinte carbone en kilograme par litre. Usage : "customAverageCO2UnitEmission:2.724" - snapSpeed : vitesse d'accrochage au graphe en kilomètre par heure. Usage : "snapSpeed:10"	oui	
maxCost	Coût maximum à ne pas dépasser dans le calcul -1 : pas de coût maximum à prendre en compte 0 : prendre la valeur par default défini dans la configuration de SmartRouting Server sinon : valeur en mètres si method=distance ou en secondes si method=time	oui	
timeOut	Time out pour le calcul (en millisecondes)	oui	

(M18) Disponible à partir de la version M18 des graphes fournis par GEOCONCEPT SAS.

En sortie

Itinéraire (routeResultV5)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l'itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l'itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	double	0/1	Distance totale de l'itinéraire en mètres.
durationSeconds	double	0/1	Durée totale de l'itinéraire en secondes.
bounds	string	0/1	Bornes (BoundingBox) de la géométrie de l'itinéraire.
wktGeometry	string	0/1	Géométrie de l'itinéraire au format WKT.
wktSimplified	string	0/1	Géométrie simplifiée de l'itinéraire au format WKT.
leg (ou legs en JSON / JSON-P)	subRouteV5 (ou array en JSON / JSON-P)	0/illimité	Liste des portions d'itinéraires.
startDateTime	string	0/1	Date et heure de départ (format : ISO8601 sans code zone : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 pour un départ le 21 janvier 2014, à 9h à Paris
finishDateTime	string	0/1	Date et heure d'arrivée (format : ISO8601 sans code zone : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 pour une arrivée le 21 janvier 2014, à 9h à Paris
timeLineItem	timeLineItemV5 (ou array en JSON / JSON-P)	0/illimité	Liste des positions calculées.
srs	string	0/1	projection passée en entrée (code EPSG comme epsg:4326 ou wgs84)
originNode	string	0/1	Identifiant du noeud de départ (renseigné si format=NODE).
destinationNode	string	0/1	Identifiant du noeud d'arrivée (renseigné si format=NODE).
waypointNodes	array de waypointNodes (string)	0/illimité	Identifiants des noeuds d'étapes (renseigné si format=NODE).
carbonFootprint	double	0/1	Empreinte carbone (émission CO2 en kilogrammes)
tollCost	Array de tollCost	0/illimité	Information sur le coût des péages (renseigné si format=tollCost).

Portion d'itinéraire (subRouteV5)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l'itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l'itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	double	0/1	Distance de la portion de l'itinéraire en mètres.
durationSeconds	double	0/1	Durée de la portion de l'itinéraire en secondes.
step ou (ou steps en JSON / JSON-P)	segmentV5 (ou array en JSON / JSON-P)	0/illimité	Liste des segments composants la portion d'itinéraire.

Segment d'itinéraire (segmentV5)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l'itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l'itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	double	0/1	Distance du segment de l'itinéraire en mètres.
durationSeconds	double	0/1	Durée du segment de l'itinéraire en secondes.
navigationInstruction	string	0/1	Code d'instruction de navigation : - F : Tout droit - FR: Tourner légèrement à droite - FL: Tourner légèrement à gauche - R: Tourner à droite - L: Tourner à gauche - BR: Tourner fortement à droite - BL: Tourner fortement à gauche - B: Demi-tour - round_about_entry: S'engager sur le rond-point - round_about_exit: Sortir du rond-point
name	string	0/1	Nom de rue du segment.
point	string	0/illimité	Liste de coordonnées séparées par la caractère ,

Position d'itinéraire (timeLineItemV5)

paramètre	type	min/max	description
durationSeconds	double	0/1	Durée de l'itinéraire en secondes jusquà cette position.
message	string	0/1	Message d'erreur pour cette position.
status	string	0/1	Status de cette position.
distanceMeters	double	0/1	Distance de l'itinéraire en mètres jusquà cette position.
location	string	0/1	Coordonnées de la position.

Coût des péages (tollCost)

paramètre	type	min/max	description
amount	double	0/1	Coût total des péages.
currency	string	0/1	Devise.
costsByCountry	Array de costsByCountry	0/ unlimited	Liste des coût des péages par pays.

Coût des péages par pays (costsByCountry)

paramètre	type	min/max	description
amount	double	0/1	Coût des péages.
country	string	0/1	Code pays (3-lettres ISO codes pays).

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/routeService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:routeV5>
         <!--Optional:-->
         <request>
            <origin>
               <x>-1.351448</x>
               <y>47.446923</y>
            </origin>
            <!--Optional:-->
            <originNode></originNode>
            <destination>
               <x>-1.34529</x>
               <y>47.4479931</y>
            </destination>
            <!--Optional:-->
            <destinationNode></destinationNode>
            <!--Optional:-->
            <waypoints>
               <!--Zero or more repetitions:-->
               <waypoint>
                  <x>-1.34981</x>
                  <y>47.44837</y>
               </waypoint>
            </waypoints>
            <!--Optional:-->
            <waypointNodes>
               <!--Zero or more repetitions:-->
               <waypointNode></waypointNode>
            </waypointNodes>
            <!--Optional:-->
            <srs>epsg:4326</srs>
            <!--Optional:-->
            <method>time</method>
            <!--Optional:-->
            <format>STANDARD</format>
            <!--Optional:-->
            <tolerance>0.</tolerance>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <timeLine>
               <!--Zero or more repetitions:-->
               <timeLineItem></timeLineItem>
            </timeLine>
            <!--Optional:-->
```

```
            <snapMethod></snapMethod>
            <!--Optional:-->
            <avoidArea></avoidArea>
            <!--Optional:-->
            <computeOptions></computeOptions>
            <!--Optional:-->
            <timeOut></timeOut>
            <!--Optional:-->
            <configName></configName>
         </request>
      </sch:routeV5>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:routeV5Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <RouteResult>
            <status>OK</status>
            <distance>640 m</distance>
            <duration>0:02:06</duration>
            <distanceMeters>639.97</distanceMeters>
            <durationSeconds>126.88</durationSeconds>
            <bounds>-1.351448,47.446922;-1.345263,47.44848</bounds>
            <wktGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, [...] ,  -1.34529
 47.447993)</wktGeometry>
            <wktSimplifiedGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922,  [...] , -1.34529
 47.447993)</wktSimplifiedGeometry>
            <leg>
               <distance>233 m</distance>
               <duration>0:00:46</duration>
               <distanceMeters>233.1</distanceMeters>
               <durationSeconds>46.28</durationSeconds>
               <step>
                  <distance>139 m</distance>
                  <duration>0:00:29</duration>
                  <distanceMeters>138.59</distanceMeters>
                  <durationSeconds>29.28</durationSeconds>
                  <name>RUE LA FONTAINE BRUNEAU</name>
               </step>
               <step>
                  <distance>91 m</distance>
                  <duration>0:00:16</duration>
                  <distanceMeters>91.33</distanceMeters>
                  <durationSeconds>16.43</durationSeconds>
                  <navigationInstruction>FR</navigationInstruction>
                  <name>RUE DES SAULES</name>
               </step>
               <step>
                  <distance>3 m</distance>
                  <duration>0:00:00</duration>
                  <distanceMeters>3.18</distanceMeters>
                  <durationSeconds>0.57</durationSeconds>
                  <navigationInstruction>round_about_entry</navigationInstruction>
                  <name />
               </step>
            </leg>
            <leg>
               <distance>407 m</distance>
               <duration>0:01:20</duration>
               <distanceMeters>406.87</distanceMeters>
```

```
                <durationSeconds>80.6</durationSeconds>
                <step>
                   <distance>18 m</distance>
                   <duration>0:00:03</duration>
                   <distanceMeters>17.64</distanceMeters>
                   <durationSeconds>3.17</durationSeconds>
                   <name />
                </step>
                <step>
                   <distance>67 m</distance>
                   <duration>0:00:15</duration>
                   <distanceMeters>66.62</distanceMeters>
                   <durationSeconds>15.05</durationSeconds>
                   <navigationInstruction>round_about_exit</navigationInstruction>
                   <name>RUE DES FOURS</name>
                </step>
                <step>
                   <distance>36 m</distance>
                   <duration>0:00:08</duration>
                   <distanceMeters>35.74</distanceMeters>
                   <durationSeconds>8.57</durationSeconds>
                   <navigationInstruction>F</navigationInstruction>
                   <name>PLACE DE L'ÉGLISE</name>
                </step>
                <step>
                   <distance>72 m</distance>
                   <duration>0:00:15</duration>
                   <distanceMeters>72.25</distanceMeters>
                   <durationSeconds>15.24</durationSeconds>
                   <navigationInstruction>F</navigationInstruction>
                   <name>RUE DES PRESSOIRS</name>
                </step>
                <step>
                   <distance>26 m</distance>
                   <duration>0:00:04</duration>
                   <distanceMeters>26.02</distanceMeters>
                   <durationSeconds>4.68</durationSeconds>
                   <navigationInstruction>round_about_entry</navigationInstruction>
                   <name />
                </step>
                <step>
                   <distance>183 m</distance>
                   <duration>0:00:32</duration>
                   <distanceMeters>182.84</distanceMeters>
                   <durationSeconds>32.91</durationSeconds>
                   <navigationInstruction>round_about_exit</navigationInstruction>
                   <name>RUE DU BOURG DRAPÉ</name>
                </step>
                <step>
                   <distance>6 m</distance>
                   <duration>0:00:00</duration>
                   <distanceMeters>5.76</distanceMeters>
                   <durationSeconds>0.98</durationSeconds>
                   <navigationInstruction>FR</navigationInstruction>
                   <name>RUE DE LA CURE</name>
                </step>
             </leg>
             <srs>epsg:4326</srs>
             <carbonFootprint>0.0</carbonFootprint>
          </RouteResult>
       </ns2:routeV5Response>
    </soap:Body>
 </soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/route/v5.json?
origin=-1.351448,47.446923&destination=-1.34529,47.4479931&waypoints=-1.34981,47.44837
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/route/v5.json?
origin=-1.351448,47.446923&destination=-1.34529,47.4479931&waypoints=-1.34981,47.44837&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/route/v5.xml?
origin=-1.351448,47.446923&destination=-1.34529,47.4479931&waypoints=-1.34981,47.44837
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
    "message": null,
    "status": "OK",
    "distance": "640 m",
    "duration": "0:02:09",
    "distanceMeters": 639.95,
    "durationSeconds": 129.48,
    "bounds": "-1.351448,47.446922;-1.345263,47.44848",
    "wktGeometry": "LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])",
    "wktSimplifiedGeometry":"LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699,
 [...])",
    "legs": [
        {
            "distance": "233 m",
            "duration": "0:00:46",
            "distanceMeters": 233.09,
            "durationSeconds": 46.84,
            "steps": [
                {
                    "distance": "139 m",
                    "duration": "0:00:29",
                    "distanceMeters": 138.59,
                    "durationSeconds": 29.83,
                    "navigationInstruction": null,
                    "name": "RUE LA FONTAINE BRUNEAU",
                    "points": []
                },
                {
                    [...]
                }
            ]
        },
        {
            "distance": "407 m",
```

```
            "duration": "0:01:22",
            "distanceMeters": 406.86,
            "durationSeconds": 82.64,
            "steps": [
                {
                    "distance": "18 m",
                    "duration": "0:00:03",
                    "distanceMeters": 17.63,
                    "durationSeconds": 3.18,
                    "navigationInstruction": null,
                    "name": "",
                    "points": []
                },
                {
                   [...]
                }
            ]
        }
    ],
    "startDateTime": null,
    "finishDateTime": null,
    "srs": null,
    "originNode": null,
    "waypointNodes": null,
    "destinationNode": null,
    "carbonFootprint": 0.195038864105688
}
```


Format JSON-P


```
myCallback(
    {
        "message": null,
        "status": "OK",
        "distance": "640 m",
        "duration": "0:02:09",
        "distanceMeters": 639.95,
        "durationSeconds": 129.48,
        "bounds": "-1.351448,47.446922;-1.345263,47.44848",
        "wktGeometry": "LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])",
        "wktSimplifiedGeometry":"LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699,
 [...])",
        "legs": [
            {
                "distance": "233 m",
                "duration": "0:00:46",
                "distanceMeters": 233.09,
                "durationSeconds": 46.84,
                "steps": [
                    {
                        "distance": "139 m",
                        "duration": "0:00:29",
                        "distanceMeters": 138.59,
                        "durationSeconds": 29.83,
                        "navigationInstruction": null,
                        "name": "RUE LA FONTAINE BRUNEAU",
                        "points": []
                    },
                    {
                        [...]
                    }
                ]
            },
```

```
            {
                "distance": "407 m",
                "duration": "0:01:22",
                "distanceMeters": 406.86,
                "durationSeconds": 82.64,
                "steps": [
                    {
                        "distance": "18 m",
                        "duration": "0:00:03",
                        "distanceMeters": 17.63,
                        "durationSeconds": 3.18,
                        "navigationInstruction": null,
                        "name": "",
                        "points": []
                    },
                    {
                       [...]
                    }
                ]
            }
        ],
        "startDateTime": null,
        "finishDateTime": null,
        "srs": null,
        "originNode": null,
        "waypointNodes": null,
        "destinationNode": null,
        "carbonFootprint": 0.195038864105688
    }
);
```


Format XML


```
<?xml version="1.0" encoding="UTF-8"?>
<routeResultV5>
        <status>OK</status>
        <distance>640 m</distance>
        <duration>0:02:09</duration>
        <distanceMeters>639.95</distanceMeters>
        <durationSeconds>129.48</durationSeconds>
        <bounds>-1.351448,47.446922;-1.345263,47.44848</bounds>
        <wktGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
wktGeometry>
        <wktSimplifiedGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699,
 [...])</wktSimplifiedGeometry>
        <leg>
                <distance>233 m</distance>
                <duration>0:00:46</duration>
                <distanceMeters>233.09</distanceMeters>
                <durationSeconds>46.84</durationSeconds>
                <step>
                        <distance>139 m</distance>
                        <duration>0:00:29</duration>
                        <distanceMeters>138.59</distanceMeters>
                        <durationSeconds>29.83</durationSeconds>
                        <name>RUE LA FONTAINE BRUNEAU</name>
                </step>
                <step>
            [...]
                </step>
        </leg>
        <leg>
                <distance>407 m</distance>
```

```
                <duration>0:01:22</duration>
                <distanceMeters>406.86</distanceMeters>
                <durationSeconds>82.64</durationSeconds>
                <step>
                        <distance>18 m</distance>
                        <duration>0:00:03</duration>
                        <distanceMeters>17.63</distanceMeters>
                        <durationSeconds>3.18</durationSeconds>
                        <name />
                </step>
                <step>
                        [...]
                </step>
        </leg>
        <carbonFootprint>0.195038864105688</carbonFootprint>
</routeResultV5>
```

API JavaScript

Inclure la librairie JavaScript.

```
var routeCtrl = new GCUI.Control.Route();
routeCtrl.route({
    url:'http://<server>/<webapp>/api/lbs/route/v5.json',
    tolerance : 100,
    origin : new OpenLayers.LonLat(0.691012, 47.384813),
    destination : new OpenLayers.LonLat(0.691012, 47.384813),
    waypoints : [ new OpenLayers.LonLat(2.344408, 49.898798) ],
    callback : function(result, options) {
        console.log(result);
    }
});
```

La variable **`result`** est au format JSON décrit ci-dessus. La fonction **`callback`** passée en paramètre est appelée à la fin du calcul d'itinéraire.

Retours possibles

Cas d'un itinéraire trouvé (routeResultV5/status est OK)

```
<?xml version="1.0" encoding="UTF-8"?>
<routeResultV5>
        <status>OK</status>
        <distance>640 m</distance>
        <duration>0:02:09</duration>
        <distanceMeters>639.95</distanceMeters>
        <durationSeconds>129.48</durationSeconds>
        <bounds>-1.351448,47.446922;-1.345263,47.44848</bounds>
        <wktGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
wktGeometry>
        <wktSimplifiedGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699,
 [...])</wktSimplifiedGeometry>
        <leg>
                <distance>233 m</distance>
                <duration>0:00:46</duration>
                <distanceMeters>233.09</distanceMeters>
                <durationSeconds>46.84</durationSeconds>
                <step>
                        <distance>139 m</distance>
                        <duration>0:00:29</duration>
```

```
                        <distanceMeters>138.59</distanceMeters>
                        <durationSeconds>29.83</durationSeconds>
                        <name>RUE LA FONTAINE BRUNEAU</name>
                </step>
                <step>
            [...]
                </step>
        </leg>
        <leg>
                <distance>407 m</distance>
                <duration>0:01:22</duration>
                <distanceMeters>406.86</distanceMeters>
                <durationSeconds>82.64</durationSeconds>
                <step>
                        <distance>18 m</distance>
                        <duration>0:00:03</duration>
                        <distanceMeters>17.63</distanceMeters>
                        <durationSeconds>3.18</durationSeconds>
                        <name />
                </step>
                <step>
                        [...]
                </step>
        </leg>
        <carbonFootprint>0.195038864105688</carbonFootprint>
</routeResultV5>
```

Cas d'un oubli de spécification du point de départ ou d'arrivée (routeResultV5/status est ERROR)

```
<routeResultV5>
   <message>Origin and destination must be not null</message>
   <status>ERROR</status>
   <distanceMeters>0.0</distanceMeters>
   <durationSeconds>0.0</durationSeconds>
</routeResultV5>
```

Cas d'un mauvais formatage du point de départ, d'arrivée ou des étapes (routeResultV5/status est ERROR)

```
<routeResultV5>
   <message>Origin, destination and waypoints should be represented by a couple of coordinates</message>
   <status>ERROR</status>
   <distanceMeters>0.0</distanceMeters>
   <durationSeconds>0.0</durationSeconds>
</routeResultV5>
```

Cas d'un mauvais typage (serviceResult/status est ERROR)

```
<serviceResult>
        <message>NumberFormatException: For input string: "AAA"</message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'une erreur d'accrochage au graphe (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in route computation
Error in smartrouting
Failed to execute calculateRoute
```

```
com.geoconcept.smartrouting.SmartRoutingNativeException: failed to connect waypoint { 146.691012, 47.384813,
 0.000000 }
failed to connect waypoint { 146.691012, 47.384813, 0.000000 }</message>
   <status>ERROR</status>
</serviceResult>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
    <message>ServiceException: Error in route computation
Error in smartrouting
datasource is null</message>
    <status>ERROR</status>
</serviceResult>
```

V4

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
origin	Coordonnées du point de départ. Les coordonnées longitude et latitude sont séparées par la caractère ,	non	
originNode	Noeud de départ. Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
destination	Coordonnées du point d'arrivée. Les coordonnées longitude et latitude sont séparées par la caractère ,	non	
destinationNode	Noeud d'arrivée. Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
waypoints	Coordonnées des étapes. Chaque couple de coordonnées d'une étape est encadré par la balise <waypoint> Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	
waypointNodes	Noeuds d'étapes. Chaque couple de coordonnées d'une étape est encadré par la balise <waypointNode> Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
format	- standard : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt / géométrie simplifiée au format wkt / liste des segments concaténés (sans géométries) - extended : résultat = résumé de l'itinéraire / bornes / géométrie simplifiée au format wkt / liste des segments non-concaténés (avec géométries) - summary : résultat = résumé de l'itinéraire - geometry : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt - simplifiedgeometry : résultat = résumé de l'itinéraire / bornes / géométrie simplifiée au format wkt - geometries : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt / géométrie simplifiée au format wkt	oui	standard

paramètre	description	optionnel	défaut
	- brief : résultat = résumé de l'itinéraire / Liste des segments avec durée et distance - standardext : résultat = résumé de l'itinéraire / bornes / liste des segments concaténés avec durée et distance et géométrie. - node : résultat = résumé de l'itinéraire + Id des noeuds accrochés		
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
tolerance	Distance de tolérance (en mètre) de simplification de la géométrie.	oui	
graphName	Nom du graphe à utiliser Ce paramètre est omis si le paramètre configName est utilisé.	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris. Attention le caractère + peut être mal interprété par les navigateurs, dans ce cas, il faut le remplacer par *%2B*.	oui	
profileId	Identifiant du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileName	Profil du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère , ou *;* (Exemple : Toll, Tunnel, Bridge)	oui	
timeLine	Liste de durées intermédiaires pour calculer des positions le long du trajet. Exprimées en secondes, séparés par le caractère *;*	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction - nodes : Accrochage direct aux noeuds fourni par les paramètres originNode, destinationNode and waypointNodes	oui	standard
avoidArea	Zone de transit interdit au format WKT (POLYGON ou MULTIPOLYGON) dans la projection (paramètre srs) demandée Exemple en wgs84 : `POLYGON ((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689))` - `MULTIPOLYGON (((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689)), ((-1.558639 47.218144, -1.558639 47.218193, [...] -1.558639 47.218144)))` Attention les géométries WKT doivent être fermées.	oui	
configName	Nom de la configuration à utiliser (défini dans Geoconcept Web - Administration / Outils / Définitions des graphes) Il remplace l'usage de graphName, profileId et profileName	oui	

En sortie

Itinéraire (routeResultV4)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l'itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)

paramètre	type	min/max	description
duration	string	0/1	Durée totale de l'itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	double	0/1	Distance totale de l'itinéraire en mètres.
durationSeconds	double	0/1	Durée totale de l'itinéraire en secondes.
bounds	string	0/1	Bornes (BoundingBox) de la géométrie de l'itinéraire.
wktGeometry	string	0/1	Géométrie de l'itinéraire au format WKT.
wktSimplified	string	0/1	Géométrie simplifiée de l'itinéraire au format WKT.
leg (ou legs en JSON / JSON-P)	subRouteV4 (ou array en JSON / JSON-P)	0/illimité	Liste des portions d'itinéraires.
startDateTime	string	0/1	Date et heure de départ (format : ISO8601 sans code zone : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 pour un départ le 21 janvier 2014, à 9h à Paris
finishDateTime	string	0/1	Date et heure d'arrivée (format : ISO8601 sans code zone : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 pour une arrivée le 21 janvier 2014, à 9h à Paris
timeLineItem	timeLineItemV4 (ou array en JSON / JSON-P)	0/illimité	Liste des positions calculées.
srs	string	0/1	projection passée en entrée (code EPSG comme epsg:4326 ou wgs84)
originNode	string	0/1	Identifiant du noeud de départ (renseigné si format=NODE).
destinationNode	string	0/1	Identifiant du noeud d'arrivée (renseigné si format=NODE).
waypointNodes	array de waypointNodes (string)	0/illimité	Identifiants des noeuds d'étapes (renseigné si format=NODE).
carbonFootprint	double	0/1	Empreinte carbone (émission CO2 en kilogrammes)

Portion d'itinéraire (subRouteV4)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l'itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l'itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	double	0/1	Distance de la portion de l'itinéraire en mètres.
durationSeconds	double	0/1	Durée de la portion de l'itinéraire en secondes.
step ou (ou steps en JSON / JSON-P)	segmentV4 (ou array en JSON / JSON-P)	0/illimité	Liste des segments composants la portion d'itinéraire.

Segment d'itinéraire (segmentV4)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l'itinéraire, formaté : - xx.xx Km

paramètre	type	min/max	description
			- xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l'itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	double	0/1	Distance du segment de l'itinéraire en mètres.
durationSeconds	double	0/1	Durée du segment de l'itinéraire en secondes.
navigationInstruction	string	0/1	Code d'instruction de navigation : - F : Tout droit - FR: Tourner légèrement à droite - FL: Tourner légèrement à gauche - R: Tourner à droite - L: Tourner à gauche - BR: Tourner fortement à droite - BL: Tourner fortement à gauche - B: Demi-tour - round_about_entry: S'engager sur le rond-point - round_about_exit: Sortir du rond-point
name	string	0/1	Nom de rue du segment.
point	string	0/illimité	Liste de coordonnées séparées par la caractère ,

Position d'itinéraire (timeLineItemV4)

paramètre	type	min/max	description
durationSeconds	double	0/1	Durée de l'itinéraire en secondes jusquà cette position.
message	string	0/1	Message d'erreur pour cette position.
status	string	0/1	Status de cette position.
distanceMeters	double	0/1	Distance de l'itinéraire en mètres jusquà cette position.
location	string	0/1	Coordonnées de la position.

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/routeService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:routeV4>
         <!--Optional:-->
         <request>
            <origin>
               <x>-1.351448</x>
               <y>47.446923</y>
            </origin>
            <!--Optional:-->
            <originNode></originNode>
            <destination>
               <x>-1.34529</x>
               <y>47.4479931</y>
```

```
            </destination>
            <!--Optional:-->
            <destinationNode></destinationNode>
            <!--Optional:-->
            <waypoints>
               <!--Zero or more repetitions:-->
               <waypoint>
                  <x>-1.34981</x>
                  <y>47.44837</y>
               </waypoint>
            </waypoints>
            <!--Optional:-->
            <waypointNodes>
               <!--Zero or more repetitions:-->
               <waypointNode></waypointNode>
            </waypointNodes>
            <!--Optional:-->
            <srs>epsg:4326</srs>
            <!--Optional:-->
            <method>time</method>
            <!--Optional:-->
            <format>STANDARD</format>
            <!--Optional:-->
            <tolerance>0.</tolerance>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <timeLine>
               <!--Zero or more repetitions:-->
               <timeLineItem></timeLineItem>
            </timeLine>
            <!--Optional:-->
            <snapMethod></snapMethod>
            <!--Optional:-->
            <avoidArea></avoidArea>
            <!--Optional:-->
            <configName></configName>
         </request>
      </sch:routeV4>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:routeV4Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <RouteResult>
            <status>OK</status>
            <distance>640 m</distance>
            <duration>0:02:06</duration>
            <distanceMeters>639.95</distanceMeters>
```

```
            <durationSeconds>126.88</durationSeconds>
            <bounds>-1.351448,47.446922;-1.345263,47.44848</bounds>
            <wktGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922,  [...] )</wktGeometry>
            <wktSimplifiedGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922,  [...] )</
wktSimplifiedGeometry>
            <leg>
               <distance>233 m</distance>
               <duration>0:00:46</duration>
               <distanceMeters>233.09</distanceMeters>
               <durationSeconds>46.28</durationSeconds>
               <step>
                  <distance>139 m</distance>
                  <duration>0:00:29</duration>
                  <distanceMeters>138.59</distanceMeters>
                  <durationSeconds>29.28</durationSeconds>
                  <name>RUE LA FONTAINE BRUNEAU</name>
               </step>
               <step>
                  <distance>91 m</distance>
                  <duration>0:00:16</duration>
                  <distanceMeters>91.33</distanceMeters>
                  <durationSeconds>16.43</durationSeconds>
                  <navigationInstruction>FR</navigationInstruction>
                  <name>RUE DES SAULES</name>
               </step>
               <step>
                  <distance>3 m</distance>
                  <duration>0:00:00</duration>
                  <distanceMeters>3.17</distanceMeters>
                  <durationSeconds>0.57</durationSeconds>
                  <navigationInstruction>round_about_entry</navigationInstruction>
                  <name/>
               </step>
            </leg>
            <leg>
               <distance>407 m</distance>
               <duration>0:01:20</duration>
               <distanceMeters>406.86</distanceMeters>
               <durationSeconds>80.6</durationSeconds>
               <step>
                  <distance>18 m</distance>
                  <duration>0:00:03</duration>
                  <distanceMeters>17.63</distanceMeters>
                  <durationSeconds>3.17</durationSeconds>
                  <name/>
               </step>
               <step>
                  <distance>67 m</distance>
                  <duration>0:00:15</duration>
                  <distanceMeters>66.62</distanceMeters>
                  <durationSeconds>15.05</durationSeconds>
                  <navigationInstruction>round_about_exit</navigationInstruction>
                  <name>RUE DES FOURS</name>
               </step>
               <step>
                  <distance>36 m</distance>
                  <duration>0:00:08</duration>
                  <distanceMeters>35.74</distanceMeters>
                  <durationSeconds>8.57</durationSeconds>
                  <navigationInstruction>F</navigationInstruction>
                  <name>PLACE DE L'ÉGLISE</name>
               </step>
               <step>
```

```
                  <distance>72 m</distance>
                  <duration>0:00:15</duration>
                  <distanceMeters>72.25</distanceMeters>
                  <durationSeconds>15.24</durationSeconds>
                  <navigationInstruction>F</navigationInstruction>
                  <name>RUE DES PRESSOIRS</name>
               </step>
               <step>
                  <distance>26 m</distance>
                  <duration>0:00:04</duration>
                  <distanceMeters>26.02</distanceMeters>
                  <durationSeconds>4.68</durationSeconds>
                  <navigationInstruction>round_about_entry</navigationInstruction>
                  <name/>
               </step>
               <step>
                  <distance>183 m</distance>
                  <duration>0:00:32</duration>
                  <distanceMeters>182.84</distanceMeters>
                  <durationSeconds>32.91</durationSeconds>
                  <navigationInstruction>round_about_exit</navigationInstruction>
                  <name>RUE DU BOURG DRAPÉ</name>
               </step>
               <step>
                  <distance>6 m</distance>
                  <duration>0:00:00</duration>
                  <distanceMeters>5.76</distanceMeters>
                  <durationSeconds>0.98</durationSeconds>
                  <navigationInstruction>FR</navigationInstruction>
                  <name>RUE DE LA CURE</name>
               </step>
            </leg>
            <timeLineItem>
               <durationSeconds>0.0</durationSeconds>
               <status>OK</status>
               <distanceMeters>0.0</distanceMeters>
               <location>-1.351448,47.446923</location>
            </timeLineItem>
            <srs>epsg:4326</srs>
            <carbonFootprint>0.0</carbonFootprint>
         </RouteResult>
      </ns2:routeV4Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/route/v4.json?
origin=-1.351448,47.446923&destination=-1.34529,47.4479931&waypoints=-1.34981,47.44837
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/route/v4.json?
origin=-1.351448,47.446923&destination=-1.34529,47.4479931&waypoints=-1.34981,47.44837&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/route/v4.xml?
origin=-1.351448,47.446923&destination=-1.34529,47.4479931&waypoints=-1.34981,47.44837
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON


```
{
    "message": null,
    "status": "OK",
    "distance": "640 m",
    "duration": "0:02:09",
    "distanceMeters": 639.95,
    "durationSeconds": 129.48,
    "bounds": "-1.351448,47.446922;-1.345263,47.44848",
    "wktGeometry": "LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])",
    "wktSimplifiedGeometry":"LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699,
 [...])",
    "legs": [
        {
            "distance": "233 m",
            "duration": "0:00:46",
            "distanceMeters": 233.09,
            "durationSeconds": 46.84,
            "steps": [
                {
                    "distance": "139 m",
                    "duration": "0:00:29",
                    "distanceMeters": 138.59,
                    "durationSeconds": 29.83,
                    "navigationInstruction": null,
                    "name": "RUE LA FONTAINE BRUNEAU",
                    "points": []
                },
                {
                    [...]
                }
            ]
        },
        {
            "distance": "407 m",
            "duration": "0:01:22",
            "distanceMeters": 406.86,
            "durationSeconds": 82.64,
            "steps": [
                {
                    "distance": "18 m",
                    "duration": "0:00:03",
                    "distanceMeters": 17.63,
                    "durationSeconds": 3.18,
                    "navigationInstruction": null,
                    "name": "",
                    "points": []
                },
                {
                   [...]
                }
            ]
```

```
        }
    ],
    "startDateTime": null,
    "finishDateTime": null,
    "srs": null,
    "originNode": null,
    "waypointNodes": null,
    "destinationNode": null,
    "carbonFootprint": 0.195038864105688
}
```

Format JSON-P

```
myCallback(
    {
        "message": null,
        "status": "OK",
        "distance": "640 m",
        "duration": "0:02:09",
        "distanceMeters": 639.95,
        "durationSeconds": 129.48,
        "bounds": "-1.351448,47.446922;-1.345263,47.44848",
        "wktGeometry": "LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])",
        "wktSimplifiedGeometry":"LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699,
 [...])",
        "legs": [
            {
                "distance": "233 m",
                "duration": "0:00:46",
                "distanceMeters": 233.09,
                "durationSeconds": 46.84,
                "steps": [
                    {
                        "distance": "139 m",
                        "duration": "0:00:29",
                        "distanceMeters": 138.59,
                        "durationSeconds": 29.83,
                        "navigationInstruction": null,
                        "name": "RUE LA FONTAINE BRUNEAU",
                        "points": []
                    },
                    {
                        [...]
                    }
                ]
            },
            {
                "distance": "407 m",
                "duration": "0:01:22",
                "distanceMeters": 406.86,
                "durationSeconds": 82.64,
                "steps": [
                    {
                        "distance": "18 m",
                        "duration": "0:00:03",
                        "distanceMeters": 17.63,
                        "durationSeconds": 3.18,
                        "navigationInstruction": null,
                        "name": "",
                        "points": []
                    },
                    {
                       [...]
```

```
                    }
                ]
            }
        ],
        "startDateTime": null,
        "finishDateTime": null,
        "srs": null,
        "originNode": null,
        "waypointNodes": null,
        "destinationNode": null,
        "carbonFootprint": 0.195038864105688
    }
);
```

Format XML

```
<?xml version="1.0" encoding="UTF-8"?>
<routeResultV4>
        <status>OK</status>
        <distance>640 m</distance>
        <duration>0:02:09</duration>
        <distanceMeters>639.95</distanceMeters>
        <durationSeconds>129.48</durationSeconds>
        <bounds>-1.351448,47.446922;-1.345263,47.44848</bounds>
        <wktGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
wktGeometry>
        <wktSimplifiedGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699,
 [...])</wktSimplifiedGeometry>
        <leg>
                <distance>233 m</distance>
                <duration>0:00:46</duration>
                <distanceMeters>233.09</distanceMeters>
                <durationSeconds>46.84</durationSeconds>
                <step>
                        <distance>139 m</distance>
                        <duration>0:00:29</duration>
                        <distanceMeters>138.59</distanceMeters>
                        <durationSeconds>29.83</durationSeconds>
                        <name>RUE LA FONTAINE BRUNEAU</name>
                </step>
                <step>
            [...]
                </step>
        </leg>
        <leg>
                <distance>407 m</distance>
                <duration>0:01:22</duration>
                <distanceMeters>406.86</distanceMeters>
                <durationSeconds>82.64</durationSeconds>
                <step>
                        <distance>18 m</distance>
                        <duration>0:00:03</duration>
                        <distanceMeters>17.63</distanceMeters>
                        <durationSeconds>3.18</durationSeconds>
                        <name />
                </step>
                <step>
                        [...]
                </step>
        </leg>
        <carbonFootprint>0.195038864105688</carbonFootprint>
</routeResultV4>
```

API JavaScript

Inclure la librairie JavaScript.

```
var routeCtrl = new GCUI.Control.Route();
routeCtrl.route({
    url:'http://<server>/<webapp>/api/lbs/route/v4.json',
    tolerance : 100,
    origin : new OpenLayers.LonLat(0.691012, 47.384813),
    destination : new OpenLayers.LonLat(0.691012, 47.384813),
    waypoints : [ new OpenLayers.LonLat(2.344408, 49.898798) ],
    callback : function(result, options) {
        console.log(result);
    }
});
```

La variable **`result`** est au format JSON décrit ci-dessus. La fonction **`callback`** passée en paramètre est appelée à la fin du calcul d'itinéraire.

Retours possibles

Cas d'un itinéraire trouvé (routeResultV4/status est OK)

```
<?xml version="1.0" encoding="UTF-8"?>
<routeResultV4>
        <status>OK</status>
        <distance>640 m</distance>
        <duration>0:02:09</duration>
        <distanceMeters>639.95</distanceMeters>
        <durationSeconds>129.48</durationSeconds>
        <bounds>-1.351448,47.446922;-1.345263,47.44848</bounds>
        <wktGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
wktGeometry>
        <wktSimplifiedGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699,
 [...])</wktSimplifiedGeometry>
        <leg>
                <distance>233 m</distance>
                <duration>0:00:46</duration>
                <distanceMeters>233.09</distanceMeters>
                <durationSeconds>46.84</durationSeconds>
                <step>
                        <distance>139 m</distance>
                        <duration>0:00:29</duration>
                        <distanceMeters>138.59</distanceMeters>
                        <durationSeconds>29.83</durationSeconds>
                        <name>RUE LA FONTAINE BRUNEAU</name>
                </step>
                <step>
            [...]
                </step>
        </leg>
        <leg>
                <distance>407 m</distance>
                <duration>0:01:22</duration>
                <distanceMeters>406.86</distanceMeters>
                <durationSeconds>82.64</durationSeconds>
                <step>
                        <distance>18 m</distance>
                        <duration>0:00:03</duration>
                        <distanceMeters>17.63</distanceMeters>
```

```
                        <durationSeconds>3.18</durationSeconds>
                        <name />
                </step>
                <step>
                        [...]
                </step>
        </leg>
        <carbonFootprint>0.195038864105688</carbonFootprint>
</routeResultV4>
```

Cas d'un oubli de spécification du point de départ ou d'arrivée (routeResultV4/status est ERROR)

```
<routeResultV4>
   <message>Origin and destination must be not null</message>
   <status>ERROR</status>
   <distanceMeters>0.0</distanceMeters>
   <durationSeconds>0.0</durationSeconds>
</routeResultV4>
```

Cas d'un mauvais formatage du point de départ, d'arrivée ou des étapes (routeResultV4/status est ERROR)

```
<routeResultV4>
   <message>Origin, destination and waypoints should be represented by a couple of coordinates</message>
   <status>ERROR</status>
   <distanceMeters>0.0</distanceMeters>
   <durationSeconds>0.0</durationSeconds>
</routeResultV4>
```

Cas d'un mauvais typage (serviceResult/status est ERROR)

```
<serviceResult>
        <message>NumberFormatException: For input string: "AAA"</message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'une erreur d'accrochage au graphe (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in route computation
Error in smartrouting
Failed to execute calculateRoute
com.geoconcept.smartrouting.SmartRoutingNativeException: failed to connect waypoint { 146.691012, 47.384813,
 0.000000 }
failed to connect waypoint { 146.691012, 47.384813, 0.000000 }</message>
   <status>ERROR</status>
</serviceResult>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
    <message>ServiceException: Error in route computation
Error in smartrouting
datasource is null</message>
    <status>ERROR</status>
</serviceResult>
```

V3

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
origin	Coordonnées du point de départ. Les coordonnées longitude et latitude sont séparées par la caractère ,	non	
destination	Coordonnées du point d'arrivée. Les coordonnées longitude et latitude sont séparées par la caractère ,	non	
waypoints	Coordonnées des étapes. Chaque couple de coordonnées d'une étape est encadré par la balise <waypoint> Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
format	- standard : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt / géométrie simplifiée au format wkt / liste des segments concaténés (sans géométries) - extended : résultat = résumé de l'itinéraire / bornes / géométrie simplifiée au format wkt / liste des segments non-concaténés (avec géométries) - summary : résultat = résumé de l'itinéraire - geometry : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt - simplifiedgeometry : résultat = résumé de l'itinéraire / bornes / géométrie simplifiée au format wkt - geometries : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt / géométrie simplifiée au format wkt - brief : résultat = résumé de l'itinéraire / Liste des segments avec durée et distance - standardext : résultat = résumé de l'itinéraire / bornes / liste des segments concaténés avec durée et distance et géométrie.	oui	standard
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
tolerance	Distance de tolérance (en mètre) de simplification de la géométrie.	oui	
graphName	Nom du graphe à utiliser	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris	oui	
profileId	Identifiant du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
profileName	Profil du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère , ou ; (Exemple : Toll, Tunnel, Bridge)	oui	
timeLine	Liste de durées intermédiaires pour calculer des positions le long du trajet. Exprimées en secondes, séparés par le caractère ;	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction	oui	standard

En sortie

Itinéraire (routeResultV3)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l’itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l’itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	double	0/1	Distance totale de l’itinéraire en mètres.
durationSeconds	double	0/1	Durée totale de l’itinéraire en secondes.
bounds	string	0/1	Bornes (BoundingBox) de la géométrie de l’itinéraire.
wktGeometry	string	0/1	Géométrie de l’itinéraire au format WKT.
wktSimplified	string	0/1	Géométrie simplifiée de l’itinéraire au format WKT.
leg (ou legs en JSON / JSON-P)	subRouteV3 (ou array en JSON / JSON-P)	0/illimité	Liste des portions d’itinéraires.
startDateTime	string	0/1	Date et heure de départ (format : ISO8601 sans code zone : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 pour un départ le 21 janvier 2014, à 9h à Paris
finishDateTime	string	0/1	Date et heure d’arrivée (format : ISO8601 sans code zone : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 pour une arrivée le 21 janvier 2014, à 9h à Paris
timeLineItem	timeLineItemV3 (ou array en JSON / JSON-P)	0/illimité	Liste des positions calculées.
srs	string	0/1	projection passée en entrée (code EPSG comme epsg:4326 ou wgs84)

Portion d’itinéraire (subRouteV3)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l’itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l’itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	double	0/1	Distance de la portion de l’itinéraire en mètres.
durationSeconds	double	0/1	Durée de la portion de l’itinéraire en secondes.
step ou (ou steps en JSON / JSON-P)	segmentV3 (ou array en JSON / JSON-P)	0/illimité	Liste des segments composants la portion d’itinéraire.

Segment d’itinéraire (segmentV3)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l’itinéraire, formaté : - xx.xx Km

paramètre	type	min/max	description
			- xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l'itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	double	0/1	Distance du segment de l'itinéraire en mètres.
durationSeconds	double	0/1	Durée du segment de l'itinéraire en secondes.
navigationInstruction	string	0/1	Code d'instruction de navigation : - F : Tout droit - FR: Tourner légèrement à droite - FL: Tourner légèrement à gauche - R: Tourner à droite - L: Tourner à gauche - BR: Tourner fortement à droite - BL: Tourner fortement à gauche - B: Demi-tour - round_about_entry: S'engager sur le rond-point - round_about_exit: Sortir du rond-point
name	string	0/1	Nom de rue du segment.
point	string	0/illimité	Liste de coordonnées séparées par la caractère ,

Position d'itinéraire (timeLineItemV3)

paramètre	type	min/max	description
durationSeconds	double	0/1	Durée de l'itinéraire en secondes jusquà cette position.
message	string	0/1	Message d'erreur pour cette position.
status	string	0/1	Status de cette position.
distanceMeters	double	0/1	Distance de l'itinéraire en mètres jusquà cette position.
location	string	0/1	Coordonnées de la position.

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/routeService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:routeV3>
         <!--Optional:-->
         <request>
            <origin>
               <x>-1.351448</x>
               <y>47.446923</y>
            </origin>
            <destination>
               <x>-1.34529</x>
               <y>47.4479931</y>
            </destination>
            <!--Optional:-->
```

```
            <waypoints>
               <!--Zero or more repetitions:-->
               <waypoint>
                  <x>-1.34981</x>
                  <y>47.44837</y>
               </waypoint>
            </waypoints>
            <!--Optional:-->
            <srs>epsg:4326</srs>
            <!--Optional:-->
            <method>time</method>
            <!--Optional:-->
            <format>STANDARD</format>
            <!--Optional:-->
            <tolerance>0.</tolerance>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <startDateTime>2015-07-29T09:00:00.000+01:00</startDateTime>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <timeLine>
               <!--Zero or more repetitions:-->
               <timeLineItem>5</timeLineItem>
               <timeLineItem>15</timeLineItem>
            </timeLine>
            <!--Optional:-->
            <snapMethod></snapMethod>
         </request>
      </sch:routeV3>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:routeV3Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <RouteResult>
            <status>OK</status>
            <distance>640 m</distance>
            <duration>0:02:06</duration>
            <distanceMeters>639.95</distanceMeters>
            <durationSeconds>126.88</durationSeconds>
            <bounds>-1.351448,47.446922;-1.345263,47.44848</bounds>
            <wktGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922,  [...] )</wktGeometry>
            <wktSimplifiedGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922,  [...] )</
wktSimplifiedGeometry>
            <leg>
               <distance>233 m</distance>
               <duration>0:00:46</duration>
               <distanceMeters>233.09</distanceMeters>
               <durationSeconds>46.28</durationSeconds>
               <step>
                  <distance>139 m</distance>
```

```
                  <duration>0:00:29</duration>
                  <distanceMeters>138.59</distanceMeters>
                  <durationSeconds>29.28</durationSeconds>
                  <name>RUE LA FONTAINE BRUNEAU</name>
               </step>
               <step>
                  <distance>91 m</distance>
                  <duration>0:00:16</duration>
                  <distanceMeters>91.33</distanceMeters>
                  <durationSeconds>16.43</durationSeconds>
                  <navigationInstruction>FR</navigationInstruction>
                  <name>RUE DES SAULES</name>
               </step>
               <step>
                  <distance>3 m</distance>
                  <duration>0:00:00</duration>
                  <distanceMeters>3.17</distanceMeters>
                  <durationSeconds>0.57</durationSeconds>
                  <navigationInstruction>round_about_entry</navigationInstruction>
                  <name/>
               </step>
            </leg>
            <leg>
               <distance>407 m</distance>
               <duration>0:01:20</duration>
               <distanceMeters>406.86</distanceMeters>
               <durationSeconds>80.6</durationSeconds>
               <step>
                  <distance>18 m</distance>
                  <duration>0:00:03</duration>
                  <distanceMeters>17.63</distanceMeters>
                  <durationSeconds>3.17</durationSeconds>
                  <name/>
               </step>
               <step>
                  <distance>67 m</distance>
                  <duration>0:00:15</duration>
                  <distanceMeters>66.62</distanceMeters>
                  <durationSeconds>15.05</durationSeconds>
                  <navigationInstruction>round_about_exit</navigationInstruction>
                  <name>RUE DES FOURS</name>
               </step>
               <step>
                  <distance>36 m</distance>
                  <duration>0:00:08</duration>
                  <distanceMeters>35.74</distanceMeters>
                  <durationSeconds>8.57</durationSeconds>
                  <navigationInstruction>F</navigationInstruction>
                  <name>PLACE DE L'ÉGLISE</name>
               </step>
               <step>
                  <distance>72 m</distance>
                  <duration>0:00:15</duration>
                  <distanceMeters>72.25</distanceMeters>
                  <durationSeconds>15.24</durationSeconds>
                  <navigationInstruction>F</navigationInstruction>
                  <name>RUE DES PRESSOIRS</name>
               </step>
               <step>
                  <distance>26 m</distance>
                  <duration>0:00:04</duration>
                  <distanceMeters>26.02</distanceMeters>
                  <durationSeconds>4.68</durationSeconds>
```

```
                  <navigationInstruction>round_about_entry</navigationInstruction>
                  <name/>
               </step>
               <step>
                  <distance>183 m</distance>
                  <duration>0:00:32</duration>
                  <distanceMeters>182.84</distanceMeters>
                  <durationSeconds>32.91</durationSeconds>
                  <navigationInstruction>round_about_exit</navigationInstruction>
                  <name>RUE DU BOURG DRAPÉ</name>
               </step>
               <step>
                  <distance>6 m</distance>
                  <duration>0:00:00</duration>
                  <distanceMeters>5.76</distanceMeters>
                  <durationSeconds>0.98</durationSeconds>
                  <navigationInstruction>FR</navigationInstruction>
                  <name>RUE DE LA CURE</name>
               </step>
            </leg>
            <startDateTime>2015-07-29T10:00:00.000+02:00</startDateTime>
            <finishDateTime>2015-07-29T10:02:06.880+02:00</finishDateTime>
            <timeLineItem>
               <durationSeconds>5.0</durationSeconds>
               <status>OK</status>
               <distanceMeters>23.66</distanceMeters>
               <location>-1.35137,47.447123</location>
            </timeLineItem>
            <timeLineItem>
               <durationSeconds>15.0</durationSeconds>
               <status>OK</status>
               <distanceMeters>71.0</distanceMeters>
               <location>-1.351204,47.447533</location>
            </timeLineItem>
            <srs>epsg:4326</srs>
         </RouteResult>
      </ns2:routeV3Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/route/v3.json?
origin=-1.351448,47.446923&destination=-1.34529,47.4479931&waypoints=-1.34981,47.44837
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/route/v3.json?
origin=-1.351448,47.446923&destination=-1.34529,47.4479931&waypoints=-1.34981,47.44837&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/route/v3.xml?
origin=-1.351448,47.446923&destination=-1.34529,47.4479931&waypoints=-1.34981,47.44837
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON


```
{
   "message":null,
   "status":"OK",
   "distance":"640 m",
   "duration":"0:01:25",
   "distanceMeters":640.0,
   "durationSeconds":85.64,
   "bounds":"-1.351448,47.446922;-1.345263,47.44848",
   "wktGeometry":"LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])",
   "wktSimplifiedGeometry":"LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699,
 [...])",
   "legs":[
      {
         "distance":"233 m",
         "duration":"0:00:36",
         "distanceMeters":233.1,
         "durationSeconds":36.33,
         "steps":[
            {
               "distance":"139 m",
               "duration":"0:00:25",
               "distanceMeters":138.59,
               "durationSeconds":25.0,
               "navigationInstruction":null,
               "name":"RUE LA FONTAINE BRUNEAU",
               "points":[

               ]
            },
            {
                [...]
            }
         ]
      },
      {
         "distance":"407 m",
         "duration":"0:00:49",
         "distanceMeters":406.90000000000003,
         "durationSeconds":49.31,
         "steps":[
            {
               "distance":"18 m",
               "duration":"0:00:02",
               "distanceMeters":17.62,
               "durationSeconds":2.11,
               "navigationInstruction":null,
               "name":"",
               "points":[

               ]
            },
            {
               [...]
            }
         ]
      }
   ],
```

```
   "startDateTime":null,
   "finishDateTime":null,
   "srs":"epsg:4326"
}
```

Format JSON-P

```
myCallback(
    {
       "message":null,
       "status":"OK",
       "distance":"640 m",
       "duration":"0:01:25",
       "distanceMeters":640.0,
       "durationSeconds":85.64,
       "bounds":"-1.351448,47.446922;-1.345263,47.44848",
       "wktGeometry":"LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])",
       "wktSimplifiedGeometry":"LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699,
 [...])",
       "legs":[
          {
             "distance":"233 m",
             "duration":"0:00:36",
             "distanceMeters":233.1,
             "durationSeconds":36.33,
             "steps":[
                {
                   "distance":"139 m",
                   "duration":"0:00:25",
                   "distanceMeters":138.59,
                   "durationSeconds":25.0,
                   "navigationInstruction":null,
                   "name":"RUE LA FONTAINE BRUNEAU",
                   "points":[

                   ]
                },
                {
                    [...]
                }
             ]
          },
          {
             "distance":"407 m",
             "duration":"0:00:49",
             "distanceMeters":406.90000000000003,
             "durationSeconds":49.31,
             "steps":[
                {
                   "distance":"18 m",
                   "duration":"0:00:02",
                   "distanceMeters":17.62,
                   "durationSeconds":2.11,
                   "navigationInstruction":null,
                   "name":"",
                   "points":[

                   ]
                },
                {
                   [...]
                }
             ]
```

```
          }
       ],
       "startDateTime":null,
       "finishDateTime":null,
       "srs":"epsg:4326"
    }
);
```

Format XML

```
<?xml version="1.0" encoding="UTF-8"?>
<routeResultV3>
   <status>OK</status>
   <distance>640 m</distance>
   <duration>0:01:25</duration>
   <distanceMeters>640.0</distanceMeters>
   <durationSeconds>85.64</durationSeconds>
   <bounds>-1.351448,47.446922;-1.345263,47.44848</bounds>
   <wktGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
wktGeometry>
   <wktSimplifiedGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
wktSimplifiedGeometry>
   <leg>
      <distance>233 m</distance>
      <duration>0:00:36</duration>
      <distanceMeters>233.1</distanceMeters>
      <durationSeconds>36.33</durationSeconds>
      <step>
         <distance>139 m</distance>
         <duration>0:00:25</duration>
         <distanceMeters>138.59</distanceMeters>
         <durationSeconds>25.0</durationSeconds>
         <name>RUE LA FONTAINE BRUNEAU</name>
      </step>
      <step>
            [...]
      </step>
   </leg>
   <leg>
      <distance>407 m</distance>
      <duration>0:00:49</duration>
      <distanceMeters>406.90000000000003</distanceMeters>
      <durationSeconds>49.31</durationSeconds>
      <step>
         <distance>18 m</distance>
         <duration>0:00:02</duration>
         <distanceMeters>17.62</distanceMeters>
         <durationSeconds>2.11</durationSeconds>
         <name />
      </step>
      <step>
            [...]
      </step>
   </leg>
   <srs>epsg:4326</srs>
</routeResultV3>
```

API JavaScript

Inclure la librairie JavaScript.

```
var routeCtrl = new GCUI.Control.Route();
```

```
routeCtrl.route({
    url:'http://<server>/<webapp>/api/lbs/route/v3.json',
    tolerance : 100,
    origin : new OpenLayers.LonLat(0.691012, 47.384813),
    destination : new OpenLayers.LonLat(0.691012, 47.384813),
    waypoints : [ new OpenLayers.LonLat(2.344408, 49.898798) ],
    callback : function(result, options) {
        console.log(result);
    }
});
```

La variable `result` est au format JSON décrit ci-dessus. La fonction `callback` passée en paramètre est appelée à la fin du calcul d'itinéraire.

Retours possibles

Cas d'un itinéraire trouvé (routeResultV3/status est OK)

```
<?xml version="1.0" encoding="UTF-8"?>
<routeResultV3>
   <status>OK</status>
   <distance>640 m</distance>
   <duration>0:01:25</duration>
   <distanceMeters>640.0</distanceMeters>
   <durationSeconds>85.64</durationSeconds>
   <bounds>-1.351448,47.446922;-1.345263,47.44848</bounds>
   <wktGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
wktGeometry>
   <wktSimplifiedGeometry>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
wktSimplifiedGeometry>
   <leg>
      <distance>233 m</distance>
      <duration>0:00:36</duration>
      <distanceMeters>233.1</distanceMeters>
      <durationSeconds>36.33</durationSeconds>
      <step>
         <distance>139 m</distance>
         <duration>0:00:25</duration>
         <distanceMeters>138.59</distanceMeters>
         <durationSeconds>25.0</durationSeconds>
         <name>RUE LA FONTAINE BRUNEAU</name>
      </step>
      <step>
            [...]
      </step>
   </leg>
   <leg>
      <distance>407 m</distance>
      <duration>0:00:49</duration>
      <distanceMeters>406.90000000000003</distanceMeters>
      <durationSeconds>49.31</durationSeconds>
      <step>
         <distance>18 m</distance>
         <duration>0:00:02</duration>
         <distanceMeters>17.62</distanceMeters>
         <durationSeconds>2.11</durationSeconds>
         <name />
      </step>
      <step>
            [...]
      </step>
   </leg>
```

```
   <srs>epsg:4326</srs>
</routeResultV3>
```

Cas d'un oubli de spécification du point de départ ou d'arrivée (routeResultV3/status est ERROR)

```
<routeResultV3>
   <message>Origin and destination must be not null</message>
   <status>ERROR</status>
   <distanceMeters>0.0</distanceMeters>
   <durationSeconds>0.0</durationSeconds>
</routeResultV3>
```

Cas d'un mauvais formatage du point de départ, d'arrivée ou des étapes (routeResultV3/status est ERROR)

```
<routeResultV3>
   <message>Origin, destination and waypoints should be represented by a couple of coordinates</message>
   <status>ERROR</status>
   <distanceMeters>0.0</distanceMeters>
   <durationSeconds>0.0</durationSeconds>
</routeResultV3>
```

Cas d'un mauvais typage (serviceResult/status est ERROR)

```
<serviceResult>
        <message>NumberFormatException: For input string: "AAA"</message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'une erreur d'accrochage au graphe (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in route computation
Error in smartrouting
Failed to execute calculateRoute
com.geoconcept.smartrouting.SmartRoutingNativeException: failed to connect waypoint { 146.691012, 47.384813,
 0.000000 }
failed to connect waypoint { 146.691012, 47.384813, 0.000000 }</message>
   <status>ERROR</status>
</serviceResult>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
    <message>ServiceException: Error in route computation
Error in smartrouting
datasource is null</message>
    <status>ERROR</status>
</serviceResult>
```

V2

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
origin	Coordonnées du point de départ. Les coordonnées longitude et latitude sont séparées par la caractère ,	non	
destination	Coordonnées du point d'arrivée. Les coordonnées longitude et latitude sont séparées par la caractère ,	non	
waypoints	Coordonnées des étapes. Chaque couple de coordonnées d'une étape est encadré par la balise <waypoint> Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
format	- standard : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt / géométrie simplifiée au format wkt / liste des segments concaténés (sans géométries) - extended : résultat = résumé de l'itinéraire / bornes / géométrie simplifiée au format wkt / liste des segments non-concaténés (avec géométries) - summary : résultat = résumé de l'itinéraire - geometry : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt - simplifiedgeometry : résultat = résumé de l'itinéraire / bornes / géométrie simplifiée au format wkt - geometries : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt / géométrie simplifiée au format wkt - brief : résultat = résumé de l'itinéraire / Liste des segments avec durée et distance - standardext : résultat = résumé de l'itinéraire / bornes / liste des segments concaténés avec durée et distance et géométrie.	oui	standard
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
tolerance	Distance de tolérance (en mètre) de simplification de la géométrie.	oui	
graphName	Nom du graphe à utiliser	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris	oui	
profileId	Identifiant du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
profileName	Profil du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère , ou *;* (Exemple : Toll, Tunnel, Bridge)	oui	
timeLine	Liste de durées intermédiaires pour calculer des positions le long du trajet. Exprimées en secondes, séparés par le caractère *;*	oui	

En sortie

Itinéraire (routeResult)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l'itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l'itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)

paramètre	type	min/max	description
distanceMeters	string	0/1	Distance totale de l'itinéraire en mètres.
durationSeconds	string	0/1	Durée totale de l'itinéraire en secondes.
bounds	string	0/1	Bornes (BoundingBox) de la géométrie de l'itinéraire.
geometryWkt	string	0/1	Géométrie de l'itinéraire au format WKT.
simplifiedWkt	string	0/1	Géométrie simplifiée de l'itinéraire au format WKT.
leg (ou legs en JSON / JSON-P)	subRoute (ou array en JSON / JSON-P)	0/illimité	Liste des portions d'itinéraires.
startDateTime	string	0/1	Date et heure de départ (format : ISO8601 sans code zone : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 pour un départ le 21 janvier 2014, à 9h à Paris
finishDateTime	string	0/1	Date et heure d'arrivée (format : ISO8601 sans code zone : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 pour une arrivée le 21 janvier 2014, à 9h à Paris
timeLineItem	timeLineItem (ou array en JSON / JSON-P)	0/illimité	Liste des positions calculées.

Portion d'itinéraire (subRoute)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l'itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l'itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	string	0/1	Distance de la portion de l'itinéraire en mètres.
durationSeconds	string	0/1	Durée de la portion de l'itinéraire en secondes.
step ou (ou steps en JSON / JSON-P)	segment (ou array en JSON / JSON-P)	0/illimité	Liste des segments composants la portion d'itinéraire.

Segment d'itinéraire (segment)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l'itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l'itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	string	0/1	Distance du segment de l'itinéraire en mètres.
durationSeconds	string	0/1	Durée du segment de l'itinéraire en secondes.
navigationInstruction	string	0/1	Code d'instruction de navigation : - F : Tout droit - FR: Tourner légèrement à droite - FL: Tourner légèrement à gauche - R: Tourner à droite - L: Tourner à gauche

paramètre	type	min/max	description
			- BR: Tourner fortement à droite - BL: Tourner fortement à gauche - B: Demi-tour - round_about_entry: S'engager sur le rond-point - round_about_exit: Sortir du rond-point
name	string	0/1	Nom de rue du segment.
point	string	0/illimité	Liste de coordonnées séparées par la caractère ,

Position d'itinéraire (timeLineItem)

paramètre	type	min/max	description
elapsedTimeSeconds	double	0/1	Durée de l'itinéraire en secondes jusquà cette position.
message	string	0/1	Message d'erreur pour cette position.
status	string	0/1	Status de cette position.
distanceMeters	double	0/1	Distance de l'itinéraire en mètres jusquà cette position.
coordinates	string	0/1	Coordonnées de la position.

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/routeService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:routeV2>
         <!--Optional:-->
         <request>
            <origin>
              <x>-1.351448</x>
              <y>47.446923</y>
            </origin>
            <destination>
               <x>-1.34529</x>
                <y>47.4479931</y>
            </destination>
            <!--Optional:-->
            <waypoints>
               <!--Zero or more repetitions:-->
               <waypoint>
                <x>-1.34981</x>
                  <y>47.44837</y>
               </waypoint>
            </waypoints>
            <!--Optional:-->
            <srs>epsg:4326</srs>
            <!--Optional:-->
            <method>time</method>
            <!--Optional:-->
            <format>STANDARD</format>
            <!--Optional:-->
```

```
            <tolerance>0.</tolerance>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <startDateTime>2015-07-29T09:00:00.000+01:00</startDateTime>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Zero or more repetitions:-->
            <rejectFlags></rejectFlags>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <timeLine>
               <!--Zero or more repetitions:-->
               <timeLineItem>5</timeLineItem>
               <timeLineItem>15</timeLineItem>
            </timeLine>
         </request>
      </sch:routeV2>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:routeV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <RouteResult>
            <status>OK</status>
            <distance>640 m</distance>
            <duration>0:01:24</duration>
            <distanceMeters>640</distanceMeters>
            <durationSeconds>84.87</durationSeconds>
            <bounds>-1.351448,47.446922;-1.345263,47.44848</bounds>
            <geometryWkt>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
geometryWkt>
            <simplifiedWkt>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
simplifiedWkt>
            <leg>
               <distance>233 m</distance>
               <duration>0:00:36</duration>
               <distanceMeters>233.1</distanceMeters>
               <durationSeconds>36.16</durationSeconds>
               <step>
                  <distance>139 m</distance>
                  <duration>0:00:24</duration>
                  <distanceMeters>138.59</distanceMeters>
                  <durationSeconds>24.83</durationSeconds>
                  <name>RUE LA FONTAINE BRUNEAU</name>
               </step>
               <step>
                    [...]
               </step>
            </leg>
            <leg>
               <distance>407 m</distance>
               <duration>0:00:48</duration>
               <distanceMeters>406.9</distanceMeters>
```

```
               <durationSeconds>48.71</durationSeconds>
               <step>
                  <distance>18 m</distance>
                  <duration>0:00:02</duration>
                  <distanceMeters>17.62</distanceMeters>
                  <durationSeconds>2.11</durationSeconds>
                  <name/>
               </step>
               <step>
                    [...]
               </step>
            </leg>
            <startDateTime>2015-07-29T13:30:00.000+05:30</startDateTime>
            <finishDateTime>2015-07-29T13:31:24.870+05:30</finishDateTime>
            <timeLineItem>
               <elapsedTimeSeconds>5.0</elapsedTimeSeconds>
               <status>OK</status>
               <distanceMeters>27.90894700354487</distanceMeters>
               <coordinates>-1.351356,47.447160</coordinates>
            </timeLineItem>
            <timeLineItem>
               <elapsedTimeSeconds>15.0</elapsedTimeSeconds>
               <status>OK</status>
               <distanceMeters>83.72684101063462</distanceMeters>
               <coordinates>-1.351156,47.447643</coordinates>
            </timeLineItem>
         </RouteResult>
      </ns2:routeV2Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/route/v2.json?
origin=-1.351448,47.446923&destination=-1.34529,47.4479931&waypoints=-1.34981,47.44837
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/route/v2.json?
origin=-1.351448,47.446923&destination=-1.34529,47.4479931&waypoints=-1.34981,47.44837&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/route/v2.xml?
origin=-1.351448,47.446923&destination=-1.34529,47.4479931&waypoints=-1.34981,47.44837
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message":null,
   "status":"OK",
```

```
    "distance":"640 m",
    "duration":"0:02:06",
    "distanceMeters":"639.97",
    "durationSeconds":"126.55",
    "bounds":"-1.351448,47.446922;-1.345263,47.44848",
    "geometryWkt":"LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])",
    "simplifiedWkt":"LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])",
    "legs":[
       {
          "distance":"233 m",
          "duration":"0:00:46",
          "distanceMeters":"233.09",
          "durationSeconds":"46.21",
          "steps":[
             {
                "distance":"139 m",
                "duration":"0:00:29",
                "distanceMeters":"138.59",
                "durationSeconds":"29.21",
                "navInstruction":null,
                "name":"RUE LA FONTAINE BRUNEAU",
                "points":[

                ]
             },
             {
                 [...]
             }
          ]
       },
       {
          "distance":"407 m",
          "duration":"0:01:20",
          "distanceMeters":"406.88",
          "durationSeconds":"80.34",
          "steps":[
             {
                "distance":"18 m",
                "duration":"0:00:03",
                "distanceMeters":"17.63",
                "durationSeconds":"3.17",
                "navInstruction":null,
                "name":"",
                "points":[

                ]
             },
             {
                 [...]
             }
          ]
       }
    ],
    "startDateTime":null,
    "finishDateTime":null
}
```


Format JSON-P


```
myCallback(
    {
       "message":null,
       "status":"OK",
```

```
        "distance":"640 m",
        "duration":"0:02:06",
        "distanceMeters":"639.97",
        "durationSeconds":"126.55",
        "bounds":"-1.351448,47.446922;-1.345263,47.44848",
        "geometryWkt":"LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])",
        "simplifiedWkt":"LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])",
        "legs":[
           {
              "distance":"233 m",
              "duration":"0:00:46",
              "distanceMeters":"233.09",
              "durationSeconds":"46.21",
              "steps":[
                 {
                    "distance":"139 m",
                    "duration":"0:00:29",
                    "distanceMeters":"138.59",
                    "durationSeconds":"29.21",
                    "navInstruction":null,
                    "name":"RUE LA FONTAINE BRUNEAU",
                    "points":[

                    ]
                 },
                 {
                     [...]
                 }
              ]
           },
           {
              "distance":"407 m",
              "duration":"0:01:20",
              "distanceMeters":"406.88",
              "durationSeconds":"80.34",
              "steps":[
                 {
                    "distance":"18 m",
                    "duration":"0:00:03",
                    "distanceMeters":"17.63",
                    "durationSeconds":"3.17",
                    "navInstruction":null,
                    "name":"",
                    "points":[

                    ]
                 },
                 {
                     [...]
                 }
              ]
           }
        ],
        "startDateTime":null,
        "finishDateTime":null
     }
);
```


Format XML


```
<?xml version="1.0" encoding="UTF-8"?>
<routeResult>
   <status>OK</status>
```

```
   <distance>640 m</distance>
   <duration>0:02:06</duration>
   <distanceMeters>639.97</distanceMeters>
   <durationSeconds>126.55</durationSeconds>
   <bounds>-1.351448,47.446922;-1.345263,47.44848</bounds>
   <geometryWkt>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
geometryWkt>
   <simplifiedWkt>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
simplifiedWkt>
   <leg>
      <distance>233 m</distance>
      <duration>0:00:46</duration>
      <distanceMeters>233.09</distanceMeters>
      <durationSeconds>46.21</durationSeconds>
      <step>
         <distance>139 m</distance>
         <duration>0:00:29</duration>
         <distanceMeters>138.59</distanceMeters>
         <durationSeconds>29.21</durationSeconds>
         <name>RUE LA FONTAINE BRUNEAU</name>
      </step>
      <step>
         [...]
      </step>
   </leg>
   <leg>
      <distance>407 m</distance>
      <duration>0:01:20</duration>
      <distanceMeters>406.88</distanceMeters>
      <durationSeconds>80.34</durationSeconds>
      <step>
         <distance>18 m</distance>
         <duration>0:00:03</duration>
         <distanceMeters>17.63</distanceMeters>
         <durationSeconds>3.17</durationSeconds>
         <name />
      </step>
      <step>
         [...]
      </step>
   </leg>
</routeResult>
```

API JavaScript

Inclure la librairie JavaScript.

```
var routeCtrl = new GCUI.Control.Route();
routeCtrl.route({
    url:'http://<server>/<webapp>/api/lbs/route/v2.json',
    tolerance : 100,
    origin : new OpenLayers.LonLat(0.691012, 47.384813),
    destination : new OpenLayers.LonLat(0.691012, 47.384813),
    waypoints : [ new OpenLayers.LonLat(2.344408, 49.898798) ],
    callback : function(result, options) {
        console.log(result);
    }
});
```

La variable **`result`** est au format JSON décrit ci-dessus. La fonction **`callback`** passée en paramètre est appelée à la fin du calcul d'itinéraire.

Retours possibles

Cas d'un itinéraire trouvé (routeResult/status est OK)

```
<?xml version="1.0" encoding="UTF-8"?>
<routeResult>
   <status>OK</status>
   <distance>640 m</distance>
   <duration>0:02:06</duration>
   <distanceMeters>639.97</distanceMeters>
   <durationSeconds>126.55</durationSeconds>
   <bounds>-1.351448,47.446922;-1.345263,47.44848</bounds>
   <geometryWkt>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
geometryWkt>
   <simplifiedWkt>LINESTRING (-1.351448 47.446923, -1.351439 47.446922, -1.35142 47.44699, [...])</
simplifiedWkt>
   <leg>
      <distance>233 m</distance>
      <duration>0:00:46</duration>
      <distanceMeters>233.09</distanceMeters>
      <durationSeconds>46.21</durationSeconds>
      <step>
         <distance>139 m</distance>
         <duration>0:00:29</duration>
         <distanceMeters>138.59</distanceMeters>
         <durationSeconds>29.21</durationSeconds>
         <name>RUE LA FONTAINE BRUNEAU</name>
      </step>
      <step>
         [...]
      </step>
   </leg>
   <leg>
      <distance>407 m</distance>
      <duration>0:01:20</duration>
      <distanceMeters>406.88</distanceMeters>
      <durationSeconds>80.34</durationSeconds>
      <step>
         <distance>18 m</distance>
         <duration>0:00:03</duration>
         <distanceMeters>17.63</distanceMeters>
         <durationSeconds>3.17</durationSeconds>
         <name />
      </step>
      <step>
         [...]
      </step>
   </leg>
</routeResult>
```

Cas d'un oubli de spécification du point de départ ou d'arrivée (routeResult/status est ERROR)

```
<routeResult>
   <message>Origin and destination must be not null</message>
   <status>ERROR</status>
</routeResult>
```

Cas d'un mauvais formatage du point de départ, d'arrivée ou des étapes (routeResult/status est ERROR)

```
<routeResult>
   <message>Origin, destination and waypoints should be represented by a couple of coordinates</message>
   <status>ERROR</status>
</routeResult>
```

Cas d'un mauvais typage (serviceResult/status est ERROR)

```
<serviceResult>
        <message>NumberFormatException: For input string: "AAA"</message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'une erreur d'accrochage au graphe (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in route computation
Error in smartrouting
Failed to execute calculateRoute
com.geoconcept.smartrouting.SmartRoutingNativeException: failed to connect waypoint { 146.691012, 47.384813,
 0.000000 }
failed to connect waypoint { 146.691012, 47.384813, 0.000000 }</message>
   <status>ERROR</status>
</serviceResult>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
    <message>ServiceException: Error in route computation
Error in smartrouting
datasource is null</message>
    <status>ERROR</status>
</serviceResult>
```

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
origin	Coordonnées du point de départ. Les coordonnées longitude et latitude sont séparées par la caractère ,	non	
destination	Coordonnées du point d'arrivée. Les coordonnées longitude et latitude sont séparées par la caractère ,	non	
waypoints	Coordonnées des étapes. Chaque couple de coordonnées d'une étape est encadré par la balise <waypoint> Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
format	- standard : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt / géométrie simplifiée au format wkt / liste des segments concaténés (sans géométries)	oui	standard

paramètre	description	optionnel	défaut
	- extended : résultat = résumé de l'itinéraire / bornes / géométrie simplifiée au format wkt / liste des segments non-concaténés (avec géométries) - summary : résultat = résumé de l'itinéraire - geometry : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt - simplifiedgeometry : résultat = résumé de l'itinéraire / bornes / géométrie simplifiée au format wkt - geometries : résultat = résumé de l'itinéraire / bornes / géométrie au format wkt / géométrie simplifiée au format wkt - brief : résultat = résumé de l'itinéraire / Liste des segments avec durée et distance - standardext : résultat = résumé de l'itinéraire / bornes / liste des segments concaténés avec durée et distance et géométrie.		
projection	Déprécié, remplacé par srs	oui	
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
tolerance	Distance de tolérance (en mètre) de simplification de la géométrie.	oui	
graphName	Nom du graphe à utiliser	oui	
map	Déprécié : Nom de la carte à utiliser	oui	
applyMapPrecisionOut	Déprécié : Précision de la carte	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris	oui	
profileId	Identifiant du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
profileName	Profil du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
rejectFlags	Déprécié, remplacé par exclusions	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère , ou ; (Exemple : Toll, Tunnel, Bridge)	oui	

En sortie

Itinéraire (routeResult)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l'itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l'itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	string	0/1	Distance totale de l'itinéraire en mètres.
durationSeconds	string	0/1	Durée totale de l'itinéraire en secondes.
bounds	string	0/1	Bornes (BoundingBox) de la géométrie de l'itinéraire.
geometryWkt	string	0/1	Géométrie de l'itinéraire au format WKT.
simplifiedWkt	string	0/1	Géométrie simplifiée de l'itinéraire au format WKT.
leg (ou legs en JSON / JSON-P)	subRoute (ou array en JSON / JSON-P)	0/illimité	Liste des portions d'itinéraires.

paramètre	type	min/max	description
startDateTime	string	0/1	Date et heure de départ (format : ISO8601 sans code zone : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 pour un départ le 21 janvier 2014, à 9h à Paris
finishDateTime	string	0/1	Date et heure d'arrivée (format : ISO8601 sans code zone : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 pour une arrivée le 21 janvier 2014, à 9h à Paris
timeLineItem	timeLineItem (ou array en JSON / JSON-P)	0/illimité	Non implémenté dans cette version.

Portion d'itinéraire (subRoute)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l'itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l'itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	string	0/1	Distance de la portion de l'itinéraire en mètres.
durationSeconds	string	0/1	Durée de la portion de l'itinéraire en secondes.
step ou (ou steps en JSON / JSON-P)	segment (ou array en JSON / JSON-P)	0/illimité	Liste des segments composants la portion d'itinéraire.

Segment d'itinéraire (segment)

paramètre	type	min/max	description
distance	string	0/1	Distance totale de l'itinéraire, formaté : - xx.xx Km - xx m (si distance inférieure à 1 km)
duration	string	0/1	Durée totale de l'itinéraire, formaté : - HH:mm:ss (HH=heures, mm=minutes, ss=secondes)
distanceMeters	string	0/1	Distance du segment de l'itinéraire en mètres.
durationSeconds	string	0/1	Durée du segment de l'itinéraire en secondes.
navigationInstruction	string	0/1	Code d'instruction de navigation : - F : Tout droit - FR: Tourner légèrement à droite - FL: Tourner légèrement à gauche - R: Tourner à droite - L: Tourner à gauche - BR: Tourner fortement à droite - BL: Tourner fortement à gauche - B: Demi-tour - round_about_entry: S'engager sur le rond-point - round_about_exit: Sortir du rond-point
name	string	0/1	Nom de rue du segment.
point	string	0/illimité	Liste de coordonnées séparées par la caractère ,

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/routeService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:route>
         <!--Optional:-->
         <request>
            <origin>
               <x>0.691012</x>
               <y>47.384813</y>
            </origin>
            <destination>
               <x>0.693012</x>
               <y>47.385813</y>
            </destination>
            <!--Optional:-->
            <waypoints>
               <!--Zero or more repetitions:-->
               <waypoint>
                  <x>0.692012</x>
                  <y>47.384813</y>
               </waypoint>
            </waypoints>
            <!--Optional:-->
            <srs></srs>
            <!--Optional:-->
            <method></method>
            <!--Optional:-->
            <format></format>
            <!--Optional:-->
            <tolerance></tolerance>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Zero or more repetitions:-->
            <rejectFlags></rejectFlags>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
         </request>
      </sch:route>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:routeResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <RouteResult>
            <status>OK</status>
            <distance>577 m</distance>
            <duration>0:02:49</duration>
            <bounds>0.69089,47.38404;0.693012,47.386078</bounds>
            <geometryWkt>LINESTRING (0.6910119767724592 47.38481290982329, 0.6910138207911578
 47.384806371938815, ...)</geometryWkt>
            <simplifiedWkt>LINESTRING (0.6910119767724592 47.38481290982329, 0.6910138207911578
 47.384806371938815, ...)</simplifiedWkt>
            <leg>
               <distance>266 m</distance>
               <duration>0:01:08</duration>
               <step>
                  <distance>10</distance>
                  <duration>2</duration>
                  <name>RUE EUPATORIA</name>
               </step>
               <step>
                  <distance>86</distance>
                  <duration>18</duration>
                  <navInstruction>L</navInstruction>
                  <name>AVENUE DE GRAMMONT</name>
               </step>
               <step>
                  <distance>13</distance>
                  <duration>2</duration>
                  <navInstruction>L</navInstruction>
                  <name>PLACE VAILLANT</name>
               </step>
               <step>
                  <distance>66</distance>
                  <duration>12</duration>
                  <navInstruction>L</navInstruction>
                  <name>AVENUE DE GRAMMONT</name>
               </step>
               <step>
                  <distance>89</distance>
                  <duration>32</duration>
                  <navInstruction>R</navInstruction>
                  <name>RUE PARMENTIER</name>
               </step>
            </leg>
            <leg>
               <distance>311 m</distance>
               <duration>0:01:40</duration>
               <step>
                  <distance>90</distance>
                  <duration>33</duration>
                  <name>RUE PARMENTIER</name>
               </step>
               <step>
                  <distance>153</distance>
                  <duration>32</duration>
                  <navInstruction>L</navInstruction>
                  <name>RUE MICHELET</name>
               </step>
               <step>
                  <distance>66</distance>
                  <duration>35</duration>
                  <navInstruction>R</navInstruction>
```

```
                  <name>RUE DUPORTAL</name>
               </step>
            </leg>
         </RouteResult>
      </ns2:routeResponse>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/route.json?
origin=0.691012,47.384813&destination=0.693012,47.385813&waypoints=0.692012,47.384813
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/route.json?
origin=0.691012,47.384813&destination=0.693012,47.385813&waypoints=0.692012,47.384813&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/route.xml?
origin=0.691012,47.384813&destination=0.693012,47.385813&waypoints=0.692012,47.384813
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON


```
{
        "message":null,"status":"OK",
        "distance":"577 m","duration":"0:02:49","bounds":"475757.89,2266229.66;475920.47,2266453.7",
        "geometryWkt":"LINESTRING (475767.17 2266316.12, 475767.3 2266315.39, ...)",
        "legs":
                [
                        {"distance":"266 m","duration":"0:01:08",
                                "steps":
                                        [

 {"distance":"10","duration":"2","navInstruction":null,"name":"RUE EUPATORIA","points":[]},

 {"distance":"86","duration":"18","navInstruction":"L","name":"AVENUE DE GRAMMONT","points":[]},

 {"distance":"13","duration":"2","navInstruction":"L","name":"PLACE VAILLANT","points":[]},

 {"distance":"66","duration":"12","navInstruction":"L","name":"AVENUE DE GRAMMONT","points":[]},

 {"distance":"89","duration":"32","navInstruction":"R","name":"RUE PARMENTIER","points":[]}
                                        ]
                        },
                        {"distance":"311 m","duration":"0:01:40",
                                "steps":
                                        [
```

```
 {"distance":"90","duration":"33","navInstruction":null,"name":"RUE PARMENTIER","points":[]},

 {"distance":"153","duration":"32","navInstruction":"L","name":"RUE MICHELET","points":[]},

 {"distance":"66","duration":"35","navInstruction":"R","name":"RUE DUPORTAL","points":[]}
                                        ]
                                }
                ],
        "startDateTime":null,"finishDateTime":null
}
```


Format JSON-P


```
myCallback(
        {
                "message":null,"status":"OK",
                "distance":"577 m","duration":"0:02:49","bounds":"475757.89,2266229.66;475920.47,2266453.7",
                "geometryWkt":"LINESTRING (475767.17 2266316.12, 475767.3 2266315.39, ...)",
                "legs":
                        [
                                {"distance":"266 m","duration":"0:01:08",
                                "steps":
                                        [

 {"distance":"10","duration":"2","navInstruction":null,"name":"RUE EUPATORIA","points":[]},

 {"distance":"86","duration":"18","navInstruction":"L","name":"AVENUE DE GRAMMONT","points":[]},

 {"distance":"13","duration":"2","navInstruction":"L","name":"PLACE VAILLANT","points":[]},

 {"distance":"66","duration":"12","navInstruction":"L","name":"AVENUE DE GRAMMONT","points":[]},

 {"distance":"89","duration":"32","navInstruction":"R","name":"RUE PARMENTIER","points":[]}
                                        ]
                                },
                                {"distance":"311 m","duration":"0:01:40",
                                "steps":
                                        [

 {"distance":"90","duration":"33","navInstruction":null,"name":"RUE PARMENTIER","points":[]},

 {"distance":"153","duration":"32","navInstruction":"L","name":"RUE MICHELET","points":[]},

 {"distance":"66","duration":"35","navInstruction":"R","name":"RUE DUPORTAL","points":[]}
                                        ]
                                }
                        ],
                "startDateTime":null,"finishDateTime":null
        }
);
```


Format XML


```
<routeResult>
        <status>OK</status>
        <distance>577 m</distance>
        <duration>0:02:49</duration>
        <bounds>475757.89,2266229.66;475920.47,2266453.7</bounds>
        <geometryWkt>LINESTRING (475767.17 2266316.12, 475767.3 2266315.39, ...)</geometryWkt>
        <simplifiedWkt>LINESTRING (475767.17 2266316.12, 475767.3 2266315.39, ...)</simplifiedWkt>
        <leg>
```

```
                <distance>266 m</distance>
                <duration>0:01:08</duration>
                <step>
                        <distance>10</distance>
                        <duration>2</duration>
                        <name>RUE EUPATORIA</name>
                </step>
                <step>
                        <distance>86</distance>
                        <duration>18</duration>
                        <navInstruction>L</navInstruction>
                        <name>AVENUE DE GRAMMONT</name>
                </step>
                <step>
                        <distance>13</distance>
                        <duration>2</duration>
                        <navInstruction>L</navInstruction>
                        <name>PLACE VAILLANT</name>
                </step>
                <step>
                        <distance>66</distance>
                        <duration>12</duration>
                        <navInstruction>L</navInstruction>
                        <name>AVENUE DE GRAMMONT</name>
                </step>
                <step>
                        <distance>89</distance>
                        <duration>32</duration>
                        <navInstruction>R</navInstruction>
                        <name>RUE PARMENTIER</name>
                </step>
        </leg>
        <leg>
                <distance>311 m</distance>
                <duration>0:01:40</duration>
                <step>
                        <distance>90</distance>
                        <duration>33</duration>
                        <name>RUE PARMENTIER</name>
                </step>
                <step>
                        <distance>153</distance>
                        <duration>32</duration>
                        <navInstruction>L</navInstruction>
                        <name>RUE MICHELET</name>
                </step>
                <step>
                        <distance>66</distance>
                        <duration>35</duration>
                        <navInstruction>R</navInstruction>
                        <name>RUE DUPORTAL</name>
                </step>
        </leg>
</routeResult>
```

API JavaScript

Inclure la librairie JavaScript.

```
var routeCtrl = new GCUI.Control.Route();
routeCtrl.route({
    url:'http://<server>/<webapp>/api/lbs/route.json',
```

```
    tolerance : 100,
    origin : new OpenLayers.LonLat(0.691012, 47.384813),
    destination : new OpenLayers.LonLat(0.691012, 47.384813),
    waypoints : [ new OpenLayers.LonLat(2.344408, 49.898798) ],
    callback : function(result, options) {
        console.log(result);
    }
});
```

La variable **`result`** est au format JSON décrit ci-dessus. La fonction **`callback`** passée en paramètre est appelée à la fin du calcul d'itinéraire.

Retours possibles

Cas d'un itinéraire trouvé (routeResult/status est OK)

```
<routeResult>
    <status>OK</status>
    <distance>1.66 Km</distance>
    <duration>0:04:53</duration>
    <bounds>2.423385,48.84452;2.43999,48.84741</bounds>
    <geometryWkt>LINESTRING (2.4233899276812516 48.84461991900793, 2.4233847309012826
 48.84462042192212, ...)</geometryWkt>
    <simplifiedWkt>LINESTRING (2.4233899276812516 48.84461991900793, 2.4233847309012826
 48.84462042192212, ...)</simplifiedWkt>
    <leg>
        <distance>1.66 Km</distance>
        <duration>0:04:53</duration>
        <step>
            <distance>147</distance>
            <duration>31</duration>
            <name>AVENUE PASTEUR</name>
        </step>
        <step>
            <distance>1108</distance>
            <duration>159</duration>
            <navInstruction>R</navInstruction>
            <name>AVENUE DE PARIS</name>
        </step>
                ...
        </step>
    </leg>
</routeResult>
```

Cas d'un oubli de spécification du point de départ ou d'arrivée (routeResult/status est ERROR)

```
<routeResult>
    <message>Origin and destination must be not null</message>
    <status>ERROR</status>
</routeResult>
```

Cas d'un mauvais typage (serviceResult/status est ERROR)

```
<serviceResult>
        <message>NumberFormatException: For input string: "AAA"</message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'une erreur d'accrochage au graphe (serviceResult/status est ERROR)

```
<serviceResult>
    <message>ServiceException: Error in route computation
Error in smartrouting
Failed to execute calculateRoute
com.geoconcept.smartrouting.SmartRoutingNativeException: failed to connect waypoint { 243966.000000,
 48.847410, 0.000000 }
failed to connect waypoint { 243966.000000, 48.847410, 0.000000 }</message>
    <status>ERROR</status>
</serviceResult>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
    <message>ServiceException: Error in route computation
Error in smartrouting
datasource is null</message>
    <status>ERROR</status>
</serviceResult>
```

FAQ

1. Est-il possible de prioriser le temps de parcours ou la distance ?

   Oui, en changeant la méthode **`method`** distance ou time.

2. Peut-on utiliser des alias à défaut des noms de fichiers .siti pour appeler une datasource ?

   Oui, cf. détails dans la FAQ du Web Service du géocodage inverse.

3. Comment utiliser les statistiques routières ?

   Vérifier que le graphe contient bien ces statistiques et utiliser les deux paramètres suivants : **`computeOptions`** avec la valeur *trafficPatterns* et **`startDateTime`** pour préciser la date/heure de départ.

4. L'ordre des points a-t-il une importance ?

   Les points, qu'ils soient de départ, d'arrivée ou de passage sont lus et utilisés dans l'ordre. Le premier point déclaré est considéré comme point de départ, le second comme point d'arrivée et les autres comme points de passage. L'ordre des points est donc très important.

5. Comment faire un calcul d'itinéraire sans péage?

   Si la contrainte *Toll* a bien été incluse dans le graphe, placer une exclusion dans **`exclusions`** :

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:route>
         <!--Optional:-->
         <request>
            <origin>
               <x>0.691012</x>
               <y>47.384813</y>
            </origin>
            <destination>
               <x>0.693012</x>
               <y>47.385813</y>
            </destination>
            <!--Zero or more repetitions:-->
```

```
            <waypoints>
                        <waypoint>
                                <x>0.692012</x>
                                <y>47.384813</y>
                        </waypoint>
            </waypoints>
            <!--Optional:-->
            <srs></srs>
            <!--Optional:-->
            <method></method>
            <!--Optional:-->
            <format></format>
            <tolerance></tolerance>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <startDateTime></startDateTime>
                        <!--Optional:-->
            <profileId></profileId>
                        <!--Optional:-->
                        <profileName></profileName>
            <!--Zero or more repetitions:-->
            <exclusions>Toll</exclusions>
         </request>
      </sch:route>
   </soapenv:Body>
</soapenv:Envelope>
```

6. Qu’est-ce que les Speed Patterns ? Comment les utiliser ?

   Afin de proposer des temps de trajets au plus près des conditions de circulation, les graphes fournis par GEOCONCEPT SAS intègrent, à partir de la version M18, pour les voitures et pour les camions 5 profils de vitesses (Speed Patterns) pour tenir compte des différents niveaux de congestion dans une journée :

   - standard *normal-speed* correspond à la vitesse d’une heure moyennement chargée (11h)
   - nuit *fast-speed* correspond à un trafic très fluide, observé le plus souvent la nuit (3h)
   - chargée *slow-speed* correspond aux heures de trafic denses (8h)
   - heure de pointe *very-slow-speed* correspond aux heures de trafic denses dans les grandes agglomérations, plus lent que le précédent (8h)
   - défaut *defaut* correspond aux vitesses moyennées sur une journée entière

   Pour les utiliser il faut passer, lors de l’appel au web service, le paramètre **`computeOptions`** avec l’option *speedPattern* et préciser la valeur de la Speed Pattern demandée sans omettre un profil de véhicule
   Exemple :

   +

   ```
   http://<server>/<webapp>/api/lbs/route/v5.xml?
   origin=-1.519363,47.215945&destination=-1.559309,47.213465&computeOptions=speedPattern:fast-
   speed&profileId=1
   ```

   + Retourne un trajet de 707 secondes contre 817 secondes pour un speedPattern *slow-speed*.

7. Comment utiliser les attributs poids lourds ?

Il faut que le graphe intègre les attributs de poids lourds (en standard dans les graphes fournis par GEOCONCEPT SAS à partir de la version M18) et soit calculer un itinéraire en utilisant un profil de véhicule utilisant les restrictions (cf. le catalogue de véhicules, éditable, définit dans le fichier SmartRoutingVehicles.xml, dans le dossier `"<*GEOCONCEPT_WEB_HOME*>"\*smartrouting*\*jee*\*smartrouting*\*conf*\ ), soit surcharger l'appel au web service en utilisant le paramètre **`computeOptions`** avec les options *length*, *width*, *height*, *weight*, *axles* et/ou *weightPerAxle*.
Exemple pour un trajet avec un véhicule de 4,5 mètres de hauteur :

```
http://<server>/<webapp>/api/lbs/route/v5.xml?
origin=-1.543363,47.215945&destination=-1.550309,47.213465&computeOptions=height:450
```

Retourne un trajet de 1529 mètres contre 624 mètres sans restriction de hauteur.

8. Comment calculer l'empreinte carbone d'un trajet ?

Le calcul de l'empreinte carbone tient compte à la fois du type du véhicule, de son carburant et de sa vitesse tronçon de voie par tronçon de voie : 10 km en ville n'a pas la même empreinte carbone que 10 km en campagne. Il est affiché dans la réponse lorsque le format est égal à *standard* (par defaut) ou *extended*.
L'empreinte carbone de l'itinéraire est automatiquement calculée soit en utilisant un profil de véhicule utilisant les restrictions (cf. le catalogue de véhicules, éditable, définit dans le fichier SmartRoutingVehicles.xml, dans le dossier `"<*GEOCONCEPT_WEB_HOME*>"\*smartrouting*\*jee*\*smartrouting*\*conf*\ ), soit en surchargant les valeurs lors de l'appel au web service avec l'utilisation de **`computeOptions`** avec les options *fuelType*, *averageConsumption* et/ou *customAverageCO2UnitEmission*.
Les valeurs disponibles sont fuelType avec comme valeurs possibles :

- NoFuel
- Diesel
- UnleadedFuel
- LGP
- CustomFuelType (permet de personnaliser une valeur en kg de CO2 par litre à préciser dans customAverageCO2UnitEmission)

averageConsumption permet d'indiquer la consommation moyenne du véhicule pour 100 kilomètres
Exemple, pour la configuration suivante :

+

```
<vehicle id="10" name="Car">
        <speedProfile>Cars</speedProfile>
        <speedProfileId>3</speedProfileId>
        <rejectFlag>Automobiles</rejectFlag>
        <snapSpeed>4</snapSpeed>
        <fuelType>UndefinedFuelType</fuelType>
</vehicle>
<vehicle id="11" name="Car UnleadedFuel">
        <speedProfile>Cars</speedProfile>
        <speedProfileId>3</speedProfileId>
        <rejectFlag>Automobiles</rejectFlag>
```

```
                        <speedPatternsProfile>Cars SpeedPatterns</speedPatternsProfile>
                        <snapSpeed>4</snapSpeed>
                        <fuelType>UnleadedFuel</fuelType>
                        <averageConsumption>7.27</averageConsumption>
                </vehicle>
                <vehicle id="12" name="Car Diesel">
                        <speedProfile>Cars</speedProfile>
                        <speedProfileId>3</speedProfileId>
                        <rejectFlag>Automobiles</rejectFlag>
                        <speedPatternsProfile>Cars SpeedPatterns</speedPatternsProfile>
                        <snapSpeed>4</snapSpeed>
                        <fuelType>Diesel</fuelType>
                        <averageConsumption>6.06</averageConsumption>
                </vehicle>
                <vehicle id="30" name="Truck CustomFuelType">
                        <speedProfile>Trucks</speedProfile>
                        <speedProfileId>5</speedProfileId>
                        <rejectFlag>Trucks</rejectFlag>
                        <snapSpeed>4</snapSpeed>
                        <fuelType>CustomFuelType</fuelType>
                        <customAverageCO2UnitEmission>5</customAverageCO2UnitEmission>
                </vehicle>
        <vehicle id="32" name="Truck Diesel">
                        <speedProfile>Trucks</speedProfile>
                        <speedProfileId>5</speedProfileId>
                        <rejectFlag>Trucks</rejectFlag>
                        <speedPatternsProfile>Trucks SpeedPatterns</speedPatternsProfile>
                        <snapSpeed>4</snapSpeed>
                        <fuelType>Diesel</fuelType>
                        <averageConsumption>34</averageConsumption>
                </vehicle>
```

+ il est obtenu :

+ Car

+

```
http://<server>/<webapp>/api/lbs/route/v5.xml?
origin=-0.978536,47.443316&destination=-2.511388,47.303726&profileId=10
```

+ retourne en kg équivalent CO2 :

+

```
<carbonFootprint>0.0</carbonFootprint>
```

+ Car UnleadedFuel

+

```
http://<server>/<webapp>/api/lbs/route/v5.xml?
origin=-0.978536,47.443316&destination=-2.511388,47.303726&profileId=11
```

+ retourne en kg équivalent CO2 :

+

```
<carbonFootprint>31.151</carbonFootprint>
```

+ Car Diesel

+

```
http://<server>/<webapp>/api/lbs/route/v5.xml?
origin=-0.978536,47.443316&destination=-2.511388,47.303726&profileId=12
```

+ retourne en kg équivalent CO2 :

+

```
<carbonFootprint>26.117</carbonFootprint>
```

+ Truck CustomFuelType

+

```
http://<server>/<webapp>/api/lbs/route/v5.xml?
origin=-0.978536,47.443316&destination=-2.511388,47.303726&profileId=30
```

+ retourne en kg équivalent CO2 :

+

```
<carbonFootprint>43.914</carbonFootprint>
```

+ Truck Diesel

+

```
http://<server>/<webapp>/api/lbs/route/v5.xml?
origin=-0.978536,47.443316&destination=-2.511388,47.303726&profileId=32
```

+ retourne en kg équivalent CO2 :

+

```
<carbonFootprint>153.081</carbonFootprint>
```

Calcul d’itinéraire (batch)

Cette page n’est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

Ce web service calcule un lot d'itinéraires entre deux points et renvoie une feuille de route complète. L'ajout d'étapes intermédiaires est une possibilité. Il s'appuie sur le graphe paramétré dont le nom a été spécifié dans l'interface d'administration de Geoconcept Web.

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
routeRequests	tableau de RouteRequestV5	Non	Tableau d'itinéraires.
timeOut	string	Oui	Time out valeur de calcul (en millisecondes).

- Itinéraire en entrée (RouteRequestV5)**

Detaillé sur Calcul d'itinéraire web service.

En sortie

parameter	type	min/max	description
result	tableau de RouteResultV5	0/illimité	Itinéraires en sortie

- Itinéraires en sortie (RouteResultV5)**

Detaillé sur Calcul d'itinéraire web service.

Possible returns

Cas d'un itinéraire trouvé (routeResultV5/status est OK)


```
{
    "message": null,
    "status": "OK",
    "results": [
        {
            "message": null,
            "status": "OK",
            "distance": "1.66 Km",
            "duration": "0:07:56",
            "distanceMeters": 1658.9,
            "durationSeconds": 476.69,
            "bounds": null,
            "wktGeometry": null,
            "wktSimplifiedGeometry": null,
            "compressedGeometry": null,
            "compressedSimplifiedGeometry": null,
            "legs": [
            ],
```

```
            "startDateTime": null,
            "finishDateTime": null,
            "srs": "epsg:4326",
            "originNode": null,
            "waypointNodes": null,
            "destinationNode": null,
            "carbonFootprint": null
        },
        {
            "message": null,
            "status": "OK",
            "distance": "159 m",
            "duration": "0:00:47",
            "distanceMeters": 158.76,
            "durationSeconds": 47.66,
            "bounds": null,
            "wktGeometry": null,
            "wktSimplifiedGeometry": null,
            "compressedGeometry": null,
            "compressedSimplifiedGeometry": null,
            "legs": [
            ],
            "startDateTime": null,
            "finishDateTime": null,
            "srs": "epsg:4326",
            "originNode": null,
            "waypointNodes": null,
            "destinationNode": null,
            "carbonFootprint": null
        }
    ]
}
```


Voir Calcul d'itinéraire web service.

FAQ

Voir Calcul d'itinéraire web service.

Optimisation (version simplifiée)

Cette page n'est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

Ce web service permet de définir des tournées optimales en minimisant les temps et distances de déplacement. L'ordonnancement des points à visiter est optimisé sur base de contraintes géographiques et opérationnelles (plage horaire, durée de la visite…). Il s'appuie sur le graphe paramétré dont le nom a été spécifié dans l'interface d'administration de Geoconcept Web.

Disponibilité

Ce web service est une option de Geoconcept Web, veuiller nous contacter pour connaître les modalités d'acquisition.

Changement de version

Les versions précédente du web service sont conservées dans Geoconcept Web pour assurer la compatibilité avec les développements antérieurs. Il est recommandé d'utiliser la version la plus récente.

Changements avec la v3

- Ajout des paramètres en entrée "graphName", "profileId", "profileName", "configName", "snapMethod", "exclusions", "startDateTime", "avoidArea", "computeOptions", "maxCost" et "timeOut".
- Ajout en sortie de la liste des étapes non visitées car non atteignables "unreachableSteps".

Changements avec la v2

- Ajout du paramètre "timewindows/timewindow".
- Les paramètres "totalDistance" et "totalTime" sont renommés respectivement "distanceMeters" et "durationSeconds".
- Le Web service n'est plus disponible en REST GET.

V3

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
origin (optimRouteStepV3)	point d'origine (id,x,y)	oui	
destination (optimRouteStepV3)	point de destination (id,x,y)	oui	
steps (optimRouteStepV3)	points à visiter(id,x,y;id,x,y;…) Plusiers plages de visites en millisecondes peuvent être précisées pour chaque point : (id,x,y,début,fin)	non	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
graphName	Nom du graphe à utiliser Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileId	Identifiant du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileName	Profil du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
configName	Nom de la configuration à utiliser (défini dans Geoconcept Web - Administration / Outils / Définitions des graphes) Il remplace l'usage de graphName, profileId et profileName	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction - nodes : Accrochage direct aux noeuds fournis par les paramètres originNode, destinationNode et waypointNodes ou, si ces premiers ne sont pas renseignés, aux noeuds les plus proches des paramètres origin, destination et waypoint	oui	standard

paramètre	description	optionnel	défaut
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère *;* (Exemple : Toll, Tunnel, Bridge)	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris. Attention le caractère + peut être mal interprété par les navigateurs, dans ce cas, il faut le remplacer par *%2B*.	oui	
avoidArea	Zone de transit interdit au format WKT (POLYGON ou MULTIPOLYGON) dans la projection (paramètre srs) demandée Exemple en wgs84 : `POLYGON ((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689))` - `MULTIPOLYGON (((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689)), ((-1.558639 47.218144, -1.558639 47.218193, [...] -1.558639 47.218144)))` Attention les géométries WKT doivent être fermées.	oui	
computeOptions	Liste des options pour le calcul, séparées par le caractère *;* - trafficPatterns : utilise les statistiques routières (il est nécessaire de renseigner le paramètre startDateTime et d'utiliser un graphe intégrant les informations de traffic patterns) - speedPattern (M18) : utilise d'une *speed pattern* tel que définis dans le fichier SmartRoutingVehicles.xml qui se trouve dans le dossier *`"<GEOCONCEPT_WEB_HOME>"\smartrouting\jee\smartrouting\conf* . Usage : "speedPattern:slow-speed" - length (M18) : longueur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "length:950" - width (M18) : largeur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "width:255" - height (M18) : hauteur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "height:360" - weight (M18) : poids maximum autorisé en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weight:18000" - axles (M18) : nombre maximum d'axe autorisé (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "axles:2" - weightPerAxle (M18) : poids maximum autorisé par axe en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weightPerAxle:9000" - snapSpeed : vitesse d'accrochage au graphe en kilomètre par heure. Usage : "snapSpeed:10"	oui	
matrixProvider	fournisseur de matrice de calcul globe : vol d'oiseau sur globe (les coordonnées doivent être en long/lat) euclidean : vol d'oiseau sur carte (les coordonnées doivent être en mètres) smartrouting : calcul sur le réseau routier par SmartRouting nokia : calcul sur le réseau routier par Nokia services provided : paramètre de matrice	oui	smartrouting
matrix	matrice de temps/distance (id1,id2,temps,distance;…)	oui	
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	wgs84
directSpeed	Vitesse par défaut lors de l'utilisation des matrices globe ou euclidean	oui	50
maxCost	Coût maximum à ne pas dépasser dans le calcul -1 : pas de coût maximum à prendre en compte	oui	

paramètre	description	optionnel	défaut
	0 : prendre la valeur par default défini dans la configuration de SmartRouting Server sinon : valeur en mètres si method=distance ou en secondes si method=time		
timeOut	Time out pour le calcul (en millisecondes)	oui	

Etapes (optimRouteStepV3)

paramètre	type	min/max	description
id	string	0/1	id du point
duration	long	1/1	durée d'une visite, en millisecondes. 0 par défaut
timeWindows/ timeWindow (optimTimeWindow)	long	1/1	Fenêtres horaires

Fenêtres horaires (optimTimeWindow)

paramètre	type	min/max	description
start	long	1/1	début de plage horaire, en millisecondes. 0 par défaut
end	long	1/1	fin de plage horaire, en millisecondes. Long.MAX_VALUE par défaut

En sortie

paramètre	type	min/max	description
steps/step	array (optimRouteStepV3)	0/illimité	liste des étapes ordonnancées
distanceMeters	long	1/1	distance totale du trajet en mètres
durationSeconds	long	1/1	temps total du trajet en secondes
unreachableSteps	array (OptimUnreachableStepV3)	0/illimité	liste des étapes non atteignables.

Etapes (optimRouteStepV3)

paramètre	type	min/max	description
id	string	0/1	id de l'étape
x	number	1/1	coordonnée X de l'étape
y	number	1/1	coordonnée Y de l'étape
duration	long	1/1	durée d'une visite, en millisecondes. 0 par défaut
effectiveStart	long	1/1	début effectif d'une visite, en millisecondes
driveDistanceBefore	int	1/1	distance de conduite avant l'étape, en mètres
driveDistanceAfter	int	1/1	distance de conduite après l'étape, en mètre
driveTimeBefore	long	1/1	temps de conduite avant l'étape, en millisecondes
driveTimeAfter	long	1/1	temps de conduite après l'étape, en millisecondes
timeWindows/ timeWindow (optimTimeWindow)	long	1/1	Fenêtres horaires

Fenêtres horaires (optimTimeWindow)

paramètre	type	min/max	description
start	long	1/1	début de plage horaire, en millisecondes. 0 par défaut
end	long	1/1	fin de plage horaire, en millisecondes. Long.MAX_VALUE par défaut

Liste des étapes non atteignables (OptimUnreachableStepV3)

paramètre	type	min/max	description
id	string	1/1	id de l'étape
x	double	1/1	Longitude de l'étape
y	double	1/1	Latitude de l'étape

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/optimService?wsdl

Requête

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header />
   <soapenv:Body>
      <sch:optimRouteV3>
         <request>
            <origin>
               <x>-1.5291995940651142</x>
               <y>47.19158811606974</y>
               <id>1</id>
               <duration>3600000</duration>
               <effectiveStart>0</effectiveStart>
               <driveDistanceBefore>0</driveDistanceBefore>
               <driveDistanceAfter>0</driveDistanceAfter>
               <driveTimeBefore>0</driveTimeBefore>
               <driveTimeAfter>0</driveTimeAfter>
            </origin>
            <destination>
               <x>-1.5405963750884433</x>
               <y>47.19752774053115</y>
               <id>5</id>
               <duration>3600000</duration>
               <effectiveStart>0</effectiveStart>
               <driveDistanceBefore>0</driveDistanceBefore>
               <driveDistanceAfter>0</driveDistanceAfter>
               <driveTimeBefore>0</driveTimeBefore>
               <driveTimeAfter>0</driveTimeAfter>
            </destination>
            <steps>
               <step>
                  <x>-1.5315788375137436</x>
                  <y>47.209701524818236</y>
                  <id>2</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>0</driveDistanceBefore>
```

```
                  <driveDistanceAfter>0</driveDistanceAfter>
                  <driveTimeBefore>0</driveTimeBefore>
                  <driveTimeAfter>0</driveTimeAfter>
                  <timeWindows>
                     <!--Zero or more repetitions:-->
                     <timeWindow>
                        <start>0</start>
                        <end>9223372036854775807</end>
                     </timeWindow>
                  </timeWindows>
               </step>
               <step>
                  <x>-1.5391738246474214</x>
                  <y>47.193576663578824</y>
                  <id>3</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>0</driveDistanceBefore>
                  <driveDistanceAfter>0</driveDistanceAfter>
                  <driveTimeBefore>0</driveTimeBefore>
                  <driveTimeAfter>0</driveTimeAfter>
                  <timeWindows>
                     <!--Zero or more repetitions:-->
                     <timeWindow>
                        <start>0</start>
                        <end>9223372036854775807</end>
                     </timeWindow>
                  </timeWindows>
               </step>
               <step>
                  <x>-1.5562968681202867</x>
                  <y>47.21808294131743</y>
                  <id>4</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>0</driveDistanceBefore>
                  <driveDistanceAfter>0</driveDistanceAfter>
                  <driveTimeBefore>0</driveTimeBefore>
                  <driveTimeAfter>0</driveTimeAfter>
               </step>
            </steps>
            <srs>epsg:4326</srs>
            <method>time</method>
         </request>
      </sch:optimRouteV3>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:optimRouteV3Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <OptimRouteResult>
            <steps>
               <step>
                  <x>-1.5291995940651142</x>
                  <y>47.19158811606974</y>
                  <id>1</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>0</driveDistanceBefore>
                  <driveDistanceAfter>1779</driveDistanceAfter>
```

```
                  <driveTimeBefore>0</driveTimeBefore>
                  <driveTimeAfter>372710</driveTimeAfter>
                  <timeWindows/>
               </step>
               <step>
                  <x>-1.5391738246474214</x>
                  <y>47.193576663578824</y>
                  <id>3</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>1779</driveDistanceBefore>
                  <driveDistanceAfter>3486</driveDistanceAfter>
                  <driveTimeBefore>372710</driveTimeBefore>
                  <driveTimeAfter>752870</driveTimeAfter>
                  <timeWindows>
                     <timeWindow>
                        <start>0</start>
                        <end>9223372036854775807</end>
                     </timeWindow>
                  </timeWindows>
               </step>
               <step>
                  <x>-1.5562968681202867</x>
                  <y>47.21808294131743</y>
                  <id>4</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>3486</driveDistanceBefore>
                  <driveDistanceAfter>3175</driveDistanceAfter>
                  <driveTimeBefore>752870</driveTimeBefore>
                  <driveTimeAfter>662230</driveTimeAfter>
                  <timeWindows/>
               </step>
               <step>
                  <x>-1.5315788375137436</x>
                  <y>47.209701524818236</y>
                  <id>2</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>3175</driveDistanceBefore>
                  <driveDistanceAfter>2713</driveDistanceAfter>
                  <driveTimeBefore>662230</driveTimeBefore>
                  <driveTimeAfter>520090</driveTimeAfter>
                  <timeWindows>
                     <timeWindow>
                        <start>0</start>
                        <end>9223372036854775807</end>
                     </timeWindow>
                  </timeWindows>
               </step>
               <step>
                  <x>-1.5405963750884433</x>
                  <y>47.19752774053115</y>
                  <id>5</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>2713</driveDistanceBefore>
                  <driveDistanceAfter>0</driveDistanceAfter>
                  <driveTimeBefore>520090</driveTimeBefore>
                  <driveTimeAfter>0</driveTimeAfter>
                  <timeWindows/>
               </step>
            </steps>
```

```
            <distanceMeters>11153</distanceMeters>
            <durationSeconds>2307</durationSeconds>
         </OptimRouteResult>
      </ns2:optimRouteV3Response>
   </soap:Body>
</soap:Envelope>
```

REST (POST)

Requête

Requête

```
http://<server>/<webapp>/api/lbs/optim/route.xml
```

Data (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<optimRouteRequestV3>
   <origin>
      <x>-1.5291995940651142</x>
      <y>47.19158811606974</y>
      <!--Optional:-->
      <id>1</id>
      <timeWindows>
         <timeWindow>
            <start>09:00</start>
            <end>20:00</end>
         </timeWindow>
      </timeWindows>
   </origin>
   <destination>
      <x>-1.5405963750884433</x>
      <y>47.19752774053115</y>
      <!--Optional:-->
      <id>5</id>
   </destination>
   <steps>
      <step>
         <x>-1.5315788375137436</x>
         <y>47.209701524818236</y>
         <id>2</id>
         <timeWindows>
            <timeWindow>
               <start>09:00</start>
               <end>20:00</end>
            </timeWindow>
         </timeWindows>
      </step>
      <step>
         <x>-1.5391738246474214</x>
         <y>47.193576663578824</y>
         <id>3</id>
      </step>
      <step>
         <x>-1.5562968681202867</x>
         <y>47.21808294131743</y>
         <id>4</id>
      </step>
   </steps>
   <method>time</method>
```

```
    <srs>epsg:4326</srs>
    <matrixProvider>SMARTROUTING</matrixProvider>
</optimRouteRequestV3>
```

Réponse

La réponse est toujours encodée en UTF-8.

Format XML

```
<optimRouteResultV3>
   <steps>
      <step>
         <x>-1.5291995940651142</x>
         <y>47.19158811606974</y>
         <id>1</id>
         <duration>0</duration>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>0</driveDistanceBefore>
         <driveDistanceAfter>1779</driveDistanceAfter>
         <driveTimeBefore>0</driveTimeBefore>
         <driveTimeAfter>372710</driveTimeAfter>
         <timeWindows>
            <timeWindow>
               <start>0</start>
               <end>0</end>
            </timeWindow>
         </timeWindows>
      </step>
      <step>
         <x>-1.5391738246474214</x>
         <y>47.193576663578824</y>
         <id>3</id>
         <duration>0</duration>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>1779</driveDistanceBefore>
         <driveDistanceAfter>3486</driveDistanceAfter>
         <driveTimeBefore>372710</driveTimeBefore>
         <driveTimeAfter>752870</driveTimeAfter>
         <timeWindows/>
      </step>
      <step>
         <x>-1.5562968681202867</x>
         <y>47.21808294131743</y>
         <id>4</id>
         <duration>0</duration>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>3486</driveDistanceBefore>
         <driveDistanceAfter>3175</driveDistanceAfter>
         <driveTimeBefore>752870</driveTimeBefore>
         <driveTimeAfter>662230</driveTimeAfter>
         <timeWindows/>
      </step>
      <step>
         <x>-1.5315788375137436</x>
         <y>47.209701524818236</y>
         <id>2</id>
         <duration>0</duration>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>3175</driveDistanceBefore>
         <driveDistanceAfter>2713</driveDistanceAfter>
         <driveTimeBefore>662230</driveTimeBefore>
```

```
         <driveTimeAfter>520090</driveTimeAfter>
         <timeWindows>
            <timeWindow>
               <start>0</start>
               <end>0</end>
            </timeWindow>
         </timeWindows>
      </step>
      <step>
         <x>-1.5405963750884433</x>
         <y>47.19752774053115</y>
         <id>5</id>
         <duration>0</duration>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>2713</driveDistanceBefore>
         <driveDistanceAfter>0</driveDistanceAfter>
         <driveTimeBefore>520090</driveTimeBefore>
         <driveTimeAfter>0</driveTimeAfter>
         <timeWindows/>
      </step>
   </steps>
   <distanceMeters>11153</distanceMeters>
   <durationSeconds>2307</durationSeconds>
</optimRouteResultV3>
```

Retours possibles

Cas d'une optimisation effectuée avec succès

```
<optimRouteResultV3>
   <steps>
      <step>
         <x>-1.5291995940651142</x>
         <y>47.19158811606974</y>
         <id>1</id>
         <duration>0</duration>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>0</driveDistanceBefore>
         <driveDistanceAfter>1779</driveDistanceAfter>
         <driveTimeBefore>0</driveTimeBefore>
         <driveTimeAfter>372710</driveTimeAfter>
         <timeWindows>
            <timeWindow>
               <start>0</start>
               <end>0</end>
            </timeWindow>
         </timeWindows>
      </step>
      <step>
         [...]
      </step>
   </steps>
   <distanceMeters>11153</distanceMeters>
   <durationSeconds>2307</durationSeconds>
</optimRouteResultV3>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… ⇒ erreur avec faultstring qui contient la description

```
<soap:Fault>
    <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
```

```
    <faultstring>Error in matrix computation
        Error in smartrouting
        datasource is null</faultstring>
</soap:Fault>
```

V2

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
origin (optimRouteStepV2)	point d'origine (id,x,y)	oui	
destination (optimRouteStepV2)	point de destination (id,x,y)	oui	
steps (optimRouteStepV2)	points à visiter(id,x,y;id,x,y;…) Plusiers plages de visites en millisecondes peuvent être précisées pour chaque point : (id,x,y,début,fin)	non	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
matrixProvider	fournisseur de matrice de calcul globe : vol d'oiseau sur globe (les coordonnées doivent être en long/lat) euclidean : vol d'oiseau sur carte (les coordonnées doivent être en mètres) smartrouting : calcul sur le réseau routier par SmartRouting nokia : calcul sur le réseau routier par Nokia services provided : paramètre de matrice	oui	smartrouting
matrix	matrice de temps/distance (id1,id2,temps,distance;…)	oui	
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	wgs84
directSpeed	Vitesse par défaut lors de l'utilisation des matrices globe ou euclidean	oui	50

Etapes (optimRouteStepV2)

paramètre	type	min/max	description
id	string	0/1	id du point
duration	long	1/1	durée d'une visite, en millisecondes. 0 par défaut
timeWindows/ timeWindow (optimTimeWindow)	long	1/1	Fenêtres horaires

Fenêtres horaires (optimTimeWindow)

paramètre	type	min/max	description
start	long	1/1	début de plage horaire, en millisecondes. 0 par défaut
end	long	1/1	fin de plage horaire, en millisecondes. Long.MAX_VALUE par défaut

En sortie

paramètre	type	min/max	description
steps/step	array (optimRouteStepV2)	0/illimité	points à visiter ordonnancés

paramètre	type	min/max	description
distanceMeters	long	1/1	distance totale du trajet en mètres
durationSeconds	long	1/1	temps total du trajet en secondes

Etapes (optimRouteStepV2)

paramètre	type	min/max	description
id	string	0/1	id du point
duration	long	1/1	durée d'une visite, en millisecondes. 0 par défaut
effectiveStart	long	1/1	début effectif d'une visite, en millisecondes
driveDistanceBefore	int	1/1	distance de conduite avant l'étape, en mètres
driveDistanceAfter	int	1/1	distance de conduite après l'étape, en mètre
driveTimeBefore	long	1/1	temps de conduite avant l'étape, en millisecondes
driveTimeAfter	long	1/1	temps de conduite après l'étape, en millisecondes
timeWindows/ timeWindow (optimTimeWindow)	long	1/1	Fenêtres horaires

Fenêtres horaires (optimTimeWindow)

paramètre	type	min/max	description
start	long	1/1	début de plage horaire, en millisecondes. 0 par défaut
end	long	1/1	fin de plage horaire, en millisecondes. Long.MAX_VALUE par défaut

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/optimService?wsdl

Requête

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header />
   <soapenv:Body>
      <sch:optimRouteV2>
         <request>
            <origin>
               <x>-1.5291995940651142</x>
               <y>47.19158811606974</y>
               <id>1</id>
               <duration>3600000</duration>
               <effectiveStart>0</effectiveStart>
               <driveDistanceBefore>0</driveDistanceBefore>
               <driveDistanceAfter>0</driveDistanceAfter>
               <driveTimeBefore>0</driveTimeBefore>
               <driveTimeAfter>0</driveTimeAfter>
            </origin>
            <destination>
               <x>-1.5405963750884433</x>
               <y>47.19752774053115</y>
```

```
               <id>5</id>
               <duration>3600000</duration>
               <effectiveStart>0</effectiveStart>
               <driveDistanceBefore>0</driveDistanceBefore>
               <driveDistanceAfter>0</driveDistanceAfter>
               <driveTimeBefore>0</driveTimeBefore>
               <driveTimeAfter>0</driveTimeAfter>
            </destination>
            <steps>
               <step>
                  <x>-1.5315788375137436</x>
                  <y>47.209701524818236</y>
                  <id>2</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>0</driveDistanceBefore>
                  <driveDistanceAfter>0</driveDistanceAfter>
                  <driveTimeBefore>0</driveTimeBefore>
                  <driveTimeAfter>0</driveTimeAfter>
                  <timeWindows>
                     <!--Zero or more repetitions:-->
                     <timeWindow>
                        <start>0</start>
                        <end>9223372036854775807</end>
                     </timeWindow>
                  </timeWindows>
               </step>
               <step>
                  <x>-1.5391738246474214</x>
                  <y>47.193576663578824</y>
                  <id>3</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>0</driveDistanceBefore>
                  <driveDistanceAfter>0</driveDistanceAfter>
                  <driveTimeBefore>0</driveTimeBefore>
                  <driveTimeAfter>0</driveTimeAfter>
                  <timeWindows>
                     <!--Zero or more repetitions:-->
                     <timeWindow>
                        <start>0</start>
                        <end>9223372036854775807</end>
                     </timeWindow>
                  </timeWindows>
               </step>
               <step>
                  <x>-1.5562968681202867</x>
                  <y>47.21808294131743</y>
                  <id>4</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>0</driveDistanceBefore>
                  <driveDistanceAfter>0</driveDistanceAfter>
                  <driveTimeBefore>0</driveTimeBefore>
                  <driveTimeAfter>0</driveTimeAfter>
               </step>
            </steps>
            <srs>epsg:4326</srs>
            <method>time</method>
         </request>
      </sch:optimRouteV2>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse


```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:optimRouteV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <OptimRouteResult>
            <steps>
               <step>
                  <x>-1.5291995940651142</x>
                  <y>47.19158811606974</y>
                  <id>1</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>0</driveDistanceBefore>
                  <driveDistanceAfter>1779</driveDistanceAfter>
                  <driveTimeBefore>0</driveTimeBefore>
                  <driveTimeAfter>372710</driveTimeAfter>
                  <timeWindows/>
               </step>
               <step>
                  <x>-1.5391738246474214</x>
                  <y>47.193576663578824</y>
                  <id>3</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>1779</driveDistanceBefore>
                  <driveDistanceAfter>3486</driveDistanceAfter>
                  <driveTimeBefore>372710</driveTimeBefore>
                  <driveTimeAfter>752870</driveTimeAfter>
                  <timeWindows>
                     <timeWindow>
                        <start>0</start>
                        <end>9223372036854775807</end>
                     </timeWindow>
                  </timeWindows>
               </step>
               <step>
                  <x>-1.5562968681202867</x>
                  <y>47.21808294131743</y>
                  <id>4</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>3486</driveDistanceBefore>
                  <driveDistanceAfter>3175</driveDistanceAfter>
                  <driveTimeBefore>752870</driveTimeBefore>
                  <driveTimeAfter>662230</driveTimeAfter>
                  <timeWindows/>
               </step>
               <step>
                  <x>-1.5315788375137436</x>
                  <y>47.209701524818236</y>
                  <id>2</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>3175</driveDistanceBefore>
                  <driveDistanceAfter>2713</driveDistanceAfter>
                  <driveTimeBefore>662230</driveTimeBefore>
                  <driveTimeAfter>520090</driveTimeAfter>
                  <timeWindows>
                     <timeWindow>
                        <start>0</start>
                        <end>9223372036854775807</end>
                     </timeWindow>
```

```
                  </timeWindows>
               </step>
               <step>
                  <x>-1.5405963750884433</x>
                  <y>47.19752774053115</y>
                  <id>5</id>
                  <duration>3600000</duration>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>2713</driveDistanceBefore>
                  <driveDistanceAfter>0</driveDistanceAfter>
                  <driveTimeBefore>520090</driveTimeBefore>
                  <driveTimeAfter>0</driveTimeAfter>
                  <timeWindows/>
               </step>
            </steps>
            <distanceMeters>11153</distanceMeters>
            <durationSeconds>2307</durationSeconds>
         </OptimRouteResult>
      </ns2:optimRouteV2Response>
   </soap:Body>
</soap:Envelope>
```

REST (POST)

Requête

Requête

```
http://<server>/<webapp>/api/lbs/optim/route.xml
```

Data (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<optimRouteRequestV2>
   <origin>
      <x>-1.5291995940651142</x>
      <y>47.19158811606974</y>
      <!--Optional:-->
      <id>1</id>
      <timeWindows>
         <timeWindow>
            <start>09:00</start>
            <end>20:00</end>
         </timeWindow>
      </timeWindows>
   </origin>
   <destination>
      <x>-1.5405963750884433</x>
      <y>47.19752774053115</y>
      <!--Optional:-->
      <id>5</id>
   </destination>
   <steps>
      <step>
         <x>-1.5315788375137436</x>
         <y>47.209701524818236</y>
         <id>2</id>
         <timeWindows>
            <timeWindow>
               <start>09:00</start>
               <end>20:00</end>
```

```
            </timeWindow>
         </timeWindows>
      </step>
      <step>
         <x>-1.5391738246474214</x>
         <y>47.193576663578824</y>
         <id>3</id>
      </step>
      <step>
         <x>-1.5562968681202867</x>
         <y>47.21808294131743</y>
         <id>4</id>
      </step>
   </steps>
   <method>time</method>
   <srs>epsg:4326</srs>
   <matrixProvider>SMARTROUTING</matrixProvider>
</optimRouteRequestV2>
```

Réponse

La réponse est toujours encodée en UTF-8.

Format XML

```
<optimRouteResultV2>
   <steps>
      <step>
         <x>-1.5291995940651142</x>
         <y>47.19158811606974</y>
         <id>1</id>
         <duration>0</duration>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>0</driveDistanceBefore>
         <driveDistanceAfter>1779</driveDistanceAfter>
         <driveTimeBefore>0</driveTimeBefore>
         <driveTimeAfter>372710</driveTimeAfter>
         <timeWindows>
            <timeWindow>
               <start>0</start>
               <end>0</end>
            </timeWindow>
         </timeWindows>
      </step>
      <step>
         <x>-1.5391738246474214</x>
         <y>47.193576663578824</y>
         <id>3</id>
         <duration>0</duration>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>1779</driveDistanceBefore>
         <driveDistanceAfter>3486</driveDistanceAfter>
         <driveTimeBefore>372710</driveTimeBefore>
         <driveTimeAfter>752870</driveTimeAfter>
         <timeWindows/>
      </step>
      <step>
         <x>-1.5562968681202867</x>
         <y>47.21808294131743</y>
         <id>4</id>
         <duration>0</duration>
         <effectiveStart>0</effectiveStart>
```

```
         <driveDistanceBefore>3486</driveDistanceBefore>
         <driveDistanceAfter>3175</driveDistanceAfter>
         <driveTimeBefore>752870</driveTimeBefore>
         <driveTimeAfter>662230</driveTimeAfter>
         <timeWindows/>
      </step>
      <step>
         <x>-1.5315788375137436</x>
         <y>47.209701524818236</y>
         <id>2</id>
         <duration>0</duration>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>3175</driveDistanceBefore>
         <driveDistanceAfter>2713</driveDistanceAfter>
         <driveTimeBefore>662230</driveTimeBefore>
         <driveTimeAfter>520090</driveTimeAfter>
         <timeWindows>
            <timeWindow>
               <start>0</start>
               <end>0</end>
            </timeWindow>
         </timeWindows>
      </step>
      <step>
         <x>-1.5405963750884433</x>
         <y>47.19752774053115</y>
         <id>5</id>
         <duration>0</duration>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>2713</driveDistanceBefore>
         <driveDistanceAfter>0</driveDistanceAfter>
         <driveTimeBefore>520090</driveTimeBefore>
         <driveTimeAfter>0</driveTimeAfter>
         <timeWindows/>
      </step>
   </steps>
   <distanceMeters>11153</distanceMeters>
   <durationSeconds>2307</durationSeconds>
</optimRouteResultV2>
```

Retours possibles

Cas d’une optimisation effectuée avec succès

```
<optimRouteResultV2>
   <steps>
      <step>
         <x>-1.5291995940651142</x>
         <y>47.19158811606974</y>
         <id>1</id>
         <duration>0</duration>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>0</driveDistanceBefore>
         <driveDistanceAfter>1779</driveDistanceAfter>
         <driveTimeBefore>0</driveTimeBefore>
         <driveTimeAfter>372710</driveTimeAfter>
         <timeWindows>
            <timeWindow>
               <start>0</start>
               <end>0</end>
            </timeWindow>
         </timeWindows>
```

```
      </step>
      <step>
         [...]
      </step>
   </steps>
   <distanceMeters>11153</distanceMeters>
   <durationSeconds>2307</durationSeconds>
</optimRouteResultV2>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… ⇒ erreur avec faultstring qui contient la description

```
<soap:Fault>
    <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
    <faultstring>Error in matrix computation
        Error in smartrouting
        datasource is null</faultstring>
</soap:Fault>
```

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
origin	point d'origine (id,x,y)	oui	
destination	point de destination (id,x,y)	oui	
steps	points à visiter(id,x,y;id,x,y;…) Les plages de visites en millisecondes peuvent être précisées pour chaque point : (id,x,y,début,fin)	non	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
matrixProvider	founrnisseur de matrice de calcul globe : vol d'oiseau sur globe (les coordonnées doivent être en long/lat) euclidean : vol d'oiseau sur carte (les coordonnées doivent être en mètres) smartrouting : calcul sur le réseau routier par SmartRouting nokia : calcul sur le réseau routier par Nokia services provided : paramètre de matrice	oui	smartrouting
matrix	matrice de temps/distance (id1,id2,temps,distance;…)	oui	
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	wgs84
directSpeed	Vitesse par défaut lors de l'utilisation des matrices globe ou euclidean	oui	50

En sortie

paramètre	type	min/max	description
steps/step	array (optimRouteStep)	0/illimité	points à visiter ordonnancés
totalDistance	long	1/1	distance totale du trajet en mètres
totalTime	long	1/1	temps total du trajet en minutes

Etapes (optimRouteStep)

paramètre	type	min/max	description
id	string	0/1	id du point
duration	long	1/1	durée d'une visite, en millisecondes. 0 par défaut
timeWindowStart	long	1/1	début de plage horaire, en millisecondes. 0 par défaut
timeWindowEnd	long	1/1	fin de plage horaire, en millisecondes. Long.MAX_VALUE par défaut
effectiveStart	long	1/1	début effectif d'une visite, en millisecondes
driveDistanceBefore	int	1/1	distance de conduite avant l'étape, en mètres
driveDistanceAfter	int	1/1	distance de conduite après l'étape, en mètre
driveTimeBefore	long	1/1	temps de conduite avant l'étape, en millisecondes
driveTimeAfter	long	1/1	temps de conduite après l'étape, en millisecondes

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/optimService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:optimRoute>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <origin>
               <x>-1.55306</x>
               <y>47.21812</y>
               <!--Optional:-->
               <id>0</id>
               <duration>0</duration>
               <timeWindowStart>0</timeWindowStart>
               <timeWindowEnd>0</timeWindowEnd>
               <effectiveStart>0</effectiveStart>
               <driveDistanceBefore>0</driveDistanceBefore>
               <driveDistanceAfter>0</driveDistanceAfter>
               <driveTimeBefore>0</driveTimeBefore>
               <driveTimeAfter>0</driveTimeAfter>
            </origin>
            <!--Optional:-->
            <destination>
               <x>-2.2164099</x>
               <y>47.2806206</y>
               <!--Optional:-->
               <id>1</id>
               <duration>0</duration>
               <timeWindowStart>0</timeWindowStart>
               <timeWindowEnd>0</timeWindowEnd>
               <effectiveStart>0</effectiveStart>
               <driveDistanceBefore>0</driveDistanceBefore>
               <driveDistanceAfter>0</driveDistanceAfter>
               <driveTimeBefore>0</driveTimeBefore>
               <driveTimeAfter>0</driveTimeAfter>
```

```
            </destination>
            <!--Optional:-->
            <steps>
               <!--Zero or more repetitions:-->
               <step>
                  <x>-2.02616</x>
                  <y>47.2835503</y>
                  <!--Optional:-->
               <id>2</id>
               <duration>0</duration>
               <timeWindowStart>0</timeWindowStart>
               <timeWindowEnd>0</timeWindowEnd>
               <effectiveStart>0</effectiveStart>
               <driveDistanceBefore>0</driveDistanceBefore>
               <driveDistanceAfter>0</driveDistanceAfter>
               <driveTimeBefore>0</driveTimeBefore>
               <driveTimeAfter>0</driveTimeAfter>
               </step>
            </steps>
            <!--Optional:-->
            <srs>wgs84</srs>
            <!--Optional:-->
            <method>time</method>
            <!--Optional:-->
            <matrixProvider>SMARTROUTING</matrixProvider>
            <!--Optional:-->
            <matrix>
               <!--Optional:-->
               <elements>
                  <!--Zero or more repetitions:-->
                  <element>
                     <!--Optional:-->
                     <origin></origin>
                     <!--Optional:-->
                     <destination></destination>
                     <time>1</time>
                     <distance>1</distance>
                  </element>
               </elements>
            </matrix>
            <directSpeed>0</directSpeed>
         </request>
      </sch:optimRoute>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:optimRouteResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <OptimRouteResult>
            <steps>
               <step>
                  <x>-1.55306</x>
                  <y>47.21812</y>
                  <id>0</id>
                  <duration>0</duration>
                  <timeWindowStart>0</timeWindowStart>
                  <timeWindowEnd>0</timeWindowEnd>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>0</driveDistanceBefore>
                  <driveDistanceAfter>47085</driveDistanceAfter>
```

```
                  <driveTimeBefore>0</driveTimeBefore>
                  <driveTimeAfter>3139890</driveTimeAfter>
               </step>
               <step>
                  <x>-2.02616</x>
                  <y>47.2835503</y>
                  <id>2</id>
                  <duration>0</duration>
                  <timeWindowStart>0</timeWindowStart>
                  <timeWindowEnd>0</timeWindowEnd>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>47085</driveDistanceBefore>
                  <driveDistanceAfter>22124</driveDistanceAfter>
                  <driveTimeBefore>3139890</driveTimeBefore>
                  <driveTimeAfter>1488050</driveTimeAfter>
               </step>
               <step>
                  <x>-2.2164099</x>
                  <y>47.2806206</y>
                  <id>1</id>
                  <duration>0</duration>
                  <timeWindowStart>0</timeWindowStart>
                  <timeWindowEnd>0</timeWindowEnd>
                  <effectiveStart>0</effectiveStart>
                  <driveDistanceBefore>22124</driveDistanceBefore>
                  <driveDistanceAfter>0</driveDistanceAfter>
                  <driveTimeBefore>1488050</driveTimeBefore>
                  <driveTimeAfter>0</driveTimeAfter>
               </step>
            </steps>
            <totalDistance>69209</totalDistance>
            <totalTime>77</totalTime>
         </OptimRouteResult>
      </ns2:optimRouteResponse>
   </soap:Body>
</soap:Envelope>
```

REST (POST)

Requête

Requête

```
http://<server>/<webapp>/api/lbs/optim/route.xml
```

Data (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<optimRouteRequest>
        <steps>
                <step>
                        <x>-1.55306</x>
                        <y>47.21812</y>
                        <id>0</id>
                </step>
                <step>
                        <x>-2.2164099</x>
                        <y>47.2806206</y>
                        <id>1</id>
                </step>
                <step>
```

```
                        <x>-2.02616</x>
                        <y>47.2835503</y>
                        <id>2</id>
                </step>
        </steps>
        <method>time</method>
        <srs>wgs84</srs>
        <matrixProvider>SMARTROUTING</matrixProvider>
</optimRouteRequest>
```

Réponse

La réponse est toujours encodée en UTF-8.

Format XML

```
<optimRouteResult>
   <steps>
      <step>
         <x>-2.02616</x>
         <y>47.2835503</y>
         <id>2</id>
         <duration>0</duration>
         <timeWindowStart>0</timeWindowStart>
         <timeWindowEnd>9223372036854775807</timeWindowEnd>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>0</driveDistanceBefore>
         <driveDistanceAfter>22124</driveDistanceAfter>
         <driveTimeBefore>0</driveTimeBefore>
         <driveTimeAfter>1488050</driveTimeAfter>
      </step>
      <step>
         <x>-2.2164099</x>
         <y>47.2806206</y>
         <id>1</id>
         <duration>0</duration>
         <timeWindowStart>0</timeWindowStart>
         <timeWindowEnd>9223372036854775807</timeWindowEnd>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>22124</driveDistanceBefore>
         <driveDistanceAfter>63139</driveDistanceAfter>
         <driveTimeBefore>1488050</driveTimeBefore>
         <driveTimeAfter>2992570</driveTimeAfter>
      </step>
      <step>
         <x>-1.55306</x>
         <y>47.21812</y>
         <id>0</id>
         <duration>0</duration>
         <timeWindowStart>0</timeWindowStart>
         <timeWindowEnd>9223372036854775807</timeWindowEnd>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>63139</driveDistanceBefore>
         <driveDistanceAfter>0</driveDistanceAfter>
         <driveTimeBefore>2992570</driveTimeBefore>
         <driveTimeAfter>0</driveTimeAfter>
      </step>
   </steps>
   <totalDistance>85263</totalDistance>
   <totalTime>74</totalTime>
</optimRouteResult>
```

REST (GET)

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/optim/route.json?
steps=0,-1.55306,47.21812;1,-2.2164099,47.2806206;2,-2.02616,47.2835503&method=time&srs=wgs84
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/optim/route.json?
steps=0,-1.55306,47.21812;1,-2.2164099,47.2806206;2,-2.02616,47.2835503&method=time&srs=wgs84&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/optim/route.xml?
steps=0,-1.55306,47.21812;1,-2.2164099,47.2806206;2,-2.02616,47.2835503&method=time&srs=wgs84
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "steps":    [
            {
         "x": -2.02616, "y": 47.2835503, "id": "2",
         "duration": 0, "timeWindowStart": 0, "timeWindowEnd": 9223372036854775807, "effectiveStart": 0,
         "driveDistanceBefore": 0, "driveDistanceAfter": 22124, "driveTimeBefore": 0, "driveTimeAfter":
 1488050
      },
            {
         "x": -2.2164099, "y": 47.2806206, "id": "1",
         "duration": 0, "timeWindowStart": 0, "timeWindowEnd": 9223372036854775807, "effectiveStart": 0,
         "driveDistanceBefore": 22124, "driveDistanceAfter": 63139, "driveTimeBefore": 1488050,
 "driveTimeAfter": 2992570
      },
            {
         "x": -1.55306, "y": 47.21812, "id": "0",
         "duration": 0, "timeWindowStart": 0, "timeWindowEnd": 9223372036854775807, "effectiveStart": 0,
         "driveDistanceBefore": 63139, "driveDistanceAfter": 0, "driveTimeBefore": 2992570,
 "driveTimeAfter": 0
      }
   ],
   "totalDistance": 85263,
   "totalTime": 74
}
```

Format JSON-P

```
myCallback({
   "steps":    [
            {
         "x": -2.02616, "y": 47.2835503, "id": "2",
```

```
         "duration": 0, "timeWindowStart": 0, "timeWindowEnd": 9223372036854775807, "effectiveStart": 0,
         "driveDistanceBefore": 0, "driveDistanceAfter": 22124, "driveTimeBefore": 0, "driveTimeAfter":
 1488050
      },
            {
         "x": -2.2164099, "y": 47.2806206, "id": "1",
         "duration": 0, "timeWindowStart": 0, "timeWindowEnd": 9223372036854775807, "effectiveStart": 0,
         "driveDistanceBefore": 22124, "driveDistanceAfter": 63139, "driveTimeBefore": 1488050,
 "driveTimeAfter": 2992570
      },
            {
         "x": -1.55306, "y": 47.21812, "id": "0",
         "duration": 0, "timeWindowStart": 0, "timeWindowEnd": 9223372036854775807, "effectiveStart": 0,
         "driveDistanceBefore": 63139, "driveDistanceAfter": 0, "driveTimeBefore": 2992570,
 "driveTimeAfter": 0
      }
   ],
   "totalDistance": 85263,
   "totalTime": 74
});
```


Format XML


```
<optimRouteResult>
   <steps>
      <step>
         <x>-2.02616</x>
         <y>47.2835503</y>
         <id>2</id>
         <duration>0</duration>
         <timeWindowStart>0</timeWindowStart>
         <timeWindowEnd>9223372036854775807</timeWindowEnd>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>0</driveDistanceBefore>
         <driveDistanceAfter>22124</driveDistanceAfter>
         <driveTimeBefore>0</driveTimeBefore>
         <driveTimeAfter>1488050</driveTimeAfter>
      </step>
      <step>
         <x>-2.2164099</x>
         <y>47.2806206</y>
         <id>1</id>
         <duration>0</duration>
         <timeWindowStart>0</timeWindowStart>
         <timeWindowEnd>9223372036854775807</timeWindowEnd>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>22124</driveDistanceBefore>
         <driveDistanceAfter>63139</driveDistanceAfter>
         <driveTimeBefore>1488050</driveTimeBefore>
         <driveTimeAfter>2992570</driveTimeAfter>
      </step>
      <step>
         <x>-1.55306</x>
         <y>47.21812</y>
         <id>0</id>
         <duration>0</duration>
         <timeWindowStart>0</timeWindowStart>
         <timeWindowEnd>9223372036854775807</timeWindowEnd>
         <effectiveStart>0</effectiveStart>
         <driveDistanceBefore>63139</driveDistanceBefore>
         <driveDistanceAfter>0</driveDistanceAfter>
         <driveTimeBefore>2992570</driveTimeBefore>
         <driveTimeAfter>0</driveTimeAfter>
```

```
      </step>
   </steps>
   <totalDistance>85263</totalDistance>
   <totalTime>74</totalTime>
</optimRouteResult>
```

Retours possibles

Cas d'une optimisation effectuée avec succès

```
<ns2:optimRouteResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
 <OptimRouteResult>
    <steps>
       <step>
          <x>-1.55306</x>
          <y>47.21812</y>
          <id>0</id>
          <duration>0</duration>
          <timeWindowStart>0</timeWindowStart>
          <timeWindowEnd>0</timeWindowEnd>
          <effectiveStart>0</effectiveStart>
          <driveDistanceBefore>0</driveDistanceBefore>
          <driveDistanceAfter>47085</driveDistanceAfter>
          <driveTimeBefore>0</driveTimeBefore>
          <driveTimeAfter>3139890</driveTimeAfter>
       </step>
       <step>
            ...
       </step>
    </steps>
    <totalDistance>69209</totalDistance>
    <totalTime>77</totalTime>
 </OptimRouteResult>
</ns2:optimRouteResponse>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… ⇒ erreur avec faultstring qui contient la description

```
<soap:Fault>
    <faultcode xmlns:ns1="geoconcept.com">ns1:8</faultcode>
    <faultstring>Error in matrix computation
        Error in smartrouting
        datasource is null</faultstring>
</soap:Fault>
```

FAQ

1. Peut-on forcer une étape de départ et/ou d'arrivée dans le calcul de la tournée ?

   Oui, il suffit d'utiliser respectivement les paramètres origin et/ou destination. Le parcours entre les étapes steps est optimisé en tenant compte des contraintes opérationnelles et sans tenir compte de contrainte logistique (départ d'un domicile, arrivée dans un dépôt, …).

2. Comment préciser, pour chaque étape, une durée et une plage horaire de visite ?

   A l'aide du paramètre duration (pour indiquer la durée prévue d'une visite), et des paramètres timeWindowStart et timeWindowEnd (pour préciser l'heure de début et l'heure de fin de visite.

Calcul d'isochrone/isodistance

Cette page n'est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

Ce web service calcule un isochrone/isodistance depuis un point et retourne la géométrie de la zone calculée. Il s'appuie sur le graphe paramétré dont le nom a été spécifié dans l'interface d'administration de Geoconcept Web.

Disponibilité

Ce web service est disponible en permanence avec Geoconcept Web et un graphe.

Changement de version

Les versions précédente du web service sont conservées dans Geoconcept Web pour assurer la compatibilité avec les développements antérieurs. Il est recommandé d'utiliser la version la plus récente.

Changements avec la V4

- Ajout des paramètres "timeOut" et "computeOptions".

Changements avec la V3

- Ajout des paramètres "avoidArea" et "configName".

Changements avec la V2

- Ajout des paramètres "startDateTime" et "snapMethod"

V4

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
id	Identifiant de l'isochrone	oui	
location	Départ (ou arrivée si le `reverse` est à true). Les coordonnées sont séparées par la caractère ,	non	
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	epsg:4326
graphName (déprécié)	Nom du graphe à utiliser Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileId (déprécié)	Identifiant du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileName (déprécié)	Profil du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère ; (Exemple : Toll, Tunnel, Bridge)	oui	

paramètre	description	optionnel	défaut
method	"time" pour isochrone ou "distance" for isodistance	non	time
time	Temps maximum d'accès, en secondes	oui	
distance	Distance maximum d'accès, en mètres	oui	
reverse	si *true*, le `location` est considéré comme arrivée	oui	false
smoothing	Lissage	oui	false
holes	Afficher les trous dans la zone résultante (la géométrie retournée est plus volumineuse lorsque ce paramètre est à *true*)	oui	false
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris. Attention le caractère + peut être mal interprété par les navigateurs, dans ce cas, il faut le remplacer par *%2B*.	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction	oui	standard
avoidArea	Zone de transit interdit au format WKT (POLYGON ou MULTIPOLYGON) dans la projection (paramètre srs) demandée Exemple en wgs84 : `POLYGON ((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689))` - `MULTIPOLYGON (((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689)), ((-1.558639 47.218144, -1.558639 47.218193, [...] -1.558639 47.218144)))` Attention les géométries WKT doivent être fermées.	oui	
configName	Nom de la configuration à utiliser (défini dans Geoconcept Web - Administration / Outils / Définitions des graphes) Il remplace l'usage de graphName, profileId et profileName	oui	
timeOut	Time out pour le calcul (en millisecondes)	oui	
computeOptions	Liste des options pour le calcul, séparées par le caractère *;* - trafficPatterns : utilise les statistiques routières (il est nécessaire de renseigner le paramètre startDateTime et d'utiliser un graphe intégrant les informations de traffic patterns) - speedPattern (M18) : utilise d'une *speed pattern* tel que définis dans le fichier SmartRoutingVehicles.xml qui se trouve dans le dossier `*'<GEOCONCEPT_WEB_HOME>'\smartrouting\jee\smartrouting\conf* . Usage : "speedPattern:slow-speed" - length (M18) : longueur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "length:950" - width (M18) : largeur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "width:255" - height (M18) : hauteur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "height:360" - weight (M18) : poids maximum autorisé en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weight:18000" - axles (M18) : nombre maximum d'axe autorisé (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "axles:2"	oui	

paramètre	description	optionnel	défaut
	- weightPerAxle (M18) : poids maximum autorisé par axe en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weightPerAxle:9000" - snapSpeed : vitesse d'accrochage au graphe en kilomètre par heure. Usage : "snapSpeed:10"		

(M18) Disponible à partir de la version M18 des graphes fournis par GEOCONCEPT SAS.

En sortie

Itinéraire (isochroneResult)

paramètre	type	min/max	description
id	string	0/1	Identifiant de l'isochrone
location	string	0/1	Départ (ou arrivée si le `reverse` est à true).
srs	string	0/1	projection
time	string	0/1	Temps maximum d'accès, en secondes
distance	string	0/1	Distance maximum d'accès, en mètres
wktGeometry	string	0/1	Géometrie de l'isochrone, au format wkt

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/isochroneService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:isochroneV4>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <id>1</id>
            <location>
               <x>-1.557189</x>
               <y>47.217122</y>
            </location>
            <!--Optional:-->
            <srs></srs>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
```

```
            <method>distance</method>
            <!--Optional:-->
            <time></time>
            <!--Optional:-->
            <distance>5000</distance>
            <!--Optional:-->
            <reverse></reverse>
            <!--Optional:-->
            <smoothing></smoothing>
            <!--Optional:-->
            <holes></holes>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <snapMethod></snapMethod>
            <!--Optional:-->
            <avoidArea></avoidArea>
            <!--Optional:-->
            <configName></configName>
            <!--Optional:-->
            <computeOptions></computeOptions>
            <!--Optional:-->
            <timeOut></timeOut>
            </request>
      </sch:isochroneV4>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:isochroneV4Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <IsochroneResult>
            <status>OK</status>
            <id>1</id>
            <location>-1.557189,47.217122</location>
            <srs />
            <distance>5000</distance>
            <wktGeometry>POLYGON ((-1.544603 47.179044, -1.545261 47.179044, [...], -1.544603 47.179044))</
wktGeometry>
         </IsochroneResult>
      </ns2:isochroneV4Response>
   </soap:Body>
</soap:Envelope>
```

REST (POST)

Requête

Requête

```
http://<server>/<webapp>/api/lbs/isochrone/v4.xml
```

Data (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<isochroneRequestV4>
  <id></id>
  <location>
```

```
    <x>-1.557189</x>
    <y>47.217122</y>
  </location>
  <srs></srs>
  <graphName></graphName>
  <profileId></profileId>
  <profileName></profileName>
  <exclusions>
    <exclusion></exclusion>
    <exclusion></exclusion>
    <!--...more "exclusion" elements...-->
  </exclusions>
  <method></method>
  <time>50</time>
  <distance></distance>
  <reverse></reverse>
  <smoothing></smoothing>
  <holes></holes>
  <startDateTime></startDateTime>
  <snapMethod></snapMethod>
  <avoidArea></avoidArea>
  <configName></configName>
</isochroneRequestV4>
```

Réponse

La réponse est toujours encodée en UTF-8.

Format XML

```
<isochroneResultV4>
   <status>OK</status>
   <id>1</id>
   <location>-1.557189,47.217122</location>
   <srs/>
   <time>50</time>
   <wktGeometry>POLYGON ((-1.556864 47.216487, -1.556864 47.216948, [...]))</wktGeometry>
</isochroneResultV4>
```

REST (GET)

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/isochrone/v4.json?
location=-1.557189,47.217122&method=distance&distance=5000
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/isochrone/v4.json?
location=-1.557189,47.217122&method=distance&distance=5000&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/isochrone/v4.xml?
location=-1.557189,47.217122&method=distance&distance=2000&holes=true
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message": null,
   "status": "OK",
   "id": null,
   "location": "-1.557189,47.217122",
   "srs": "epsg:4326",
   "time": null,
   "distance": "5000",
   "wktGeometry": "POLYGON ((-1.545312 47.178178, -1.545312 47.180418, ...))"
}
```

Format JSON-P

```
myCallback(
    {
       "message": null,
       "status": "OK",
       "id": null,
       "location": "-1.557189,47.217122",
       "srs": "EPSG:4326",
       "time": null,
       "distance": "5000",
       "wktGeometry": "POLYGON ((-1.545312 47.178178, -1.545312 47.180418, ...))"
    }
);
```

Format XML

```
<isochroneResultV4>
   <status>OK</status>
   <location>-1.557189,47.217122</location>
   <srs>EPSG:4326</srs>
   <distance>2000</distance>
   <wktGeometry>POLYGON ((-1.546926 47.202309, -1.546926 47.202752, ...))</wktGeometry>
</isochroneResultV4>
```

Retours possibles

Cas d'un isochrone/isodistance trouvé (isochroneResultV2/status est OK)

```
<isochroneResultV4>
   <status>OK</status>
   <location>-1.557189,47.217122</location>
   <srs>EPSG:4326</srs>
   <distance>2000</distance>
   <wktGeometry>POLYGON ((-1.546926 47.202309, -1.546926 47.202752, ...))</wktGeometry>
</isochroneResultV4>
```

Cas de method method = distance et distance non renseigné (isochroneResult/status est ERROR)

```
<isochroneResultV4>
    <message>distance parameter must not be null or 0 !</message>
```

```
    <status>ERROR</status>
</isochroneResultV4>
```

Cas de method + time non renseigné (isochroneResult/status est ERROR)

```
<isochroneResultV4>
    <message>time parameter must not be null or 0 !</message>
    <status>ERROR</status>
</isochroneResultV4>
```

Cas ou la balise location est manquante (isochroneResult/status est ERROR)

```
<isochroneResultV4>
    <message>Location must be not null</message>
    <status>ERROR</status>
</isochroneResultV4>
```

Cas ou le point de départ est mal renseigné (isochroneResult/status est ERROR)

```
<isochroneResultV4>
   <message>Location point must have 2 components separated with a ,</message>
   <status>ERROR</status>
</isochroneResultV4>
```

Cas d'une erreur d'accrochage au graphe (serviceResult/status est ERROR)

```
<serviceResult>
        <message>
                ServiceException: Error in isochrone computation Error in smartrouting Failed to execute
 calculateConcentricReachableAreas com.geoconcept.smartrouting.SmartRoutingNativeException: failed to
 connect isochrone orign { 52.324230, 48.803256, 0.000000 } failed to connect isochrone orign { 52.324230,
 48.803256, 0.000000 }
        </message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in isochrone computation
        Error in smartrouting
        datasource is null
    </message>
   <status>ERROR</status>
</serviceResult>
```

V3

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
id	Identifiant de l'isochrone	oui	

paramètre	description	optionnel	défaut
location	Départ (ou arrivée si le `reverse` est à true). Les coordonnées sont séparées par la caractère ,	non	
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	epsg:4326
graphName	Nom du graphe à utiliser Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileId	Identifiant du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileName	Profil du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère , ou *;* (Exemple : "Toll", "Tunnel", "Bridge")	oui	
method	"time" pour isochrone ou "distance" for isodistance	non	time
time	Temps maximum d'accès, en secondes	oui	
distance	Distance maximum d'accès, en mètres	oui	
reverse	si *true*, le `location` est considéré comme arrivée	oui	false
smoothing	Lissage	oui	false
holes	Afficher les trous dans la zone résultante (la géométrie retournée est plus volumineuse lorsque ce paramètre est à *true*)	oui	false
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris. Attention le caractère + peut être mal interprété par les navigateurs, dans ce cas, il faut le remplacer par *%2B*.	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction	oui	standard
avoidArea	Zone de transit interdit au format WKT (POLYGON ou MULTIPOLYGON) dans la projection (paramètre srs) demandée Exemple en wgs84 : `POLYGON ((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689))` - `MULTIPOLYGON (((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689)), ((-1.558639 47.218144, -1.558639 47.218193, [...] -1.558639 47.218144)))` Attention les géométries WKT doivent être fermées.	oui	
configName	Nom de la configuration à utiliser (défini dans Geoconcept Web - Administration / Outils / Définitions des graphes) Il remplace l'usage de graphName, profileId et profileName	oui	

En sortie

Itinéraire (isochroneResult)

paramètre	type	min/max	description
id	string	0/1	Identifiant de l'isochrone
location	string	0/1	Départ (ou arrivée si le `reverse` est à true).

paramètre	type	min/max	description
srs	string	0/1	projection
time	string	0/1	Temps maximum d'accès, en secondes
distance	string	0/1	Distance maximum d'accès, en mètres
wktGeometry	string	0/1	Géometrie de l'isochrone, au format wkt

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/isochroneService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:isochroneV3>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <id>1</id>
            <location>
               <x>-1.557189</x>
               <y>47.217122</y>
            </location>
            <!--Optional:-->
            <srs></srs>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <method>distance</method>
            <!--Optional:-->
            <time></time>
            <!--Optional:-->
            <distance>5000</distance>
            <!--Optional:-->
            <reverse></reverse>
            <!--Optional:-->
            <smoothing></smoothing>
            <!--Optional:-->
            <holes></holes>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <snapMethod></snapMethod>
            <!--Optional:-->
            <avoidArea></avoidArea>
            <!--Optional:-->
```

```
            <configName></configName>
         </request>
      </sch:isochroneV3>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:isochroneV3Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <IsochroneResult>
            <status>OK</status>
            <id>1</id>
            <location>-1.557189,47.217122</location>
            <srs/>
            <distance>5000</distance>
            <wktGeometry>POLYGON ((-1.544603 47.179044, -1.545261 47.179044, [...]))</wktGeometry>
         </IsochroneResult>
      </ns2:isochroneV3Response>
   </soap:Body>
</soap:Envelope>
```

REST (POST)

Requête

Requête

```
http://<server>/<webapp>/api/lbs/isochrone/v3.xml
```

Data (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<isochroneRequestV3>
  <id></id>
  <location>
    <x>-1.557189</x>
    <y>47.217122</y>
  </location>
  <srs></srs>
  <graphName></graphName>
  <profileId></profileId>
  <profileName></profileName>
  <exclusions>
    <exclusion></exclusion>
    <exclusion></exclusion>
    <!--...more "exclusion" elements...-->
  </exclusions>
  <method></method>
  <time>50</time>
  <distance></distance>
  <reverse></reverse>
  <smoothing></smoothing>
  <holes></holes>
  <startDateTime></startDateTime>
  <snapMethod></snapMethod>
  <avoidArea></avoidArea>
  <configName></configName>
```

```
</isochroneRequestV3>
```

Réponse

La réponse est toujours encodée en UTF-8.

Format XML

```
<isochroneResultV3>
   <status>OK</status>
   <id>1</id>
   <location>-1.557189,47.217122</location>
   <srs/>
   <time>50</time>
   <wktGeometry>POLYGON ((-1.556864 47.216487, -1.556864 47.216948, [...]))</wktGeometry>
</isochroneResultV3>
```

REST (GET)

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/isochrone/v3.json?
location=-1.557189,47.217122&method=distance&distance=5000
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/isochrone/v3.json?
location=-1.557189,47.217122&method=distance&distance=5000&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/isochrone/v3.xml?
location=-1.557189,47.217122&method=distance&distance=2000&holes=true
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message": null,
   "status": "OK",
   "id": null,
   "location": "-1.557189,47.217122",
   "srs": "EPSG:4326",
   "time": null,
   "distance": "5000",
   "wktGeometry": "POLYGON ((-1.545312 47.178178, -1.545312 47.180418, ...))"
}
```

Format JSON-P

```
myCallback(
    {
       "message": null,
       "status": "OK",
       "id": null,
       "location": "-1.557189,47.217122",
       "srs": "EPSG:4326",
       "time": null,
       "distance": "5000",
       "wktGeometry": "POLYGON ((-1.545312 47.178178, -1.545312 47.180418, ...))"
    }
);
```

Format XML

```
<isochroneResultV3>
   <status>OK</status>
   <location>-1.557189,47.217122</location>
   <srs>EPSG:4326</srs>
   <distance>2000</distance>
   <wktGeometry>POLYGON ((-1.546926 47.202309, -1.546926 47.202752, ...))</wktGeometry>
</isochroneResultV3>
```

Retours possibles

Cas d'un isochrone/isodistance trouvé (isochroneResultV2/status est OK)

```
<isochroneResultV3>
   <status>OK</status>
   <location>-1.557189,47.217122</location>
   <srs>EPSG:4326</srs>
   <distance>2000</distance>
   <wktGeometry>POLYGON ((-1.546926 47.202309, -1.546926 47.202752, ...))</wktGeometry>
</isochroneResultV3>
```

Cas de method method = distance et distance non renseigné (isochroneResult/status est ERROR)

```
<isochroneResultV3>
    <message>distance parameter must not be null or 0 !</message>
    <status>ERROR</status>
</isochroneResultV3>
```

Cas de method + time non renseigné (isochroneResult/status est ERROR)

```
<isochroneResultV3>
    <message>time parameter must not be null or 0 !</message>
    <status>ERROR</status>
</isochroneResultV3>
```

Cas ou la balise location est manquante (isochroneResult/status est ERROR)

```
<isochroneResultV3>
    <message>Location must be not null</message>
    <status>ERROR</status>
</isochroneResultV3>
```

Cas ou le point de départ est mal renseigné (isochroneResult/status est ERROR)

```
<isochroneResultV3>
   <message>Location point must have 2 components separated with a ,</message>
   <status>ERROR</status>
</isochroneResultV3>
```

Cas d'une erreur d'accrochage au graphe (serviceResult/status est ERROR)

```
<serviceResult>
        <message>
                ServiceException: Error in isochrone computation Error in smartrouting Failed to execute
 calculateConcentricReachableAreas com.geoconcept.smartrouting.SmartRoutingNativeException: failed to
 connect isochrone orign { 52.324230, 48.803256, 0.000000 } failed to connect isochrone orign { 52.324230,
 48.803256, 0.000000 }
        </message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in isochrone computation
        Error in smartrouting
        datasource is null
    </message>
   <status>ERROR</status>
</serviceResult>
```

V2

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
id	Identifiant de l'isochrone	oui	
location	Départ (ou arrivée si le **`reverse`** est à true). Les coordonnées sont séparées par la caractère ,	non	
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	epsg:4326
graphName	Nom du graphe à utiliser	oui	
profileId	Identifiant du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
profileName	Profil du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère , ou ; (Exemple : "Toll", "Tunnel", "Bridge")	oui	
method	"time" pour isochrone ou "distance" for isodistance	non	time
time	Temps maximum d'accès, en secondes	oui	
distance	Distance maximum d'accès, en mètres	oui	
reverse	si *true*, le **`location`** est considéré comme arrivée	oui	false
smoothing	Lissage	oui	false

paramètre	description	optionnel	défaut
holes	Afficher les trous dans la zone résultante (la géométrie retournée est plus volumineuse lorsque ce paramètre est à *true*)	oui	false
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction	oui	standard

En sortie

Itinéraire (isochroneResult)

paramètre	type	min/max	description
id	string	0/1	Identifiant de l'isochrone
location	string	0/1	Départ (ou arrivée si le **`reverse`** est à true).
srs	string	0/1	projection
time	string	0/1	Temps maximum d'accès, en secondes
distance	string	0/1	Distance maximum d'accès, en mètres
wktGeometry	string	0/1	Géometrie de l'isochrone, au format wkt

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/isochroneService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:isochroneV2>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <id>1</id>
            <location>
               <x>-1.557189</x>
               <y>47.217122</y>
            </location>
            <!--Optional:-->
            <srs></srs>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
```

```
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <method>distance</method>
            <!--Optional:-->
            <time></time>
            <!--Optional:-->
            <distance>5000</distance>
            <!--Optional:-->
            <reverse></reverse>
            <!--Optional:-->
            <smoothing></smoothing>
            <!--Optional:-->
            <holes></holes>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <snapMethod></snapMethod>
         </request>
      </sch:isochroneV2>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:isochroneV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <IsochroneResult>
            <status>OK</status>
            <id/>
            <location>-1.557189,47.217122</location>
            <srs/>
            <distance>5000</distance>
            <wktGeometry>POLYGON ((-1.544603 47.179044, -1.545261 47.179044, [...]))</wktGeometry>
         </IsochroneResult>
      </ns2:isochroneV2Response>
   </soap:Body>
</soap:Envelope>
```

REST (POST)

Requête

Requête

```
http://<server>/<webapp>/api/lbs/isochrone/v2.xml
```

Data (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<isochroneRequestV2>
  <id></id>
  <location>
    <x>-1.557189</x>
    <y>47.217122</y>
  </location>
  <srs></srs>
```

```
  <graphName></graphName>
  <profileId></profileId>
  <profileName></profileName>
  <exclusions>
    <exclusion></exclusion>
    <exclusion></exclusion>
    <!--...more "exclusion" elements...-->
  </exclusions>
  <method></method>
  <time>50</time>
  <distance></distance>
  <reverse></reverse>
  <smoothing></smoothing>
  <holes></holes>
  <startDateTime></startDateTime>
  <snapMethod></snapMethod>
</isochroneRequestV2>
```

Réponse

La réponse est toujours encodée en UTF-8.

Format XML

```
<isochroneResultV2>
   <status>OK</status>
   <id>1</id>
   <location>-1.557189,47.217122</location>
   <srs/>
   <time>50</time>
   <wktGeometry>POLYGON ((-1.556864 47.216487, -1.556864 47.216948, [...]))</wktGeometry>
</isochroneResultV2>
```

REST (GET)

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/isochrone/v2.json?
location=-1.557189,47.217122&method=distance&distance=5000
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/isochrone/v2.json?
location=-1.557189,47.217122&method=distance&distance=5000&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/isochrone/v2.xml?
location=-1.557189,47.217122&method=distance&distance=2000&holes=true
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message": null,
   "status": "OK",
   "id": null,
   "location": "-1.557189,47.217122",
   "srs": "EPSG:4326",
   "time": null,
   "distance": "5000",
   "wktGeometry": "POLYGON ((-1.545312 47.178178, -1.545312 47.180418, ...))"
}
```

Format JSON-P

```
myCallback(
    {
       "message": null,
       "status": "OK",
       "id": null,
       "location": "-1.557189,47.217122",
       "srs": "EPSG:4326",
       "time": null,
       "distance": "5000",
       "wktGeometry": "POLYGON ((-1.545312 47.178178, -1.545312 47.180418, ...))"
    }
);
```

Format XML

```
<isochroneResultV2>
   <status>OK</status>
   <location>-1.557189,47.217122</location>
   <srs>EPSG:4326</srs>
   <distance>2000</distance>
   <wktGeometry>POLYGON ((-1.546926 47.202309, -1.546926 47.202752, ...))</wktGeometry>
</isochroneResultV2>
```

Retours possibles

Cas d'un isochrone/isodistance trouvé (isochroneResultV2/status est OK)

```
<isochroneResultV2>
   <status>OK</status>
   <location>-1.557189,47.217122</location>
   <srs>EPSG:4326</srs>
   <distance>2000</distance>
   <wktGeometry>POLYGON ((-1.546926 47.202309, -1.546926 47.202752, ...))</wktGeometry>
</isochroneResultV2>
```

Cas de method method = distance et distance non renseigné (isochroneResult/status est ERROR)

```
<isochroneResultV2>
    <message>distance parameter must not be null or 0 !</message>
    <status>ERROR</status>
</isochroneResultV2>
```

Cas de method + time non renseigné (isochroneResult/status est ERROR)

```
<isochroneResultV2>
    <message>time parameter must not be null or 0 !</message>
    <status>ERROR</status>
</isochroneResultV2>
```

Cas ou la balise location est manquante (isochroneResult/status est ERROR)

```
<isochroneResultV2>
    <message>Location must be not null</message>
    <status>ERROR</status>
</isochroneResultV2>
```

Cas ou le point de départ est mal renseigné (isochroneResult/status est ERROR)

```
<isochroneResultV2>
   <message>Location point must have 2 components separated with a ,</message>
   <status>ERROR</status>
</isochroneResultV2>
```

Cas d'une erreur d'accrochage au graphe (serviceResult/status est ERROR)

```
<serviceResult>
        <message>
                ServiceException: Error in isochrone computation Error in smartrouting Failed to execute
 calculateConcentricReachableAreas com.geoconcept.smartrouting.SmartRoutingNativeException: failed to
 connect isochrone orign { 52.324230, 48.803256, 0.000000 } failed to connect isochrone orign { 52.324230,
 48.803256, 0.000000 }
        </message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in isochrone computation
        Error in smartrouting
        datasource is null
    </message>
   <status>ERROR</status>
</serviceResult>
```

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
id	Identifiant de l'isochrone	oui	
location	Départ (ou arrivée si le `reverse` est à true). Les coordonnées sont séparées par la caractère ,	non	
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	epsg:4326
graphName	Nom du graphe à utiliser	oui	

paramètre	description	optionnel	défaut
profileId	Identifiant du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
profileName	Profil du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère , ou ; (Exemple : "Toll", "Tunnel", "Bridge")	oui	
method	"time" pour isochrone ou "distance" for isodistance	non	time
time	Temps maximum d'accès, en secondes	oui	
distance	Distance maximum d'accès, en mètres	oui	
reverse	si *true*, le **`location`** est considéré comme arrivée	oui	false
smoothing	Lissage	oui	false
holes	Afficher les trous dans la zone résultante (la géométrie retournée est plus volumineuse lorsque ce paramètre est à *true*)	oui	false

En sortie

Itinéraire (isochroneResult)

paramètre	type	min/max	description
id	string	0/1	Identifiant de l'isochrone
location	string	0/1	Départ (ou arrivée si le **`reverse`** est à true).
srs	string	0/1	projection
time	string	0/1	Temps maximum d'accès, en secondes
distance	string	0/1	Distance maximum d'accès, en mètres
wktGeometry	string	0/1	Géometrie de l'isochrone, au format wkt

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/isochroneService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:isochrone>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <id/>
            <location>
               <x>-1.557189</x>
               <y>47.217122</y>
            </location>
            <!--Optional:-->
            <srs/>
            <!--Optional:-->
            <graphName/>
            <!--Optional:-->
```

```
            <profileId/>
            <!--Optional:-->
            <profileName/>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion/>
            </exclusions>
            <!--Optional:-->
            <method>distance</method>
            <!--Optional:-->
            <distance>5000</distance>
            <!--Optional:-->
            <reverse/>
            <!--Optional:-->
            <smoothing/>
            <!--Optional:-->
            <holes/>
         </request>
      </sch:isochrone>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:isochroneResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <IsochroneResult>
            <status>OK</status>
            <id/>
            <location>-1.557189,47.217122</location>
            <srs/>
            <distance>5000</distance>
            <wktGeometry>POLYGON ((-1.544603 47.179044, -1.545261 47.179044, [...]))</wktGeometry>
         </IsochroneResult>
      </ns2:isochroneResponse>
   </soap:Body>
</soap:Envelope>
```

REST (POST)

Requête

Requête

```
http://<server>/<webapp>/api/lbs/isochrone.xml
```

Data (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<isochroneRequest>
  <id>1</id>
  <location>
    <x>-1.557189</x>
    <y>47.217122</y>
  </location>
  <srs></srs>
  <graphName></graphName>
```

```
  <profileId></profileId>
  <profileName></profileName>
  <exclusions>
    <exclusion></exclusion>
    <exclusion></exclusion>
    <!--...more "exclusion" elements...-->
  </exclusions>
  <method></method>
  <time>50</time>
  <distance></distance>
  <reverse></reverse>
  <smoothing></smoothing>
  <holes></holes>
</isochroneRequest>
```

Réponse

La réponse est toujours encodée en UTF-8.

Format XML

```
<isochroneResult>
   <status>OK</status>
   <id>1</id>
   <location>-1.557189,47.217122</location>
   <srs/>
   <time>50</time>
   <wktGeometry>POLYGON ((-1.556864 47.216487, -1.556864 47.216948, ...))</wktGeometry>
</isochroneResult>
```

REST (GET)

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/isochrone.json?location=-1.557189,47.217122&method=distance&distance=5000
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/isochrone.json?
location=-1.557189,47.217122&method=distance&distance=5000&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/isochrone.xml??
location=-1.557189,47.217122&method=distance&distance=2000&holes=true
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message": null,
```

```
   "status": "OK",
   "id": null,
   "location": "-1.557189,47.217122",
   "srs": "EPSG:4326",
   "time": null,
   "distance": "5000",
   "wktGeometry": "POLYGON ((-1.545312 47.178178, -1.545312 47.180418, ...))"
}
```

Format JSON-P

```
myCallback(
    {
       "message": null,
       "status": "OK",
       "id": null,
       "location": "-1.557189,47.217122",
       "srs": "EPSG:4326",
       "time": null,
       "distance": "5000",
       "wktGeometry": "POLYGON ((-1.545312 47.178178, -1.545312 47.180418, ...))"
    }
);
```

Format XML

```
<isochroneResult>
   <status>OK</status>
   <location>-1.557189,47.217122</location>
   <srs>EPSG:4326</srs>
   <distance>2000</distance>
   <wktGeometry>POLYGON ((-1.546926 47.202309, -1.546926 47.202752, ...))</wktGeometry>
</isochroneResult>
```

Retours possibles

Cas d'un isochrone/isodistance trouvé (isochroneResult/status est OK)

```
<isochroneResult>
        <status>OK</status>
        <location>2.32423,48.803256</location>
        <srs>epsg:4326</srs>
        <distance>2000,000000</distance>
        <wktGeometry>
                POLYGON ((2.317525987661223 48.786657549807174, 2.317525987661223 48.787103131779986, ...))
        </wktGeometry>
</isochroneResult>
```

Cas ou le point de départ est vide (isochroneResult/status est ERROR)

```
<isochroneResult>
    <message>Location must be not null</message>
    <status>ERROR</status>
</isochroneResult>
```

Cas ou le point de départ est mal renseigné (isochroneResult/status est ERROR)

```
<isochroneResult>
```

```
    <message>Location point must have 2 components separated with a ,</message>
    <status>ERROR</status>
</isochroneResult>
```

Cas d'un mauvais typage

```
<isochroneResult>
        <message>NumberFormatException: For input string: "AAA"</message>
        <status>ERROR</status>
</isochroneResult>
```

Cas d'une erreur d'accrochage au graphe (isochroneResult/status est ERROR)

```
<serviceResult>
        <message>
                ServiceException: Error in isochrone computation Error in smartrouting Failed to execute
 calculateConcentricReachableAreas com.geoconcept.smartrouting.SmartRoutingNativeException: failed to
 connect isochrone orign { 52.324230, 48.803256, 0.000000 } failed to connect isochrone orign { 52.324230,
 48.803256, 0.000000 }
        </message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (isochroneResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in isochrone computation
        Error in smartrouting
        datasource is null
    </message>
   <status>ERROR</status>
</serviceResult>
```

FAQ

1. Est-il possible de prioriser le temps de parcours ou la distance ?

   Oui, en changeant la méthode **`method`** "time" pour isochrone ou "distance" for isodistance

2. Peut-on utiliser des alias à défaut des noms de fichiers .siti pour appeler une datasource ?

   Oui, cf. détails dans la FAQ du Web Service du géocodage inverse.

3. Comment utiliser les statistiques routières ?

   Vérifier que le graphe contient bien ces statistiques et utiliser les deux paramètres suivants : **`computeOptions`** avec la valeur *trafficPatterns* et **`startDateTime`** pour préciser la date/heure de départ.

4. Comment faire un calcul d'isochrone/isodistance sans péage?

   Si la contrainte *Toll* a bien été incluse dans le graphe, placer une exclusion dans **`exclusions`** :

   ```
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
   geoconcept.com/gc/schemas">
      <soapenv:Header/>
      <soapenv:Body>
         <sch:isochroneV2>
            <!--Optional:-->
   ```

```
         <request>
            <!--Optional:-->
            <id>1</id>
            <location>
               <x>-1.557189</x>
               <y>47.217122</y>
            </location>
            <!--Optional:-->
            <srs></srs>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusions>Toll</exclusions>
            </exclusions>
            <!--Optional:-->
            <method>distance</method>
            <!--Optional:-->
            <time></time>
            <!--Optional:-->
            <distance>5000</distance>
            <!--Optional:-->
            <reverse></reverse>
            <!--Optional:-->
            <smoothing></smoothing>
            <!--Optional:-->
            <holes></holes>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <snapMethod></snapMethod>
         </request>
      </sch:isochroneV2>
   </soapenv:Body>
</soapenv:Envelope>
```

5. Qu'est-ce que les Speed Patterns ? Comment les utiliser ?

   Afin de proposer des temps de trajets au plus près des conditions de circulation, les graphes fournis par GEOCONCEPT SAS intègrent, à partir de la version M18, pour les voitures et pour les camions 5 profils de vitesses (Speed Patterns) pour tenir compte des différents niveaux de congestion dans une journée :

   - standard *normal-speed* correspond à la vitesse d'une heure moyennement chargée (11h)
   - nuit *fast-speed* correspond à un trafic très fluide, observé le plus souvent la nuit (3h)
   - chargée *slow-speed* correspond aux heures de trafic denses (8h)
   - heure de pointe *very-slow-speed* correspond aux heures de trafic denses dans les grandes agglomérations, plus lent que le précédent (8h)
   - défaut *defaut* correspond aux vitesses moyennées sur une journée entière

   Pour les utiliser il faut passer, lors de l'appel au web service, le paramètre **`computeOptions`** avec l'option *speedPattern* et préciser la valeur de la Speed Pattern demandée sans omettre un profil de véhicule

Exemple :

```
&computeOptions=speedPattern:fast-speed&profileId=1
```

6. Comment utiliser les attributs poids lourds ?

   Il faut que le graphe intègre les attributs de poids lourds (en standard dans les graphes fournis par GEOCONCEPT SAS à partir de la version M18) et soit calculer un itinéraire en utilisant un profil de véhicule utilisant les restrictions (cf. le catalogue de véhicules, éditable, définit dans le fichier SmartRoutingVehicles.xml, dans le dossier `'*<GEOCONCEPT_WEB_HOME>*'\*smartrouting*\*jee*\*smartrouting*\*conf*\ ), soit surcharger l'appel au web service en utilisant le paramètre **`computeOptions`** avec les options *length*, *width*, *height*, *weight*, *axles* et/ou *weightPerAxle*.
   Exemple pour un trajet avec un véhicule de 4,5 mètres de hauteur :

   ```
   &computeOptions=height:450
   ```

Calcul de matrice

Cette page n'est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

Ce web service calcule une matrice d'itinéraires pour un ensemble de points et retourne un distancier. Il s'appuie sur le graphe paramétré dont le nom a été spécifié dans l'interface d'administration de Geoconcept Web. Voir aussi le web service de calcul de matrice compacte.

Disponibilité

Ce web service est disponible en permanence avec Geoconcept Web et un graphe.

Changement de version

Les versions précédente du web service sont conservées dans Geoconcept Web pour assurer la compatibilité avec les développements antérieurs. Il est recommandé d'utiliser la version la plus récente.

Changements avec la V4

- Ajout des paramètres "timeOut" et "computeOptions".
- Ajout dans la snapMethod "nodes" de l'accrochage aux noeuds les plus proches.

Changements avec la V3

- Ajout de la notion de noeud, plus rapide, pour s'accrocher aux noeuds du graphe plutôt qu'a des coordonnées géographiques. Ajout des éléments suivants : "originNodes", "destinationNodes" et de "nodes" dans les snapMethod.
- Ajout du paramètre "maxCost".

Changements avec la V2

- Ajout du paramètre "snapMethod"
- Ajout du paramètre "startDateTime"
- Suppression du paramètre "RejectFlags", remplacé par "exclusions"
- Les paramètres "distance" et "duration" sont renommés respectivement "distanceMeters" et "durationSeconds"

V4

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
origins	Liste des coordonnées des points d'origines. Les coordonnées longitude et latitude sont séparées par la caractère ,	oui *	
originNodes	Liste des noeuds ids d'origines. Les Noeuds ids sont séparées par la caractère ,. Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
destinations	Liste des coordonnées des points de destinations. Les coordonnées longitude et latitude sont séparées par la caractère ,	oui *	
destinationNodes	Liste des noeuds ids de destinations. Les Noeuds ids sont séparées par la caractère ,. Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
graphName (déprécié)	Nom du graphe à utiliser Ce paramètre est omis si le paramètre configName est utilisé.	oui	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
profileId (déprécié)	Identifiant du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileName (déprécié)	Profil du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère *;* (Exemple : Toll, Tunnel, Bridge)	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris. Attention le caractère + peut être mal interprété par les navigateurs, dans ce cas, il faut le remplacer par *%2B*.	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction - nodes : Accrochage direct aux noeuds fournis par les paramètres originNode, destinationNode et waypointNodes ou, si ces premiers ne sont pas renseignés, aux noeuds les plus proches des paramètres origin, destination et waypoint	oui	standard

paramètre	description	optionnel	défaut
avoidArea	Zone de transit interdit au format WKT (POLYGON ou MULTIPOLYGON) dans la projection (paramètre srs) demandée Exemple en wgs84 : `POLYGON ((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689))` - `MULTIPOLYGON (((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689)), ((-1.558639 47.218144, -1.558639 47.218193, [...] -1.558639 47.218144)))` Attention les géométries WKT doivent être fermées.	oui	
configName	Nom de la configuration à utiliser (défini dans Geoconcept Web - Administration / Outils / Définitions des graphes) Il remplace l'usage de graphName, profileId et profileName	oui	
computeOptions	Liste des options pour le calcul, séparées par le caractère *;* - trafficPatterns : utilise les statistiques routières (il est nécessaire de renseigner le paramètre startDateTime et d'utiliser un graphe intégrant les informations de traffic patterns) - speedPattern (M18) : utilise d'une *speed pattern* tel que définis dans le fichier SmartRoutingVehicles.xml qui se trouve dans le dossier *`"<GEOCONCEPT_WEB_HOME>"\smartrouting\jee\smartrouting\conf* . Usage : "speedPattern:slow-speed" - length (M18) : longueur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "length:950" - width (M18) : largeur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "width:255" - height (M18) : hauteur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "height:360" - weight (M18) : poids maximum autorisé en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weight:18000" - axles (M18) : nombre maximum d'axe autorisé (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "axles:2" - weightPerAxle (M18) : poids maximum autorisé par axe en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weightPerAxle:9000" - snapSpeed : vitesse d'accrochage au graphe en kilomètre par heure. Usage : "snapSpeed:10"	oui	
maxCost	Coût maximum à ne pas dépasser dans les résultats -1 : pas de coût maximum à prendre en compte 0 : prendre la valeur par default défini dans la configuration de SmartRouting Server sinon : valeur en mètres si method=distance ou en secondes si method=time	oui	
timeOut	Time out pour le calcul (en millisecondes)	oui	

(*) Au moins l'un des deux couples de paramètres origins/destinations ou originNodes/destinationsNodes doit être renseigné.

(M18) Disponible à partir de la version M18 des graphes fournis par GEOCONCEPT SAS.

En sortie

Matrice (matrixResultV3)

paramètre	type	min/max	description
origins/origin (ou origins en JSON / JSON-P)	string (ou array of origins/origin en JSON / JSON-P)	0/illimité	positions d'origine.
destinations/destination (ou destinations en JSON / JSON-P)	string (ou array of destinations/destination en JSON / JSON-P)	0/illimité	positions de destination.
rows/row (ou rows en JSON / JSON-P)	matrixRowV3 (ou array of rows/row (matrixRow) en JSON / JSON-P)	0/illimité	Ligne de matrice

Ligne de matrice (matrixRowV3)

paramètre	type	min/max	description
cells/cell (ou cells en JSON / JSON-P)	matrixCellV3 (ou array of cells/cell (matrixCellV3) en JSON / JSON-P)	0/illimité	Cellule de matrice

Cellule de matrice (matrixCellV3)

paramètre	type	min/max	description
distanceMeters	double	1/1	Distance de cellule de matrice (en mètres)
durationSeconds	double	1/1	Durée de cellule de matrice (en secondes)
status	string	0/1	Status cellule de matrice : - OK : l'itinéraire a bien été trouvé pour cette cellule - KO : pas d'itinéraire trouvé pour cette cellule

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/matrixService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:matrixV4>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <srs>epsg:27572</srs>
            <!--Optional:-->
            <origins>
               <!--1 or more repetitions:-->
               <origin>
                  <x>315846.96</x>
                  <y>2254268.35</y>
               </origin>
                <origin>
                            <x>313584.77</x>
```

```
                  <y>2251648.97</y>
               </origin>
            </origins>
            <!--Optional:-->
            <originNodes>
               <!--1 or more repetitions:-->
               <originNode></originNode>
            </originNodes>
            <!--Optional:-->
            <destinations>
               <!--1 or more repetitions:-->
               <destination>
                  <x>321442.89</x>
                  <y>2251013.98</y>
               </destination>
                <destination>
                  <x>318982.27</x>
                  <y>2248315.22</y>
               </destination>
            </destinations>
            <!--Optional:-->
            <destinationNodes>
               <!--1 or more repetitions:-->
               <destinationNode></destinationNode>
            </destinationNodes>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <method>time</method>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <snapMethod></snapMethod>
            <!--Optional:-->
            <avoidArea></avoidArea>
            <!--Optional:-->
            <configName></configName>
            <!--Optional:-->
            <computeOptions></computeOptions>
            <!--Optional:-->
            <timeOut></timeOut>
                        <!--Optional:-->
                        <maxCost>0</maxCost>
         </request>
      </sch:matrixV4>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:matrixV4Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <MatrixResult>
```

```
            <status>OK</status>
            <origins>
               <origin>315846.96,2254268.35</origin>
               <origin>313584.77,2251648.97</origin>
            </origins>
            <destinations>
               <destination>321442.89,2251013.98</destination>
               <destination>318982.27,2248315.22</destination>
            </destinations>
            <srs>epsg:27572</srs>
            <rows>
               <row>
                  <cells>
                     <cell>
                        <distanceMeters>8109.0</distanceMeters>
                        <durationSeconds>564.66</durationSeconds>
                        <status>OK</status>
                     </cell>
                     <cell>
                        <distanceMeters>11249.39</distanceMeters>
                        <durationSeconds>816.38</durationSeconds>
                        <status>OK</status>
                     </cell>
                  </cells>
               </row>
               <row>
                  <cells>
                     <cell>
                        <distanceMeters>11736.89</distanceMeters>
                        <durationSeconds>856.77</durationSeconds>
                        <status>OK</status>
                     </cell>
                     <cell>
                        <distanceMeters>7636.69</distanceMeters>
                        <durationSeconds>526.1</durationSeconds>
                        <status>OK</status>
                     </cell>
                  </cells>
               </row>
            </rows>
         </MatrixResult>
      </ns2:matrixV4Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/matrix/v4.json?
origins=315846.96,2254268.35;313584.77,2251648.97&destinations=321442.89,2251013.98;318982.27,2248315.22&srs=epsg:27572&metho
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/matrix/v4.json?
origins=315846.96,2254268.35;313584.77,2251648.97&destinations=321442.89,2251013.98;318982.27,2248315.22&srs=epsg:27572&metho
```

Requête XML

```
http://<server>/<webapp>/api/lbs/matrix/v4.xml?
origins=315846.96,2254268.35;313584.77,2251648.97&destinations=321442.89,2251013.98;318982.27,2248315.22&srs=epsg:27572&me
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON


```
{
   "message":null,
   "status":"OK",
   "origins":[
      "315846.96,2254268.35",
      "313584.77,2251648.97"
   ],
   "destinations":[
      "321442.89,2251013.98",
      "318982.27,2248315.22"
   ],
   "srs":"epsg:27572",
   "rows":[
      {
         "cells":[
            {
               "distanceMeters":8109.0,
               "durationSeconds":656.74,
               "status":"OK"
            },
            {
               "distanceMeters":11249.37,
               "durationSeconds":888.57,
               "status":"OK"
            }
         ]
      },
      {
         "cells":[
            {
               "distanceMeters":11736.89,
               "durationSeconds":959.9,
               "status":"OK"
            },
            {
               "distanceMeters":7636.67,
               "durationSeconds":609.35,
               "status":"OK"
            }
         ]
      }
   ]
}
```


Format JSON-P


```
MyCallBack({
   "message":null,
   "status":"OK",
   "origins":[
      "315846.96,2254268.35",
```

```
      "313584.77,2251648.97"
   ],
   "destinations":[
      "321442.89,2251013.98",
      "318982.27,2248315.22"
   ],
   "srs":"epsg:27572",
   "rows":[
      {
         "cells":[
            {
               "distanceMeters":8109.0,
               "durationSeconds":656.74,
               "status":"OK"
            },
            {
               "distanceMeters":11249.37,
               "durationSeconds":888.57,
               "status":"OK"
            }
         ]
      },
      {
         "cells":[
            {
               "distanceMeters":11736.89,
               "durationSeconds":959.9,
               "status":"OK"
            },
            {
               "distanceMeters":7636.67,
               "durationSeconds":609.35,
               "status":"OK"
            }
         ]
      }
   ]
});
```

Format XML

```
<?xml version="1.0" encoding="UTF-8"?>
<matrixResultV4>
   <status>OK</status>
   <origins>
      <origin>315846.96,2254268.35</origin>
      <origin>313584.77,2251648.97</origin>
   </origins>
   <destinations>
      <destination>321442.89,2251013.98</destination>
      <destination>318982.27,2248315.22</destination>
   </destinations>
   <srs>epsg:27572</srs>
   <rows>
      <row>
         <cells>
            <cell>
               <distanceMeters>8109.0</distanceMeters>
               <durationSeconds>656.74</durationSeconds>
               <status>OK</status>
            </cell>
            <cell>
               <distanceMeters>11249.37</distanceMeters>
```

```
                <durationSeconds>888.57</durationSeconds>
                <status>OK</status>
             </cell>
          </cells>
       </row>
       <row>
          <cells>
             <cell>
                <distanceMeters>11736.89</distanceMeters>
                <durationSeconds>959.9</durationSeconds>
                <status>OK</status>
             </cell>
             <cell>
                <distanceMeters>7636.67</distanceMeters>
                <durationSeconds>609.35</durationSeconds>
                <status>OK</status>
             </cell>
          </cells>
       </row>
    </rows>
</matrixResultV4>
```

Retours possibles

Cas d'un itinéraire trouvé (matrixResultV3/status est OK)

```
<?xml version="1.0" encoding="UTF-8"?>
<matrixResultV4>
   <status>OK</status>
   <origins>
      <origin>315846.96,2254268.35</origin>
      <origin>313584.77,2251648.97</origin>
   </origins>
   <destinations>
      <destination>321442.89,2251013.98</destination>
      <destination>318982.27,2248315.22</destination>
   </destinations>
   <srs>epsg:27572</srs>
   <rows>
      <row>
         <cells>
            <cell>
               <distanceMeters>8109.0</distanceMeters>
               <durationSeconds>656.74</durationSeconds>
               <status>OK</status>
            </cell>
            <cell>
               <distanceMeters>11249.37</distanceMeters>
               <durationSeconds>888.57</durationSeconds>
               <status>OK</status>
            </cell>
         </cells>
      </row>
      <row>
         <cells>
            <cell>
               <distanceMeters>11736.89</distanceMeters>
               <durationSeconds>959.9</durationSeconds>
               <status>OK</status>
            </cell>
            <cell>
               <distanceMeters>7636.67</distanceMeters>
```

```
               <durationSeconds>609.35</durationSeconds>
               <status>OK</status>
            </cell>
         </cells>
      </row>
   </rows>
</matrixResultV4>
```

Cas d'un oubli de spécification d'origine ou de destination (matrixResultV3/status est ERROR)

```
<matrixResultV4>
   <message>Origins and destinations must be not null</message>
   <status>ERROR</status>
</matrixResultV4>
```

Cas d'un mauvais typage (serviceResult/status est ERROR)

```
<serviceResult>
        <message>NumberFormatException: For input string: "AAA"</message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'une erreur d'accrochage au graphe (matrixResultV3/status est OK) / (cell/status est KO)

```
<matrixResultV4>
   <status>OK</status>
   <origins>
      <origin>-0.36147,49.18392</origin>
      <origin>7.26132,43.70629</origin>
   </origins>
   <destinations>
      <destination>146.08821,48.43253</destination>
      <destination>2.34878,48.86473</destination>
   </destinations>
   <rows>
      <row>
         <cells>
            <cell>
               <distanceMeters>0.0</distanceMeters>
               <durationSeconds>0.0</durationSeconds>
               <status>KO</status>
            </cell>
            <cell>
               <distanceMeters>234196.77000000002</distanceMeters>
               <durationSeconds>9451.57</durationSeconds>
               <status>OK</status>
            </cell>
         </cells>
      </row>
      <row>
         <cells>
            <cell>
               <distanceMeters>0.0</distanceMeters>
               <durationSeconds>0.0</durationSeconds>
               <status>KO</status>
            </cell>
            <cell>
               <distanceMeters>930934.64</distanceMeters>
               <durationSeconds>31773.440000000002</durationSeconds>
               <status>OK</status>
```

```
            </cell>
          </cells>
        </row>
    </rows>
</matrixResultV4>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in matrix computation
Error in smartrouting
datasource is null</message>
   <status>ERROR</status>
</serviceResult>
```

V3

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
origins	Liste des coordonnées des points d'origines. Les coordonnées longitude et latitude sont séparées par la caractère ,	oui *	
originNodes	Liste des noeuds ids d'origines. Les Noeuds ids sont séparées par la caractère ,. Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
destinations	Liste des coordonnées des points de destinations. Les coordonnées longitude et latitude sont séparées par la caractère ,	oui *	
destinationNodes	Liste des noeuds ids de destinations. Les Noeuds ids sont séparées par la caractère ,. Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
graphName	Nom du graphe à utiliser Ce paramètre est omis si le paramètre configName est utilisé.	oui	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
profileId	Identifiant du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileName	Profil du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère , ou ; (Exemple : "Toll", "Tunnel", "Bridge")	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris. Attention le caractère + peut être mal interprété par les navigateurs, dans ce cas, il faut le remplacer par *%2B*.	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche	oui	standard

paramètre	description	optionnel	défaut
	- extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction - nodes : accrochage direct aux noeuds indiqués par locationNode et les paramètres *node* des ressources		
avoidArea	Zone de transit interdit au format WKT (POLYGON ou MULTIPOLYGON) dans la projection (paramètre srs) demandée Exemple en wgs84 : `POLYGON ((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689))` - `MULTIPOLYGON (((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689)), ((-1.558639 47.218144, -1.558639 47.218193, [...] -1.558639 47.218144)))` Attention les géométries WKT doivent être fermées.	oui	
configName	Nom de la configuration à utiliser (défini dans Geoconcept Web - Administration / Outils / Définitions des graphes) Il remplace l'usage de graphName, profileId et profileName	oui	
maxCost	Coût maximum à ne pas dépasser dans les résultats -1 : pas de coût maximum à prendre en compte 0 : prendre la valeur par default défini dans la configuration de SmartRouting Server sinon : valeur en mètres si method=distance ou en secondes si method=time	oui	

(*) Au moins l'un des deux couples de paramètres origins/destinations ou originNodes/destinationsNodes doit être renseigné.

En sortie

Matrice (matrixResultV3)

paramètre	type	min/max	description
origins/origin (ou origins en JSON / JSON-P)	string (ou array of origins/origin en JSON / JSON-P)	0/illimité	positions d'origine.
destinations/destination (ou destinations en JSON / JSON-P)	string (ou array of destinations/destination en JSON / JSON-P)	0/illimité	positions de destination.
rows/row (ou rows en JSON / JSON-P)	matrixRowV3 (ou array of rows/row (matrixRow) en JSON / JSON-P)	0/illimité	Ligne de matrice

Ligne de matrice (matrixRowV3)

paramètre	type	min/max	description
cells/cell (ou cells en JSON / JSON-P)	matrixCellV3 (ou array of cells/cell (matrixCellV3) en JSON / JSON-P)	0/illimité	Cellule de matrice

Cellule de matrice (matrixCellV3)

paramètre	type	min/max	description
distanceMeters	double	1/1	Distance de cellule de matrice (en mètres)
durationSeconds	double	1/1	Durée de cellule de matrice (en secondes)
status	string	0/1	Status cellule de matrice : - OK : l'itinéraire a bien été trouvé pour cette cellule - KO : pas d'itinéraire trouvé pour cette cellule

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/matrixService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:matrixV3>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <srs>epsg:27572</srs>
            <!--Optional:-->
            <origins>
               <!--1 or more repetitions:-->
               <origin>
                  <x>315846.96</x>
                  <y>2254268.35</y>
               </origin>
                <origin>
                              <x>313584.77</x>
                  <y>2251648.97</y>
               </origin>
            </origins>
            <!--Optional:-->
            <originNodes>
               <!--1 or more repetitions:-->
               <originNode></originNode>
            </originNodes>
            <!--Optional:-->
            <destinations>
               <!--1 or more repetitions:-->
               <destination>
                  <x>321442.89</x>
                  <y>2251013.98</y>
               </destination>
                <destination>
                  <x>318982.27</x>
                  <y>2248315.22</y>
               </destination>
            </destinations>
            <!--Optional:-->
            <destinationNodes>
               <!--1 or more repetitions:-->
               <destinationNode></destinationNode>
            </destinationNodes>
            <!--Optional:-->
```

```
            <graphName></graphName>
            <!--Optional:-->
            <method>time</method>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <snapMethod></snapMethod>
            <!--Optional:-->
            <avoidArea></avoidArea>
            <!--Optional:-->
            <configName></configName>
            <!--Optional:-->
            <maxCost>0</maxCost>
         </request>
      </sch:matrixV3>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:matrixV3Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <MatrixResult>
            <status>OK</status>
            <origins>
               <origin>315846.96,2254268.35</origin>
               <origin>313584.77,2251648.97</origin>
            </origins>
            <destinations>
               <destination>321442.89,2251013.98</destination>
               <destination>318982.27,2248315.22</destination>
            </destinations>
            <srs>epsg:27572</srs>
            <rows>
               <row>
                  <cells>
                     <cell>
                        <distanceMeters>8109.0</distanceMeters>
                        <durationSeconds>564.66</durationSeconds>
                        <status>OK</status>
                     </cell>
                     <cell>
                        <distanceMeters>11249.37</distanceMeters>
                        <durationSeconds>816.38</durationSeconds>
                        <status>OK</status>
                     </cell>
                  </cells>
               </row>
               <row>
                  <cells>
                     <cell>
                        <distanceMeters>11736.89</distanceMeters>
                        <durationSeconds>856.77</durationSeconds>
```

```
                        <status>OK</status>
                     </cell>
                     <cell>
                        <distanceMeters>7636.67</distanceMeters>
                        <durationSeconds>526.1</durationSeconds>
                        <status>OK</status>
                     </cell>
                  </cells>
               </row>
            </rows>
         </MatrixResult>
      </ns2:matrixV3Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/matrix/v3.json?
origins=315846.96,2254268.35;313584.77,2251648.97&destinations=321442.89,2251013.98;318982.27,2248315.22&srs=epsg:27572&me
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/matrix/v3.json?
origins=315846.96,2254268.35;313584.77,2251648.97&destinations=321442.89,2251013.98;318982.27,2248315.22&srs=epsg:27572&me
```

Requête XML

```
http://<server>/<webapp>/api/lbs/matrix/v3.xml?
origins=315846.96,2254268.35;313584.77,2251648.97&destinations=321442.89,2251013.98;318982.27,2248315.22&srs=epsg:27572&me
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message":null,
   "status":"OK",
   "origins":[
      "315846.96,2254268.35",
      "313584.77,2251648.97"
   ],
   "destinations":[
      "321442.89,2251013.98",
      "318982.27,2248315.22"
   ],
   "srs":"epsg:27572",
   "rows":[
      {
         "cells":[
            {
               "distanceMeters":8109.0,
               "durationSeconds":656.74,
```

```
                "status":"OK"
             },
             {
                "distanceMeters":11249.37,
                "durationSeconds":888.57,
                "status":"OK"
             }
          ]
       },
       {
          "cells":[
             {
                "distanceMeters":11736.89,
                "durationSeconds":959.9,
                "status":"OK"
             },
             {
                "distanceMeters":7636.67,
                "durationSeconds":609.35,
                "status":"OK"
             }
          ]
       }
    ]
}
```


Format JSON-P


```
MyCallBack({
   "message":null,
   "status":"OK",
   "origins":[
      "315846.96,2254268.35",
      "313584.77,2251648.97"
   ],
   "destinations":[
      "321442.89,2251013.98",
      "318982.27,2248315.22"
   ],
   "srs":"epsg:27572",
   "rows":[
      {
         "cells":[
            {
               "distanceMeters":8109.0,
               "durationSeconds":656.74,
               "status":"OK"
            },
            {
               "distanceMeters":11249.37,
               "durationSeconds":888.57,
               "status":"OK"
            }
         ]
      },
      {
         "cells":[
            {
               "distanceMeters":11736.89,
               "durationSeconds":959.9,
               "status":"OK"
            },
            {
```

```
               "distanceMeters":7636.67,
               "durationSeconds":609.35,
               "status":"OK"
            }
         ]
      }
   ]
});
```

Format XML

```
<?xml version="1.0" encoding="UTF-8"?>
<matrixResultV3>
   <status>OK</status>
   <origins>
      <origin>315846.96,2254268.35</origin>
      <origin>313584.77,2251648.97</origin>
   </origins>
   <destinations>
      <destination>321442.89,2251013.98</destination>
      <destination>318982.27,2248315.22</destination>
   </destinations>
   <srs>epsg:27572</srs>
   <rows>
      <row>
         <cells>
            <cell>
               <distanceMeters>8109.0</distanceMeters>
               <durationSeconds>656.74</durationSeconds>
               <status>OK</status>
            </cell>
            <cell>
               <distanceMeters>11249.37</distanceMeters>
               <durationSeconds>888.57</durationSeconds>
               <status>OK</status>
            </cell>
         </cells>
      </row>
      <row>
         <cells>
            <cell>
               <distanceMeters>11736.89</distanceMeters>
               <durationSeconds>959.9</durationSeconds>
               <status>OK</status>
            </cell>
            <cell>
               <distanceMeters>7636.67</distanceMeters>
               <durationSeconds>609.35</durationSeconds>
               <status>OK</status>
            </cell>
         </cells>
      </row>
   </rows>
</matrixResultV3>
```

Retours possibles

Cas d’un itinéraire trouvé (matrixResultV3/status est OK)

```
<?xml version="1.0" encoding="UTF-8"?>
<matrixResultV3>
```

```
   <status>OK</status>
   <origins>
      <origin>315846.96,2254268.35</origin>
      <origin>313584.77,2251648.97</origin>
   </origins>
   <destinations>
      <destination>321442.89,2251013.98</destination>
      <destination>318982.27,2248315.22</destination>
   </destinations>
   <srs>epsg:27572</srs>
   <rows>
      <row>
         <cells>
            <cell>
               <distanceMeters>8109.0</distanceMeters>
               <durationSeconds>656.74</durationSeconds>
               <status>OK</status>
            </cell>
            <cell>
               <distanceMeters>11249.37</distanceMeters>
               <durationSeconds>888.57</durationSeconds>
               <status>OK</status>
            </cell>
         </cells>
      </row>
      <row>
         <cells>
            <cell>
               <distanceMeters>11736.89</distanceMeters>
               <durationSeconds>959.9</durationSeconds>
               <status>OK</status>
            </cell>
            <cell>
               <distanceMeters>7636.67</distanceMeters>
               <durationSeconds>609.35</durationSeconds>
               <status>OK</status>
            </cell>
         </cells>
      </row>
   </rows>
</matrixResultV3>
```

Cas d'un oubli de spécification d'origine ou de destination (matrixResultV3/status est ERROR)

```
<matrixResultV3>
   <message>Origins and destinations must be not null</message>
   <status>ERROR</status>
</matrixResultV3>
```

Cas d'un mauvais typage (serviceResult/status est ERROR)

```
<serviceResult>
        <message>NumberFormatException: For input string: "AAA"</message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'une erreur d'accrochage au graphe (matrixResultV3/status est OK) / (cell/status est KO)

```
<matrixResultV3>
   <status>OK</status>
   <origins>
```

```
      <origin>-0.36147,49.18392</origin>
      <origin>7.26132,43.70629</origin>
   </origins>
   <destinations>
      <destination>146.08821,48.43253</destination>
      <destination>2.34878,48.86473</destination>
   </destinations>
   <rows>
      <row>
         <cells>
            <cell>
               <distanceMeters>0.0</distanceMeters>
               <durationSeconds>0.0</durationSeconds>
               <status>KO</status>
            </cell>
            <cell>
               <distanceMeters>234196.77000000002</distanceMeters>
               <durationSeconds>9451.57</durationSeconds>
               <status>OK</status>
            </cell>
         </cells>
      </row>
      <row>
         <cells>
            <cell>
               <distanceMeters>0.0</distanceMeters>
               <durationSeconds>0.0</durationSeconds>
               <status>KO</status>
            </cell>
            <cell>
               <distanceMeters>930934.64</distanceMeters>
               <durationSeconds>31773.440000000002</durationSeconds>
               <status>OK</status>
            </cell>
         </cells>
      </row>
   </rows>
</matrixResultV3>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in matrix computation
Error in smartrouting
datasource is null</message>
   <status>ERROR</status>
</serviceResult>
```

V2

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
origins	Liste des coordonnées des points d'origines. Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	

paramètre	description	optionnel	défaut
destinations	Liste des coordonnées des points de destinations. Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	
graphName	Nom du graphe à utiliser	oui	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
profileId	Identifiant du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
profileName	Profil du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère , ou *;* (Exemple : "Toll", "Tunnel", "Bridge")	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction	oui	standard

En sortie

Matrice (matrixResultV2)

paramètre	type	min/max	description
origins/origin (ou origins en JSON / JSON-P)	string (ou array of origins/origin en JSON / JSON-P)	0/illimité	positions d'origine.
destinations/destination (ou destinations en JSON / JSON-P)	string (ou array of destinations/destination en JSON / JSON-P)	0/illimité	positions de destination.
rows/row (ou rows en JSON / JSON-P)	matrixRowV2 (ou array of rows/row (matrixRow) en JSON / JSON-P)	0/illimité	Ligne de matrice

Ligne de matrice (matrixRowV2)

paramètre	type	min/max	description
cells/cell (ou cells en JSON / JSON-P)	matrixCellV2 (ou array of cells/cell (matrixCellV2) en JSON / JSON-P)	0/illimité	Cellule de matrice

Cellule de matrice (matrixCellV2)

paramètre	type	min/max	description
distanceMeters	double	1/1	Distance de cellule de matrice (en mètres)
durationSeconds	double	1/1	Durée de cellule de matrice (en secondes)
status	string	0/1	Status cellule de matrice : - OK : l'itinéraire a bien été trouvé pour cette cellule

paramètre	type	min/max	description
			- KO : pas d'itinéraire trouvé pour cette cellule

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/matrixService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:matrixV2>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <srs>epsg:27572</srs>
            <!--Optional:-->
            <origins>
               <!--1 or more repetitions:-->
               <origin>
                  <x>315846.96</x>
                  <y>2254268.3500000001</y>
               </origin>
                <origin>
                           <x>313584.77</x>
                  <y>2251648.9700000002</y>
               </origin>
            </origins>
            <!--Optional:-->
            <destinations>
               <!--1 or more repetitions:-->
               <destination>
                  <x>321442.89</x>
                  <y>2251013.98</y>
               </destination>
                <destination>
                  <x>318982.27</x>
                  <y>2248315.2200000002</y>
               </destination>
            </destinations>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <method>time</method>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <snapMethod></snapMethod>
```

```
         </request>
      </sch:matrixV2>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:matrixV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <MatrixResult>
            <status>OK</status>
            <origins>
               <origin>315846.96,2254268.35</origin>
               <origin>313584.77,2251648.97</origin>
            </origins>
            <destinations>
               <destination>321442.89,2251013.98</destination>
               <destination>318982.27,2248315.22</destination>
            </destinations>
            <srs>epsg:27572</srs>
            <rows>
               <row>
                  <cells>
                     <cell>
                        <distanceMeters>8109.04</distanceMeters>
                        <durationSeconds>894.48</durationSeconds>
                        <status>OK</status>
                     </cell>
                     <cell>
                        <distanceMeters>11127.53</distanceMeters>
                        <durationSeconds>873.21</durationSeconds>
                        <status>OK</status>
                     </cell>
                  </cells>
               </row>
               <row>
                  <cells>
                     <cell>
                        <distanceMeters>11736.87</distanceMeters>
                        <durationSeconds>1158.04</durationSeconds>
                        <status>OK</status>
                     </cell>
                     <cell>
                        <distanceMeters>7514.82</distanceMeters>
                        <durationSeconds>615.79</durationSeconds>
                        <status>OK</status>
                     </cell>
                  </cells>
               </row>
            </rows>
         </MatrixResult>
      </ns2:matrixV2Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/matrix/v2.json?
origins=315846.96,2254268.3500000001;313584.77,2251648.9700000002&destinations=321442.89,2251013.98;318982.27,2248315.2200
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/matrix/v2.json?
origins=315846.96,2254268.3500000001;313584.77,2251648.9700000002&destinations=321442.89,2251013.98;318982.27,2248315.2200
```

Requête XML

```
http://<server>/<webapp>/api/lbs/matrix/v2.xml?
origins=315846.96,2254268.3500000001;313584.77,2251648.9700000002&destinations=321442.89,2251013.98;318982.27,2248315.2200
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON


```
{
   "message":null,
   "status":"OK",
   "origins":[
      "315846.96,2254268.35",
      "313584.77,2251648.97"
   ],
   "destinations":[
      "321442.89,2251013.98",
      "318982.27,2248315.22"
   ],
   "srs":"epsg:27572",
   "rows":[
      {
         "cells":[
            {
               "distanceMeters":8109.04,
               "durationSeconds":894.48,
               "status":"OK"
            },
            {
               "distanceMeters":11127.53,
               "durationSeconds":873.21,
               "status":"OK"
            }
         ]
      },
      {
         "cells":[
            {
               "distanceMeters":11736.87,
               "durationSeconds":1158.04,
               "status":"OK"
            },
            {
               "distanceMeters":7514.82,
               "durationSeconds":615.79,
               "status":"OK"
            }
         ]
```

```
      }
   ]
}
```

Format JSON-P

```
MyCallBack({
   "message":null,
   "status":"OK",
   "origins":[
      "315846.96,2254268.35",
      "313584.77,2251648.97"
   ],
   "destinations":[
      "321442.89,2251013.98",
      "318982.27,2248315.22"
   ],
   "srs":"epsg:27572",
   "rows":[
      {
         "cells":[
            {
               "distanceMeters":8109.04,
               "durationSeconds":894.48,
               "status":"OK"
            },
            {
               "distanceMeters":11127.53,
               "durationSeconds":873.21,
               "status":"OK"
            }
         ]
      },
      {
         "cells":[
            {
               "distanceMeters":11736.87,
               "durationSeconds":1158.04,
               "status":"OK"
            },
            {
               "distanceMeters":7514.82,
               "durationSeconds":615.79,
               "status":"OK"
            }
         ]
      }
   ]
});
```

Format XML

```
<matrixResultV2>
   <status>OK</status>
   <origins>
      <origin>-0.36147,49.18392</origin>
      <origin>7.26132,43.70629</origin>
   </origins>
   <destinations>
      <destination>0.08821,48.43253</destination>
      <destination>2.34878,48.86473</destination>
   </destinations>
```

```
    <rows>
       <row>
          <cells>
             <cell>
                <distanceMeters>107526.06</distanceMeters>
                <durationSeconds>4906.25</durationSeconds>
                <status>OK</status>
             </cell>
             <cell>
                <distanceMeters>234196.77000000002</distanceMeters>
                <durationSeconds>9451.57</durationSeconds>
                <status>OK</status>
             </cell>
          </cells>
       </row>
       <row>
          <cells>
             <cell>
                <distanceMeters>1103695.95</distanceMeters>
                <durationSeconds>37887.43</durationSeconds>
                <status>OK</status>
             </cell>
             <cell>
                <distanceMeters>930934.64</distanceMeters>
                <durationSeconds>31773.440000000002</durationSeconds>
                <status>OK</status>
             </cell>
          </cells>
       </row>
    </rows>
</matrixResultV2>
```

Retours possibles

Cas d'un itinéraire trouvé (matrixResultV2/status est OK)

```
<?xml version="1.0" encoding="UTF-8"?>
<matrixResultV2>
   <status>OK</status>
   <origins>
      <origin>315846.96,2254268.35</origin>
      <origin>313584.77,2251648.97</origin>
   </origins>
   <destinations>
      <destination>321442.89,2251013.98</destination>
      <destination>318982.27,2248315.22</destination>
   </destinations>
   <srs>epsg:27572</srs>
   <rows>
      <row>
         <cells>
            <cell>
               <distanceMeters>8109.04</distanceMeters>
               <durationSeconds>894.48</durationSeconds>
               <status>OK</status>
            </cell>
            <cell>
               <distanceMeters>11127.53</distanceMeters>
               <durationSeconds>873.21</durationSeconds>
               <status>OK</status>
            </cell>
         </cells>
```

```
      </row>
      <row>
         <cells>
            <cell>
               <distanceMeters>11736.87</distanceMeters>
               <durationSeconds>1158.04</durationSeconds>
               <status>OK</status>
            </cell>
            <cell>
               <distanceMeters>7514.82</distanceMeters>
               <durationSeconds>615.79</durationSeconds>
               <status>OK</status>
            </cell>
         </cells>
      </row>
   </rows>
</matrixResultV2>
```

Cas d'un oubli de spécification d'origine ou de destination (matrixResultV2/status est ERROR)

```
<matrixResultV2>
   <message>Origins and destinations must be not null</message>
   <status>ERROR</status>
</matrixResultV2>
```

Cas d'un mauvais typage (serviceResult/status est ERROR)

```
<serviceResult>
        <message>NumberFormatException: For input string: "AAA"</message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'une erreur d'accrochage au graphe (matrixResultV2/status est OK) / (cell/status est KO)

```
<matrixResultV2>
   <status>OK</status>
   <origins>
      <origin>-0.36147,49.18392</origin>
      <origin>7.26132,43.70629</origin>
   </origins>
   <destinations>
      <destination>146.08821,48.43253</destination>
      <destination>2.34878,48.86473</destination>
   </destinations>
   <rows>
      <row>
         <cells>
            <cell>
               <distanceMeters>0.0</distanceMeters>
               <durationSeconds>0.0</durationSeconds>
               <status>KO</status>
            </cell>
            <cell>
               <distanceMeters>234196.77000000002</distanceMeters>
               <durationSeconds>9451.57</durationSeconds>
               <status>OK</status>
            </cell>
         </cells>
      </row>
      <row>
```

```
          <cells>
             <cell>
                <distanceMeters>0.0</distanceMeters>
                <durationSeconds>0.0</durationSeconds>
                <status>KO</status>
             </cell>
             <cell>
                <distanceMeters>930934.64</distanceMeters>
                <durationSeconds>31773.440000000002</durationSeconds>
                <status>OK</status>
             </cell>
          </cells>
       </row>
    </rows>
 </matrixResultV2>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in matrix computation
Error in smartrouting
datasource is null</message>
   <status>ERROR</status>
</serviceResult>
```

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
origins	Liste des coordonnées des points d'origines. Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	
destinations	Liste des coordonnées des points de destinations. Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	
graphName	Nom du graphe à utiliser	oui	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
profileId	Identifiant du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
profileName	Profil du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
rejectFlags	Déprécié, remplacé par exclusions		
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère , ou ; (Exemple : "Toll", "Tunnel", "Bridge")	oui	

En sortie

Matrice (matrixResult)

paramètre	type	min/max	description
originLocations/ originLocation (ou	string (ou array of originLocations/	0/illimité	positions d'origine

paramètre	type	min/max	description
originLocations en JSON / JSON-P)	originLocation en JSON / JSON-P)		
destinationLocations/ destinationLocation (ou destinationLocations en JSON / JSON-P)	string (ou array of destinationLocations/ destinationLocation en JSON / JSON-P)	0/illimité	positions de destination
rows/row (ou rows en JSON / JSON-P)	matrixRow (ou array of rows/row (matrixRow) en JSON / JSON-P)	0/illimité	Ligne de matrice

Ligne de matrice (matrixRow)

paramètre	type	min/max	description
cells/cell (ou cells en JSON / JSON-P)	matrixCell (ou array of cells/cell (matrixCell) en JSON / JSON-P)	0/illimité	Cellule de matrice

Cellule de matrice (matrixCell)

paramètre	type	min/max	description
distance	double	1/1	Distance de cellule de matrice (en mètres)
duration	double	1/1	Durée de cellule de matrice (en secondes)
status	string	0/1	Status cellule de matrice : - OK : l'itinéraire a bien été trouvé pour cette cellule - KO : pas d'itinéraire trouvé pour cette cellule

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/matrixService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:matrix>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <srs></srs>
            <!--Optional:-->
            <origins>
               <!--1 or more repetitions:-->
               <origin>
                  <x>-0.361470</x>
                  <y>49.183920</y>
               </origin>
               <origin>
                  <x>7.261320</x>
                  <y>43.706290</y>
```

```
               </origin>
            </origins>
            <!--Optional:-->
            <destinations>
               <!--1 or more repetitions:-->
               <destination>
                  <x>0.088210</x>
                  <y>48.432530</y>
               </destination>
               <destination>
                  <x>2.348780</x>
                  <y>48.864730</y>
               </destination>
            </destinations>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <method></method>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions></exclusions>
         </request>
      </sch:matrix>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:matrixResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <MatrixResult>
            <status>OK</status>
            <originLocations>
                              <originLocation>-0,361470;49,183920</originLocation>
                              <originLocation>7,261320;43,706290</originLocation>
            </originLocations>
            <destinationLocations>
                              <destinationLocation>0,088210;48,432530</destinationLocation>
                              <destinationLocation>2,348780;48,864730</destinationLocation>
            </destinationLocations>
            <rows>
               <row>
                  <cells>
                     <cell>
                        <distance>107898.07</distance>
                        <duration>3592.48</duration>
                        <status>OK</status>
                     </cell>
                     <cell>
                        <distance>233393.72</distance>
                        <duration>7063.8</duration>
                        <status>OK</status>
                     </cell>
                  </cells>
               </row>
               <row>
                  <cells>
                     <cell>
                        <distance>1105102.84</distance>
```

```
                        <duration>32465.86</duration>
                        <status>OK</status>
                     </cell>
                     <cell>
                        <distance>931394.85</distance>
                        <duration>27277.25</duration>
                        <status>OK</status>
                     </cell>
                  </cells>
               </row>
            </rows>
         </MatrixResult>
      </ns2:matrixResponse>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/matrix.json?
origins=-0.36147,49.18392;7.26132,43.70629&destinations=0.08821,48.43253;2.34878,48.86473
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/matrix.json?
origins=-0.36147,49.18392;7.26132,43.70629&destinations=0.08821,48.43253;2.34878,48.86473&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/matrix.xml?
origins=-0.36147,49.18392;7.26132,43.70629&destinations=0.08821,48.43253;2.34878,48.86473
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message":null,
   "status":"OK",
   "originLocations":[
      "-0,361470;49,183920",
      "7,261320;43,706290"
   ],
   "destinationLocations":[
      "0,088210;48,432530",
      "2,348780;48,864730"
   ],
   "rows":[
      {
         "cells":[
            {
               "distance":107528.37,
               "duration":5139.2300000000005,
```

```
               "status":"OK"
            },
            {
               "distance":234732.19,
               "duration":9880.54,
               "status":"OK"
            }
         ]
      },
      {
         "cells":[
            {
               "distance":1103920.65,
               "duration":38946.42,
               "status":"OK"
            },
            {
               "distance":930731.03,
               "duration":32689.65,
               "status":"OK"
            }
         ]
      }
   ]
}
```


Format JSON-P


```
myCallback(
        {
           "message":null,
           "status":"OK",
           "originLocations":[
                  "-0,361470;49,183920",
                  "7,261320;43,706290"
           ],
           "destinationLocations":[
                  "0,088210;48,432530",
                  "2,348780;48,864730"
           ],
           "rows":[
                  {
                         "cells":[
                                {
                                   "distance":107528.37,
                                   "duration":5139.2300000000005,
                                   "status":"OK"
                                },
                                {
                                   "distance":234732.19,
                                   "duration":9880.54,
                                   "status":"OK"
                                }
                         ]
                  },
                  {
                         "cells":[
                                {
                                   "distance":1103920.65,
                                   "duration":38946.42,
                                   "status":"OK"
                                },
                                {
```

```
                                    "distance":930731.03,
                                    "duration":32689.65,
                                    "status":"OK"
                                }
                            ]
                        }
                    ]
                }
);
```

Format XML

```
<matrixResult>
        <status>OK</status>
        <originLocations>
                <originLocation>-0,361470;49,183920</originLocation>
                <originLocation>7,261320;43,706290</originLocation>
        </originLocations>
        <destinationLocations>
                <destinationLocation>0,088210;48,432530</destinationLocation>
                <destinationLocation>2,348780;48,864730</destinationLocation>
        </destinationLocations>
        <rows>
                <row>
                        <cells>
                                <cell>
                                        <distance>107528.37</distance>
                                        <duration>5139.2300000000005</duration>
                                        <status>OK</status>
                                </cell>
                                <cell>
                                        <distance>234732.19</distance>
                                        <duration>9880.54</duration>
                                        <status>OK</status>
                                </cell>
                        </cells>
                </row>
                <row>
                        <cells>
                                <cell>
                                        <distance>1103920.65</distance>
                                        <duration>38946.42</duration>
                                        <status>OK</status>
                                </cell>
                                <cell>
                                        <distance>930731.03</distance>
                                        <duration>32689.65</duration>
                                        <status>OK</status>
                                </cell>
                        </cells>
                </row>
        </rows>
</matrixResult>
```

Retours possibles

Cas d’un itinéraire trouvé (matrixResult/status est OK)

```
<matrixResult>
        <status>OK</status>
        <originLocations>
```

```
                <originLocation>-0,361470;49,183920</originLocation>
                <originLocation>7,261320;43,706290</originLocation>
        </originLocations>
        <destinationLocations>
                <destinationLocation>0,088210;48,432530</destinationLocation>
                <destinationLocation>2,348780;48,864730</destinationLocation>
        </destinationLocations>
        <rows>
                <row>
                        <cells>
                                <cell>
                                        <distance>107528.37</distance>
                                        <duration>5139.2300000000005</duration>
                                        <status>OK</status>
                                </cell>
                                <cell>
                                        <distance>234732.19</distance>
                                        <duration>9880.54</duration>
                                        <status>OK</status>
                                </cell>
                        </cells>
                </row>
                <row>
                        <cells>
                                <cell>
                                        <distance>1103920.65</distance>
                                        <duration>38946.42</duration>
                                        <status>OK</status>
                                </cell>
                                <cell>
                                        <distance>930731.03</distance>
                                        <duration>32689.65</duration>
                                        <status>OK</status>
                                </cell>
                        </cells>
                </row>
        </rows>
</matrixResult>
```

Cas d’un oubli de spécification d’origine et de destination (matrixResult/status est OK)

```
<matrixResult>
        <status>OK</status>
        <originLocations/>
        <destinationLocations/>
</matrixResult>
```

Cas d’un mauvais typage (serviceResult/status est ERROR)

```
<serviceResult>
        <message>NumberFormatException: For input string: "AAA"</message>
        <status>ERROR</status>
</serviceResult>
```

Cas d’une erreur d’accrochage au graphe (matrixResult/status est OK) / (matrixCell/status est KO)

```
<matrixResult>
        <status>OK</status>
        <originLocations>
                <originLocation>-0,361470;49,183920</originLocation>
```

```
                <originLocation>57,261320;43,706290</originLocation>
        </originLocations>
        <destinationLocations>
                <destinationLocation>0,088210;48,432530</destinationLocation>
                <destinationLocation>2,348780;48,864730</destinationLocation>
        </destinationLocations>
        <rows>
                <row>
                        <cells>
                                <cell>
                                        <distance>107528.37</distance>
                                        <duration>5139.2300000000005</duration>
                                        <status>OK</status>
                                </cell>
                                <cell>
                                        <distance>234732.19</distance>
                                        <duration>9880.54</duration>
                                        <status>OK</status>
                                </cell>
                        </cells>
                </row>
                <row>
                        <cells>
                                <cell>
                                        <distance>0.0</distance>
                                        <duration>0.0</duration>
                                        <status>KO</status>
                                </cell>
                                <cell>
                                        <distance>0.0</distance>
                                        <duration>0.0</duration>
                                        <status>KO</status>
                                </cell>
                        </cells>
                </row>
        </rows>
</matrixResult>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in matrix computation
Error in smartrouting
datasource is null</message>
   <status>ERROR</status>
</serviceResult>
```

FAQ

1. Peut-on utiliser des alias à défaut des noms de fichiers .siti pour appeler une datasource ?

   Oui, cf. détails dans la FAQ du Web Service du géocodage inverse.

2. Comment utiliser les statistiques routières ?

   Vérifier que le graphe contient bien ces statistiques et utiliser les deux paramètres suivants : `computeOptions` avec la valeur *trafficPatterns* et `startDateTime` pour préciser la date/heure de départ.

3. Comment faire un calcul de matrice d'itinéraires sans péage?

Si la contrainte *Toll* a bien été incluse dans le graphe, placer une exclusion dans **`exclusions`** :

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:matrix>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <srs></srs>
            <!--Optional:-->
            <origins>
               <!--1 or more repetitions:-->
               <origin>
                  <x>-0.361470</x>
                  <y>49.183920</y>
               </origin>
               <origin>
                  <x>7.261320</x>
                  <y>43.706290</y>
               </origin>
            </origins>
            <!--Optional:-->
            <destinations>
               <!--1 or more repetitions:-->
               <destination>
                  <x>0.088210</x>
                  <y>48.432530</y>
               </destination>
               <destination>
                  <x>2.348780</x>
                  <y>48.864730</y>
               </destination>
            </destinations>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <method></method>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions>Toll</exclusions>
         </request>
      </sch:matrix>
   </soapenv:Body>
</soapenv:Envelope>
```

4. Qu’est-ce que les Speed Patterns ? Comment les utiliser ?

   Afin de proposer des temps de trajets au plus près des conditions de circulation, les graphes fournis par GEOCONCEPT SAS intègrent, à partir de la version M18, pour les voitures et pour les camions 5 profils de vitesses (Speed Patterns) pour tenir compte des différents niveaux de congestion dans une journée :

   - standard *normal-speed* correspond à la vitesse d’une heure moyennement chargée (11h)
   - nuit *fast-speed* correspond à un trafic très fluide, observé le plus souvent la nuit (3h)
   - chargée *slow-speed* correspond aux heures de trafic denses (8h)

- heure de pointe *very-slow-speed* correspond aux heures de trafic denses dans les grandes agglomérations, plus lent que le précédent (8h)
- défaut *defaut* correspond aux vitesses moyennées sur une journée entière

Pour les utiliser il faut passer, lors de l'appel au web service, le paramètre **`computeOptions`** avec l'option *speedPattern* et préciser la valeur de la Speed Pattern demandée sans omettre un profil de véhicule
Exemple :

```
&computeOptions=speedPattern:fast-speed&profileId=1
```

5. Comment utiliser les attributs poids lourds ?

   Il faut que le graphe intègre les attributs de poids lourds (en standard dans les graphes fournis par GEOCONCEPT SAS à partir de la version M18) et soit calculer un itinéraire en utilisant un profil de véhicule utilisant les restrictions (cf. le catalogue de véhicules, éditable, définit dans le fichier SmartRoutingVehicles.xml, dans le dossier `*'<GEOCONCEPT_WEB_HOME>'\smartrouting\jee\smartrouting\conf\* ), soit surcharger l'appel au web service en utilisant le paramètre **`computeOptions`** avec les options *length*, *width*, *height*, *weight*, *axles* et/ou *weightPerAxle*.
   Exemple pour un trajet avec un véhicule de 4,5 mètres de hauteur :

   ```
   &computeOptions=height:450
   ```

Calcul de matrice compacte

Cette page n'est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

Ce web service calcule une matrice d'itinéraires pour un ensemble de points et retourne un distancier.
Il s'appuie sur le graphe paramétré dont le nom a été spécifié dans l'interface d'administration de Geoconcept Web.

Ce web service est une variation du web service de calcul de matrice qui retourne exactement les mêmes résultats mais dans un autre formalisme.

Disponibilité

Ce web service est disponible en permanence avec Geoconcept Web et un graphe.

Changement de version

Les versions précédente du web service sont conservées dans Geoconcept Web pour assurer la compatibilité avec les développements antérieurs. Il est recommandé d'utiliser la version la plus récente.

Changements avec la V3

- Ajout des paramètres "timeOut" et "computeOptions".
- Ajout dans la snapMethod "nodes" de l'accrochage aux noeuds les plus proches.

Changements avec la V2

- Ajout de la notion de noeud, plus rapide, pour s'accrocher aux noeuds du graphe plutôt qu'a des coordonnées géographiques. Ajout des éléments suivants : "originNodes", "destinationNodes" et de "nodes" dans les snapMethod.
- Ajout du paramètre "maxCost".

V3

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
origins	Liste des coordonnées des points d'origines. Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	
originNodes	Liste des noeuds ids d'origines. Les Noeuds ids sont séparées par la caractère ,. Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui	
destinations	Liste des coordonnées des points de destinations. Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	
destinationNodes	Liste des noeuds ids de destinations. Les Noeuds ids sont séparées par la caractère ,. Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui	
graphName (déprécié)	Nom du graphe à utiliser Ce paramètre est omis si le paramètre configName est utilisé.	oui	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
profileId (déprécié)	Identifiant du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileName (déprécié)	Profil du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère *;* (Exemple : Toll, Tunnel, Bridge)	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris. Attention le caractère + peut être mal interprété par les navigateurs, dans ce cas, il faut le remplacer par *%2B*.	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction - nodes : Accrochage direct aux noeuds fournis par les paramètres originNode, destinationNode et waypointNodes ou, si ces premiers ne	oui	standard

paramètre	description	optionnel	défaut
	sont pas renseignés, aux noeuds les plus proches des paramètres origin, destination et waypoint		
avoidArea	Zone de transit interdit au format WKT (POLYGON ou MULTIPOLYGON) dans la projection (paramètre srs) demandée Exemple en wgs84 : `POLYGON ((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689))` - `MULTIPOLYGON (((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689)), ((-1.558639 47.218144, -1.558639 47.218193, [...] -1.558639 47.218144)))` Attention les géométries WKT doivent être fermées.	oui	
configName	Nom de la configuration à utiliser (défini dans Geoconcept Web - Administration / Outils / Définitions des graphes) Il remplace l'usage de graphName, profileId et profileName	oui	
computeOptions	Liste des options pour le calcul, séparées par le caractère *;* - trafficPatterns : utilise les statistiques routières (il est nécessaire de renseigner le paramètre startDateTime et d'utiliser un graphe intégrant les informations de traffic patterns) - speedPattern (M18) : utilise d'une *speed pattern* tel que définis dans le fichier SmartRoutingVehicles.xml qui se trouve dans le dossier `*"<GEOCONCEPT_WEB_HOME>"\smartrouting\jee\smartrouting\conf* . Usage : "speedPattern:slow-speed" - length (M18) : longueur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "length:950" - width (M18) : largeur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "width:255" - height (M18) : hauteur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "height:360" - weight (M18) : poids maximum autorisé en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weight:18000" - axles (M18) : nombre maximum d'axe autorisé (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "axles:2" - weightPerAxle (M18) : poids maximum autorisé par axe en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weightPerAxle:9000" - snapSpeed : vitesse d'accrochage au graphe en kilomètre par heure. Usage : "snapSpeed:10"	oui	
maxCost	Coût maximum à ne pas dépasser dans les résultats -1 : pas de coût maximum à prendre en compte 0 : prendre la valeur par default défini dans la configuration de SmartRouting Server sinon : valeur en mètres si method=distance ou en secondes si method=time	oui	
timeOut	Time out pour le calcul (en millisecondes)	oui	

(*) Au moins l'un des deux couples de paramètres origins/destinations ou originNodes/destinationsNodes doit être renseigné.

(M18) Disponible à partir de la version M18 des graphes fournis par GEOCONCEPT SAS.

En sortie

Matrice (compactMatrixResultV3)

paramètre	type	min/max	description
distanceMeters/ distanceMeter	array (compactRowV3)	0/illimité	Résultat de la matrice de distance (en mètres)
durationSeconds/ durationSecond	array (compactRowV3)	0/illimité	Résultat de la matrice en durée (en secondes)

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/compactMatrixService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:compactMatrixV3>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <srs>epsg:27572</srs>
            <!--Optional:-->
            <origins>
               <!--1 or more repetitions:-->
               <origin>
                  <x>315846.96</x>
                  <y>2254268.35</y>
               </origin>
                <origin>
                  <x>313584.77</x>
                  <y>2251648.97</y>
               </origin>
            </origins>
            <!--Optional:-->
            <originNodes>
               <!--1 or more repetitions:-->
               <originNode></originNode>
            </originNodes>
            <!--Optional:-->
            <destinations>
               <!--1 or more repetitions:-->
               <destination>
                  <x>321442.89</x>
                  <y>2251013.98</y>
               </destination>
                <destination>
                  <x>318982.27</x>
                  <y>2248315.22</y>
               </destination>
            </destinations>
            <!--Optional:-->
            <destinationNodes>
               <!--1 or more repetitions:-->
               <destinationNode></destinationNode>
            </destinationNodes>
```

```
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <method>time</method>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <snapMethod></snapMethod>
            <!--Optional:-->
            <avoidArea></avoidArea>
            <!--Optional:-->
            <configName></configName>
            <!--Optional:-->
            <computeOptions></computeOptions>
            <!--Optional:-->
            <timeOut></timeOut>
            <!--Optional:-->
            <maxCost></maxCost>
         </request>
      </sch:compactMatrixV3>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:compactMatrixV3Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <CompactMatrixResult>
            <status>OK</status>
            <distanceMeters>
               <distanceMeter>8109,11249</distanceMeter>
               <distanceMeter>11737,7637</distanceMeter>
            </distanceMeters>
            <durationSeconds>
               <durationSecond>565,816</durationSecond>
               <durationSecond>857,526</durationSecond>
            </durationSeconds>
         </CompactMatrixResult>
      </ns2:compactMatrixV3Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/compactMatrix/v3.json?
origins=315846.96,2254268.35;313584.77,2251648.97&destinations=321442.89,2251013.98;318982.27,2248315.22&srs=epsg:27572&metho
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/compactMatrix/v3.json?
origins=315846.96,2254268.35;313584.77,2251648.97&destinations=321442.89,2251013.98;318982.27,2248315.22&srs=epsg:27572&me
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
        "message": null,
        "status": "OK",
        "distanceMeters": [
                [
                        8109,
                        11249
                ],
                [
                        11737,
                        7637
                ]
        ],
        "durationSeconds": [
                [
                        657,
                        889
                ],
                [
                        960,
                        609
                ]
        ]
}
```

Format JSON-P

```
myCallback(
    {
        "message": null,
        "status": "OK",
        "distanceMeters": [
            [
                8109,
                11249
            ],
            [
                11737,
                7637
            ]
        ],
        "durationSeconds": [
            [
                657,
                889
            ],
            [
                960,
                609
```

```
            ]
        ]
    }
);
```

Retours possibles

Cas d'un itinéraire trouvé (compactMatrixResultV2/status est OK)

```
<compactMatrixResultV3>
   <status>OK</status>
   <distanceMeters>
      <distanceMeter>8109,11249</distanceMeter>
      <distanceMeter>11737,7637</distanceMeter>
   </distanceMeters>
   <durationSeconds>
      <durationSecond>657,889</durationSecond>
      <durationSecond>960,609</durationSecond>
   </durationSeconds>
</compactMatrixResultV3>
```

Cas d'un oubli de spécification d'origine ou de destination (compactMatrixResultV2/status est ERROR)

```
<compactMatrixResultV3>
   <message>Origins and destinations must be not null</message>
   <status>ERROR</status>
</compactMatrixResultV3>
```

Cas d'un mauvais typage (serviceResult/status est ERROR)

```
<serviceResult>
        <message>NumberFormatException: For input string: "AAA"</message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'une erreur d'accrochage au graphe (compactMatrixResultV2/status est OK) / (cell/status est KO)

```
<compactMatrixResultV3>
   <status>OK</status>
   <origins>
      <origin>-0.36147,49.18392</origin>
      <origin>7.26132,43.70629</origin>
   </origins>
   <destinations>
      <destination>146.08821,48.43253</destination>
      <destination>2.34878,48.86473</destination>
   </destinations>
   <rows>
      <row>
         <cells>
            <cell>
               <distanceMeters>0.0</distanceMeters>
               <durationSeconds>0.0</durationSeconds>
               <status>KO</status>
            </cell>
            <cell>
               <distanceMeters>234196.77000000002</distanceMeters>
               <durationSeconds>9451.57</durationSeconds>
```

```
                <status>OK</status>
             </cell>
          </cells>
       </row>
       <row>
          <cells>
             <cell>
                <distanceMeters>0.0</distanceMeters>
                <durationSeconds>0.0</durationSeconds>
                <status>KO</status>
             </cell>
             <cell>
                <distanceMeters>930934.64</distanceMeters>
                <durationSeconds>31773.440000000002</durationSeconds>
                <status>OK</status>
             </cell>
          </cells>
       </row>
    </rows>
 </compactMatrixResultV3>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in matrix computation
Error in smartrouting
datasource is null</message>
   <status>ERROR</status>
</serviceResult>
```

V2

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
origins	Liste des coordonnées des points d'origines. Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	
originNodes	Liste des noeuds ids d'origines. Les Noeuds ids sont séparées par la caractère ,. Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui	
destinations	Liste des coordonnées des points de destinations. Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	
destinationNodes	Liste des noeuds ids de destinations. Les Noeuds ids sont séparées par la caractère ,. Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui	
graphName	Nom du graphe à utiliser Ce paramètre est omis si le paramètre configName est utilisé.	oui	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
profileId	Identifiant du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	

paramètre	description	optionnel	défaut
profileName	Profil du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère , ou *;* (Exemple : "Toll", "Tunnel", "Bridge")	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris. Attention le caractère + peut être mal interprété par les navigateurs, dans ce cas, il faut le remplacer par *%2B*.	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction - nodes : accrochage direct aux noeuds indiqués par locationNode et les paramètres *node* des ressources	oui	standard
avoidArea	Zone de transit interdit au format WKT (POLYGON ou MULTIPOLYGON) dans la projection (paramètre srs) demandée Exemple en wgs84 : `POLYGON ((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689))` - `MULTIPOLYGON (((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689)), ((-1.558639 47.218144, -1.558639 47.218193, [...] -1.558639 47.218144)))` Attention les géométries WKT doivent être fermées.	oui	
configName	Nom de la configuration à utiliser (défini dans Geoconcept Web - Administration / Outils / Définitions des graphes) Il remplace l'usage de graphName, profileId et profileName	oui	
maxCost	Coût maximum à ne pas dépasser dans les résultats -1 : pas de coût maximum à prendre en compte 0 : prendre la valeur par default défini dans la configuration de SmartRouting Server sinon : valeur en mètres si method=distance ou en secondes si method=time	oui	

(*) Au moins l'un des deux couples de paramètres origins/destinations ou originNodes/destinationsNodes doit être renseigné.

En sortie

Matrice (compactMatrixResultV2)

paramètre	type	min/max	description
distanceMeters/ distanceMeter	array (compactRowV2)	0/illimité	Résultat de la matrice de distance (en mètres)
durationSeconds/ durationSecond	array (compactRowV2)	0/illimité	Résultat de la matrice en durée (en secondes)

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/compactMatrixService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:compactMatrixV2>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <srs>epsg:27572</srs>
            <!--Optional:-->
            <origins>
               <!--1 or more repetitions:-->
               <origin>
                  <x>315846.96</x>
                  <y>2254268.35</y>
               </origin>
                <origin>
                  <x>313584.77</x>
                  <y>2251648.97</y>
               </origin>
            </origins>
            <!--Optional:-->
            <originNodes>
               <!--1 or more repetitions:-->
               <originNode></originNode>
            </originNodes>
            <!--Optional:-->
            <destinations>
               <!--1 or more repetitions:-->
               <destination>
                  <x>321442.89</x>
                  <y>2251013.98</y>
               </destination>
                <destination>
                  <x>318982.27</x>
                  <y>2248315.22</y>
               </destination>
            </destinations>
            <!--Optional:-->
            <destinationNodes>
               <!--1 or more repetitions:-->
               <destinationNode></destinationNode>
            </destinationNodes>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <method>time</method>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
```

```
            <snapMethod></snapMethod>
            <!--Optional:-->
            <avoidArea></avoidArea>
            <!--Optional:-->
            <configName></configName>
            <!--Optional:-->
            <maxCost></maxCost>
         </request>
      </sch:compactMatrixV2>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:compactMatrixV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <CompactMatrixResult>
            <status>OK</status>
            <distanceMeters>
               <distanceMeter>8109,11249</distanceMeter>
               <distanceMeter>11737,7637</distanceMeter>
            </distanceMeters>
            <durationSeconds>
               <durationSecond>565,816</durationSecond>
               <durationSecond>857,526</durationSecond>
            </durationSeconds>
         </CompactMatrixResult>
      </ns2:compactMatrixV2Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/compactMatrix/v2.json?
origins=315846.96,2254268.35;313584.77,2251648.97&destinations=321442.89,2251013.98;318982.27,2248315.22&srs=epsg:27572&metho
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/compactMatrix/v2.json?
origins=315846.96,2254268.35;313584.77,2251648.97&destinations=321442.89,2251013.98;318982.27,2248315.22&srs=epsg:27572&metho
```

Requête XML

```
http://<server>/<webapp>/api/lbs/compactMatrix/v2.xml?
origins=315846.96,2254268.35;313584.77,2251648.97&destinations=321442.89,2251013.98;318982.27,2248315.22&srs=epsg:27572&metho
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
        "message": null,
        "status": "OK",
        "distanceMeters": [
                [
                        8109,
                        11249
                ],
                [
                        11737,
                        7637
                ]
        ],
        "durationSeconds": [
                [
                        657,
                        889
                ],
                [
                        960,
                        609
                ]
        ]
}
```


Format JSON-P


```
myCallback(
    {
        "message": null,
        "status": "OK",
        "distanceMeters": [
            [
                8109,
                11249
            ],
            [
                11737,
                7637
            ]
        ],
        "durationSeconds": [
            [
                657,
                889
            ],
            [
                960,
                609
            ]
        ]
    }
);
```


Format XML


```
<compactMatrixResultV2>
   <status>OK</status>
   <distanceMeters>
      <distanceMeter>8109,11249</distanceMeter>
      <distanceMeter>11737,7637</distanceMeter>
```

```
   </distanceMeters>
   <durationSeconds>
      <durationSecond>657,889</durationSecond>
      <durationSecond>960,609</durationSecond>
   </durationSeconds>
</compactMatrixResultV2>
```

Retours possibles

Cas d'un itinéraire trouvé (compactMatrixResultV2/status est OK)

```
<compactMatrixResultV2>
   <status>OK</status>
   <distanceMeters>
      <distanceMeter>8109,11249</distanceMeter>
      <distanceMeter>11737,7637</distanceMeter>
   </distanceMeters>
   <durationSeconds>
      <durationSecond>657,889</durationSecond>
      <durationSecond>960,609</durationSecond>
   </durationSeconds>
</compactMatrixResultV2>
```

Cas d'un oubli de spécification d'origine ou de destination (compactMatrixResultV2/status est ERROR)

```
<compactMatrixResultV2>
   <message>Origins and destinations must be not null</message>
   <status>ERROR</status>
</compactMatrixResultV2>
```

Cas d'un mauvais typage (serviceResult/status est ERROR)

```
<serviceResult>
        <message>NumberFormatException: For input string: "AAA"</message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'une erreur d'accrochage au graphe (compactMatrixResultV2/status est OK) / (cell/status est KO)

```
<compactMatrixResultV2>
   <status>OK</status>
   <origins>
      <origin>-0.36147,49.18392</origin>
      <origin>7.26132,43.70629</origin>
   </origins>
   <destinations>
      <destination>146.08821,48.43253</destination>
      <destination>2.34878,48.86473</destination>
   </destinations>
   <rows>
      <row>
         <cells>
            <cell>
               <distanceMeters>0.0</distanceMeters>
               <durationSeconds>0.0</durationSeconds>
               <status>KO</status>
            </cell>
            <cell>
               <distanceMeters>234196.77000000002</distanceMeters>
```

```
                <durationSeconds>9451.57</durationSeconds>
                <status>OK</status>
             </cell>
          </cells>
       </row>
       <row>
          <cells>
             <cell>
                <distanceMeters>0.0</distanceMeters>
                <durationSeconds>0.0</durationSeconds>
                <status>KO</status>
             </cell>
             <cell>
                <distanceMeters>930934.64</distanceMeters>
                <durationSeconds>31773.440000000002</durationSeconds>
                <status>OK</status>
             </cell>
          </cells>
       </row>
    </rows>
 </compactMatrixResultV2>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in matrix computation
Error in smartrouting
datasource is null</message>
   <status>ERROR</status>
</serviceResult>
```

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
origins	Liste des coordonnées des points d'origines. Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	
destinations	Liste des coordonnées des points de destinations. Les coordonnées longitude et latitude sont séparées par la caractère ,	oui	
graphName	Nom du graphe à utiliser	oui	
method	itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
profileId	Identifiant du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
profileName	Profil du véhicule (enregistré dans les profils de véhicule) à utiliser	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère , ou ; (Exemple : "Toll", "Tunnel", "Bridge")	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris	oui	
snapMethod	Méthode d'accrochage au graphe	oui	standard

paramètre	description	optionnel	défaut
	- standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction		

En sortie

Matrice (matrixResultV2)

paramètre	type	min/max	description
origins/origin (ou origins en JSON / JSON-P)	string (ou array of origins/origin en JSON / JSON-P)	0/illimité	positions d'origine.
destinations/destination (ou destinations en JSON / JSON-P)	string (ou array of destinations/destination en JSON / JSON-P)	0/illimité	positions de destination.
rows/row (ou rows en JSON / JSON-P)	matrixRowV2 (ou array of rows/row (matrixRow) en JSON / JSON-P)	0/illimité	Ligne de matrice

Ligne de matrice (matrixRowV2)

paramètre	type	min/max	description
cells/cell (ou cells en JSON / JSON-P)	matrixCellV2 (ou array of cells/cell (matrixCellV2) en JSON / JSON-P)	0/illimité	Cellule de matrice

Cellule de matrice (matrixCellV2)

paramètre	type	min/max	description
distanceMeters	double	1/1	Distance de cellule de matrice (en mètres)
durationSeconds	double	1/1	Durée de cellule de matrice (en secondes)
status	string	0/1	Status cellule de matrice : - OK : l'itinéraire a bien été trouvé pour cette cellule - KO : pas d'itinéraire trouvé pour cette cellule

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/compactMatrixService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:compactMatrixV1>
         <!--Optional:-->
```

```
         <request>
            <!--Optional:-->
            <srs>epsg:27572</srs>
            <!--Optional:-->
            <origins>
               <!--1 or more repetitions:-->
               <origin>
                  <x>315846.96</x>
                  <y>2254268.3500000001</y>
               </origin>
                <origin>
                           <x>313584.77</x>
                  <y>2251648.9700000002</y>
               </origin>
            </origins>
            <!--Optional:-->
            <destinations>
               <!--1 or more repetitions:-->
               <destination>
                  <x>321442.89</x>
                  <y>2251013.98</y>
               </destination>
                <destination>
                  <x>318982.27</x>
                  <y>2248315.2200000002</y>
               </destination>
            </destinations>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <method>time</method>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <snapMethod></snapMethod>
         </request>
      </sch:compactMatrixV1>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:compactMatrixV1Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <MatrixResult>
            <status>OK</status>
            <origins>
               <origin>315846.96,2254268.35</origin>
               <origin>313584.77,2251648.97</origin>
            </origins>
            <destinations>
               <destination>321442.89,2251013.98</destination>
               <destination>318982.27,2248315.22</destination>
```

```
            </destinations>
            <srs>epsg:27572</srs>
            <rows>
               <row>
                  <cells>
                     <cell>
                        <distanceMeters>8109.04</distanceMeters>
                        <durationSeconds>894.48</durationSeconds>
                        <status>OK</status>
                     </cell>
                     <cell>
                        <distanceMeters>11127.53</distanceMeters>
                        <durationSeconds>873.21</durationSeconds>
                        <status>OK</status>
                     </cell>
                  </cells>
               </row>
               <row>
                  <cells>
                     <cell>
                        <distanceMeters>11736.87</distanceMeters>
                        <durationSeconds>1158.04</durationSeconds>
                        <status>OK</status>
                     </cell>
                     <cell>
                        <distanceMeters>7514.82</distanceMeters>
                        <durationSeconds>615.79</durationSeconds>
                        <status>OK</status>
                     </cell>
                  </cells>
               </row>
            </rows>
         </MatrixResult>
      </ns2:compactMatrixV1Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/compactMatrix/v1.json?
origins=315846.96,2254268.3500000001;313584.77,2251648.9700000002&destinations=321442.89,2251013.98;318982.27,2248315.2200000
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/compactMatrix/v1.json?
origins=315846.96,2254268.3500000001;313584.77,2251648.9700000002&destinations=321442.89,2251013.98;318982.27,2248315.2200000
```

Requête XML

```
http://<server>/<webapp>/api/lbs/compactMatrix/v1.xml?
origins=315846.96,2254268.3500000001;313584.77,2251648.9700000002&destinations=321442.89,2251013.98;318982.27,2248315.2200000
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON


```
{
   "message": null,
   "status": "OK",
   "distanceMeters":    [
            [
         8109,
         11249
      ],
            [
         11737,
         7637
      ]
   ],
   "durationSeconds":    [
            [
         657,
         889
      ],
            [
         960,
         609
      ]
   ]
}
```


Format JSON-P


```
myCallback(
    {
       "message": null,
       "status": "OK",
       "distanceMeters":    [
                [
             8109,
             11249
          ],
                [
             11737,
             7637
          ]
       ],
       "durationSeconds":    [
                [
             657,
             889
          ],
                [
             960,
             609
          ]
       ]
    }
);
```


Format XML


```
<compactMatrixResult>
   <status>OK</status>
   <distanceMeters>
```

```
      <distanceMeter>8109,11249</distanceMeter>
      <distanceMeter>11737,7637</distanceMeter>
   </distanceMeters>
   <durationSeconds>
      <durationSecond>657,889</durationSecond>
      <durationSecond>960,609</durationSecond>
   </durationSeconds>
</compactMatrixResult>
```

Retours possibles

Cas d'un itinéraire trouvé (matrixResultV2/status est OK)

```
<compactMatrixResult>
   <status>OK</status>
   <distanceMeters>
      <distanceMeter>8109,11249</distanceMeter>
      <distanceMeter>11737,7637</distanceMeter>
   </distanceMeters>
   <durationSeconds>
      <durationSecond>657,889</durationSecond>
      <durationSecond>960,609</durationSecond>
   </durationSeconds>
</compactMatrixResult>
```

Cas d'un oubli de spécification d'origine ou de destination (compactMatrixResultV1/status est ERROR)

```
<comcompactMatrixResultV1>
   <message>Origins and destinations must be not null</message>
   <status>ERROR</status>
</compactMatrixResultV1>
```

Cas d'un mauvais typage (serviceResult/status est ERROR)

```
<serviceResult>
        <message>NumberFormatException: For input string: "AAA"</message>
        <status>ERROR</status>
</serviceResult>
```

Cas d'une erreur d'accrochage au graphe (compactMatrixResultV1/status est OK) / (cell/status est KO)

```
<compactMatrixResultV1>
   <status>OK</status>
   <origins>
      <origin>-0.36147,49.18392</origin>
      <origin>7.26132,43.70629</origin>
   </origins>
   <destinations>
      <destination>146.08821,48.43253</destination>
      <destination>2.34878,48.86473</destination>
   </destinations>
   <rows>
      <row>
         <cells>
            <cell>
               <distanceMeters>0.0</distanceMeters>
               <durationSeconds>0.0</durationSeconds>
               <status>KO</status>
            </cell>
```

```
            <cell>
               <distanceMeters>234196.77000000002</distanceMeters>
               <durationSeconds>9451.57</durationSeconds>
               <status>OK</status>
            </cell>
         </cells>
      </row>
      <row>
         <cells>
            <cell>
               <distanceMeters>0.0</distanceMeters>
               <durationSeconds>0.0</durationSeconds>
               <status>KO</status>
            </cell>
            <cell>
               <distanceMeters>930934.64</distanceMeters>
               <durationSeconds>31773.440000000002</durationSeconds>
               <status>OK</status>
            </cell>
         </cells>
      </row>
   </rows>
</compactMatrixResultV1>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (serviceResult/status est ERROR)

```
<serviceResult>
   <message>ServiceException: Error in matrix computation
Error in smartrouting
datasource is null</message>
   <status>ERROR</status>
</serviceResult>
```

FAQ

1. Peut-on utiliser des alias à défaut des noms de fichiers .siti pour appeler une datasource ?

   Oui, cf. détails dans la FAQ du Web Service du géocodage inverse.

2. Comment utiliser les statistiques routières ?

   Vérifier que le graphe contient bien ces statistiques et utiliser les deux paramètres suivants : **`computeOptions`** avec la valeur *trafficPatterns* et **`startDateTime`** pour préciser la date/heure de départ.

3. Comment faire un calcul de matrice d'itinéraires sans péage?

   Si la contrainte *Toll* a bien été incluse dans le graphe, placer une exclusion dans **`exclusions`** :

   ```
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
   geoconcept.com/gc/schemas">
      <soapenv:Header/>
      <soapenv:Body>
         <sch:matrix>
            <!--Optional:-->
            <request>
               <!--Optional:-->
               <srs></srs>
               <!--Optional:-->
               <origins>
   ```

```
               <!--1 or more repetitions:-->
               <origin>
                  <x>-0.361470</x>
                  <y>49.183920</y>
               </origin>
               <origin>
                  <x>7.261320</x>
                  <y>43.706290</y>
               </origin>
            </origins>
            <!--Optional:-->
            <destinations>
               <!--1 or more repetitions:-->
               <destination>
                  <x>0.088210</x>
                  <y>48.432530</y>
               </destination>
               <destination>
                  <x>2.348780</x>
                  <y>48.864730</y>
               </destination>
            </destinations>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <method></method>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <exclusions>Toll</exclusions>
         </request>
      </sch:matrix>
   </soapenv:Body>
</soapenv:Envelope>
```

4. Qu’est-ce que les Speed Patterns ? Comment les utiliser ?

   Afin de proposer des temps de trajets au plus près des conditions de circulation, les graphes fournis par GEOCONCEPT SAS intègrent, à partir de la version M18, pour les voitures et pour les camions 5 profils de vitesses (Speed Patterns) pour tenir compte des différents niveaux de congestion dans une journée :

   - standard *normal-speed* correspond à la vitesse d’une heure moyennement chargée (11h)
   - nuit *fast-speed* correspond à un trafic très fluide, observé le plus souvent la nuit (3h)
   - chargée *slow-speed* correspond aux heures de trafic denses (8h)
   - heure de pointe *very-slow-speed* correspond aux heures de trafic denses dans les grandes agglomérations, plus lent que le précédent (8h)
   - défaut *defaut* correspond aux vitesses moyennées sur une journée entière

   Pour les utiliser il faut passer, lors de l’appel au web service, le paramètre **`computeOptions`** avec l’option *speedPattern* et préciser la valeur de la Speed Pattern demandée sans omettre un profil de véhicule
   Exemple :

```
&computeOptions=speedPattern:fast-speed&profileId=1
```

5. Comment utiliser les attributs poids lourds ?

   Il faut que le graphe intègre les attributs de poids lourds (en standard dans les graphes fournis par GEOCONCEPT SAS à partir de la version M18) et soit calculer un itinéraire en utilisant un profil de véhicule utilisant les restrictions (cf. le catalogue de véhicules, éditable, définit dans le fichier SmartRoutingVehicles.xml, dans le dossier `'<*GEOCONCEPT_WEB_HOME*>"\*smartrouting*\*jee*\*smartrouting*\*conf*\ ), soit surcharger l'appel au web service en utilisant le paramètre **`computeOptions`** avec les options *length*, *width*, *height*, *weight*, *axles* et/ou *weightPerAxle*. Exemple pour un trajet avec un véhicule de 4,5 mètres de hauteur :

   ```
   &computeOptions=height:450
   ```

Search Around

Cette page n'est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

Ce web service permet de classer par temps ou par distance une liste de ressources positionnées autour d'un point. Il prend en entrée un point de départ (target) et une liste de ressources avec des positions géographiques et des priorités 1 et 2 (optionnelles). Les ressources sont des enregistrements stockés dans une base de données à la discrétion du client en fonction de ses besoins. La sélection de ces ressources dans la base de données est à la charge du client. Les ressources sont renvoyées triées d'abord par l'attribut priority1, puis par l'attribut priority2, puis par temps ou par distance (suivant le critère choisi). Les priorités sont optionnelles. La valeur par défaut des priorités est 0. Il s'appuie sur le graphe paramétré dont le nom a été spécifié dans l'interface d'administration de Geoconcept Web.

Disponibilité

Ce web service est disponible en permanence avec Geoconcept Web et un graphe.

Changement de version

Les versions précédente du web service sont conservées dans Geoconcept Web pour assurer la compatibilité avec les développements antérieurs. Il est recommandé d'utiliser la version la plus récente.

Changements avec la v4

- Ajout des paramètres "maxCost", "timeOut" et "computeOptions".
- Ajout dans la snapMethod "nodes" de l'accrochage aux noeuds les plus proches.

Changements avec la v3

- Ajout de la notion de noeud, plus rapide, pour s'accrocher aux noeuds du graphe plutôt qu'a des coordonnées géographiques. Ajout des éléments suivants : "locationNode", du paramètre "node" dans les ressources et de "nodes" dans les snapMethod.
- Ajout des paramètres "graphName", "profileId", "profileName", "avoidArea" et "configName".

Changements avec la v2

- Ajout du paramètre "snapMethod"
- Suppression des paramètres "targetX" et "targetY" typés double, remplacé par "location" typé string
- Suppression du paramètre "searchMethod", remplacé par "method"
- Suppression du paramètre "rejectFlags", remplacé par "exclusions"
- Suppression du paramètre "projection", remplacé par "srs"
- Le paramètre "distance" est renommé "distanceMeters " et son typage passe de int à double
- Le paramètre "time" est renommé "durationSeconds " et son typage passe de int à double

V4

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
location	Coordonnées du point de départ (ou d'arrivée) en SOAP, les points sont stockées dans les paramètres "x", et "y" eux même dans le paramètre "geographicPoint" en REST, les points sont séparés par le caractère ,	oui *	
locationNode	Noeud de départ (ou d'arrivée). Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
method	Itinéraire le plus court (distance), le plus rapide (time) ou à vol d'oiseau (flying)	oui	time
reverse	si *true*, la cible est considérée comme arrivée	oui	false
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère *;* (Exemple : Toll, Tunnel, Bridge)	oui	
resources	Liste des ressources, séparés par le caractère ";". Chaque ressources à la forme "id,x,y,node,priority1,priority2"	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris. Attention le caractère + peut être mal interprété par les navigateurs, dans ce cas, il faut le remplacer par *%2B*.	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction - nodes : Accrochage direct aux noeuds fournis par le paramètre locationNode ou, s'il n'est pas renseigné, au noeud les plus proche du paramètre location	oui	standard
graphName (déprécié)	Nom du graphe à utiliser Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileId (déprécié)	Identifiant du véhicule (enregistré dans les profils de véhicule)	oui	

paramètre	description	optionnel	défaut
	Ce paramètre est omis si le paramètre configName est utilisé.		
profileName (déprécié)	Profil du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
avoidArea	Zone de transit interdit au format WKT (POLYGON ou MULTIPOLYGON) dans la projection (paramètre srs) demandée Exemple en wgs84 : `POLYGON ((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689))` - `MULTIPOLYGON (((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689)), ((-1.558639 47.218144, -1.558639 47.218193, [...] -1.558639 47.218144)))` Attention les géométries WKT doivent être fermées.	oui	
configName	Nom de la configuration à utiliser (défini dans Geoconcept Web - Administration / Outils / Définitions des graphes) Il remplace l'usage de graphName, profileId et profileName	oui	
computeOptions	Liste des options pour le calcul, séparées par le caractère *;* - trafficPatterns : utilise les statistiques routières (il est nécessaire de renseigner le paramètre startDateTime et d'utiliser un graphe intégrant les informations de traffic patterns) - speedPattern (M18) : utilise d'une *speed pattern* tel que définis dans le fichier SmartRoutingVehicles.xml qui se trouve dans le dossier `*'<GEOCONCEPT_WEB_HOME>'\smartrouting\jee\smartrouting\conf* . Usage : "speedPattern:slow-speed" - length (M18) : longueur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "length:950" - width (M18) : largeur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "width:255" - height (M18) : hauteur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "height:360" - weight (M18) : poids maximum autorisé en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weight:18000" - axles (M18) : nombre maximum d'axe autorisé (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "axles:2" - weightPerAxle (M18) : poids maximum autorisé par axe en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weightPerAxle:9000" - snapSpeed : vitesse d'accrochage au graphe en kilomètre par heure. Usage : "snapSpeed:10"	oui	
maxCost	Coût maximum à ne pas dépasser dans le calcul -1 : pas de coût maximum à prendre en compte 0 : prendre la valeur par default défini dans la configuration de SmartRouting Server sinon : valeur en mètres si method=distance ou en secondes si method=time	oui	
timeOut	Time out pour le calcul (en millisecondes)	oui	

(*) Au moins l'un des deux paramètres location, et locationNode doit être renseigné.

(M18) Disponible à partir de la version M18 des graphes fournis par GEOCONCEPT SAS.

En sortie

Search Around (searchAroundWebResultsV4)

paramètre	type	min/max	description
location	double	0/1	Coordonnées du point de départ (ou d'arrivée) en SOAP, les points sont stockées dans les paramètres "x", et "y" eux même dans le paramètre "geographicPoint" en REST, les points sont séparés par le caractère ,
method	string	0/1	Méthode précisée en entrée (time, distance ou flying), time par défaut.
srs	string	0/1	Projection précisée en entrée.
exclusions	string	0/illimité	Liste des règles de restrictions précisées en entrée.
searchAroundResult	searchAroundWebResultV4 (ou array en JSON / JSON-P)	0/illimité	Liste des résultats pour chaque candidat.

Search Around élément (searchAroundWebResultV4)

paramètre	type	min/max	description
id	string	0/1	Identifiant du candidat
distanceMeters	double	1/1	Distance en mètre
durationSeconds	double	1/1	Temps en seconde

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/searchAroundService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:searchAroundV4>
         <!--Optional:-->
         <request>
            <location>
               <x>-1.553927</x>
               <y>47.218580</y>
            </location>
            <!--Optional:-->
            <locationNode></locationNode>
            <!--Optional:-->
            <method>time</method>
            <reverse></reverse>
            <!--Optional:-->
            <srs></srs>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
```

```
            <resources>
               <!--Zero or more repetitions:-->
               <resource>
                  <x>-1.593927</x>
                  <y>47.188580</y>
                  <!--Optional:-->
                  <id>1</id>
                  <!--Optional:-->
                  <node></node>
                  <!--Optional:-->
                  <priority1>1</priority1>
                  <!--Optional:-->
                  <priority2>2</priority2>
               </resource>
               <resource>
                  <x>-1.556927</x>
                  <y>47.219580</y>
                  <!--Optional:-->
                  <id>2</id>
                  <!--Optional:-->
                  <node></node>
                  <!--Optional:-->
                  <priority1>1</priority1>
                  <!--Optional:-->
                  <priority2>2</priority2>
               </resource>
            </resources>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <snapMethod></snapMethod>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <profileId></profileId>
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <avoidArea></avoidArea>
            <!--Optional:-->
            <configName></configName>
            <!--Optional:-->
            <computeOptions></computeOptions>
            <!--Optional:-->
            <maxCost></maxCost>
            <!--Optional:-->
            <timeOut></timeOut>
         </request>
      </sch:searchAroundV4>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:searchAroundV4Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <SearchAroundResult>
            <status>OK</status>
            <location>-1.553927,47.21858</location>
            <method>time</method>
            <srs/>
            <exclusions/>
```

```
            <searchAroundResult>
               <id>2</id>
               <distanceMeters>424.0</distanceMeters>
               <durationSeconds>100.0</durationSeconds>
            </searchAroundResult>
            <searchAroundResult>
               <id>1</id>
               <distanceMeters>6682.0</distanceMeters>
               <durationSeconds>942.0</durationSeconds>
            </searchAroundResult>
         </SearchAroundResult>
      </ns2:searchAroundV4Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/searchAround/v4.json?
method=time&location=-1.593927,47.218580&resources=1,-1.593927,47.188580,1,2;2,-1.556927,47.219580,1,1
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/searchAround/v4.json?
method=time&location=-1.593927,47.218580&resources=1,-1.593927,47.188580,1,2;2,-1.556927,47.219580,1,1&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/searchAround/v4.xml?
method=time&location=-1.593927,47.218580&resources=1,-1.593927,47.188580,1,2;2,-1.556927,47.219580,1,1
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message": null,
   "status": "OK",
   "location": "-1.593927,47.21858",
   "locationNode": null,
   "method": "time",
   "srs": null,
   "exclusions": [],
   "searchAroundResult":    [
            {
         "id": "2",
         "distanceMeters": 4474,
         "durationSeconds": 796
      },
            {
         "id": "1",
         "distanceMeters": 10453,
```

```
          "durationSeconds": 1114
      }
   ]
}
```

Format JSON-P

```
myCallback(
    {
       "message": null,
       "status": "OK",
       "location": "-1.593927,47.21858",
       "method": "time",
       "srs": null,
       "exclusions": [],
       "searchAroundResult":    [
                {
             "id": "2",
             "distanceMeters": 4474,
             "durationSeconds": 796
          },
                {
             "id": "1",
             "distanceMeters": 10453,
             "durationSeconds": 1114
          }
       ]
    }
);
```

Format XML

```
<searchAroundResponseV4>
   <status>OK</status>
   <location>-1.593927,47.21858</location>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distanceMeters>4474.0</distanceMeters>
      <durationSeconds>796.0</durationSeconds>
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distanceMeters>10453.0</distanceMeters>
      <durationSeconds>1114.0</durationSeconds>
   </searchAroundResult>
</searchAroundResponseV4>
```

Retours possibles

Cas d’une recherche de proximité trouvée (searchAroundResponse/status est OK)

```
<searchAroundResponseV4>
   <status>OK</status>
   <location>-1.593927,47.21858</location>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distanceMeters>4474.0</distanceMeters>
```

```
      <durationSeconds>796.0</durationSeconds>
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distanceMeters>10453.0</distanceMeters>
      <durationSeconds>1114.0</durationSeconds>
   </searchAroundResult>
</searchAroundResponseV4>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (searchAroundResponse/status est OK)

```
<searchAroundResponseV4>
   <status>OK</status>
   <location>-1.593927,47.21858</location>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
</searchAroundResponseV4>
```

Cas d'une erreur d'accrochage au graphe (searchAroundResponse/status est OK)

```
<searchAroundResponseV4>
   <status>OK</status>
   <location>-1.593927,47.21858</location>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
</searchAroundResponseV4>
```

Cas d'une absence du paramètre location (searchAroundResponse/status est ERROR)

```
<searchAroundResponseV4>
   <message>Location must be not null</message>
   <status>ERROR</status>
</searchAroundResponseV4>
```

Cas ou la paramètre location est incomplet ou avec un mauvais séparateur (searchAroundResponse/status est ERROR)

```
<searchAroundResponseV4>
   <message>Location point must have 2 components separated with a ,</message>
```

```
   <status>ERROR</status>
</searchAroundResponseV4>
```

Cas d'absence de ressource (searchAroundResponse/status est ERROR)

```
<searchAroundResponseV4>
    <message>resources not defined</message>
    <status>ERROR</status>
</searchAroundResponseV4>
```

Cas d'une ressource mal formatée (searchAroundResponse/status est ERROR)

```
<serviceResult>
   <message>ServiceException: not enough fields in candidate 1.-1.5939</message>
   <status>ERROR</status>
</serviceResult>
```

V3

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
location	Coordonnées du point de départ (ou d'arrivée) en SOAP, les points sont stockées dans les paramètres "x", et "y" eux même dans le paramètre "geographicPoint" en REST, les points sont séparés par le caractère ,	oui *	
locationNode	Noeud de départ (ou d'arrivée). Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
method	Itinéraire le plus court (distance), le plus rapide (time) ou à vol d'oiseau (flying)	oui	time
reverse	si *true*, la cible est considérée comme arrivée	oui	false
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère *;* (Exemple : "Toll", "Tunnel", "Bridge")	oui	
resources	Liste des ressources, séparés par le caractère ";". Chaque ressources à la forme "id,x,y,node,priority1,priority2"	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris. Attention le caractère + peut être mal interprété par les navigateurs, dans ce cas, il faut le remplacer par *%2B*.	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction - nodes : accrochage direct aux noeuds indiqués par locationNode et les paramètres *node* des ressources	oui	standard

paramètre	description	optionnel	défaut
graphName	Nom du graphe à utiliser Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileId	Identifiant du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileName	Profil du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
avoidArea	Zone de transit interdit au format WKT (POLYGON ou MULTIPOLYGON) dans la projection (paramètre srs) demandée Exemple en wgs84 : `POLYGON ((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689))` - `MULTIPOLYGON (((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689)), ((-1.558639 47.218144, -1.558639 47.218193, [...] -1.558639 47.218144)))` Attention les géométries WKT doivent être fermées.	oui	
configName	Nom de la configuration à utiliser (défini dans Geoconcept Web - Administration / Outils / Définitions des graphes) Il remplace l'usage de graphName, profileId et profileName	oui	

(*) Au moins l'un des deux paramètres location, et locationNode doit être renseigné.

En sortie

Search Around (searchAroundWebResultsV3)

paramètre	type	min/max	description
location	double	0/1	Coordonnées du point de départ (ou d'arrivée) en SOAP, les points sont stockées dans les paramètres "x", et "y" eux même dans le paramètre "geographicPoint" en REST, les points sont séparés par le caractère ,
method	string	0/1	Méthode précisée en entrée (time, distance ou flying), time par défaut.
srs	string	0/1	Projection précisée en entrée.
exclusions	string	0/illimité	Liste des règles de restrictions précisées en entrée.
searchAroundResult	searchAroundWebResultV3 (ou array en JSON / JSON-P)	0/illimité	Liste des résultats pour chaque candidat.

Search Around élément (searchAroundWebResultV3)

paramètre	type	min/max	description
id	string	0/1	Identifiant du candidat
distanceMeters	double	1/1	Distance en mètre
durationSeconds	double	1/1	Temps en seconde

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/searchAroundService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:searchAroundV3>
         <!--Optional:-->
         <request>
            <location>
               <x>-1.553927</x>
               <y>47.218580</y>
            </location>
            <!--Optional:-->
            <locationNode></locationNode>
            <!--Optional:-->
            <method>time</method>
            <reverse></reverse>
            <!--Optional:-->
            <srs></srs>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <resources>
               <!--Zero or more repetitions:-->
               <resource>
                  <x>-1.593927</x>
                  <y>47.188580</y>
                  <!--Optional:-->
                  <id>1</id>
                  <!--Optional:-->
                  <node></node>
                  <!--Optional:-->
                  <priority1>1</priority1>
                  <!--Optional:-->
                  <priority2>2</priority2>
               </resource>
               <resource>
                  <x>-1.556927</x>
                  <y>47.219580</y>
                  <!--Optional:-->
                  <id>2</id>
                  <!--Optional:-->
                  <node></node>
                  <!--Optional:-->
                  <priority1>1</priority1>
                  <!--Optional:-->
                  <priority2>2</priority2>
               </resource>
            </resources>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <snapMethod></snapMethod>
            <!--Optional:-->
            <graphName></graphName>
            <!--Optional:-->
            <profileId></profileId>
```

```
            <!--Optional:-->
            <profileName></profileName>
            <!--Optional:-->
            <avoidArea></avoidArea>
            <!--Optional:-->
            <configName></configName>
         </request>
      </sch:searchAroundV3>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:searchAroundV3Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <SearchAroundResult>
            <status>OK</status>
            <location>-1.553927,47.21858</location>
            <method>time</method>
            <srs/>
            <exclusions/>
            <searchAroundResult>
               <id>2</id>
               <distanceMeters>424.0</distanceMeters>
               <durationSeconds>147.0</durationSeconds>
            </searchAroundResult>
            <searchAroundResult>
               <id>1</id>
               <distanceMeters>6682.0</distanceMeters>
               <durationSeconds>1059.0</durationSeconds>
            </searchAroundResult>
         </SearchAroundResult>
      </ns2:searchAroundV3Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/searchAround/v3.json?
method=time&location=-1.593927,47.218580&resources=1,-1.593927,47.188580,1,2;2,-1.556927,47.219580,1,1
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/searchAround/v3.json?
method=time&location=-1.593927,47.218580&resources=1,-1.593927,47.188580,1,2;2,-1.556927,47.219580,1,1&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/searchAround/v3.xml?
method=time&location=-1.593927,47.218580&resources=1,-1.593927,47.188580,1,2;2,-1.556927,47.219580,1,1
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON


```
{
   "message": null,
   "status": "OK",
   "location": "-1.593927,47.21858",
   "locationNode": null,
   "method": "time",
   "srs": null,
   "exclusions": [],
   "searchAroundResult":    [
            {
         "id": "2",
         "distanceMeters": 4474,
         "durationSeconds": 796
      },
            {
         "id": "1",
         "distanceMeters": 10453,
         "durationSeconds": 1114
      }
   ]
}
```


Format JSON-P


```
myCallback(
    {
       "message": null,
       "status": "OK",
       "location": "-1.593927,47.21858",
       "method": "time",
       "srs": null,
       "exclusions": [],
       "searchAroundResult":    [
                {
             "id": "2",
             "distanceMeters": 4474,
             "durationSeconds": 796
          },
                {
             "id": "1",
             "distanceMeters": 10453,
             "durationSeconds": 1114
          }
       ]
    }
);
```


Format XML


```
<searchAroundResponseV3>
   <status>OK</status>
   <location>-1.593927,47.21858</location>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distanceMeters>4474.0</distanceMeters>
      <durationSeconds>796.0</durationSeconds>
```

```
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distanceMeters>10453.0</distanceMeters>
      <durationSeconds>1114.0</durationSeconds>
   </searchAroundResult>
</searchAroundResponseV3>
```

Retours possibles

Cas d'une recherche de proximité trouvée (searchAroundResponse/status est OK)

```
<searchAroundResponseV3>
   <status>OK</status>
   <location>-1.593927,47.21858</location>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distanceMeters>4474.0</distanceMeters>
      <durationSeconds>796.0</durationSeconds>
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distanceMeters>10453.0</distanceMeters>
      <durationSeconds>1114.0</durationSeconds>
   </searchAroundResult>
</searchAroundResponseV3>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (searchAroundResponse/status est OK)

```
<searchAroundResponseV3>
   <status>OK</status>
   <location>-1.593927,47.21858</location>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
</searchAroundResponseV3>
```

Cas d'une erreur d'accrochage au graphe (searchAroundResponse/status est OK)

```
<searchAroundResponseV3>
   <status>OK</status>
   <location>-1.593927,47.21858</location>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
   <searchAroundResult>
```

```
      <id>1</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
</searchAroundResponseV3>
```

Cas d'une absence du paramètre location (searchAroundResponse/status est ERROR)

```
<searchAroundResponseV3>
   <message>Location must be not null</message>
   <status>ERROR</status>
</searchAroundResponseV3>
```

Cas ou la paramètre location est incomplet ou avec un mauvais séparateur (searchAroundResponse/status est ERROR)

```
<searchAroundResponseV3>
   <message>Location point must have 2 components separated with a ,</message>
   <status>ERROR</status>
</searchAroundResponseV3>
```

Cas d'une absence de ressource (searchAroundResponse/status est OK)

```
<searchAroundResponseV3>
   <status>OK</status>
</searchAroundResponseV3>
```

Cas d'une ressource mal formatée (searchAroundResponse/status est ERROR)

```
<serviceResult>
   <message>ServiceException: not enough fields in candidate 1.-1.5939</message>
   <status>ERROR</status>
</serviceResult>
```

V2

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
location	Coordonnées du point de départ (ou d'arrivée) en SOAP, les points sont stockées dans les paramètres "x", et "y" eux même dans le paramètre "geographicPoint" en REST, les points sont séparés par le caractère ,	non	
method	Itinéraire le plus court (distance), le plus rapide (time) ou à vol d'oiseau (flying)	oui	time
reverse	si *true*, la cible est considérée comme arrivée	oui	false
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère *;* (Exemple : "Toll", "Tunnel", "Bridge")	oui	
resources	Liste des ressources, séparés par le caractère ";".	oui	

paramètre	description	optionnel	défaut
	Chaque ressources à la forme "id,x,y,priority1,priority2"		
startDateTime	Date et heure de départ (format ISO8601 : heure locale) exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction	oui	standard

En sortie

Search Around (searchAroundWebResultsV2)

paramètre	type	min/max	description
location	double	0/1	Coordonnées du point de départ (ou d'arrivée) en SOAP, les points sont stockées dans les paramètres "x", et "y" eux même dans le paramètre "geographicPoint" en REST, les points sont séparés par le caractère ,
method	string	0/1	Méthode précisée en entrée (time, distance ou flying), time par défaut.
srs	string	0/1	Projection précisée en entrée.
exclusions	string	0/illimité	Liste des règles de restrictions précisées en entrée.
searchAroundResult	searchAroundWebResultV2 (ou array en JSON / JSON-P)	0/illimité	Liste des résultats pour chaque candidat.

Search Around élément (searchAroundWebResultV2)

paramètre	type	min/max	description
id	string	0/1	Identifiant du candidat
distanceMeters	double	1/1	Distance en mètre
durationSeconds	double	1/1	Temps en seconde

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/searchAroundService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:searchAroundV2>
         <!--Optional:-->
         <request>
            <location>
```

```
               <x>-1.553927</x>
               <y>47.218580</y>
            </location>
            <!--Optional:-->
            <method>time</method>
            <reverse></reverse>
            <!--Optional:-->
            <srs></srs>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <resources>
               <!--Zero or more repetitions:-->
               <resource>
                  <x>-1.593927</x>
                  <y>47.188580</y>
                  <!--Optional:-->
                  <id>1</id>
                  <!--Optional:-->
                  <priority1>1</priority1>
                  <!--Optional:-->
                  <priority2>2</priority2>
               </resource>
               <resource>
               <x>-1.556927</x>
               <y>47.219580</y>
                  <!--Optional:-->
                  <id>2</id>
                  <!--Optional:-->
                  <priority1>1</priority1>
                  <!--Optional:-->
                  <priority2>2</priority2>
               </resource>
            </resources>
            <!--Optional:-->
            <startDateTime></startDateTime>
            <!--Optional:-->
            <snapMethod></snapMethod>
         </request>
      </sch:searchAroundV2>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:searchAroundV2Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <SearchAroundResult>
            <status>OK</status>
            <location>-1.553927,47.21858</location>
            <method>time</method>
            <srs/>
            <exclusions/>
            <searchAroundResult>
               <id>2</id>
               <distanceMeters>424.0</distanceMeters>
               <durationSeconds>147.0</durationSeconds>
            </searchAroundResult>
            <searchAroundResult>
```

```
               <id>1</id>
               <distanceMeters>6682.0</distanceMeters>
               <durationSeconds>1059.0</durationSeconds>
            </searchAroundResult>
         </SearchAroundResult>
      </ns2:searchAroundV2Response>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/searchAround/v2.json?
method=time&location=-1.593927,47.218580&resources=1,-1.593927,47.188580,1,2;2,-1.556927,47.219580,1,1
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/searchAround/v2.json?
method=time&location=-1.593927,47.218580&resources=1,-1.593927,47.188580,1,2;2,-1.556927,47.219580,1,1&callback=myCallback
```

Requête XML

```
http://<server>/<webapp>/api/lbs/searchAround/v2.xml?
method=time&location=-1.593927,47.218580&resources=1,-1.593927,47.188580,1,2;2,-1.556927,47.219580,1,1
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message": null,
   "status": "OK",
   "location": "-1.593927,47.21858",
   "method": "time",
   "srs": null,
   "exclusions": [],
   "searchAroundResult":    [
            {
         "id": "2",
         "distanceMeters": 3376,
         "durationSeconds": 735
      },
            {
         "id": "1",
         "distanceMeters": 10453,
         "durationSeconds": 1114
      }
   ]
}
```

Format JSON-P

```
myCallback(
    {
       "message": null,
       "status": "OK",
       "location": "-1.593927,47.21858",
       "method": "time",
       "srs": null,
       "exclusions": [],
       "searchAroundResult":    [
               {
            "id": "2",
            "distanceMeters": 3376,
            "durationSeconds": 735
         },
               {
            "id": "1",
            "distanceMeters": 10453,
            "durationSeconds": 1114
         }
       ]
    }
);
```

Format XML

```
<searchAroundResponse>
   <status>OK</status>
   <location>-1.593927,47.21858</location>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distanceMeters>3376.0</distanceMeters>
      <durationSeconds>735.0</durationSeconds>
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distanceMeters>10453.0</distanceMeters>
      <durationSeconds>1114.0</durationSeconds>
   </searchAroundResult>
</searchAroundResponse>
```

Retours possibles

Cas d’une recherche de proximité trouvée (searchAroundResponse/status est OK)

```
<searchAroundResponse>
   <status>OK</status>
   <location>-1.593927,47.21858</location>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distanceMeters>3376.0</distanceMeters>
      <durationSeconds>735.0</durationSeconds>
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distanceMeters>10453.0</distanceMeters>
      <durationSeconds>1114.0</durationSeconds>
   </searchAroundResult>
</searchAroundResponse>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (searchAroundResponse/status est OK)

```
<searchAroundResponse>
   <status>OK</status>
   <location>-1.593927,47.21858</location>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
</searchAroundResponse>
```

Cas d'une erreur d'accrochage au graphe (searchAroundResponse/status est OK)

```
<searchAroundResponse>
   <status>OK</status>
   <location>-1.593927,47.21858</location>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
</searchAroundResponse>
```

Cas d'une absence du paramètre location (searchAroundResponse/status est ERROR)

```
<searchAroundResponse>
   <message>Location must be not null</message>
   <status>ERROR</status>
</searchAroundResponse>
```

Cas ou la paramètre location est incomplet ou avec un mauvais séparateur (searchAroundResponse/status est ERROR)

```
<searchAroundResponse>
   <message>Location point must have 2 components separated with a ,</message>
   <status>ERROR</status>
</searchAroundResponse>
```

Cas d'une absence de ressource (searchAroundResponse/status est OK)

```
<searchAroundResponse>
   <status>OK</status>
</searchAroundResponse>
```

Cas d'une ressource mal formatée (searchAroundResponse/status est ERROR)

```
<serviceResult>
   <message>ServiceException: not enough fields in candidate 1.-1.5939</message>
   <status>ERROR</status>
</serviceResult>
```

V1

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
targetX	Coordonnées X ou longitude de la target	oui	
targetY	Coordonnées Y ou latitude de la target	oui	
searchMethod	Déprécié, remplacé par method	oui	time
method	Itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
reverse	si *true*, la target est considéré comme arrivée	oui	false
projection	Déprécié, remplacé par srs	oui	
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
rejectFlags	Déprécié, remplacé par exclusions	oui	
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère *;* (Exemple : "Toll", "Tunnel", "Bridge")	oui	
resources	Liste des ressources, séparés par le caractère ";". Chaque ressources à la forme "id,x,y,priority1,priority2"	oui	

En sortie

Search Around (searchAroundWebResults)

paramètre	type	min/max	description
targetX	double	1/1	Coordonnées X ou longitude de la target
targetY	double	1/1	Coordonnées Y ou latitude de la target
method	string	0/1	Méthode précisée en entrée
projection	string	0/1	Déprécié, remplacé par srs
srs	string	0/1	Projection précisée en entrée
exclusions	string	0/illimité	Liste des règles de restrictions précisées en entrée
searchAroundResult	array	0/illimité	Liste des réponses

Search Around élément (searchAroundWebResult)

paramètre	type	min/max	description
id	string	0/1	Identifiant du candidat
distance	int	1/1	Distance en mètre
time	int	1/1	Temps en seconde

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/searchAroundService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header/>
   <soapenv:Body>
      <sch:searchAround>
         <!--Optional:-->
         <request>
            <!--Optional:-->
            <target>
               <x>-1.553927</x>
               <y>47.218580</y>
            </target>
            <!--Optional:-->
            <searchMethod></searchMethod>
            <!--Optional:-->
            <method>time</method>
            <reverse></reverse>
            <!--Optional:-->
            <srs></srs>
            <!--Optional:-->
            <exclusions>
               <!--Zero or more repetitions:-->
               <exclusion></exclusion>
            </exclusions>
            <!--Optional:-->
            <rejectFlags>
               <!--Zero or more repetitions:-->
               <rejectFlag></rejectFlag>
            </rejectFlags>
            <!--Optional:-->
            <resources>
               <!--Zero or more repetitions:-->
               <resource>
               <x>-1.593927</x>
               <y>47.188580</y>
                  <!--Optional:-->
                  <id>1</id>
                  <!--Optional:-->
                  <priority1>1</priority1>
                  <!--Optional:-->
                  <priority2>2</priority2>
               </resource>
               <resource>
               <x>-1.556927</x>
               <y>47.219580</y>
                  <!--Optional:-->
                  <id>2</id>
                  <!--Optional:-->
                  <priority1>1</priority1>
                  <!--Optional:-->
                  <priority2>2</priority2>
               </resource>
            </resources>
         </request>
```

```
      </sch:searchAround>
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:searchAroundResponse xmlns:ns2="http://geoconcept.com/gc/schemas">
         <SearchAroundResult>
            <status>OK</status>
            <targetX>-1.553927</targetX>
            <targetY>47.21858</targetY>
            <method>time</method>
            <projection/>
            <srs/>
            <exclusions/>
            <searchAroundResult>
               <id>2</id>
               <distance>485</distance>
               <time>87</time>
            </searchAroundResult>
            <searchAroundResult>
               <id>1</id>
               <distance>6788</distance>
               <time>526</time>
            </searchAroundResult>
         </SearchAroundResult>
      </ns2:searchAroundResponse>
   </soap:Body>
</soap:Envelope>
```

REST

Requête

Requête JSON

```
http://<server>/<webapp>/api/lbs/searchAround.json?
method=time&targetX=-1.593927&targetY=47.218580&resources=1,-1.593927,47.188580,1,2;2,-1.556927,47.219580,1,1
```

Requête JSON-P

```
http://<server>/<webapp>/api/lbs/searchAround.json?
method=time&targetX=-1.593927&targetY=47.218580&resources=1,-1.593927,47.188580,1,2;2,-1.556927,47.219580,1,1&callback=myC
```

Requête XML

```
http://<server>/<webapp>/api/lbs/searchAround.xml?
method=time&targetX=-1.593927&targetY=47.218580&resources=1,-1.593927,47.188580,1,2;2,-1.556927,47.219580,1,1
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
   "message": null,
   "status": "OK",
   "targetX": -1.593927,
   "targetY": 47.21858,
   "method": "time",
   "projection": null,
   "srs": null,
   "exclusions": [],
   "searchAroundResult":    [
            {
         "id": "2",
         "distance": 3880,
         "time": 777
      },
            {
         "id": "1",
         "distance": 10448,
         "time": 1178
      }
   ]
}
```


Format JSON-P


```
myCallback(
        {
           "message": null,
           "status": "OK",
           "targetX": -1.593927,
           "targetY": 47.21858,
           "method": "time",
           "projection": null,
           "srs": null,
           "exclusions": [],
           "searchAroundResult":    [
                                 {
                        "id": "2",
                        "distance": 3880,
                        "time": 777
                  },
                                 {
                        "id": "1",
                        "distance": 10448,
                        "time": 1178
                  }
           ]
        }
);
```


Format XML


```
<searchAroundResponse>
   <status>OK</status>
   <targetX>-1.593927</targetX>
   <targetY>47.21858</targetY>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distance>3880</distance>
      <time>777</time>
```

```
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distance>10448</distance>
      <time>1178</time>
   </searchAroundResult>
</searchAroundResponse>
```

Retours possibles

Cas d'une recherche de proximité trouvée (searchAroundResponse/status est OK)

```
<searchAroundResponse>
   <status>OK</status>
   <targetX>-1.593927</targetX>
   <targetY>47.21858</targetY>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distance>3880</distance>
      <time>777</time>
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distance>10448</distance>
      <time>1178</time>
   </searchAroundResult>
</searchAroundResponse>
```

Cas d'un problème avec le graphe : fichier absent, mauvais chemin, etc… (searchAroundResponse/status est OK)

```
<searchAroundResponse>
   <status>OK</status>
   <targetX>-1.593927</targetX>
   <targetY>47.21858</targetY>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
</searchAroundResponse>
```

Cas d'une erreur d'accrochage au graphe (searchAroundResponse/status est OK)

```
<searchAroundResponse>
   <status>OK</status>
   <targetX>-51.593927</targetX>
   <targetY>47.21858</targetY>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distance>-1</distance>
```

```
      <time>-1</time>
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
</searchAroundResponse>
```

Cas d'un oubli de spécification du point target (searchAroundResponse/status est OK)

```
<searchAroundResponse>
   <status>OK</status>
   <targetX>0.0</targetX>
   <targetY>0.0</targetY>
   <method>time</method>
   <searchAroundResult>
      <id>2</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
   <searchAroundResult>
      <id>1</id>
      <distance>-1</distance>
      <time>-1</time>
   </searchAroundResult>
</searchAroundResponse>
```

Cas d'une absence de ressource (searchAroundResponse/status est OK)

```
<searchAroundResponse>
   <status>OK</status>
   <targetX>0.0</targetX>
   <targetY>0.0</targetY>
</searchAroundResponse>
```

Cas d'une ressource mal formatée (searchAroundResponse/status est ERROR)

```
<serviceResult>
   <message>ServiceException: not enough fields in candidate 2,-1.556927,</message>
   <status>ERROR</status>
</serviceResult>
```

FAQ

1. Est-il possible de prioriser le temps de parcours ou la distance ?

   Oui, en changeant la méthode distance, time ou flying = à vol d'oiseau (avec une vitesse de 30 km/h.

2. Peut-on utiliser des alias à défaut des noms de fichiers .siti pour appeler une datasource ?

   Oui, cf. détails dans la FAQ du Web Service du géocodage inverse.

3. Comment utiliser les statistiques routières ?

   Vérifier que le graphe contient bien ces statistiques et utiliser les deux paramètres suivants : **`computeOptions`** avec la valeur *trafficPatterns* et **`startDateTime`** pour préciser la date/heure de départ.

4. Comment s'organise le classement des adresses retournées, par rapport à la distance, au temps, aux priorités ?

   Le classement est fait par priorité 1 croissante, puis par priorité 2 croissante, puis par temps ou distance ou distance à vol d'oiseau croissant.

5. Quel format doit avoir la priorité et est-il possible de définir un ordre de classement ? (croissant/décroissant)

   La priorité est un entier, actuellement le résultat est toujours par ordre croissant. Pour obtenir l'ordre inverse, il faut mettre n - priorité dans l'attribut.

6. Si une seule adresse de la liste a une priorité à 0, la priorité est elle prise en compte pour une des adresses de la liste ?

   Actuellement, la priorité 0 n'est pas traitée de manière spéciale. Il faut donc mettre toutes les priorités à 0 si on veut ignorer le critère.

7. Comment faire un calcul d'itinéraire sans péage?

   Si la contrainte *Toll* a bien été incluse dans le graphe, placer une exclusion dans **`exclusions`** :

```
<sch:searchAroundV2>
 <!--Optional:-->
 <request>
    <location>
       <x>-1.553927</x>
       <y>47.218580</y>
    </location>
    <!--Optional:-->
    <method>time</method>
    <reverse></reverse>
    <!--Optional:-->
    <srs></srs>
    <!--Optional:-->
    <exclusions>
       <!--Zero or more repetitions:-->
       <exclusion>Toll</exclusion>
    </exclusions>
    <!--Optional:-->
    <resources>
       <!--Zero or more repetitions:-->
       <resource>
          <x>-1.593927</x>
          <y>47.188580</y>
          <!--Optional:-->
          <id>1</id>
          <!--Optional:-->
          <priority1>1</priority1>
          <!--Optional:-->
          <priority2>2</priority2>
       </resource>
    </resources>
    <!--Optional:-->
    <startDateTime></startDateTime>
 </request>
</sch:searchAroundV2>
```

8. Qu'est-ce que les Speed Patterns ? Comment les utiliser ?

Afin de proposer des temps de trajets au plus près des conditions de circulation, les graphes fournis par GEOCONCEPT SAS intègrent, à partir de la version M18, pour les voitures et pour les camions 5 profils de vitesses (Speed Patterns) pour tenir compte des différents niveaux de congestion dans une journée :

- standard *normal-speed* correspond à la vitesse d'une heure moyennement chargée (11h)
- nuit *fast-speed* correspond à un trafic très fluide, observé le plus souvent la nuit (3h)
- chargée *slow-speed* correspond aux heures de trafic denses (8h)
- heure de pointe *very-slow-speed* correspond aux heures de trafic denses dans les grandes agglomérations, plus lent que le précédent (8h)
- défaut *defaut* correspond aux vitesses moyennées sur une journée entière

Pour les utiliser il faut passer, lors de l'appel au web service, le paramètre **`computeOptions`** avec l'option *speedPattern* et préciser la valeur de la Speed Pattern demandée sans omettre un profil de véhicule
Exemple :

```
&computeOptions=speedPattern:fast-speed&profileId=1
```

9. Comment utiliser les attributs poids lourds ?

   Il faut que le graphe intègre les attributs de poids lourds (en standard dans les graphes fournis par GEOCONCEPT SAS à partir de la version M18) et soit calculer un itinéraire en utilisant un profil de véhicule utilisant les restrictions (cf. le catalogue de véhicules, éditable, définit dans le fichier SmartRoutingVehicles.xml, dans le dossier `'<*GEOCONCEPT_WEB_HOME*>"\*smartrouting*\jee\smartrouting\conf\* ), soit surcharger l'appel au web service en utilisant le paramètre **`computeOptions`** avec les options *length*, *width*, *height*, *weight*, *axles* et/ou *weightPerAxle*.
   Exemple pour un trajet avec un véhicule de 4,5 mètres de hauteur :

   ```
   &computeOptions=height:450
   ```

Search Along

Cette page n'est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

Le web service de Search Along permet d'identifier les meilleurs candidats pour ajouter une étape à proximité d'un trajet pré-existant.

L'algorithme explore les solutions pour retourner parmi les candidats celui qui à le meilleur score en fonction des critères demandées

Disponibilité

Ce web service est disponible en permanence avec Geoconcept Web et un graphe.

Paramètres / propriétés

En entrée

paramètre	description	optionnel	défaut
routes array of routes/route (inputRoute)	Tableau des trajets en entrée. Trajets pré-existant dans lesquels les candidats doivent être insérés.	oui *	
routeNodes array of routeNodes/routeNode (inputRouteNode)	Tableau des trajets en entrée basés sur les noeuds du graphe (calcul plus rapide). Attention : un noeud physique n'a pas le même ID dans un autre graphe.	oui *	
resources	Tableau des candidats	oui	
method	Itinéraire le plus court (distance) ou le plus rapide (time)	oui	time
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère *;* (Exemple : Toll, Tunnel, Bridge)	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction - nodes : Accrochage direct aux noeuds fournis par le paramètre locationNode ou, s'il n'est pas renseigné, au noeud les plus proche du paramètre location	oui	standard
avoidArea	Zone de transit interdit au format WKT (POLYGON ou MULTIPOLYGON) dans la projection (paramètre srs) demandée Exemple en wgs84 : `POLYGON ((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689))` - `MULTIPOLYGON (((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689)), ((-1.558639 47.218144, -1.558639 47.218193, [...] -1.558639 47.218144)))` Attention les géométries WKT doivent être fermées.	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris. Attention le caractère + peut être mal interprété par les navigateurs, dans ce cas, il faut le remplacer par *%2B*.	oui	
graphName (déprécié)	Nom du graphe à utiliser Ce paramètre est omis si le paramètre configName est utilisé.	oui	
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
profileId (déprécié)	Identifiant du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileName (déprécié)	Profil du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
configName	Nom de la configuration à utiliser (défini dans Geoconcept Web - Administration / Outils / Définitions des graphes) Il remplace l'usage de graphName, profileId et profileName	oui	
computeOptions	Liste des options pour le calcul, séparées par le caractère *;* - trafficPatterns : utilise les statistiques routières (il est nécessaire de renseigner le paramètre startDateTime et d'utiliser un graphe intégrant les informations de traffic patterns)	oui	

paramètre	description	optionnel	défaut
	- speedPattern (M18) : utilise d'une *speed pattern* tel que définis dans le fichier SmartRoutingVehicles.xml qui se trouve dans le dossier `'<*GEOCONCEPT_WEB_HOME*>'*smartrouting**jee**smartrouting**conf*\ . Usage : "speedPattern:slow-speed" - length (M18) : longueur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "length:950" - width (M18) : largeur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "width:255" - height (M18) : hauteur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "height:360" - weight (M18) : poids maximum autorisé en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weight:18000" - axles (M18) : nombre maximum d'axe autorisé (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "axles:2" - weightPerAxle (M18) : poids maximum autorisé par axe en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weightPerAxle:9000" - snapSpeed : vitesse d'accrochage au graphe en kilomètre par heure. Usage : "snapSpeed:10"		
maxDetourDurationSeconds	Filtre : détour maximum autorisé (en secondes)	oui	
maxDetourDistanceMeters	Filtre : détour maximum autorisé (en mètres)	oui	
maxDetourOriginDurationSeconds	Filtre : durée maximale autorisée entre le point de départ du trajet et le candidat (en secondes)	oui	
maxDetourOriginDistanceMeters	Filtre : détour maximum autorisé entre le candidat et le point d'arrivée du trajet (en mètres)	oui	
maxDetourDestinationDurationSeconds	Filtre : durée maximale autorisée entre le candidat et le point de d'arrivée du trajet (en secondes)	oui	
maxDetourDestinationDistanceMeters	Filtre : détour maximum autorisé depuis le point d'arrivée (en mètres)	oui	
timeOut	Time out pour le calcul (en millisecondes)	oui	

(*) Au moins l'un des deux paramètres routes, et routeNodes doit être renseigné.

(M18) Disponible à partir de la version M18 des graphes fournis par GEOCONCEPT SAS.

Trajets en entrée (inputRoute)

paramètre	description	optionnel	défaut
id	Identifiant du trajet	Non	
departurePoint (geographicPoint)	Coordonnées du point de départ du trajet	Non	
arrivalPoint (geographicPoint)	Coordonnées du point d'arrivée du trajet	Non	

Coordonnées (geographicPoint)

paramètre	description	optionnel	défaut
x	Première coordonnée ou longitude	Non	
y	Deuxième coordonnée ou latitude	Non	

Trajets en entrée basé sur les noeuds du graphe (inputRouteNode)

paramètre	description	optionnel	défaut
id	Identifiant du trajet	Non	
departureNode (geographicPoint)	Noeud de départ du trajet	Non	
arrivalNode (geographicPoint)	Noeud d'arrivée du trajet	Non	

Candidats (searchAlongResource)

paramètre	description	optionnel	défaut
id	Identifiant du candidat	Non	
x	Première coordonnée ou longitude	Non	
y	Deuxième coordonnée ou latitude	Non	
node	Noeud du candidat	Oui	
priority1	Priorité 1	Oui	0
priority2	Priorité 2	Oui	0

En sortie

paramètre	type	min/max	description
routes	array of routes/route (searchAlongRouteResult)	0/illimité	Liste des trajets calculés.

Liste des trajets calculés (searchAlongRouteResult)

paramètre	type	min/max	description
id	string	0/1	Identifiant du trajet
directDistanceMeters	double	1/1	Distance totale sans détour (en mètres)
directDurationSeconds	double	1/1	Durée totale sans détour (en secondes)
resources	(searchAlongResourceResult)	0/illimité	Liste de candidats

Candidats (searchAlongResourceResult)

paramètre	type	min/max	description
id	string	0/1	Identifiant du candidat
totalDistanceMeters	double	1/1	Distance totale avec détour (en mètres)
totalDurationSeconds	double	1/1	Durée totale avec détour (en secondes)
detourDistanceMeters	double	1/1	Distance du détour (en mètres)
detourDurationSeconds	double	1/1	Durée du détour (en secondes)
detourOriginDistanceMeters	double	1/1	Distance du détour depuis le point de départ du trajet jusqu'au candidat (en mètres)
detourOriginDurationSeconds	double	1/1	Durée du détour depuis le point de départ du trajet jusqu'au candidat (en secondes)
detourDestinationDistanceMeters	double	1/1	Distance du détour depuis le candidat jusqu'au point d'arrivée (en mètres)

paramètre	type	min/max	description
detourDestinationDurationSeconds	Double	1/1	Durée du détour depuis le candidat jusqu'au point d'arrivée (en secondes)

SOAP

WSDL

http://*<server>*/*<webapp>*/api/ws/searchAlongService?wsdl

Requête

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:sch="http://
geoconcept.com/gc/schemas">
   <soapenv:Header />
   <soapenv:Body>
      <sch:searchAlongV1>
         <!--Optional:-->
         <request>
            <routes>
               <route>
                  <id>dep</id>
                  <departurePoint>
                     <x>-1.553927</x>
                     <y>47.21858</y>
                  </departurePoint>
                  <arrivalPoint>
                     <x>-1.593927</x>
                     <y>47.18858</y>
                  </arrivalPoint>
               </route>
            </routes>
            <resources>
               <resource>
                  <id>res1</id>
                  <node />
                  <priority1>1</priority1>
                  <priority2>2</priority2>
                  <x>-1.511092</x>
                  <y>47.208355</y>
               </resource>
               <resource>
                  <id>res2</id>
                  <node />
                  <priority1>1</priority1>
                  <priority2>1</priority2>
                  <x>-1.549524</x>
                  <y>47.195484</y>
               </resource>
            </resources>
            <srs>epsg:4326</srs>
            <maxDetourDestinationDistanceMeters>-1</maxDetourDestinationDistanceMeters>
            <maxDetourDestinationDurationSeconds>-1</maxDetourDestinationDurationSeconds>
            <maxDetourDistanceMeters>-1</maxDetourDistanceMeters>
            <maxDetourDurationSeconds>-1</maxDetourDurationSeconds>
            <maxDetourOriginDistanceMeters>-1</maxDetourOriginDistanceMeters>
            <maxDetourOriginDurationSeconds>-1</maxDetourOriginDurationSeconds>
            <method>time</method>
                        <timeOut></timeOut>
         </request>
      </sch:searchAlongV1>
```

```
   </soapenv:Body>
</soapenv:Envelope>
```

Réponse

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:searchAlongV1Response xmlns:ns2="http://geoconcept.com/gc/schemas">
         <SearchAlongResult>
            <status>OK</status>
            <routes>
               <route>
                  <id>dep</id>
                  <directDistanceMeters>6850.71</directDistanceMeters>
                  <directDurationSeconds>973.14</directDurationSeconds>
                  <resources>
                     <id>res2</id>
                     <totalDistanceMeters>7808.36</totalDistanceMeters>
                     <totalDurationSeconds>1197.28</totalDurationSeconds>
                     <detourDistanceMeters>957.65</detourDistanceMeters>
                     <detourDurationSeconds>224.14</detourDurationSeconds>
                     <detourOriginDistanceMeters>3684.98</detourOriginDistanceMeters>
                     <detourOriginDurationSeconds>752.34</detourOriginDurationSeconds>
                     <detourDestinationDistanceMeters>4123.38</detourDestinationDistanceMeters>
                     <detourDestinationDurationSeconds>444.94</detourDestinationDurationSeconds>
                  </resources>
                  <resources>
                     <id>res1</id>
                     <totalDistanceMeters>13714.6</totalDistanceMeters>
                     <totalDurationSeconds>2261.52</totalDurationSeconds>
                     <detourDistanceMeters>6863.89</detourDistanceMeters>
                     <detourDurationSeconds>1288.38</detourDurationSeconds>
                     <detourOriginDistanceMeters>5731.27</detourOriginDistanceMeters>
                     <detourOriginDurationSeconds>1156.94</detourOriginDurationSeconds>
                     <detourDestinationDistanceMeters>7983.33</detourDestinationDistanceMeters>
                     <detourDestinationDurationSeconds>1104.58</detourDestinationDurationSeconds>
                  </resources>
               </route>
            </routes>
            <computationTime>118.72</computationTime>
         </SearchAlongResult>
      </ns2:searchAlongV1Response>
   </soap:Body>
</soap:Envelope>
```

REST (POST JSON)

Requête

Requête

```
http://<server>/<webapp>/api/lbs/searchAlong.json
```

Data (JSON)

```
{
   "routes":[
      {
         "id":"dep",
         "departurePoint":{
```

```
            "x":-1.553927,
            "y":47.218580
         },
         "arrivalPoint":{
            "x":-1.593927,
            "y":47.188580
         }
      }
   ],
   "resources":[
      {
         "id":"res1",
         "node":"",
         "priority1":1,
         "priority2":2,
         "x":-1.511092,
         "y":47.208354
      }
   ],
   "resources":[
      {
         "id":"res2",
         "node":"",
         "priority1":1,
         "priority2":1,
         "x":-1.549524,
         "y":47.195483
      }
   ],
   "method":"time",
   "srs":"epsg:4326",
   "maxDetourDurationSeconds":-1,
   "maxDetourDistanceMeters":-1,
   "maxDetourOriginDurationSeconds":-1,
   "maxDetourOriginDistanceMeters":-1,
   "maxDetourDestinationDurationSeconds":-1,
   "maxDetourDestinationDistanceMeters":-1
}
```

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
    "message": null,
    "status": "OK",
    "routes": [
        {
            "id": "dep",
            "directDistanceMeters": 6850.71,
            "directDurationSeconds": 973.14,
            "resources": [
                {
                    "id": "res2",
                    "totalDistanceMeters": 7808.36,
                    "totalDurationSeconds": 1197.28,
                    "detourDistanceMeters": 957.65,
                    "detourDurationSeconds": 224.14,
                    "detourOriginDistanceMeters": 3684.98,
                    "detourOriginDurationSeconds": 752.34,
```

```
                    "detourDestinationDistanceMeters": 4123.38,
                    "detourDestinationDurationSeconds": 444.94
                }
            ]
        }
    ]
}
```

REST (POST XML)

Requête

Requête

```
http://<server>/<webapp>/api/lbs/searchAlong.xml
```

Data (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<searchAlongRequest>
   <routes>
      <route>
         <id>dep</id>
         <departurePoint>
            <x>-1.553927</x>
            <y>47.21858</y>
         </departurePoint>
         <arrivalPoint>
            <x>-1.593927</x>
            <y>47.18858</y>
         </arrivalPoint>
      </route>
   </routes>
   <resources>
      <resource>
         <id>res1</id>
         <node />
         <priority1>1</priority1>
         <priority2>2</priority2>
         <x>-1.511092</x>
         <y>47.208355</y>
      </resource>
      <resource>
         <id>res2</id>
         <node />
         <priority1>1</priority1>
         <priority2>1</priority2>
         <x>-1.549524</x>
         <y>47.195484</y>
      </resource>
   </resources>
   <srs>epsg:4326</srs>
   <maxDetourDestinationDistanceMeters>-1</maxDetourDestinationDistanceMeters>
   <maxDetourDestinationDurationSeconds>-1</maxDetourDestinationDurationSeconds>
   <maxDetourDistanceMeters>-1</maxDetourDistanceMeters>
   <maxDetourDurationSeconds>-1</maxDetourDurationSeconds>
   <maxDetourOriginDistanceMeters>-1</maxDetourOriginDistanceMeters>
   <maxDetourOriginDurationSeconds>-1</maxDetourOriginDurationSeconds>
   <method>time</method>
</searchAlongRequest>
```

Réponse

La réponse est toujours encodée en UTF-8.

Format XML

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<searchAlongResult>
    <status>OK</status>
    <routes>
        <route>
            <id>dep</id>
            <directDistanceMeters>6850.71</directDistanceMeters>
            <directDurationSeconds>973.14</directDurationSeconds>
            <resources>
                <id>res2</id>
                <totalDistanceMeters>7808.36</totalDistanceMeters>
                <totalDurationSeconds>1197.28</totalDurationSeconds>
                <detourDistanceMeters>957.65</detourDistanceMeters>
                <detourDurationSeconds>224.14</detourDurationSeconds>
                <detourOriginDistanceMeters>3684.98</detourOriginDistanceMeters>
                <detourOriginDurationSeconds>752.34</detourOriginDurationSeconds>
                <detourDestinationDistanceMeters>4123.38</detourDestinationDistanceMeters>
                <detourDestinationDurationSeconds>444.94</detourDestinationDurationSeconds>
            </resources>
            <resources>
                <id>res1</id>
                <totalDistanceMeters>13714.6</totalDistanceMeters>
                <totalDurationSeconds>2261.52</totalDurationSeconds>
                <detourDistanceMeters>6863.89</detourDistanceMeters>
                <detourDurationSeconds>1288.38</detourDurationSeconds>
                <detourOriginDistanceMeters>5731.27</detourOriginDistanceMeters>
                <detourOriginDurationSeconds>1156.94</detourOriginDurationSeconds>
                <detourDestinationDistanceMeters>7983.33</detourDestinationDistanceMeters>
                <detourDestinationDurationSeconds>1104.58</detourDestinationDurationSeconds>
            </resources>
        </route>
    </routes>
</searchAlongResult>
```

Retours possibles

Cas d’une recherche de proximité trouvée (searchAlongResult/status est OK)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<searchAlongResult>
    <status>OK</status>
    <routes>
        <route>
            <id>dep</id>
            <directDistanceMeters>6850.71</directDistanceMeters>
            <directDurationSeconds>973.14</directDurationSeconds>
            <resources>
                <id>res2</id>
                <totalDistanceMeters>7808.36</totalDistanceMeters>
                <totalDurationSeconds>1197.28</totalDurationSeconds>
                <detourDistanceMeters>957.65</detourDistanceMeters>
                <detourDurationSeconds>224.14</detourDurationSeconds>
                <detourOriginDistanceMeters>3684.98</detourOriginDistanceMeters>
                <detourOriginDurationSeconds>752.34</detourOriginDurationSeconds>
                <detourDestinationDistanceMeters>4123.38</detourDestinationDistanceMeters>
```

```
                <detourDestinationDurationSeconds>444.94</detourDestinationDurationSeconds>
            </resources>
            <resources>
                <id>res1</id>
                <totalDistanceMeters>13714.6</totalDistanceMeters>
                <totalDurationSeconds>2261.52</totalDurationSeconds>
                <detourDistanceMeters>6863.89</detourDistanceMeters>
                <detourDurationSeconds>1288.38</detourDurationSeconds>
                <detourOriginDistanceMeters>5731.27</detourOriginDistanceMeters>
                <detourOriginDurationSeconds>1156.94</detourOriginDurationSeconds>
                <detourDestinationDistanceMeters>7983.33</detourDestinationDistanceMeters>
                <detourDestinationDurationSeconds>1104.58</detourDestinationDurationSeconds>
            </resources>
        </route>
    </routes>
</searchAlongResult>
```

Cas d'un SRS incorrect

```
{"message":"NullPointerException: null","status":"ERROR"}
```

Cas d'une absence de ressource (searchAroundResponse/status est OK)

```
{"message":"Input resources list must not be null and empty!","status":"ERROR","routes":[]}
```

FAQ

1. Est-il possible de prioriser le temps de parcours ou la distance ?

   Oui, en changeant la méthode **`method`** distance ou time

2. Peut-on utiliser des alias à défaut des noms de fichiers .siti pour appeler une datasource ?

   Oui, cf. détails dans la FAQ du Web Service du géocodage inverse.

3. Comment utiliser les statistiques routières ?

   Vérifier que le graphe contient bien ces statistiques et utiliser les deux paramètres suivants : **`computeOptions`** avec la valeur *trafficPatterns* et **`startDateTime`** pour préciser la date/heure de départ.

4. Comment s'organise le classement des adresses retournées, par rapport à la distance, au temps, aux priorités ?

   Le classement est fait par priorité 1 croissante, puis par priorité 2 croissante, puis par temps ou distance ou distance à vol d'oiseau croissant.

5. Quel format doit avoir la priorité et est-il possible de définir un ordre de classement ? (croissant/décroissant)

   La priorité est un entier, actuellement le résultat est toujours par ordre croissant. Pour obtenir l'ordre inverse, il faut mettre n - priorité dans l'attribut.

6. Si une seule adresse de la liste a une priorité à 0, la priorité est elle prise en compte pour une des adresses de la liste ?

   Actuellement, la priorité 0 n'est pas traitée de manière spéciale. Il faut donc mettre toutes les priorités à 0 si on veut ignorer le critère.

7. Comment faire un calcul d'itinéraire sans péage?

   Si la contrainte *Toll* a bien été incluse dans le graphe, placer une exclusion dans **`exclusions`**

8. Qu'est-ce que les Speed Patterns ? Comment les utiliser ?

   Afin de proposer des temps de trajets au plus près des conditions de circulation, les graphes fournis par GEOCONCEPT SAS intègrent, à partir de la version M18, pour les voitures et pour les camions 5 profils de vitesses (Speed Patterns) pour tenir compte des différents niveaux de congestion dans une journée :

   - standard *normal-speed* correspond à la vitesse d'une heure moyennement chargée (11h)
   - nuit *fast-speed* correspond à un trafic très fluide, observé le plus souvent la nuit (3h)
   - chargée *slow-speed* correspond aux heures de trafic denses (8h)
   - heure de pointe *very-slow-speed* correspond aux heures de trafic denses dans les grandes agglomérations, plus lent que le précédent (8h)
   - défaut *defaut* correspond aux vitesses moyennées sur une journée entière

Pour les utiliser il faut passer, lors de l'appel au web service, le paramètre **`computeOptions`** avec l'option *speedPattern* et préciser la valeur de la Speed Pattern demandée sans omettre un profil de véhicule
Exemple :

+

```
&computeOptions=speedPattern:fast-speed&profileId=1
```

Comment utiliser les attributs poids lourds ?

Il faut que le graphe intègre les attributs de poids lourds (en standard dans les graphes fournis par GEOCONCEPT SAS à partir de la version M18) et soit calculer un itinéraire en utilisant un profil de véhicule utilisant les restrictions (cf. le catalogue de véhicules, éditable, définit dans le fichier SmartRoutingVehicles.xml, dans le dossier `*'<GEOCONCEPT_WEB_HOME>'*\*smartrouting*\*jee*\*smartrouting*\*conf*\ ), soit surcharger l'appel au web service en utilisant le paramètre **`computeOptions`** avec les options *length*, *width*, *height*, *weight*, *axles* et/ou *weightPerAxle*.
Exemple pour un trajet avec un véhicule de 4,5 mètres de hauteur :

```
&computeOptions=height:450
```

Pickup and Delivery

Cette page n'est plus maintenue depuis la version 2022 de Geoconcept Web. Pour accéder à la documentation à jour de ce web service, veuillez suivre ce lien [https://mygeoconcept.com/doc/geoapi/docs/en/geoptimization-api-book/geoptimization-api-intro.html].

Principe

Le web service pickup and delivery peut être utilisé pour lister les meilleures solutions de ramassage/collecte et dépose/livraison que ce soit pour du transport de personnes ou du transport de biens.

Le web service prend en entrée deux types de données :

- Les trajets existants
- Points de ramassage/collecte et points de dépose/livraison éventuels

L'algorithme explore les solutions pour trouver un couple comprenant un point de ramassage et un point de dépose dans les trajets existants. Il retourne, pour chaque trajet, les meilleurs détours possibles parmi les candidats (points de ramassage et de dépose).

Dans le cas où un seul candidat (ramassage ou dépose) est disponible, le web service essaye d'insérer une étape dans les trajets existants au lieu de deux.

Disponibilité

Ce web service est une option de Geoconcept Web, veuiller nous contacter pour connaître les modalités d'acquisition.

Il nécessite l'usage d'un graphe accéléré.

Paramètres / propriétés

En entrée

paramètre	type	optionnel	description
routes	array of routes/route (pickupDeliveryRouteInput)	Non	Tableau des trajets en entrée. Trajets pré-existant dans lesquels les candidats doivent être insérés.
pickupPoints	array of pickupPoints/pickupPoint (wayPoint)	Oui	Tableau des candidats de ramassage à inserer dans les trajets.
deliveryPoints	array of deliveryPoints/deliveryPoint (wayPoint)	Oui	Tableau des possible candidats de dépose à inserer dans les trajets.
constraints	(pickupDeliveryConstraints)	Oui	Définition des contraintes pour filtrer

paramètre	type	optionnel	description
			les trajets à utiliser.
sortOptions	(pickupDeliverySortingOptions)	Oui	Définition du tri des résultats.
srs	projection (code EPSG comme epsg:4326 ou wgs84)	oui	
graphName (déprécié)	Nom du graphe à utiliser Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileId (déprécié)	Identifiant du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
profileName (déprécié)	Profil du véhicule (enregistré dans les profils de véhicule) Ce paramètre est omis si le paramètre configName est utilisé.	oui	
configName	Nom de la configuration à utiliser (défini dans Geoconcept Web - Administration / Outils / Définitions des graphes) Il remplace l'usage de graphName, profileId et profileName	oui	
snapMethod	Méthode d'accrochage au graphe - standard : au tronçon connectable le plus proche - extended : via les tronçons restreints (piétons…) - nearest : uniquement au tronçon le plus proche - unrestricted : sans règles de restriction - nodes : Accrochage direct aux noeuds fournis par les paramètres originNode, destinationNode et waypointNodes ou, si ces premiers ne sont pas renseignés, aux noeuds les plus proches des paramètres origin, destination et waypoint	oui	standard
exclusions	Liste des règles de restrictions à utiliser, séparés par le caractère *;* (Exemple : Toll, Tunnel, Bridge)	oui	
startDateTime	Date et heure de départ (format ISO8601 : heure locale) Exemple : 2014-01-21T09:00:00.000+01:00 (ou 2014-01-21T09:00:00.000%2B01:00) pour un départ le 21 janvier 2014, à 9h à Paris. Attention le caractère + peut être mal interprété par les navigateurs, dans ce cas, il faut le remplacer par *%2B*.	oui	
avoidArea	Zone de transit interdit au format WKT (POLYGON ou MULTIPOLYGON) dans la projection (paramètre srs) demandée Exemple en wgs84 : `POLYGON ((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689))` - `MULTIPOLYGON (((-1.556892 47.21689, -1.556892 47.216904, [...] -1.556892 47.21689)), ((-1.558639 47.218144, -1.558639 47.218193, [...] -1.558639 47.218144)))` Attention les géométries WKT doivent être fermées.	oui	
size	(pickupDeliverySizes)	Non	Nombre de trajets et de candidats à retourner.
dataVersionHash	string	Oui	Identifiant de graphe (non utilisé)

paramètre	type	optionnel	description
computeOptions	Liste des options pour le calcul, séparées par le caractère *;* - trafficPatterns : utilise les statistiques routières (il est nécessaire de renseigner le paramètre startDateTime et d'utiliser un graphe intégrant les informations de traffic patterns) - speedPattern (M18) : utilise d'une *speed pattern* tel que définis dans le fichier SmartRoutingVehicles.xml qui se trouve dans le dossier *`"<GEOCONCEPT_WEB_HOME>"\smartrouting\jee\smartrouting\conf* . Usage : "speedPattern:slow-speed" - length (M18) : longueur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "length:950" - width (M18) : largeur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "width:255" - height (M18) : hauteur maximale autorisée en centimètre (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "height:360" - weight (M18) : poids maximum autorisé en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weight:18000" - axles (M18) : nombre maximum d'axe autorisé (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "axles:2" - weightPerAxle (M18) : poids maximum autorisé par axe en kilogramme (il est nécessaire d'utiliser un graphe intégrant les attributs poids lourds). Usage : "weightPerAxle:9000" - snapSpeed : vitesse d'accrochage au graphe en kilomètre par heure. Usage : "snapSpeed:10"	oui	

Trajets en entrée (pickupDeliveryRouteInput)

paramètre	type	optionnel	description
id	string	Non	Identifiant du trajet
departure	(location)	Non	Coordonnées du point de départ
arrival	(location)	Non	Coordonnées du point d'arrivée
distance	long	Oui	Distance du trajet (en mètre)
duration	long	Oui	Durée du trajet (en secondes)
waypoints	Array of wayPoint	Oui	Liste des étapes du trajet

Etapes (wayPoint)

paramètre	type	optionnel	description
id	string	Non	Identifiant de l'étape

paramètre	type	optionnel	description
location	(location)	Non	Coordonnées de l'étape
stopoverPosition	long	Oui	Position de l'étape sur le trajet, si l'étape est un arrêt. Si non renseigné cette étape n'est qu'un point de passage.

Coordonnées (location)

paramètre	type	optionnel	description
lat	double	Non	Latitude du point (en wgs84)
long	double	Non	Longitude du point (en wgs84)
nodeId	string	Oui	Noeud du graphe. Attention : un noeud physique n'a pas le même ID dans un autre graphe.

Contraintes (pickupDeliveryConstraints)

paramètre	type	optionnel	description
maxDetourDuration	long	Oui	Durée maximale du détour (en secondes). Seuls les trajets résultants pour lesquels

paramètre	type	optionnel	description
			la durée ajoutée est inférieure à cette limite seront renvoyés par le web service. La durée du détour est définie comme : (durée avec détour) - (durée sans détour)
minSharedDistance	long	Oui	Valeur minimale du ratio de distance commun. Seuls les trajets résultants pour lesquels le ratio de distance commun est supérieur à cette limite seront renvoyés par le web service. Le ratio de distance commun est défini comme : (distance commune) / (distance totale avec détour)

Tri (pickupDeliverySortingOptions)

paramètre	type	optionnel	description
routesAndCandidatesSortCriteria	(sortCriteriaEnum)	Oui	Options de tri : - MIN_DETOUR_DURATION = tri (ascendant) les résultats selon la durée du détour - MIN_DETOUR_DISTANCE = tri (ascendant) les résultats selon la distance du détour - MAX_SHARED_DISTANCE = tri (descendant) les résultats selon le ratio de distance commun

Nombre de retour (pickupDeliverySizes)

paramètre	type	optionnel	description
routes	long	Non	Nombre maximal de trajet à renvoyer.
candidates	long	Non	Nombre maximal de candidats à renvoyer pour chaque trajet.

En sortie

paramètre	type	description
results	array of results/result (pickupDeliveryRouteResult)	Liste des trajets calculés.

Liste des trajets calculés (pickupDeliveryRouteResult)

paramètre	type	description
routeId	string	Liste des trajets calculés.
candidates	array of candidates/ candidate (pickupDeliveryRouteCandidate)	Liste des détours possibles pour le trajet.

Liste des détours possibles pour le trajet (pickupDeliveryRouteCandidate)

paramètre	type	description
pickup	(meetingPointCandidate)	Candidat pour le ramassage.
delivery	(meetingPointCandidate)	Candidat pour la dépose.
sharedRouteDuration	long	Durée du trajet commun (en secondes).
sharedRouteDistance	long	Distance du trajet commun (en mètres).
totalDurationWithDetour	long	Durée totale du trajet incluant le détour (en secondes).
totalDistanceWithDetour	long	Distance totale du trajet incluant le détour (en mètres).
detourDuration	long	Durée du détour = (durée avec détour) - (durée du trajet original)
detourDistance	long	Distance du détour = (distance avec détour) - (distance du trajet original)
sharedDistance	double	Distance commune = ratio entre (distance commune) et (Distance totale du trajet incluant le détour)

Candidats (meetingPointCandidate)

paramètre	type	description
meetingPointId	string	Identifiant du lieu de ramassage/dépose
duration	long	Pour un lien de ramassage : durée depuis le point de départ du trajet jusqu'au lieu de ramassage (en secondes) Pour un lieu de dépose : durée depuis le lieu de dépose jusqu'au point d'arrivée du trajet (en secondes)
distance	long	Pour un lien de ramassage : distance depuis le point de départ du trajet jusqu'au lieu de ramassage (en mètres) Pour un lieu de dépose : distance depuis le lieu de dépose jusqu'au point d'arrivée du trajet (en mètres)
meetingPointSubPath	int	Position du segment optimal (dans le trajet) du lieu de ramassage/ dépose.

REST (POST)

Requête

Requête

```
http://<server>/<webapp>/api/lbs/pickupDelivery.json
```

Data (JSON)

```
{
  "routes" : [ {
    "id" : "trip_1",
    "departure" : {
      "lat" : "47.333990",
      "lon" : "-1.805604"
    },
    "arrival" : {
      "lat" : "47.479740",
      "lon" : "-1.095251"
    },
    "waypoints" : [ {
      "id" : "waypoint_1",
      "location" : {
        "lat" : "47.157530",
        "lon" : "-1.421958"
      }
    }]
  }],
  "pickupPoints" : [ {
    "id" : "depMeet_1",
    "location" : {
      "lat" : "47.228659",
      "lon" : "-1.600995"
    }
  }],
  "deliveryPoints" : [ {
    "id" : "arrMeet_1",
    "location" : {
      "lat" : "47.293823",
      "lon" : "-1.480789"
    }
  }],
  "constraints" : {
    "maxDetourDuration" : 3600,
    "minSharedDistance" : 0.1
  },
  "sortOptions" : {
    "routesAndCandidatesSortCriterium" : "MIN_DETOUR_DURATION"
  },
  "size" : {
    "routes" : 1,
    "candidates" : 1
  }
}
```

Requête

Réponse

La réponse est toujours encodée en UTF-8.

Format JSON

```
{
    "message": null,
    "status": "OK",
    "results": [
        {
            "routeId": "trip_1",
            "candidates": [
                {
```

```
                    "pickup": {
                        "meetingPointId": "depMeet_1",
                        "duration": 1349,
                        "distance": 23426,
                        "meetingPointSubPath": 0
                    },
                    "delivery": {
                        "meetingPointId": "arrMeet_1",
                        "duration": 2177,
                        "distance": 43159,
                        "meetingPointSubPath": 0
                    },
                    "sharedRouteDuration": 1333,
                    "sharedRouteDistance": 18767,
                    "totalDurationWithDetour": 4860,
                    "totalDistanceWithDetour": 85352,
                    "detourDuration": -577,
                    "detourDistance": -19590,
                    "sharedDistance": 0.21
                }
            ]
        }
    ]
}
```


FAQ

1. Comment utiliser les statistiques routières ?

   Vérifier que le graphe contient bien ces statistiques et utiliser les deux paramètres suivants : `computeOptions` avec la valeur *trafficPatterns* et `startDateTime` pour préciser la date/heure de départ.

2. Comment faire un calcul d'itinéraire sans péage?

   Si la contrainte *Toll* a bien été incluse dans le graphe, placer une exclusion dans `exclusions`

3. Qu'est-ce que les Speed Patterns ? Comment les utiliser ?

   Afin de proposer des temps de trajets au plus près des conditions de circulation, les graphes fournis par GEOCONCEPT SAS intègrent, à partir de la version M18, pour les voitures et pour les camions 5 profils de vitesses (Speed Patterns) pour tenir compte des différents niveaux de congestion dans une journée :

   - standard *normal-speed* correspond à la vitesse d'une heure moyennement chargée (11h)
   - nuit *fast-speed* correspond à un trafic très fluide, observé le plus souvent la nuit (3h)
   - chargée *slow-speed* correspond aux heures de trafic denses (8h)
   - heure de pointe *very-slow-speed* correspond aux heures de trafic denses dans les grandes agglomérations, plus lent que le précédent (8h)
   - défaut *defaut* correspond aux vitesses moyennées sur une journée entière

Pour les utiliser il faut passer, lors de l'appel au web service, le paramètre `computeOptions` avec l'option *speedPattern* et préciser la valeur de la Speed Pattern demandée sans omettre un profil de véhicule Exemple :

+

```
&computeOptions=speedPattern:fast-speed&profileId=1
```

Comment utiliser les attributs poids lourds ?

Il faut que le graphe intègre les attributs de poids lourds (en standard dans les graphes fournis par GEOCONCEPT SAS à partir de la version M18) et soit calculer un itinéraire en utilisant un profil de véhicule utilisant les restrictions (cf. le catalogue de véhicules, éditable, définit dans le fichier SmartRoutingVehicles.xml, dans le dossier `'<*GEOCONCEPT_WEB_HOME*>''\*smartrouting*\*jee* \*smartrouting*\*conf*\ ), soit surcharger l'appel au web service en utilisant le paramètre **`computeOptions`** avec les options *length*, *width*, *height*, *weight*, *axles* et/ou *weightPerAxle*.
Exemple pour un trajet avec un véhicule de 4,5 mètres de hauteur :

```
&computeOptions=height:450
```

Optimisation (version complète)

Principe

Ce web service permet de définir des tournées optimales en minimisant les coûts d'exploitations en tenant en compte d'une part des données cartographiques routières et de leurs caractéristiques en fonction des profils des véhicules utilisés et d'autre part des contraintes « Métiers » qui concernent, d'un côté les « Clients » que l'on va livrer, visiter ou chez lesquels on intervient, et de l'autre côté les « Ressources » qui vont effectuer ces différents types de tâches.

Le service d'optimisation permet de déterminer la meilleure répartition possible des clients par rapport aux ressources disponibles. Il s'appuie :

- sur les positions géographiques de l'ensemble des clients et des ressources : un distancier est préalablement calculé par les fonctions de calculs d'itinéraires et tient compte du profil des véhicules, des attributs poids lourds, …
- sur des contraintes des clients : créneaux horaires, quantités à livrer, particularités requises, …
- sur des paramètres des ressources : temps de travail, coût de la ressource, …

L'optimisation se fait par la recherche du moindre coût global de la solution sur la base des coûts des ressources fournies par l'utilisateur ou en utilisant les coûts par défaut fournis par le module d'optimisation.

Disponibilité

Ce web service est une option de Geoconcept Web, veuiller nous contacter pour connaître les modalités d'acquisition et obtenir la documentation complète.

Pages pop-up/iframe

Introduction

Architecture Geoconcept Call Center

Geoconcept Call Center est un application Client léger permettant la recherche géographique de ressources. C'est une application J2EE, utilisant une base de données, et reposant sur les services web de Geoconcept Web.

Principe de fonctionnement

La solution Geoconcept Web publie des web services permettant une intégration facile dans des applications nécessitant une recherche géographique.

Nous prendrons pour exemple une application de gestion de sinistre nécessitant la recherche d'un prestataire proche du sinistre au moyen de Web services. Dans cette application, les prestataires figurent dans une base de données classique (Oracle, SQL Server…).

Utilisation des pop-up et principe d'intégration

L'intégration repose sur des écrans Web de type *pop-up* appelés à partir de l'application principale.
A certains moments, des fenêtres issues du serveur Geoconcept Web permettent de présenter à l'utilisateur des donnés géographiques. Le choix de l'utilisateur est renvoyé par un formulaire POST à l'application maître.

On privilégie les échanges de données par XML pour les données de taille supérieure à 100 caractères.

Géocodage

L'application affiche un outil de reconnaissance d'adresse.

Pour cela, en fonction d'une adresse sous la forme (numéro et rue, ville, code postal), on propose une liste avec les adresses les plus proches trouvées (éventuellement une seule, s'il n'y a pas d'erreur), et leur localisation.

Disponibilité

Disponible en permanence avec Geoconcept Web.

La pop-up

L'url de la pop-up est : http://<serveur>/callcenter/Ext/geocode.do

La pop-up est appelée par la page de test présente à l'adresse :

http://<serveur>/callcenter/Ext/geocodetest.do

La couche affichée dans la carte est configurée dans le paramètre serveur popup.geographics.map.layer

Formulaire et structure du XML

Il faut envoyer à la pop-up les variables ci dessous par méthode POST :

urlPost : URL recevant les résultats du géocodage

userData : n'importe quelle donnée, elle sera soumise sans modification au formulaire de réception des résultats. Cette donnée permet à l'application maître de passer un contexte, qui pourra être utile pour le retour.

xmlData : les données à géocoder en xml. Le format est celui d'une requête SOAP de géocodage.

Exemple :

```
<?xml version="1.0" encoding="UTF-8" ?>
       <gc:GeocodeRequest xmlns:gc="http://geoconcept.com/gc/schemas">
          <gc:Address>
             <gc:CountryCode>fr</gc:CountryCode>
             <gc:City>Paris</gc:City>
             <gc:PostalCode>75013</gc:PostalCode>
             <gc:AddressLine>25 rue de Toldiac</gc:AddressLine>
          </gc:Address>
       </gc:GeocodeRequest>
```

Récupération des résultats du géocodage

Le choix de l'utilisateur est renvoyé par une requête POST contenant les paramètres suivants :

userData : les données utilisateurs passées précédemment

address : l'adresse effectivement reconnue (attention : ne comporte pas le numéro si le numéro n'existait pas dans le base de géocodage)

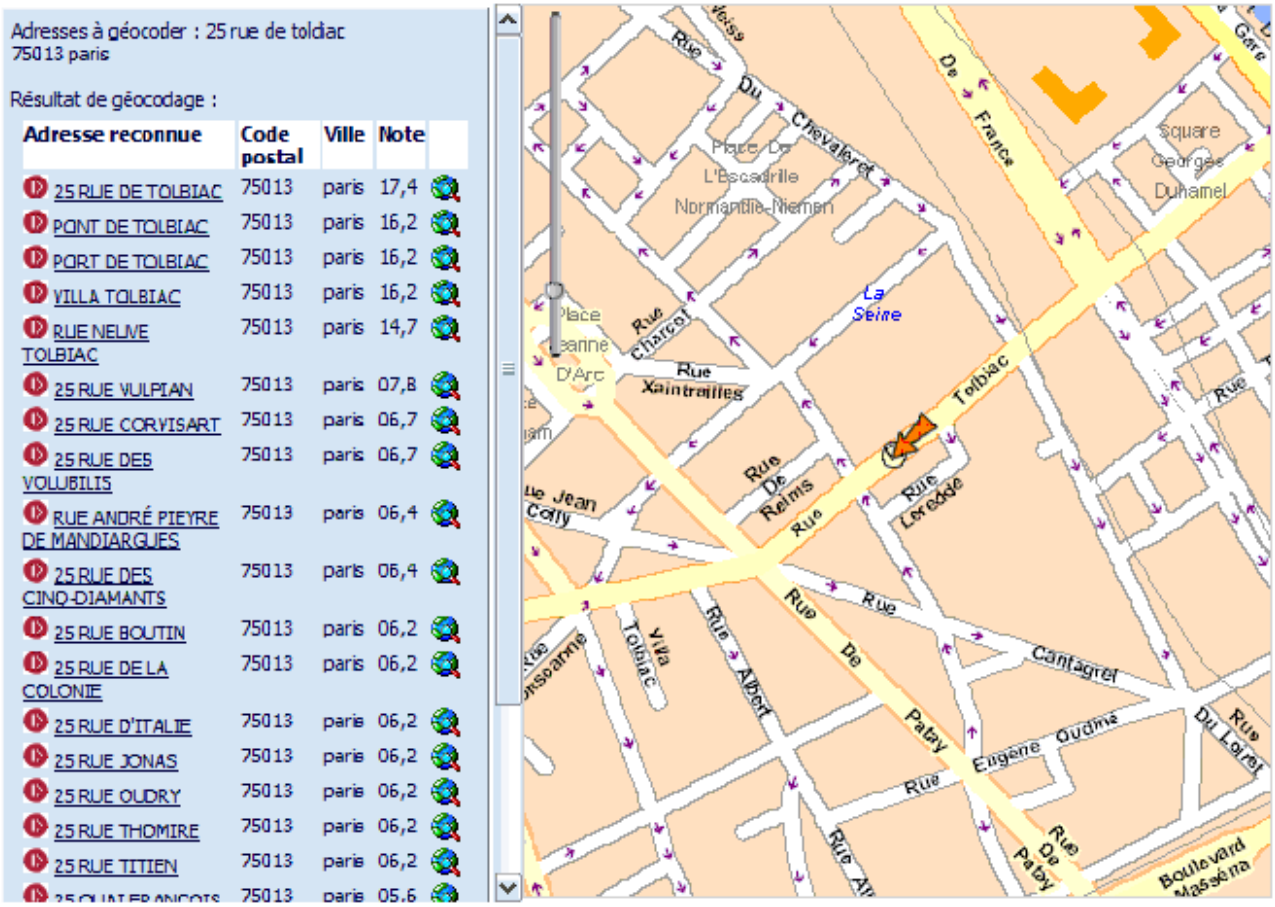
city : la ville reconnue.

postalCode : le code postal reconnu

x : position X (dans la projection configurée dans le paramètre serveur, ou en longitude si on a demandé une projection WGS 84)

y : position Y (idem)

Résultat



Recherche de ressource

Suite au clic sur un bouton «recherche», l'application maître affiche une fenêtre pop-up qui appelle une page de recherche sur le serveur Geoconcept Web. Cette page fait une première requête pour sélectionner un certain nombre de candidats répondant aux critères indiqués dans la base de données des prestataires. Elle peut en particulier rechercher des ressources dans un carré autour du point de recherche, et intégrant des critères métiers. Elle pourra en particulier utiliser des procédures stockées.

Disponibilité

Cette partie est en cours de rédaction.

La pop-up

L'url de la pop-up est :http://<serveur>/callcenter/Ext/search.do

La pop-up est appelé par la page de test présente à l'adresse :

http://<serveur>/callcenter/Ext/searchtest.do

La couche affichée dans la carte est configurée dans le paramètre serveur popup.geographics.map.layer

Paramètres

- **`urlPost`** : URL recevant les résultats du géocodage
- **`userData`** : n'importe quelle donnée, elle sera soumise sans modification au formulaire de réception des résultats. Cette donnée permet à l'application maître de passer un contexte, qui pourra être utile pour le retour.
- **`xmlData`** : les données à géocoder en xml. Le format est celui d'une requête SOAP de SearchAround.

Si la projection n'est pas configurée dans gc:Options, la projection utilisée est celle configurée dans le paramètre serveur popup.ws.defaultSrs

Toutefois, on peut ajouter les informations suivantes, utilisées uniquement pour l'affichage, dans les paramètres décrivant la cible, ou une ressource à rechercher :

- **`Name`** : nom de la ressource
- **`Html`** : un code html à afficher sous le nom de la ressource
- **`sheetHtml`** : un code html à afficher sur la mini-fiche décrivant la ressource (apparaît sur la carte quand on clique sur la ressource)
- **`ImageUrl`** : url d'image de la ressource

Exemple de requête

```
<gc:SearchAroundRequest xmlns:gc="http://geoconcept.com/gc/schemas">
          <gc:Options>
          </gc:Options>
          <gc:Target>
             <gc:X>602725</gc:X>
             <gc:Y>2425604</gc:Y>
             <gc:Id>1</gc:Id>
          </gc:Target>
          <gc:Resource>
             <gc:X>599266</gc:X>
             <gc:Y>2425096</gc:Y>
             <gc:Id>2</gc:Id>
<gc:Name>car1</gc:Name>
<gc:Address>12 rue de tolbaic</gc:Address>

<gc:Html><![CDATA[<b>ici</b>]]></gc:Html>
          </gc:Resource>
          <gc:Resource>
             <gc:X>599366</gc:X>
             <gc:Y>2425196</gc:Y>
             <gc:Id>2</gc:Id>
<gc:Name>nom1</gc:Name>
<gc:Html><![CDATA[<b>decription html</b>]]></gc:Html>
          </gc:Resource>
</gc:SearchAroundRequest>
```

Affichage de la pop-up

Récupération du résultat

Le choix de l'utilisateur est renvoyé par une requête POST contenant les paramètres suivants :

- `userData` : les données utilisateurs passées précédemment
- `resourceId` : l'identifiant de la ressource choisie
- `resourceName` : le nom de la ressource choisie
- `distance` : la distance à la ressource en mètres
- `time` : le temps d'accès en secondes

Définition de zone

Principe d'intégration

L'intégration repose sur des écrans Web de type *pop-up* appelés à partir de l'application principale. A certains moments, des fenêtres issues du serveur GeoConcept LBS Platform permettent de présenter à l'utilisateur des donnés géographiques. Le choix de l'utilisateur est renvoyé par un formulaire POST à l'application maître.

On privilégie les échanges de données par XML pour les données de taille supérieure à 100 caractères.

La pop-up

L'url de la pop-up est : http://<serveur>/callcenter/Ext/zonedefine.do

La pop-up est appelé par la page de test présente à l'adresse :

http://<serveur>/callcenter/Ext/zonedefinetest.do

La couche affichée dans la carte est configurée dans le paramètre serveur popup.geographics.map.layer

Si le nom des entitiés à sélectionner est dans un champ unicode mettre le paramètre serveur popup.geographics.entityNameUnicode à true (false sinon)

Paramètres

- **`urlPost`** : URL recevant les résultats
- **`userData`** : n'importe quelle donnée, elle sera soumise sans modification au formulaire de réception des résultats. Cette donnée permet à l'application maître de passer un contexte, qui pourra être utile pour le retour.
- **`xmlData`** : les données de la zone structure xml, de racine « DefineZoneRequest » Cet élément xml peut contenir les informations suivantes :
- **`CurrentZone`** : liste d'éléments géographiques référencés, définie par les attributs type, subtype, et code
- **`Entity`** : élément géographique défini par Id
- **`DisplayElements`** : liste d'éléments à afficher (il peut y avoir plusieurs listes)

Note : on pourra envoyer les coordonnées géographiques en projection MAP (la projection utilisée est celle configurée dans le paramètre serveur popup.ws.defaultSrs) ou en coordonnées WGS 84 (code epsg:4326).

Exemple de requête

```
<gc:DefineZoneRequest xmlns:gc="http://geoconcept.com/gc/schemas">
  <!-- The Map must have a tab  type.subtype where type/subtype is selectable. code is a field of  type/
subtype -->
  <gc:CurrentZone type="Administrative unit" subtype="Order8" code="Government code">
    <!-- select Bagneux -->
    <gc:Entity id="92007"/>
    <!-- select Cachan -->
    <gc:Entity id="94016"/>
    <!-- select Châtillon -->
    <gc:Entity id="92020"/>
  </gc:CurrentZone>
  <gc:DisplayElements img="http://en.geoconcept.com/sites/all/themes/geoconcept/images/flags/fr.png"
 projection="EPSG:4326">
    <gc:Symbol x="2.31035" y="48.80165" />
    <gc:Symbol x="2.31035" y="48.89165" />
  </gc:DisplayElements>
  <!-- An image must be called Pin_user_gcblue  -->
  <gc:DisplayElements img="Pin_user_gcblue">
    <gc:Symbol x="601725" y="2424604" />
  </gc:DisplayElements>
```

```
</gc:DefineZoneRequest >
```

Affichage de la page

La page affiche une carte avec des outils de sélection. Par défaut, l'outil de sélection est l'outil de retouche de la sélection.

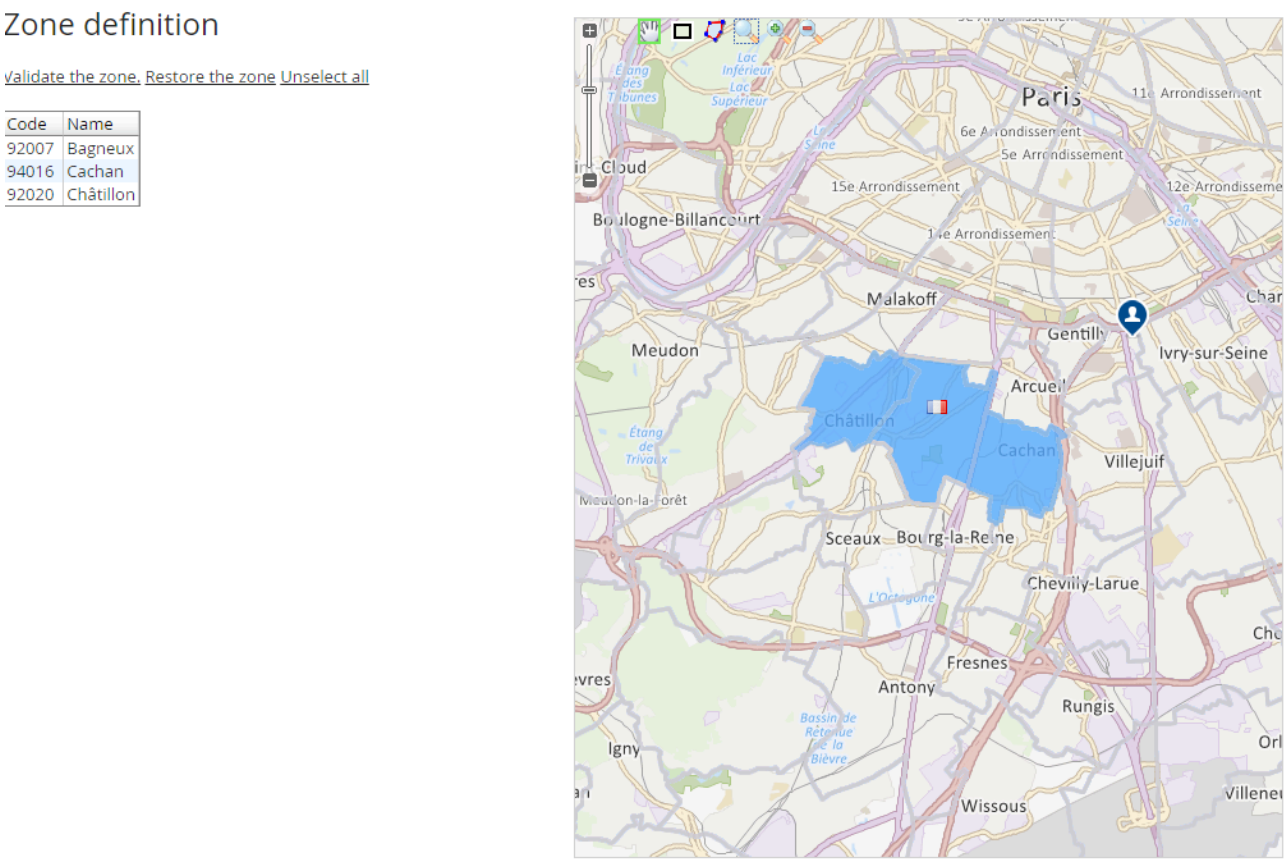
Les entités passées en paramètre sont coloriées.

Les symboles passés dans la liste sont affichés avec l'image dont le nom est indiqué.

- outil déplacer : permet de déplacer la carte
- outil dessiner une zone : permet de saisir un polygone point par point
- les entités intégralement ou partiellement comprises dans le dessin sont alors sélectionnées, le dessin s'efface.
- outil retoucher la sélection : Un clic sur une entité la sélectionne/déselectionne suivant son état.

un bouton permet de terminer la saisie.

Exemple d'interface



Récupération du résultat

Le choix de l'utilisateur est renvoyé par une requête POST contenant les paramètres suivants :

```
<gc:DefineZoneResult xmlns:gc="http://geoconcept.com/gc/schemas">
  <gc:Zone type="Administrative unit" subtype="Order8" code="Government code">
     <gc:Entity id="92007"/>
     <gc:Entity id="94016"/>
     <gc:Entity id="92020"/>
  </gc:Zone>
</gc:DefineZoneResult>
```

Exemples

Exemple d'intégration des web services : application de gestion de sinistre

Cet exemple permet d'utiliser les différents composants de Geoconcept Web et de les associer entre eux afin de construire une application complète.

Nous prendrons pour exemple une application de gestion de sinistre nécessitant la recherche d'un prestataire proche du sinistre. Dans cette application, les prestataires figurent dans une base de données classique (Oracle, SQL Server, …).

Géocodage des prestataires

Ajouter dans la table des prestataires deux colonnes X, Y (ou longitude, latitude) de type numérique.

Pour un géocodage massif (plusieurs milliers de lignes) rapide, utiliser l'application Universal Geocoder en mode desktop, ou en mode batch.

Pour un géocodage plus modeste, on peut aussi utiliser le web service de géocodage décrit dans la documentation. Ce web service prend en entrée une adresse et retourne la ou les meilleures adresses trouvées, avec leur position X,Y.

Indexer les champs longitude, latitude.

Modifications sur l'application

Ajout d'une fonctionnalité Géocodage du sinistre

Après la saisie de l'adresse du sinistre va un formulaire, on ajoute une étape *géocoder*, qui appelle le web service de géocodage. Il est également possible d'afficher une carte de l'endroit avec le composant Javascript, en utilisant le composant GCIS. Suite à ce géocodage, on obtient une position du sinistre (X,Y). Grâce aux fonctionnalités Javascript, il est possible de centrer la carte affichée sur la position X Y retournée.

Rechercher des prestataires candidats

Dans un premier temps, il est nécessaire de faire une première sélection des prestataires les plus proches, compris dans les limites d'une Bounding Box à définir.

Faire une première requête approchée dans la table des prestataires, pour rechercher les prestataires dans un carré de x kilomètres autour du sinistre. On calcule alors (formule approchée) :

```
longitude0 : longitude sinistre
latitude0 : latitude sinistre
l : distance de recherche (en km)
R = 6371 (en km)
delta_latitude = (180 / PI) * l / R
delta_longitude = (180 / PI) * l / R / cos(latitude0)
```

ce qui permet d’effectuer la requête SQL simple :

```
SELECT * FROM prestataire
WHERE longitude > longitude0 - delta_longitude
AND longitude < longitude0 + delta_longitude
AND latitude > latitude0 - delta_latitude
AND latitude < latitude0 + delta_latitude
```

Cet exemple de requête détermine un certain nombre de candidats parmi les plus proches. Si le nombre de candidats est trop grand (plus de 100), on pourra la recommencer en divisant la distance par deux. Si le nombre de candidats est trop faible, on pourra également la recommencer en multipliant la distance par deux, jusqu'à obtenir un nombre suffisant de candidats (au moins 10). Il faut ensuite pouvoir classer ces candidats potentiels en fonction de la méthode choisie, afin de faire ressortir les plus proches.

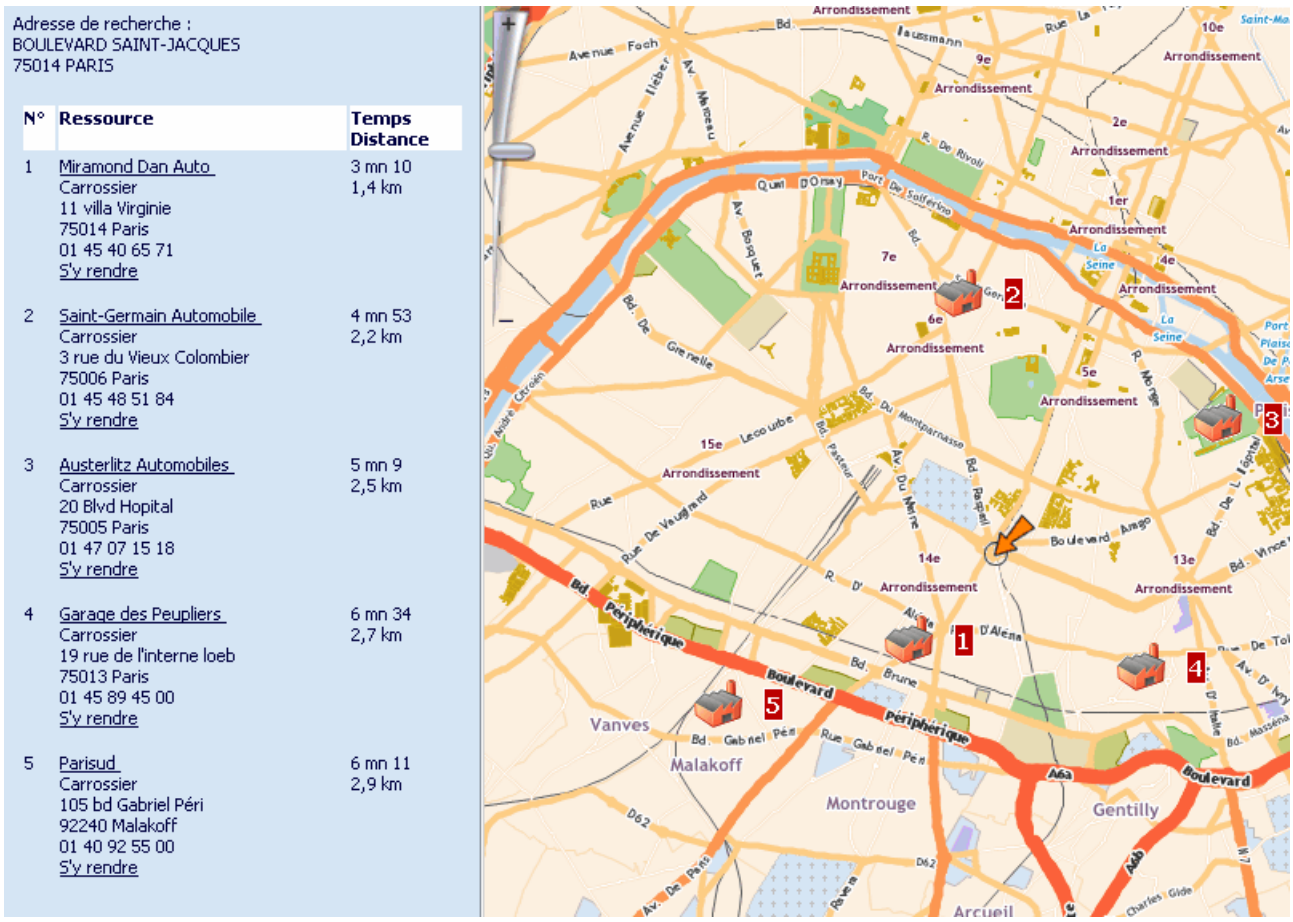
Calculs de distance, temps, et tri des candidats

On utilise le Web Service « SearchAround » décrit dans la documentation pour classer les candidats en fonction de la méthode choisie.

Il suffit alors de présenter le résultat des prestataires classés à l’utilisateur pour qu’il choisisse le prestataire le plus adéquat (le plus proche en distance ou le plus proche en temps).

Affichage des candidats sur une carte

On utilisera l’API Javascript pour inclure la carte dans l’application web existante et afficher les prestataires sur un fond de plan.



Adresse de recherche :
BOULEVARD SAINT-JACQUES
75014 PARIS

N°	Ressource	Temps Distance
1	Miramond Dan Auto Carrossier 11 villa Virginie 75014 Paris 01 45 40 65 71 S'y rendre	3 mn 10 1,4 km
2	Saint-Germain Automobile Carrossier 3 rue du Vieux Colombier 75006 Paris 01 45 48 51 84 S'y rendre	4 mn 53 2,2 km
3	Austerlitz Automobiles Carrossier 20 Blvd Hopital 75005 Paris 01 47 07 15 18 S'y rendre	5 mn 9 2,5 km
4	Garage des Peupliers Carrossier 19 rue de l'interne loeb 75013 Paris 01 45 89 45 00 S'y rendre	6 mn 34 2,7 km
5	Parisud Carrossier 105 bd Gabriel Péri 92240 Malakoff 01 40 92 55 00 S'y rendre	6 mn 11 2,9 km

Calcul d'itinéraires

On utilise le web service de calcul d'itinéraire, qui renvoie une feuille de route en format xml entre l'adresse de départ (renseignée par l'utilisateur via un formulaire) et le prestataire choisi par l'utilisateur.

Il est ensuite possible d'utiliser cette feuille afin de tracer un itinéraire entre le point de départ et le point d'arrivée.

Calcul d'itinéraires

Exemple de scripts en PHP 5

Géocodage

```
<h1>Geocoding service</h1>

<?php
// Display all errors:
error_reporting(E_ALL);
ini_set('display_errors', '1');

////////////////////////////////////////////////////////////////////////
//////////////////////////  CREATE REQUEST //////////////////////////
////////////////////////////////////////////////////////////////////////

class AddressType{

   function __construct($countryCode,$city,$postalCode,$addressLine,$srs="epsg:4326",$maxResponses=2)  {
     $this->countryCode=$countryCode;
     $this->city=$city;
     $this->postalCode=$postalCode;
     $this->addressLine=$addressLine;
     $this->srs=$srs;
     $this->maxResponses=$maxResponses;
   }
```

```
};


$addressType=new AddressType("fr","Paris","75013","25 rue de Toldiac","wgs84");
$request=array(
  "geocode" => $addressType
);

//////////////////////////////////////////////////////////////////////
//////////////////////////////  CALL SOAP  //////////////////////////////
//////////////////////////////////////////////////////////////////////

try{
  $clientOptions=array(
    'trace'=>true,
    'exceptions'=>true,
    'encoding'=>'utf-8'
  );

  $client = new SoapClient('http://gcweb.geoconcept.com/gws/api/ws/geocodeService?wsdl', $clientOptions);


  echo "<h2>Request</h2>";
  echo "<pre>";
  var_dump($request);
  echo "</pre>";

  $response = $client->__soapCall("geocode",$request);


  echo "<h2>Response (initialAddress)</h2>";
  echo "<pre>";
  var_dump($response->GeocodeResult->initialAddress);
  echo "</pre>";

  echo "<h2>Response (geocodedAddress)</h2>";
  echo "<pre>";
  var_dump($response->GeocodeResult->geocodedAddress);
  echo "</pre>";
}
catch (SoapFault $e) {
  echo $e;
}
?>
```

Calcul d’itinéraire

```
<h1>Route service</h1>

<?php
// Display all errors:
error_reporting(E_ALL);
ini_set('display_errors', '1');

//////////////////////////////////////////////////////////////////////
//////////////////////////  CREATE REQUEST //////////////////////////
//////////////////////////////////////////////////////////////////////

class RoutePointType {

  function __construct($x,$y)  {
    $this->x=$x;
```

```
    $this->y=$y;
  }
}

$step1=new RoutePointType(0.691012,47.384813);
$step2=new RoutePointType(0.693012,47.385813);

$route=array(
  "request" => array(
     "origin" => $step1,
     "destination" => $step2,
     "srs" => "epsg:4326",
     "method" => "time",
     "format" => "STANDARD",
     "rejectFlags" => array("Toll"),
     "tolerance" => array()
    )

);

$request=array(
   "route" => $route
);

//////////////////////////////////////////////////////////////////////
//////////////////////////  CALL SOAP  ///////////////////////////////
//////////////////////////////////////////////////////////////////////

try{
  $clientOptions=array(
    'trace'=>true,
    'exceptions'=>true,
    'encoding'=>'utf-8'
  );

  $client = new SoapClient('http://pouget/geoconcept-web/api/ws/routeService?wsdl', $clientOptions);


  echo "<h2>Request</h2>";
  echo "<pre>";
  var_dump($request);
  echo "</pre>";

  $response = $client->__soapCall("route",$request);

  echo "<h2>Response</h2>";
  echo "<pre>";
  var_dump($response);
  echo "</pre>";
}
catch (SoapFault $e) {
  echo $e;
}
?>
```

Search Around

```
<h1>SearchAround service</h1>

<?php
// Display all errors:
error_reporting(E_ALL);
```

```
ini_set('display_errors', '1');

////////////////////////////////////////////////////////////////////
//////////////////////////  CREATE REQUEST //////////////////////////
////////////////////////////////////////////////////////////////////

class SearchAroundTargetType {

  function __construct($x,$y)  {
    $this->x=$x;
    $this->y=$y;
  }
}

class SearchAroundPointType {

  function __construct($x,$y,$id="",$priority1="",$priority2="")  {
    $this->x=$x;
    $this->y=$y;
    $this->id=$id;
    $this->priority1=$priority1;
    $this->priority2=$priority2;
  }
}

$target=new SearchAroundTargetType(-1.593927,47.218580);
$resource1=new SearchAroundPointType(-1.593927,47.188580,2,1,1);
$resource2=new SearchAroundPointType(-1.556927,47.188580,3,1,2);
$resource3=new SearchAroundPointType(-1.557927,47.189580,4,1,1);

$searchAround=array(
       "request" => array (
                "target" => $target,
                "method" => "time",
                "reverse"   => "",
                "srs" => "epsg:4326",
                 "exclusions" => array(),
                "resources" => array($resource1,$resource2,$resource3),
       )

);

$request = array("searchAround" => $searchAround);


////////////////////////////////////////////////////////////////////
//////////////////////////  CALL SOAP  //////////////////////////
////////////////////////////////////////////////////////////////////

try{
  $clientOptions=array(
    'trace'=>true,
    'exceptions'=>true,
    'encoding'=>'utf-8'
  );

  $client = new SoapClient('http://pouget/geoconcept-web/api/ws/searchAroundService?wsdl', $clientOptions);

  echo "<h2>Request</h2>";
  echo "<pre>";
  var_dump($request);
  echo "</pre>";
```

```
  $response = $client->__soapCall("searchAround",$request);

  echo "<h2>Response</h2>";
  echo "<pre>";
  var_dump($response);
  echo "</pre>";
}
catch (SoapFault $e) {
  echo $e;
}
?>
```

UGC Server

Installation de UGC Server

Ce document présente les étapes d'installation du logiciel Universal Geocoder Server. Le choix des composants à installer est nécessaire lors de l'installation, parmi :

- UGC JEE,
- UGC Command Line,
- UGC .NET. (non disponible avec Geoconcept Web)

Ces composants sont indépendants entre eux et répondent à des problématiques différentes. Par exemple, le composant UGC JEE est nécessaire pour le widget « Geocoder » dans les portails.

Ce document décrit également le montage de d'adaptateur de ressource UGC en fonction du serveur d'application cible.

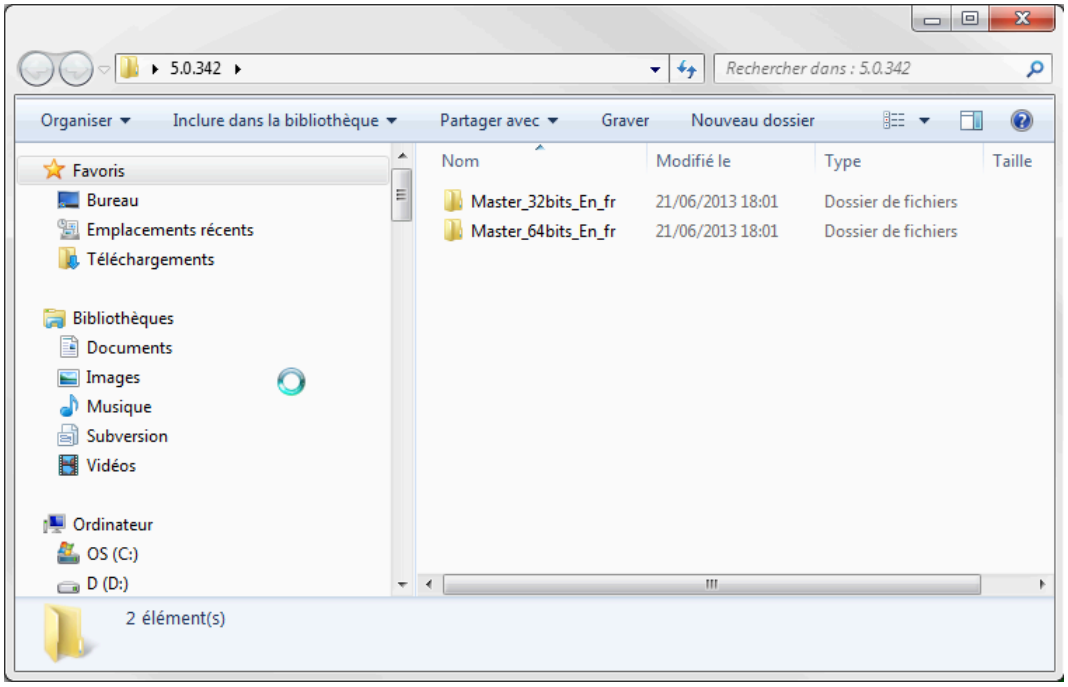
Enfin, l'installation des fichiers nécessaires à l'autocomplétion est décrite dans ce document. Pour utiliser le web service d'autocomplétion, merci de vous reporter à la partie API et Web Services de Geoconcept Web.

Installation du logiciel UGC Server

La procédure suivante ne concerne pas Geoconcept Web cas son installateur se charge d'installer et de paramétrer le composant UGC JEE.

Après avoir dézippé le dossier téléchargé, vous trouvez deux répertoires : un pour l'installateur 32 bits et un pour l'installateur 64 bits. Choisissez celui qui convient à votre architecture.

Choix du master 32 bits ou 64 bits



Dans le cas d'utilisation de UGC JEE, la version 32 bits ou 64 bits est à choisir en fonction de votre architecture Java et Apache Tomcat :

- Choisir le master 32 bits si votre JRE est une version 32 bits (Apache Tomcat est alors en 32 bits),
- Choisir le master 64 bits si votre JRE est une version 64 bits (Apache Tomcat est alors en 64 bits).

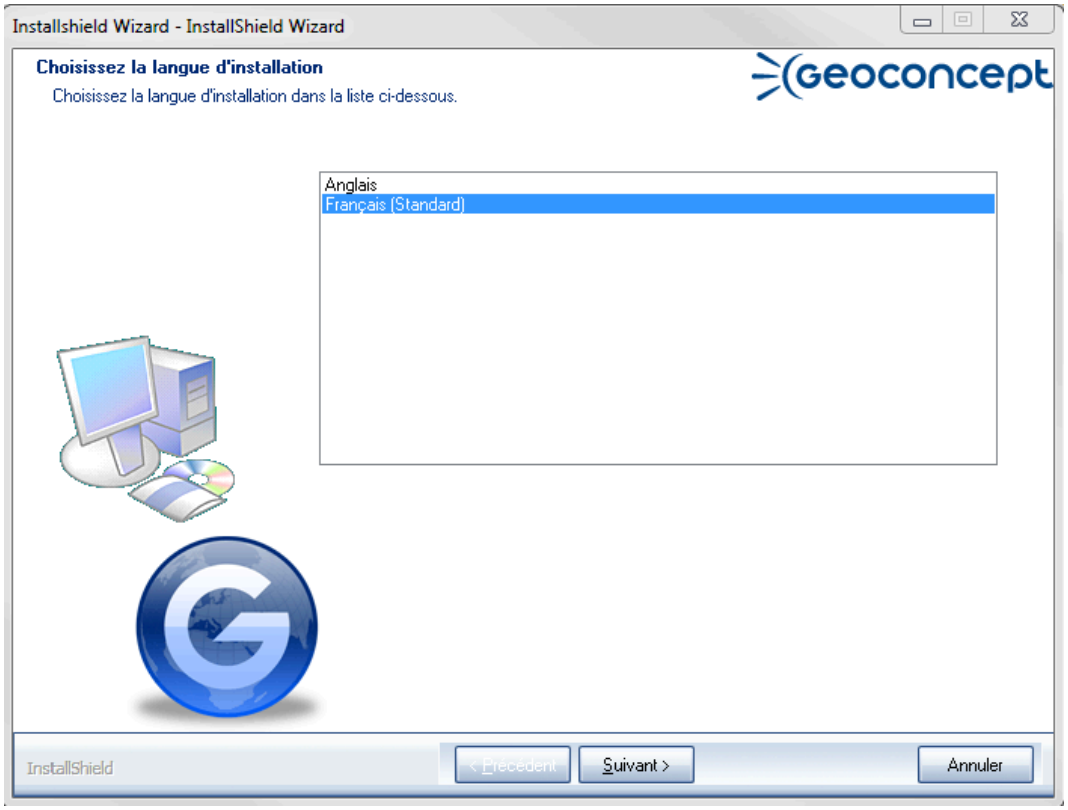
Ce choix a son importance avec les DLL utilisées pour faire fonctionner UGC Server.

Un UGC JEE 32 bits déployé sur un environnement 64 bits ne fonctionnera pas, et réciproquement.

Nous recommandons d'installer la version 64 bits.

Lancer l'installateur setup.exe, puis suivre les différentes étapes proposées par l'assistant d'installation.

Choix de la langue



Installer une nouvelle copie

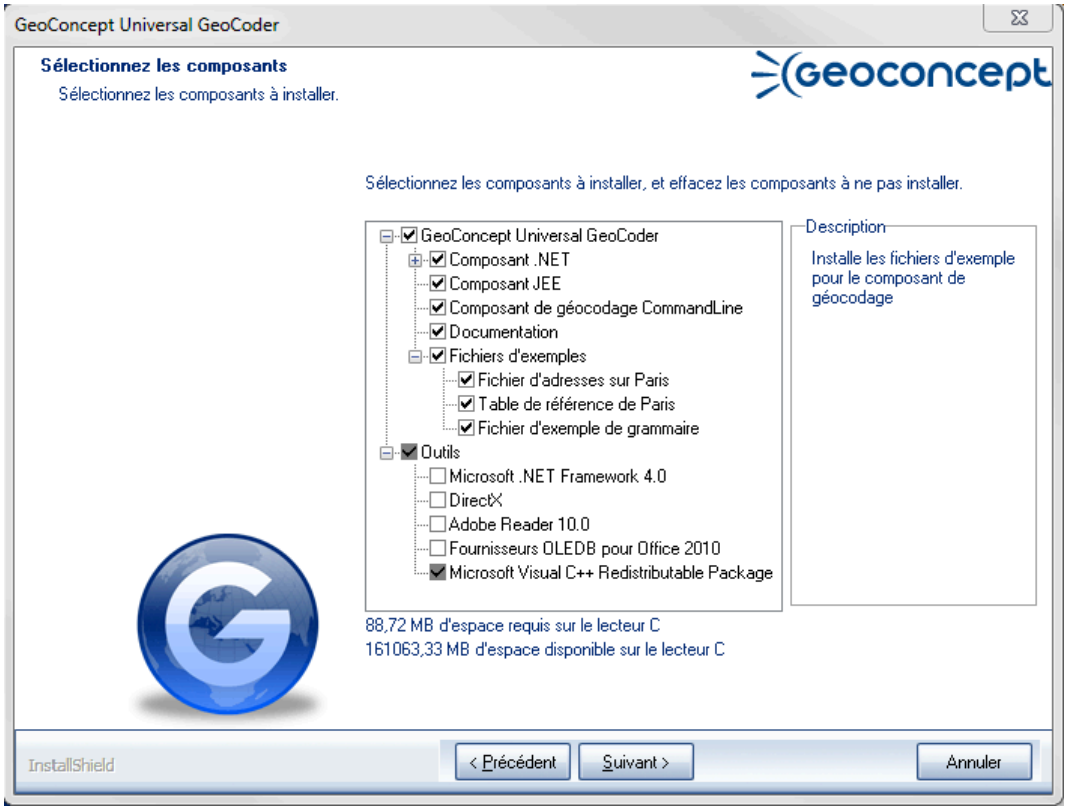
Accepter les conditions d'utilisation

Saisir la clé

Vous devez choisir les composants que vous souhaitez installer, en fonction de l'utilisation que vous souhaitez mettre en place :

- Composant .NET,
- Composant JEE,
- Composant CommandLine.

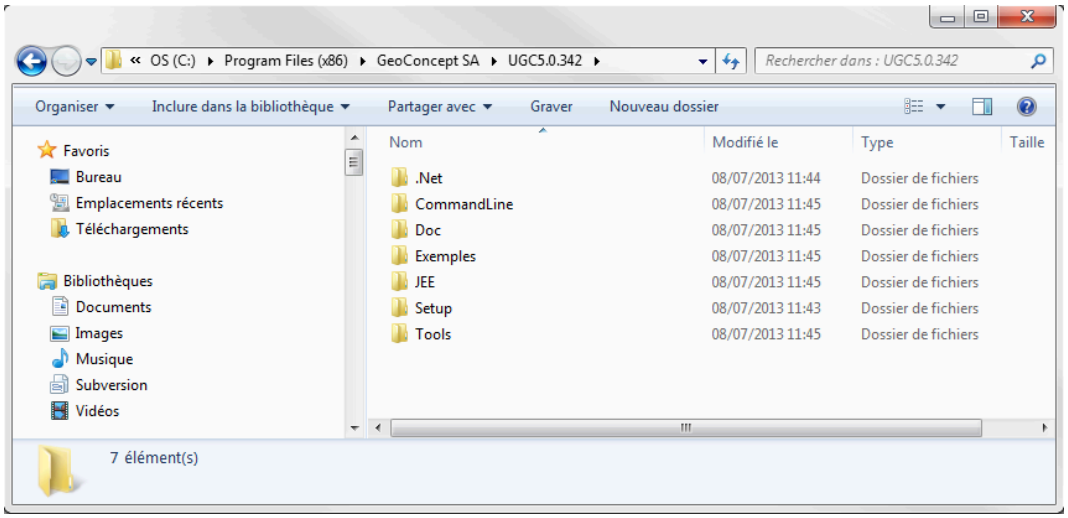
Choix des composants



Choix du répertoire d'installation

Fin de l'installation

Répertoires déployés sur le serveur



Les fichiers du référentiel de géocodage, portant des extensions .ugc.xxi, sont à placer dans le répertoire « <UGC_HOME> »\JEE\ugc\reftables (répertoire par défaut) ou dans un répertoire spécifique de type D:\TableRef. Ce répertoire spécifique devra être paramétré dans le fichier *server.xml* (merci de vous reporter au paragraphe « Configuration pour UGC JEE » dans le manuel de Geoconcept Web).

Déploiement et configuration du serveur d'application

Ce chapitre décrit le montage de l'adaptateur de ressource UGC. Le mode de déploiement dépend du Serveur d'application cible.

Configuration de Apache Tomcat

> L'utilisation du serveur d'application Apache Tomcat dans le cadre de l'installation de Geoconcept Web est décrite dans le manuel de Geoconcept Web.

La suite de ce paragraphe décrit la configuration dans un cadre autre que Geoconcept Web. Certains éléments peuvent être similaires à une utilisation faite dans le cadre de Geoconcept Web.

Le répertoire dans lequel est installé Tomcat est noté par la suite %tomcat%.

Common Lib

Dans %tomcat%/common/lib, copier les fichiers stockés dans %livrable%/tomcat/lib :

- ugc.jar : librairie principale de l'adaptateur de ressource ugc-jee,
- jdom.jar : manipulation de documents XML,
- activation.jar : nécessaire, le copier s'il n'est pas présent,
- javax-resource.jar : nécessaire pour l'adaptateur de ressources.

Configuration server.xml

Au niveau du fichier de configuration principal de Tomcat, il faut déclarer le fournisseur de service primaire en lui associant un nom logique (JNDI) : dans tomcat%/conf/server.xml, éditez le tag GlobalNamingResources et ajoutez :

- pour Tomcat 7 et 8 :

```
<Resource
        name="geoconcept/ugc/default"
        type="com.geoconcept.ugc.connect.tomcat.ConnectionFactory"
        scope="Shareable"
        description="UGC connection factory - local dll"
        auth="Container"
        RootDirectory="%geoconcept%\ugc\"
        factory="com.geoconcept.ugc.connect.tomcat.ConnectionFactory"
        ConnectionMode="LocalDll"
/>
```

> Pour toutes les versions de Tomcat, le paramètre RootDirectory doit correspondre à la racine de l'arborescence ugc externe au serveur d'application (contenant les répertoires conf, native et reftables).

Vous pouvez ajouter le paramètre `RefTablesDirectory` qui permet de spécifier un répertoire dans lequel vous pouvez avoir les tables de référence. Il sera de la forme : `RefTablesDirectory="D:\UGC\TablesRef"` .

Dans la section Server, il est nécessaire d'ajouter :

```
<Listener className="com.geoconcept.ugc.connect.tomcat.LifeCycleListener" resource="geoconcept/ugc/
default" />
```

L'ajout du `LifeCycleListener` n'est pas indispensable sous Windows mais est toutefois conseillé. Par contre sous Linux c'est indispensable (sinon le tomcat hôte ne s'arrête pas correctement).

Dans le cas où le nom logique utilisé n'est pas celui par défaut (geoconcept/ugc/defaut) adapter la configuration.

Création des liens ressources / webapps dans Tomcat

Pour Tomcat, il est nécessaire, pour les webapps qui utilisent une ressource, de la déclarer au niveau du descripteur de contexte de webapp. Pour Tomcat 7 et 8 les fichiers de configuration de webapp peuvent être externalisés de conf/server.xml et être placés dans conf/catalina/localhost (par défaut).

Donc pour chaque webapp utilisant le fournisseur primaire (référencé via JNDI), comme ugc-admin ou ugc-samples, des fichiers de contexte sont fournis (pour Tomcat 7 et 8), de ce fait pour chaque webapp devant être déployée il faut :

- copier le war dans /webapps
- copier le fichier xml de description de contexte dans /conf/catalina/localhost

De même, si vous créez votre propre webapp qui utilise le fournisseur primaire, vous devez déclarer ce lien :

- par un fichier séparé pour Tomcat 7 et 8

Pour Tomcat 7 et 8 :

- Copier le fichier ugc-admin.xml qui est dans %livrable%\tomcat\conf dans le répertoire %tomcat%/conf/catalina/localhost ;
- Copier le fichier axis.xml qui est dans %livrable%\tomcat\webapps dans le répertoire %tomcat%/conf/catalina/localhost (utile uniquement pour utiliser Universal Geocoder JEE est mode d'accès distant).

Pour définir d'autres fichiers de contexte (fichier.xml) pour d'autres applications web, il est nécessaire d'ajouter les lignes suivantes :

```
<ResourceLink
        global="geoconcept/ugc/default"
        name="geoconcept/ugc/default"
        type="com.geoconcept.ugc.connect.tomcat.ConnectionFactory"
/>
```

Axis

Cette partie de l'installation est utile uniquement pour utiliser Universal Geocoder JEE en mode d'accès distant.

Copier le répertoire axis qui est dans install\tomcat\webapps dans le répertoire %tomcat%\webapps.

> La documentation d'installation axis est accessible : http://ws.apache.org/axis/java/install.html#webapp. Pour tester axis, appeler http://localhost:8080/axis puis valider.
>
> HappyAxis est un auto-test. Vérifier qu'il n'indique pas de problème de librairies.

Les librairies optionnelles manquantes ne sont pas gênantes (mail, attachements, xml security, etc).

Installation AddressFinderService

Dans %tomcat%/webapps/axis/WEB-INF/lib, copier le fichier ugc-axis.jar qui est dans install\tomcat\axis-std.

Déploiement du service AddressFinder :

1. Lancer Tomcat,
2. Ouvrir une fenêtre ligne de commande dans le répertoire \install\tomcat\axis-std puis exécuter deploy (si le port du serveur n'est pas 8080, modifier le fichier deploy.bat. Faire de même pour l'hôte si ce n'est pas localhost).

> Les librairies Axis doivent être dans le CLASSPATH.

Si le service est déployé correctement, le serveur répond :

```
...
Processing file AddressFinderServiceAxis.wsdd
<Admin>Done processing</Admin>
```

Dans le cas contraire, il indique une erreur.

Configuration de JBoss

Il existe deux modes de déploiement :

- Soit l'adaptateur de ressource est déployé en standalone (seul) et dans ce cas le service est accessible à tout autre module déployé sur la même instance Jboss,
- Soit l'adaptateur de ressource est déployé dans une application d'entreprise qui contiens des modules de publication et dans ce cas le service n'est pas accessible à un autre module déployé sur la même instance Jboss (il s'agit en quelque sorte d'un pré-déploiement complet).

Adaptateur standalone

- Copier le fichier ugc.rar qui est dans %livrable%\jboss\standalone\ra\deploy dans le répertoire %jboss %/deploy,
- Copier le fichier ugc-ds.xml qui est dans %livrable%\jboss\standalone\ra\deploy dans le répertoire %jboss%/deploy ; puis éditez le afin de modifier la valeur de RootDirectory,
- Copier le fichier ugc-common.jar qui est dans %livrable%\jboss\standalone\ra\lib dans le répertoire %jboss%/lib.

Sous forme d'archive d'application d'entreprise (EAR)

- Copier le fichier ugc-all.ear qui est dans %livrable%\jboss\standalone\ra\deploy dans le répertoire %jboss%/deploy,
- Copier le fichier ugc-ds.xml qui est dans %livrable%\jboss\standalone\ra\deploy dans le répertoire %jboss%/deploy ; puis editez le afin de modifier la valeur de RootDirectory.

Configuration de Jonas

L'adaptateur de ressource (ugc.rar) se déploie en mode standalone (seul), le service est accessible à tout autre module déployé sur la même instance Jboss (qu'il s'agisse d'un module fourni comme ugc-admin ou ugc-ws ou un module spécifique écrit par l'utilisateur).

La procédure de déploiement et de configuration suit les préconisations Jonas : il faut modifier la configuration du module puis le déployer via une des méthodes proposées par Jonas.

Livrable

Si le livrable est sous la forme d'une archive (.zip ou .tar.gz) il faut l'extraire dans un répertoire de travail (par exemple via « tar xvfz ugc-jee-5.0.112.tar.gz » sous linux), si il sous forme explosée le recopier dans un répertoire de travail.

Modifier la configuration de ugc.rar pour l'adapter à l'installation locale

Il est nécessaire de modifier la configuration de l'adaptateur de ressource UGC (ugc.rar situé dans jonas/standalone/ra/deploy) afin d'indiquer le chemin d'installation local.

Pour ce faire, la procédure Jonas est la suivante : il faut extraire le fichier jonas-ra.xml de l'archive (.rar) puis le modifier et enfin mettre à jour l'archive avec le fichier modifié. L'outil RAConfig de Jonas permet

de générer le fichier jonas-ra.xml à partir du fichier ra.xml standard, mais içi cela n'a pas d'intérêt car le fichier jonas-ra.xml est fourni. Donc l'outil jar du jdk java suffit pour extraire et mettre à jour l'archive. Si JAVA_HOME/bin n'est pas dans le PATH il faut l'ajouter temporairement (ou bien invoquer l'outil jar en incluant ce répertoire, eg /etc/java/jdk1.5.0_22/bin/jar)

1. Extraction du fichier jonas-ra.xml : jar xvf ugc.rar META-INF/jonas-ra.xml. Cela permet de créer le sous-répertoire META-INF et le fichier jonas-ra.xml à l'intérieur,
2. Edition du fichier jonas-ra.xml : modifier le fichier jonas-ra.xml extrait afin d'indiquer dans la valeur de la propriété RootDirectory le répertoire dans lequel UGC a été installé. Si vous souhaitez utiliser un répertoire de tables de référence différent du répertoire par défaut (%ugc%/reftables) indiquez le dans la propriété optionelle RefTablesDirectory, sinon ce n'est pas nécessaire.
3. Mise à jour de l'archive (.rar) : jar uvf ugc.rar META-INF/jonas-ra.xml

L'archive mise à jour est prête a être déployée sur une instance Jonas.

Déploiement sur Jonas 4.x

Le service ressource de Jonas doit être configuré et démarré au lancement (ce qui est le cas dans la configuration par défaut). C'est à dire que dans le fichier jonas.properties la valeur ressource doit être présente dans la liste des valeurs de la propriété jonas.services.

Il existe plusieurs façons de déployer un fichier RAR sur une instance Jonas :

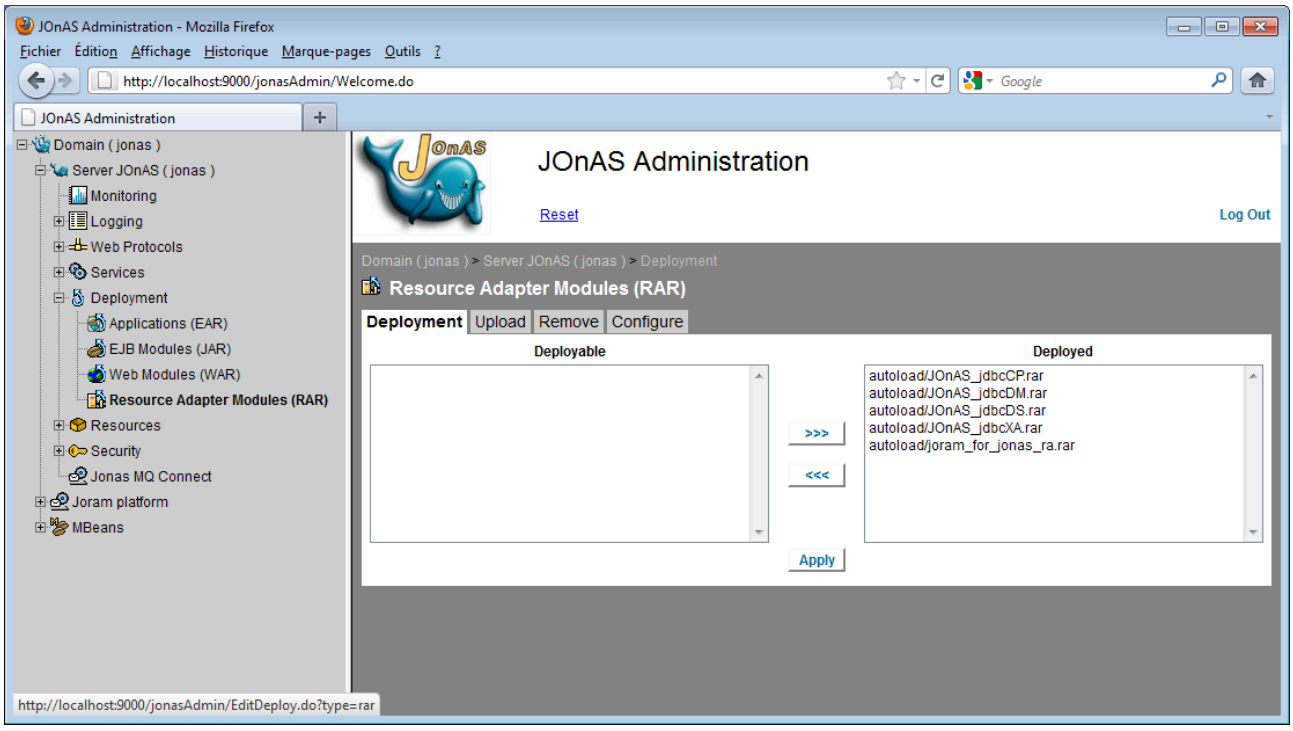
- modifier les valeurs de jonas.service.resource.resources property du fichier jonas.properties : pour y ajouter ugc (le suffixe .rar n'est pas obligatoire).Exemple : jonas.service.resource.resources ugc,
- placer le fichier dans le répertoire autoload des connecteurs, qui par défaut est $JONAS_BASE/rars/autoload,
- en ligne de commande via la commande jonas admin -a ugc.rar,
- en utilisant la console d'administration web (jonasAdmin) accessible par défaut via http://localhost:9000/jonasAdmin,

> Pour plus de détails consulter http://jonas.ow2.org/JOnAS_4_7/doc/PG_Connector.html#PG_Connector-Use

Le déploiement via le répertoire rars/autoload et le déploiement via jonasAdmin sont décrits ci-après.

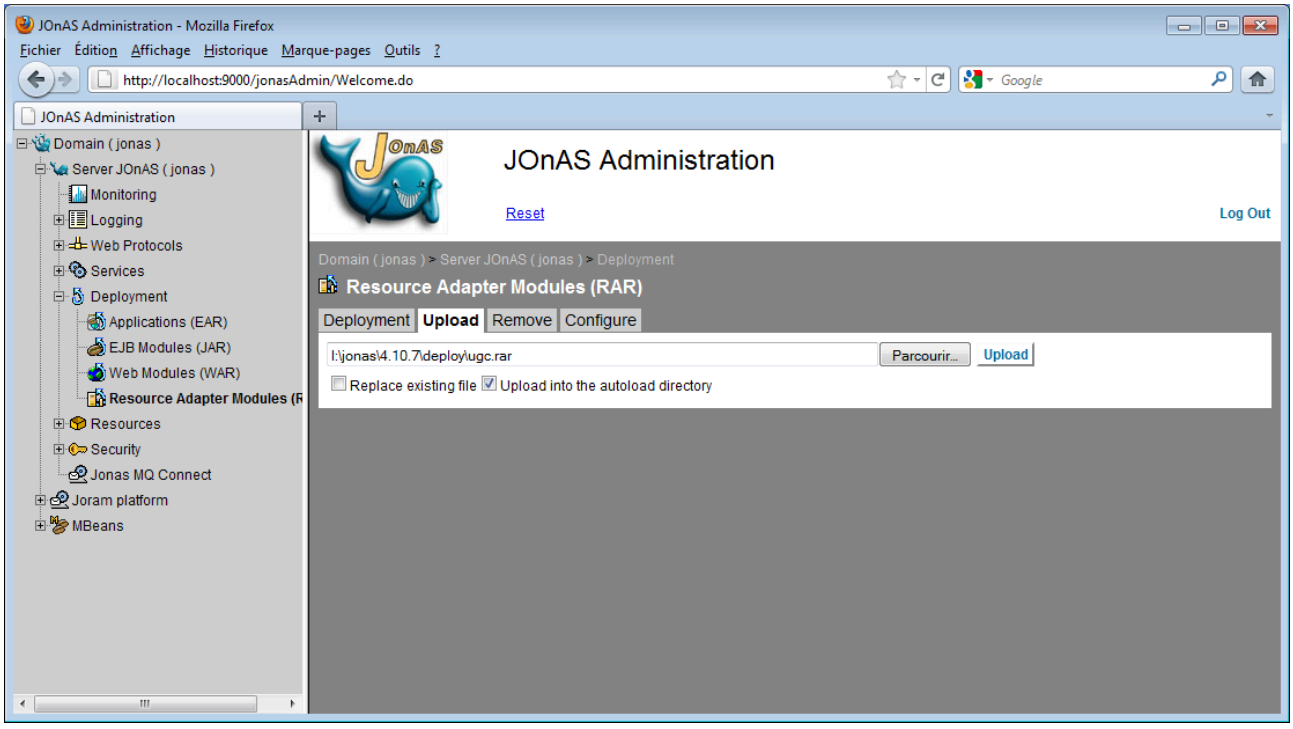
- Déploiement par le répertoire rars/autoload : copier le fichier ugc.rar préparé dans $JONAS_BASE/rars/autoload et de relancer Jonas.
- Déploiement via jonasAdmin
  - Sélectionner Deployment > Ressource Adapter Modules (RAR) dans l'arborescence ,

Jonas Deployment



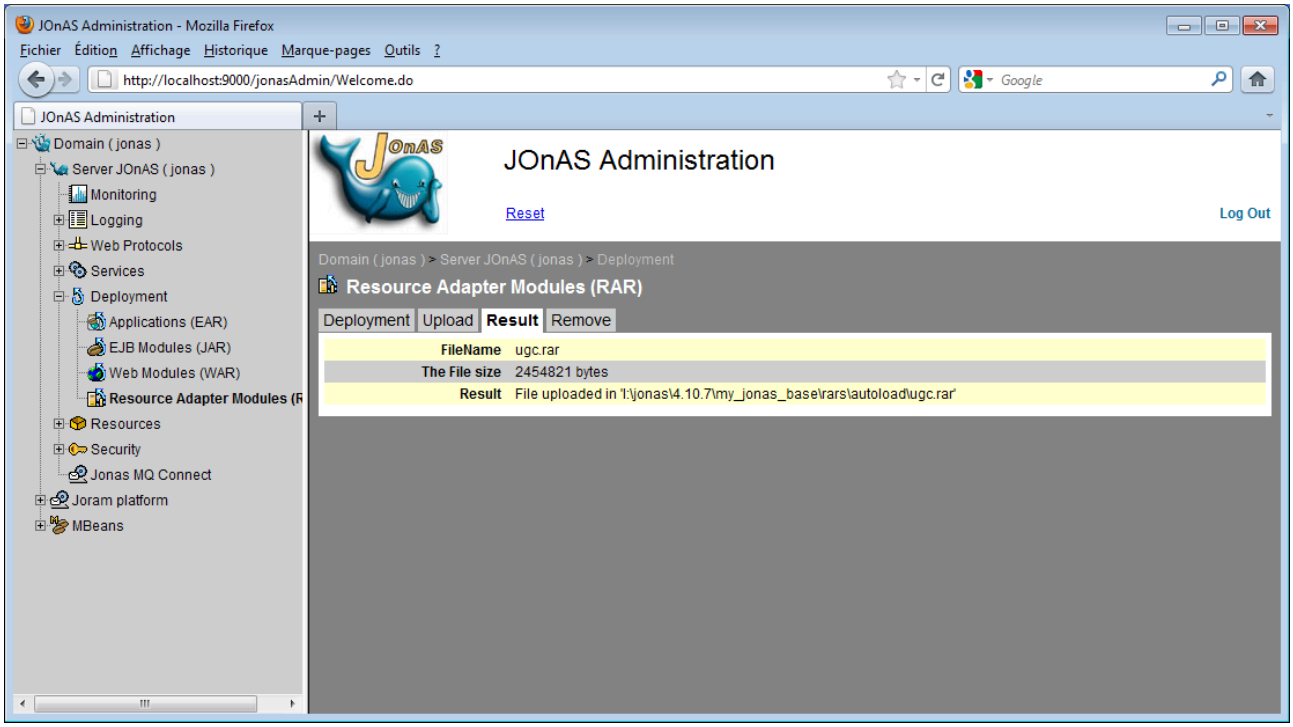
- Sélectionner l'onglet Uplodad,

Onglet Upload



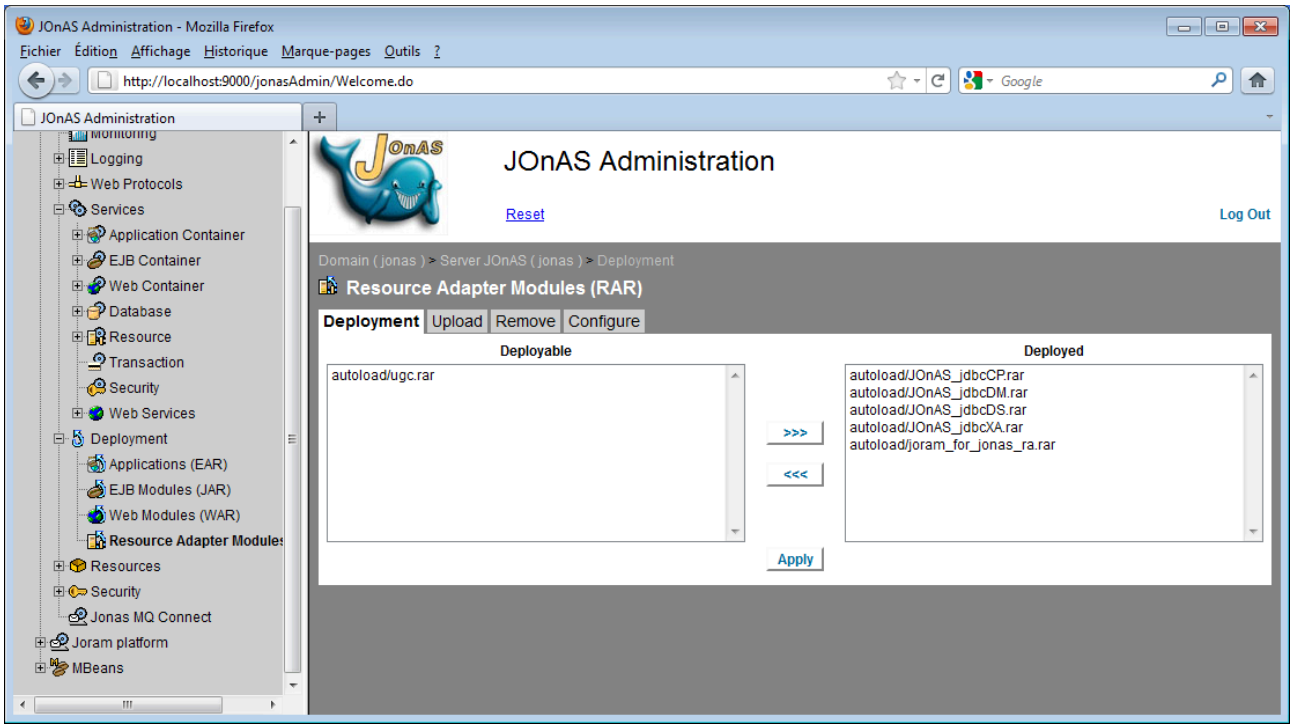
- Uploader le fichier depuis l'endroit où il a été préparé sur le disque. Cocher upload into the autoload directory sinon le déploiement n'aura lieu que pour la session courante de l'instance Jonas. Puis valider via le bouton Upload.

Onglet Résultat



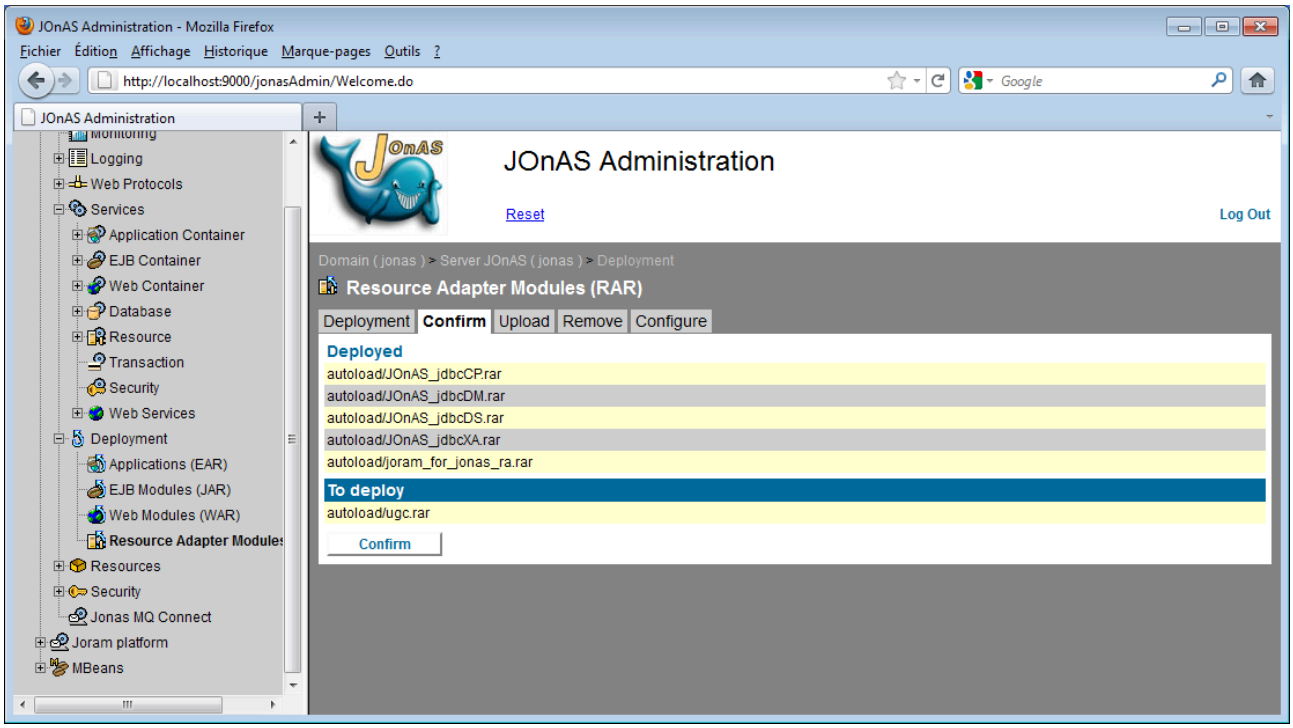
Cela aura pour effet de copier ugc.rar dans le répertoire JONAS_BASE/rars/autoload.

Onglet Deployment



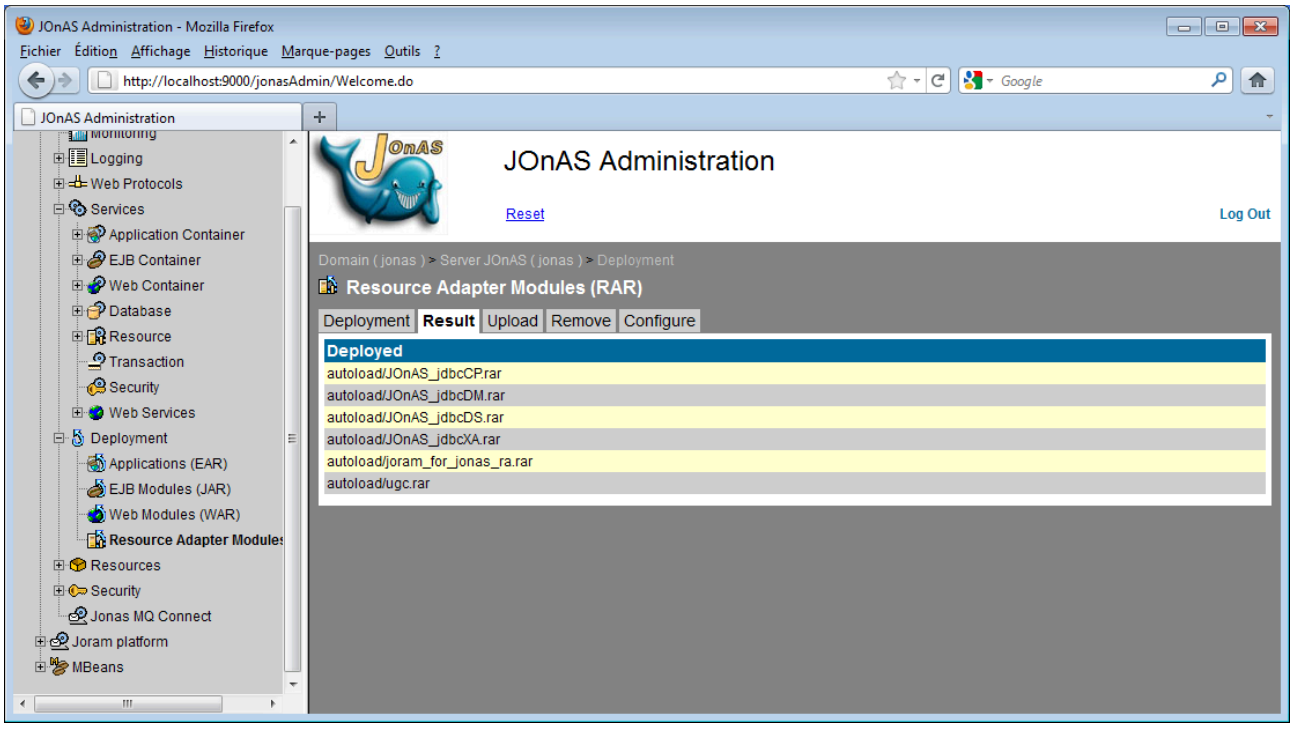
Le module est maintenant « deployable ».

Onglet Confirm



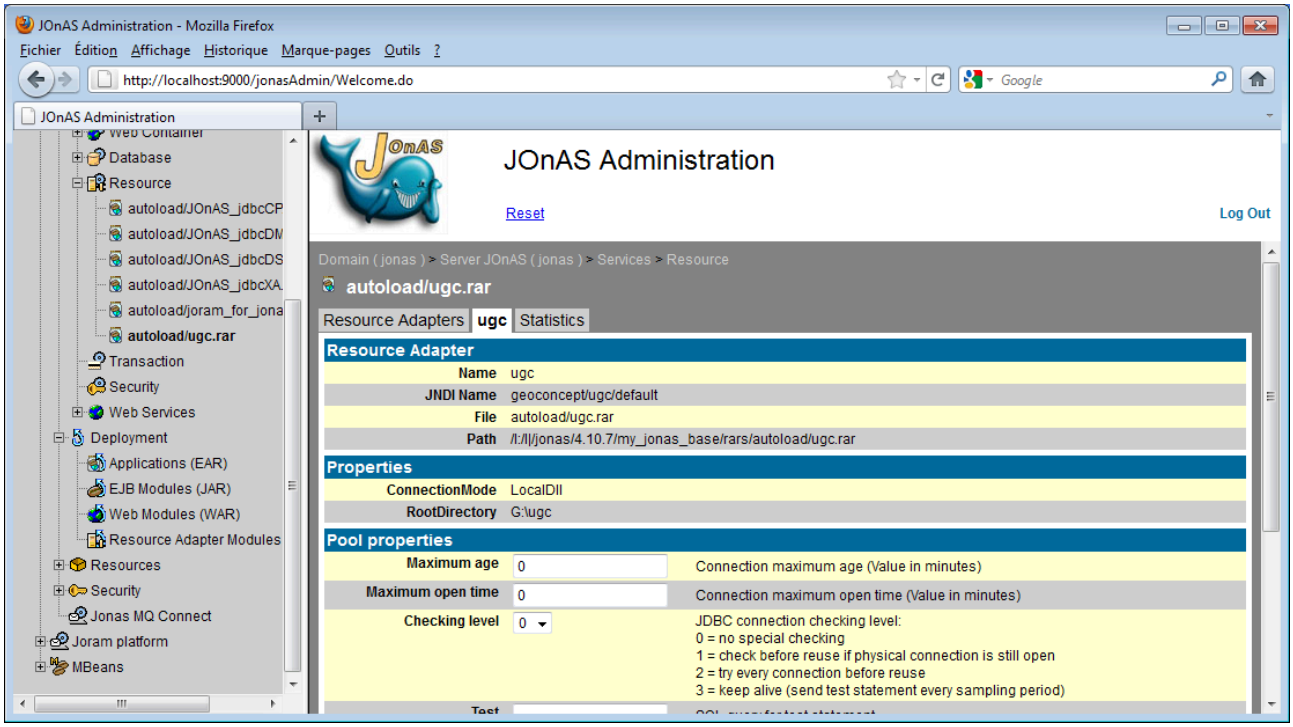
- D) Ajouter ugc.rar aux modules déployés et confirmer.

Onglet Résultat



Le module est désormais déployé. Au niveau de l'arborescence du serveur (à droite) il apparaît dans Services/Resource :

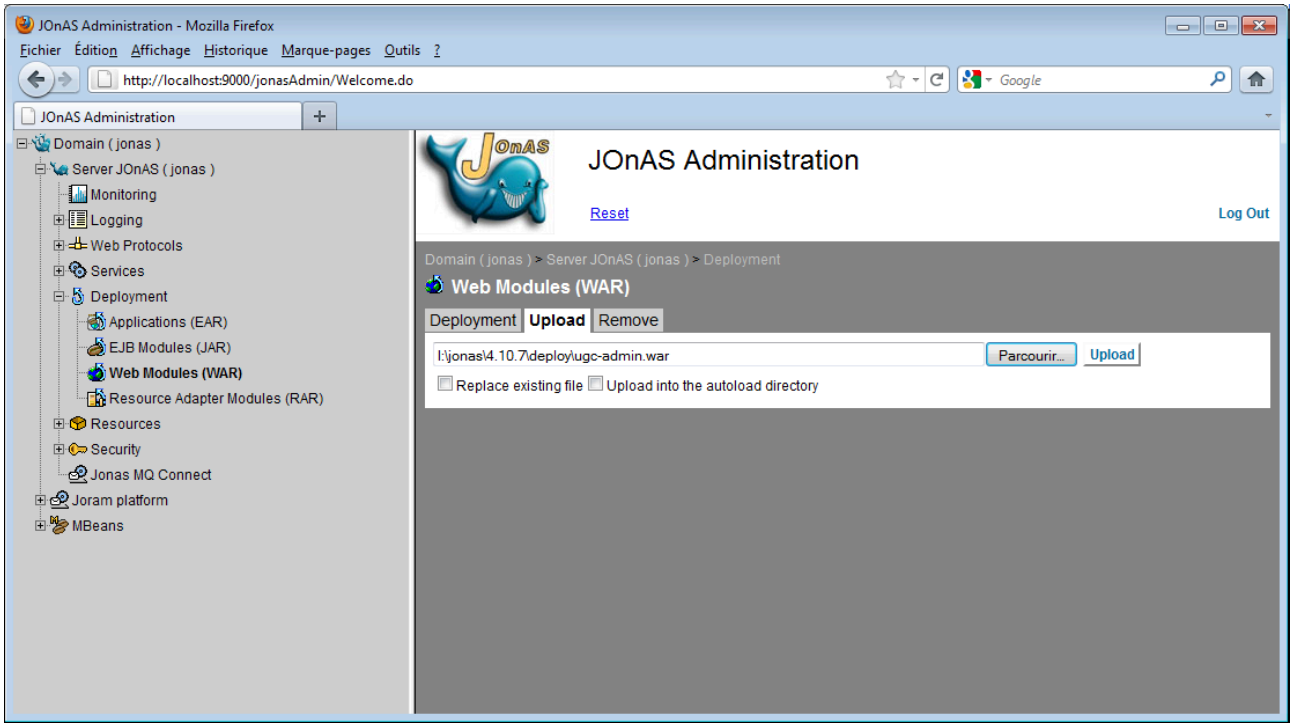
Onglet ugc



Déploiement de ugc-admin (optionnel) :

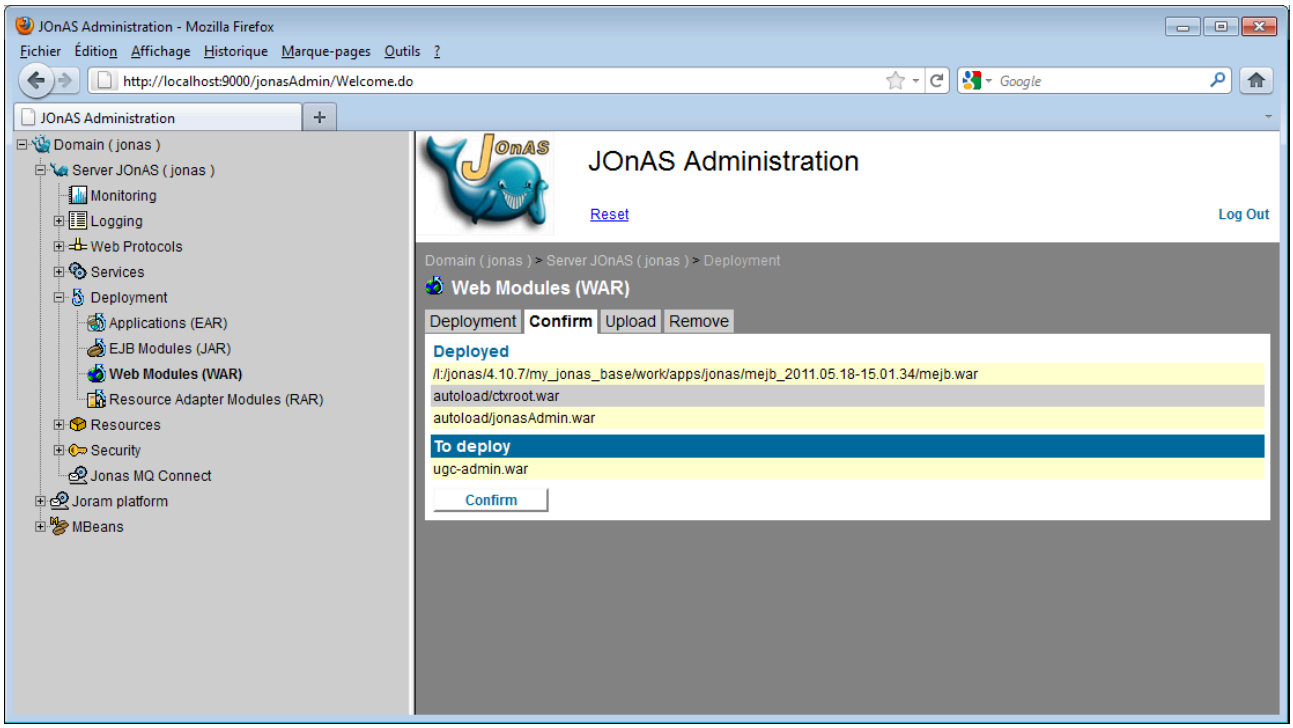
ugc-admin est une webapp d'administration qui permet (entre autres) de tester le bon fonctionnement d'ugc-jee. Son déploiement est optionnel. Le principe du déploiement est le même que pour l'adaptateur de ressource (ugc.rar) si ce n'est qu'il s'agit ici de la section Web Modules (WAR) et que ce module ne nécessite pas de configuration.

Upload du fichier ugc-admin.war

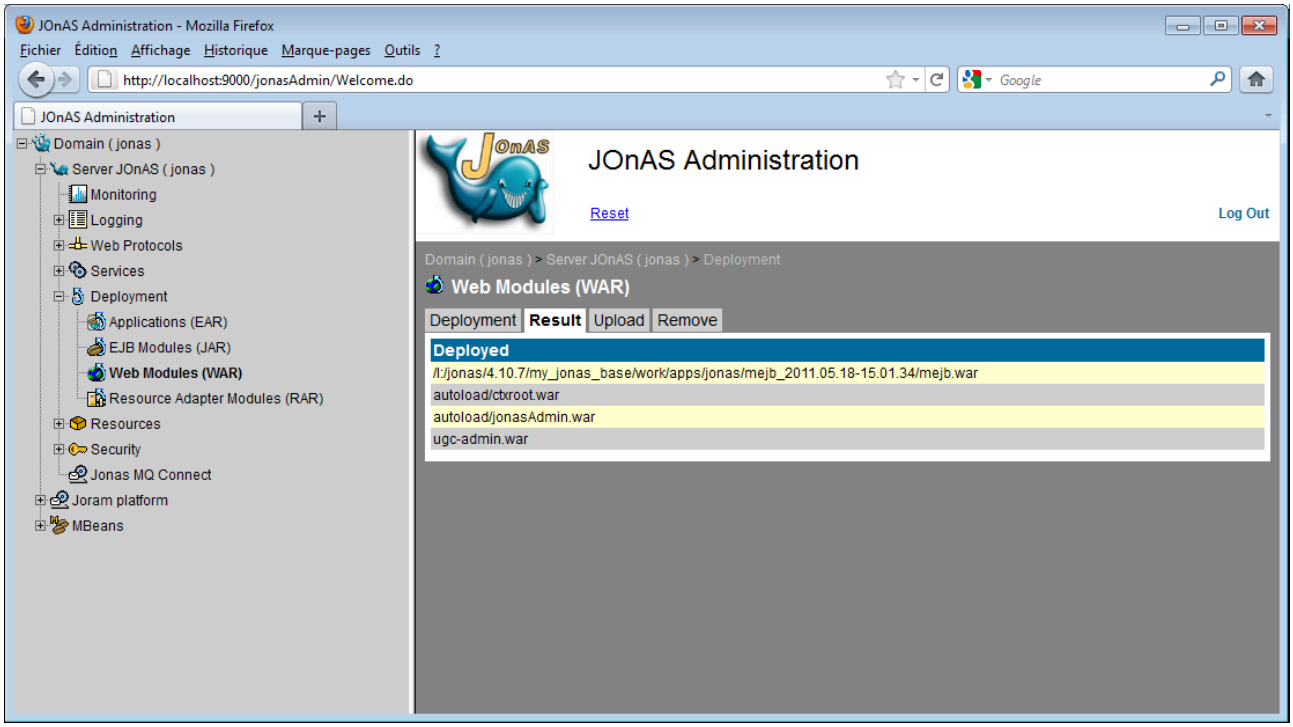


Puis déploiement :

Déploiement du fichier ugc-admin.war

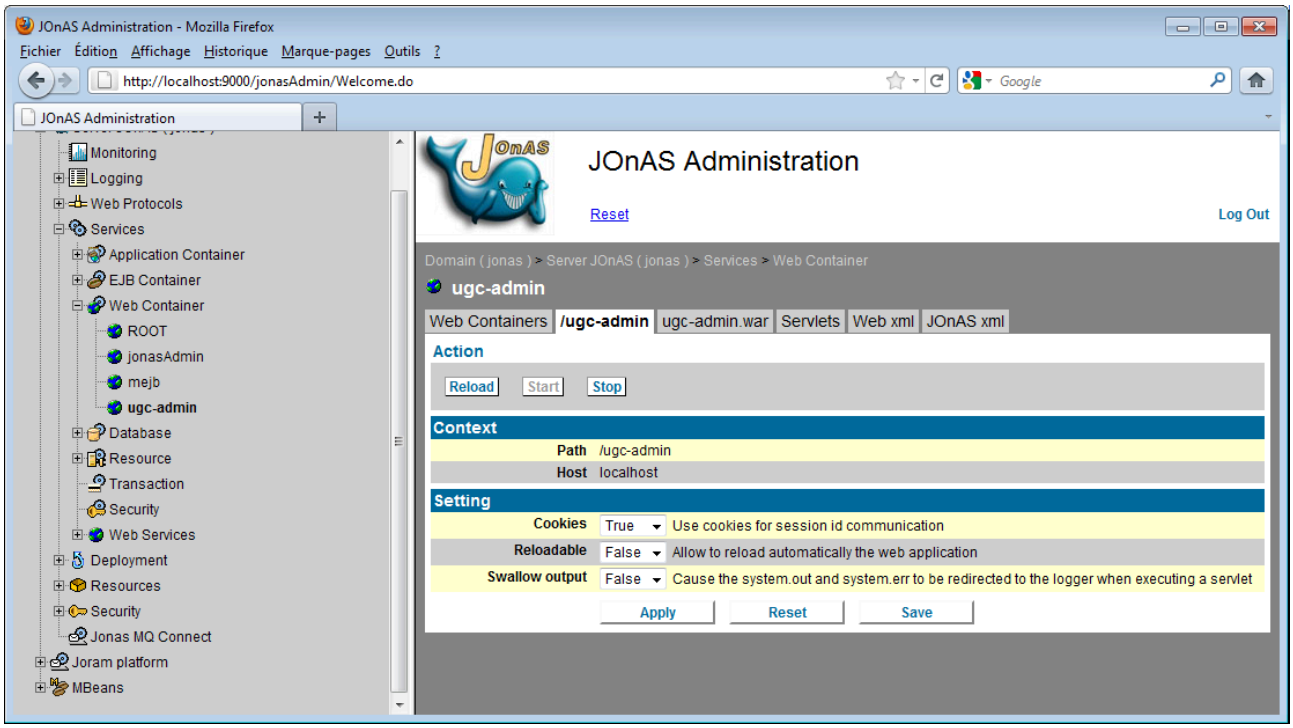


Déploiement du fichier ugc-admin.war



La webapp est déployée.

Déploiement du fichier ugc-admin.war



Montage d’un fournisseur client web service sur Jonas 4.x

Jonas 4.x ne fournissant pas d'implémentation saaj 1.3, il est nécessaire de placer la librairie saaj-impl-1.3.2.jar (qui peut être prise dans redist/providers-libs/ws-saas-client-saaj13/libs) ou une version plus récente dans JONAS_BASE/libs/apps.

Exemple de configuration de fournisseur de type client web service saas basé sur saaj (extrait du fichier jonas-ra.xml) :

```
<jonas-config-property>
        <jonas-config-property-name>ConnectionMode</jonas-config-property-name>
        <jonas-config-property-value>WssClient</jonas-config-property-value>
</jonas-config-property>
<jonas-config-property>
        <jonas-config-property-name>ServerUri</jonas-config-property-name>
        <jonas-config-property-value>http://owss.geoconcept.com:2010/service/soap11</jonas-config-property-
value>
</jonas-config-property>
<jonas-config-property>
        <jonas-config-property-name>Key</jonas-config-property-name>
        <jonas-config-property-value>123456</jonas-config-property-value>
</jonas-config-property>
```

Déploiement sur Jonas 5.x

Le déploiement est identique à Jonas 4.x excepté le fait que le répertoire $JONAS_BASE/deploy remplace les répertoires différenciés ($JONAS_BASE/rars/autoload, $JONAS_BASE/webapps/autoload) pour le déploiement.

Déploiement et configuration BES

L'environnement de déploiement décrit est Borland Enterprise Server, AppServer Edition version 5.2.1.

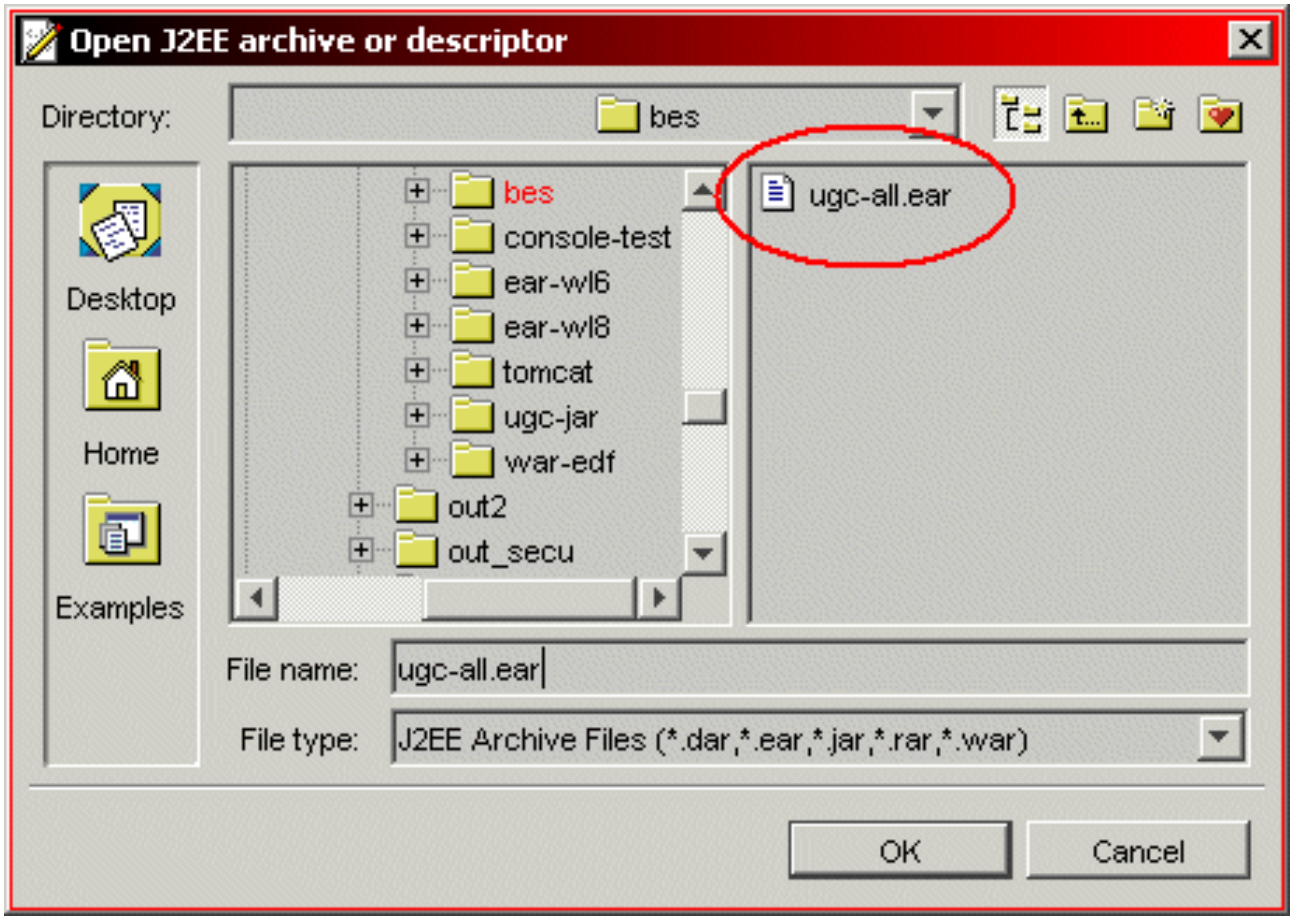
L'adaptateur de ressource (ugc.rar) et un ensemble de modules de publication (ugc-admin.war, ugc-samples.war, ugc-axis-fusion.war, etc) se déploient sous la forme d'une application d'entreprise (archive .ear) : ugc-all.ear.

Configuration de ugc-all.ear : avant de déployer ugc-all.ear, il est nécessaire de le configurer pour mettre à jour les chemins d'installation locaux.

Pour ce faire le plus simple est d'utiliser Borland Deployment Descriptor Editor (DDEditor).

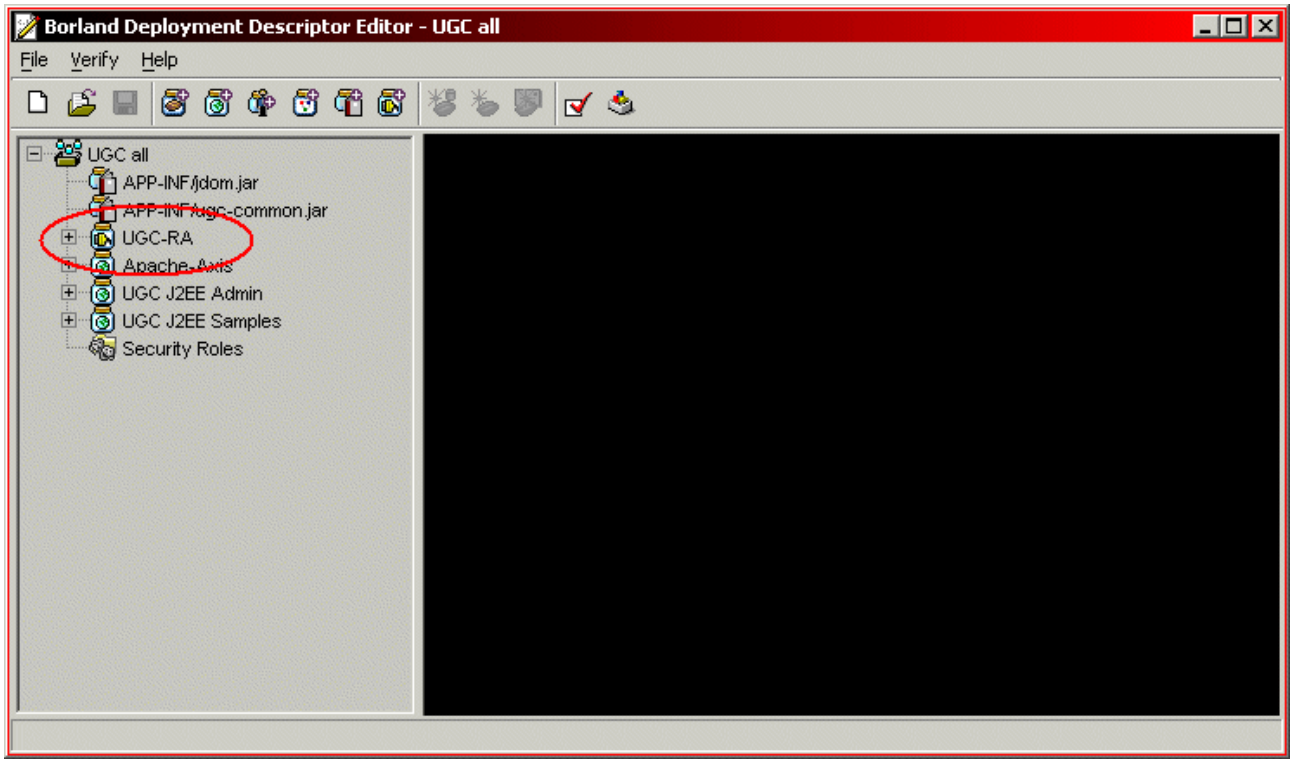
1. Exécutez ddeditor et ouvrez ugc-all.ear

ugc-all.ear



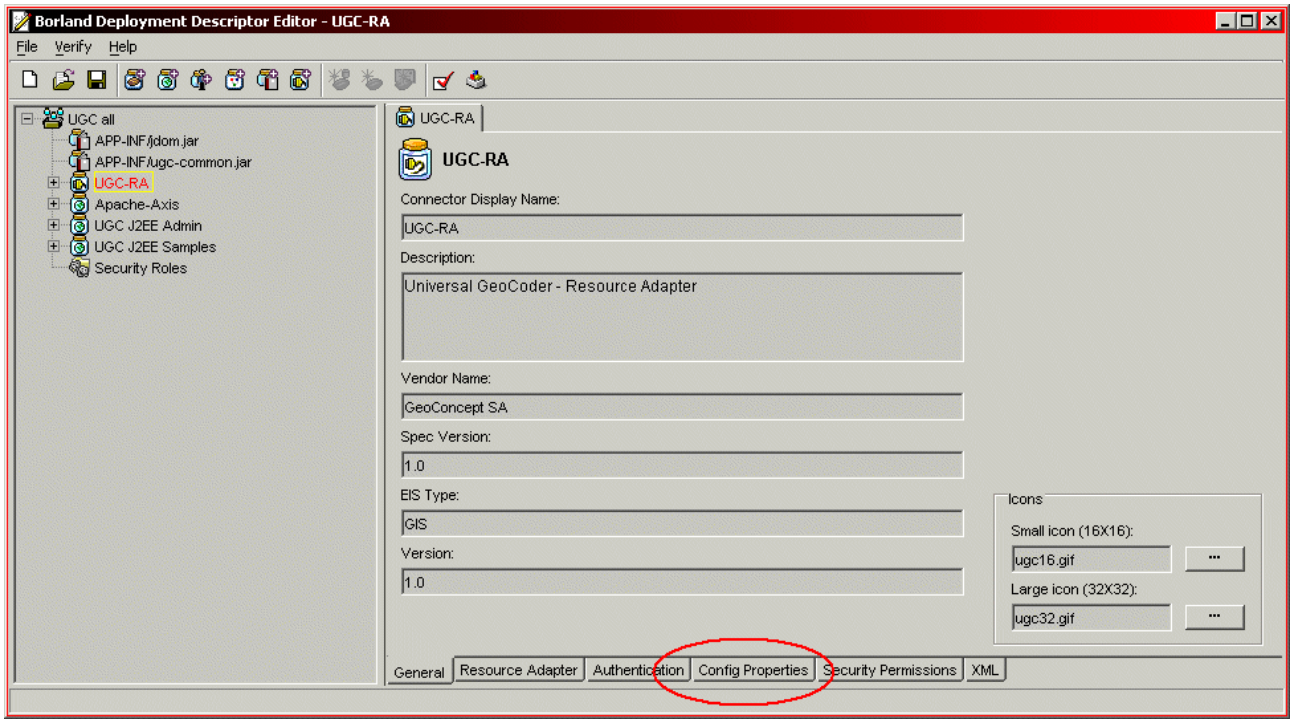
1. Sélectionnez le module adaptateur de ressource (rar) dans l'archive d'application d'entreprise (ear)

Module adaptateur de ressource



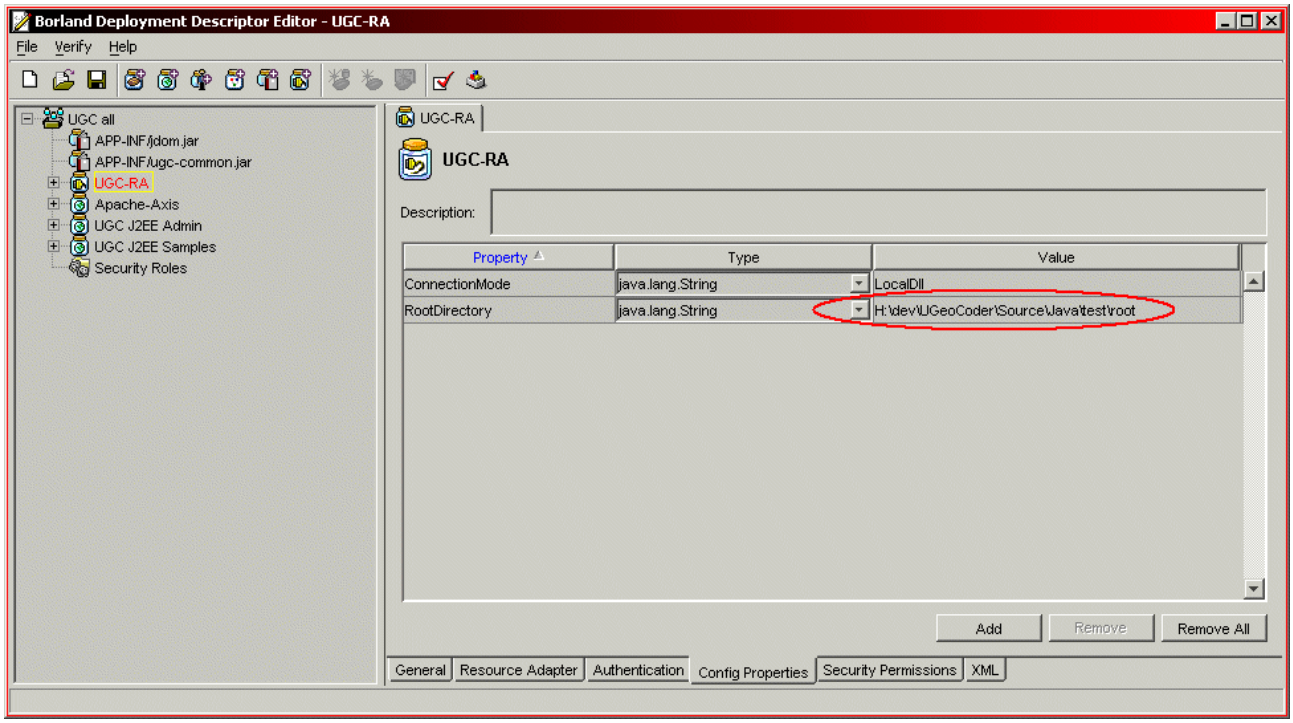
1. Sélectionnez l'onglet config properties du module adaptateur de ressource

Onglet config properties



1. Indiquez le répertoire d'installation des fichiers externes au serveur d'application

Onglet config properties

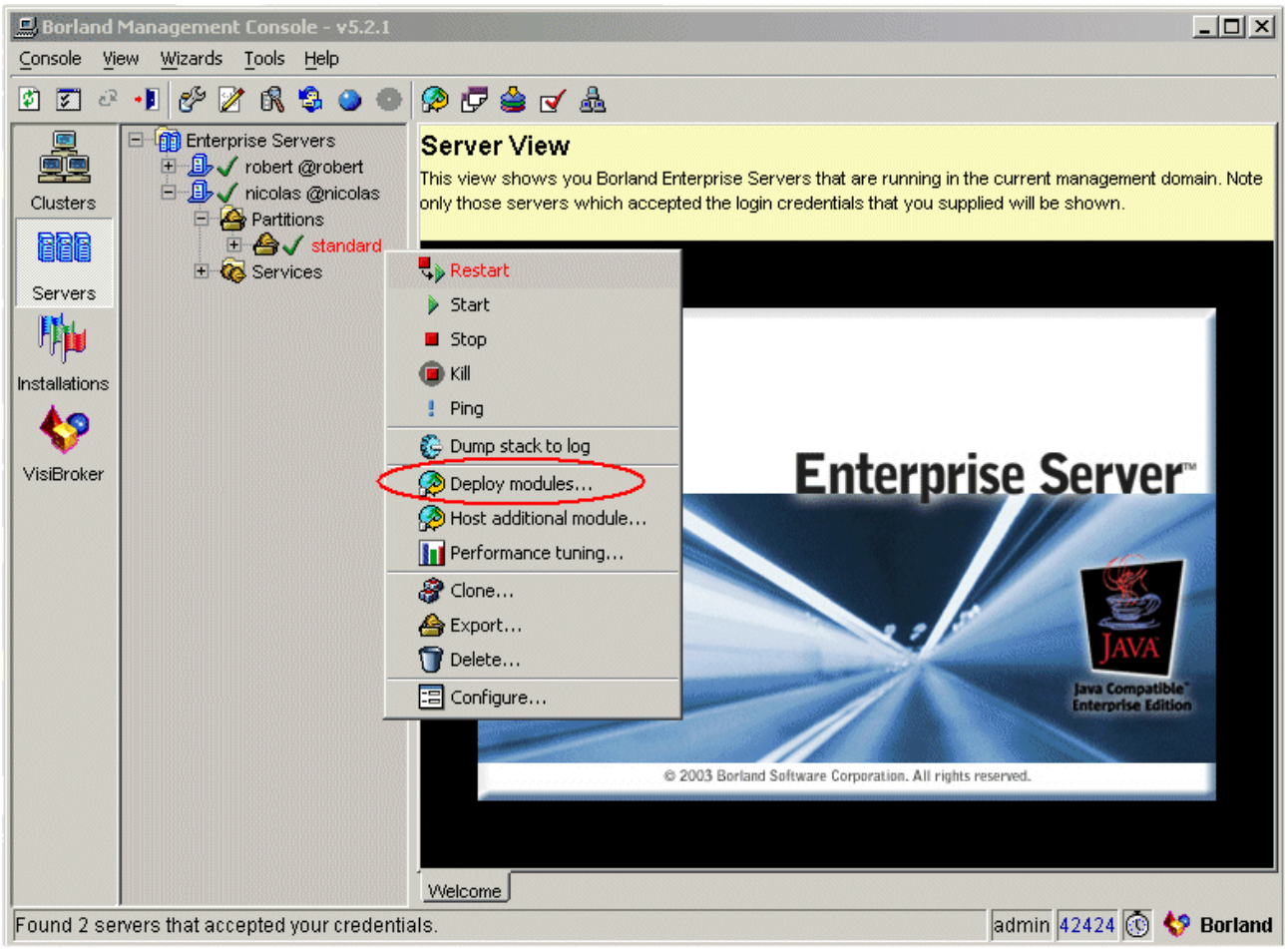


1. Sauvegarder les modifications effectuées

Déploiement de l'application d'entreprise (ugc-all.ear) dans Borland Management Console

Sélectionnez la partition sur laquelle vous voulez déployer :

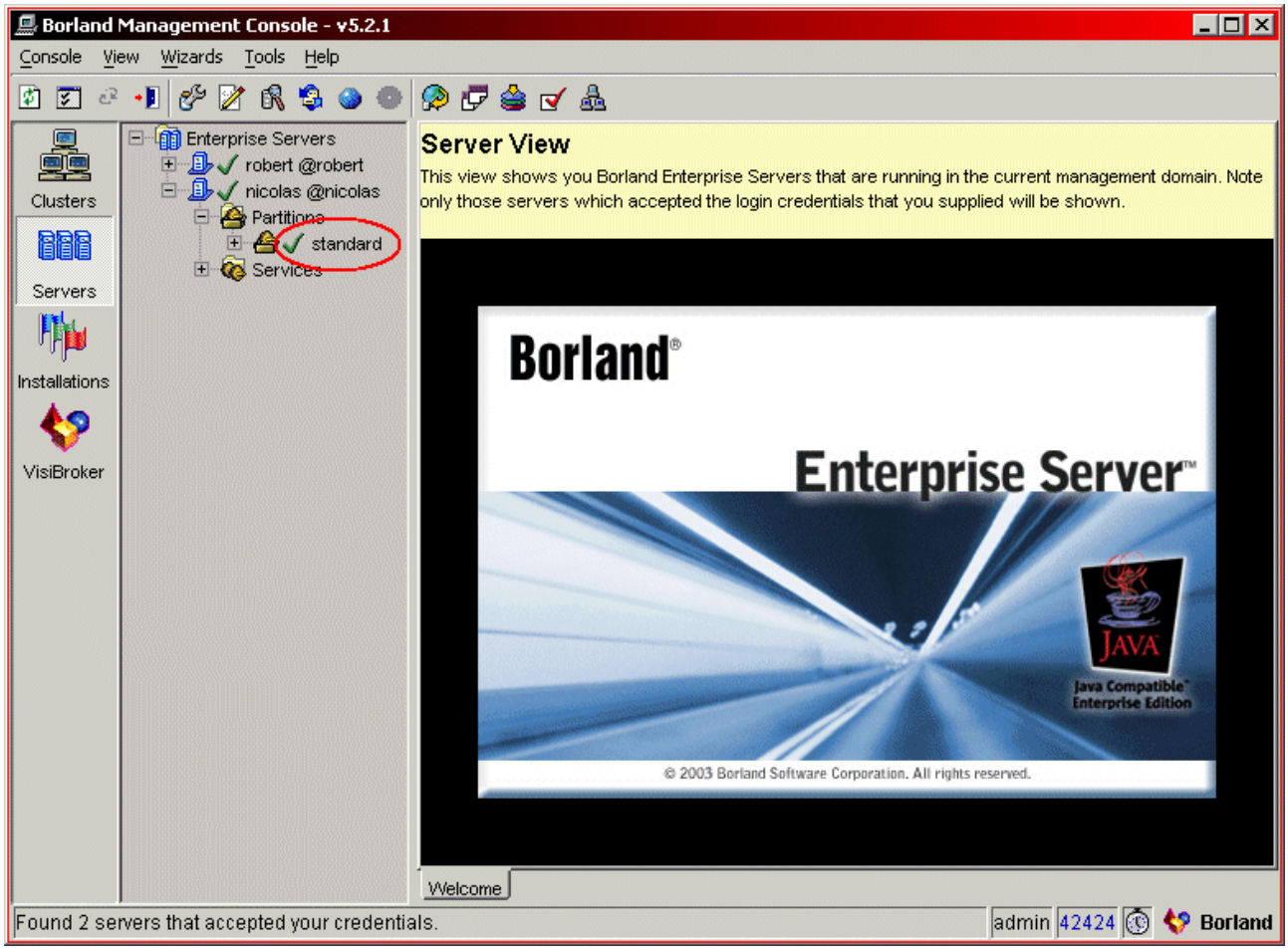
Sélection de la partition



> VisiConnect doit être activé dans les PartitionServices de la partition cible.

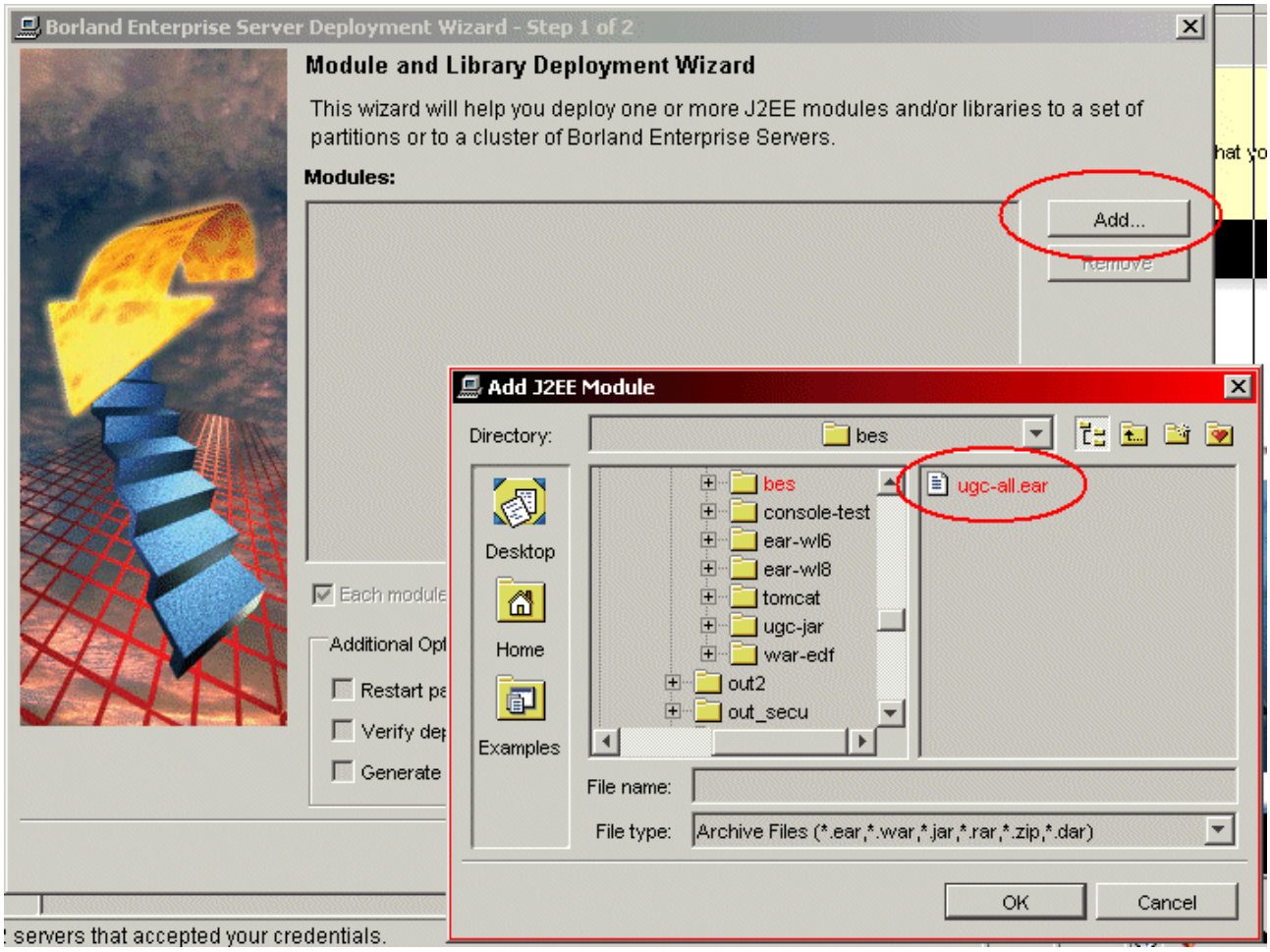
Effectuez un clic droit sur la partition et choisissez « Deploy modules … »

Deploy modules



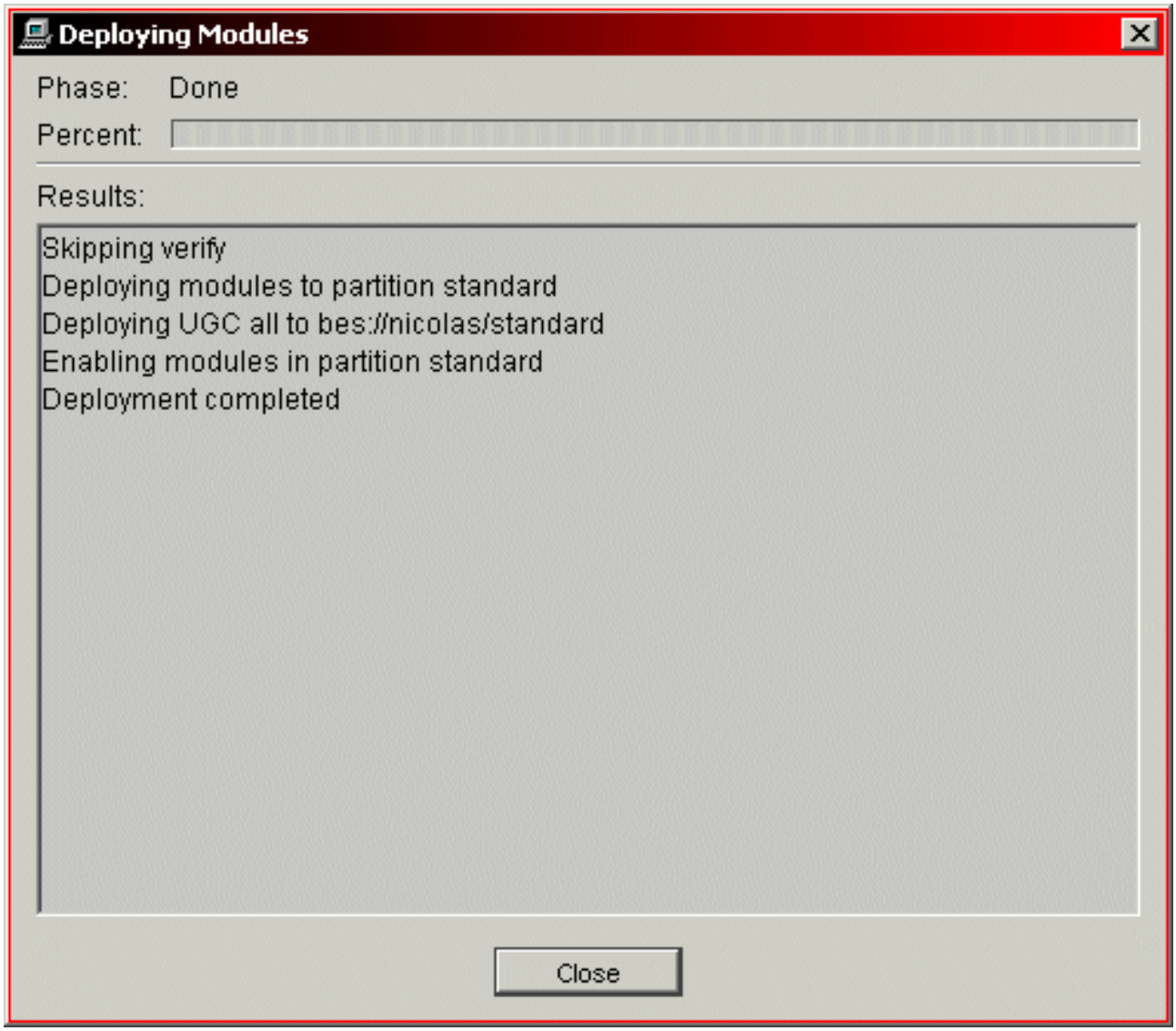
Sélectionnez le fichier ugc-all.ear que vous avez modifié à l'étape précédente et validez :

Validez le fichier



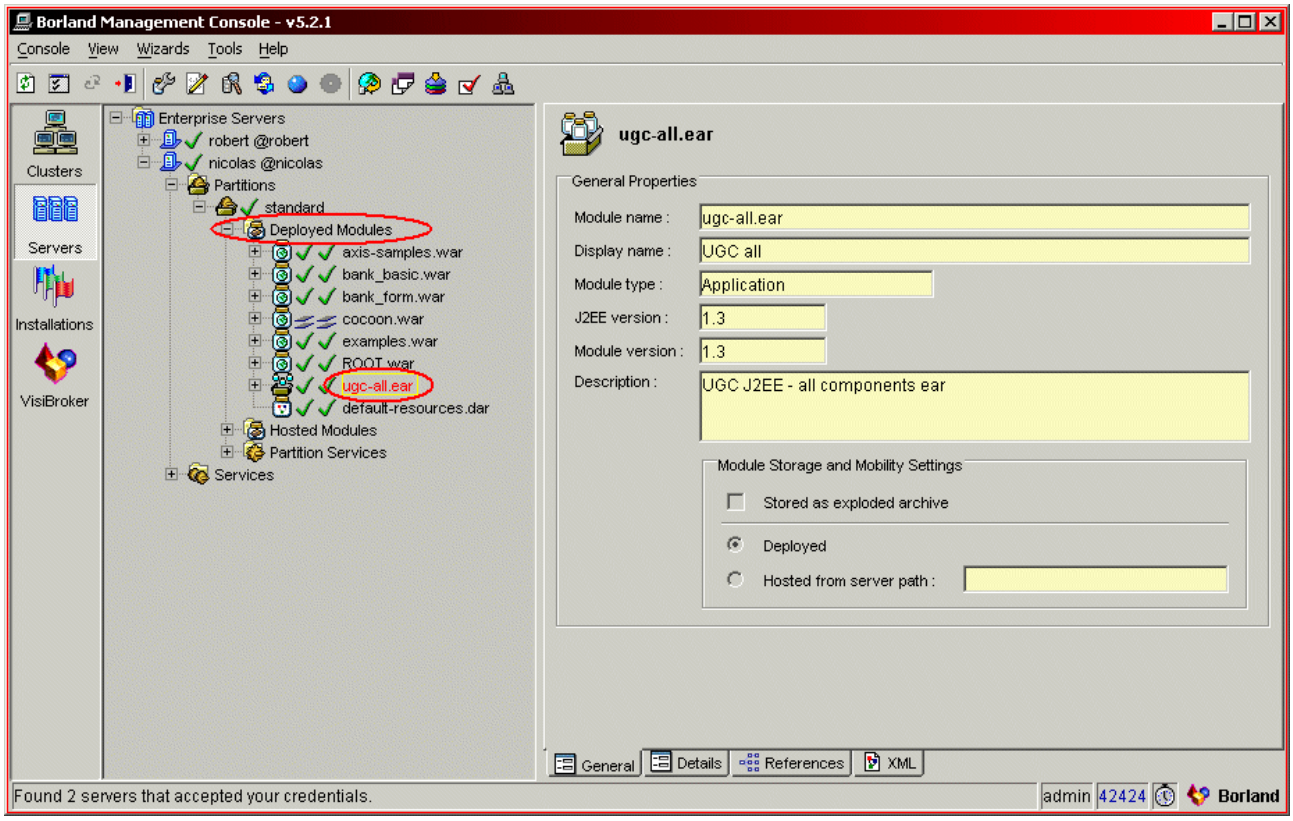
Sélectionner finish et le déploiement se termine :

Déploiement



L’application ugc-all apparaît maintenant dans la liste des modules déployés pour la partition sélectionnée :

Liste des modules déployés



Déploiement et configuration WebSphere

L'environnement de déploiement décrit est IBM WebSphere Application Server version 8.5.

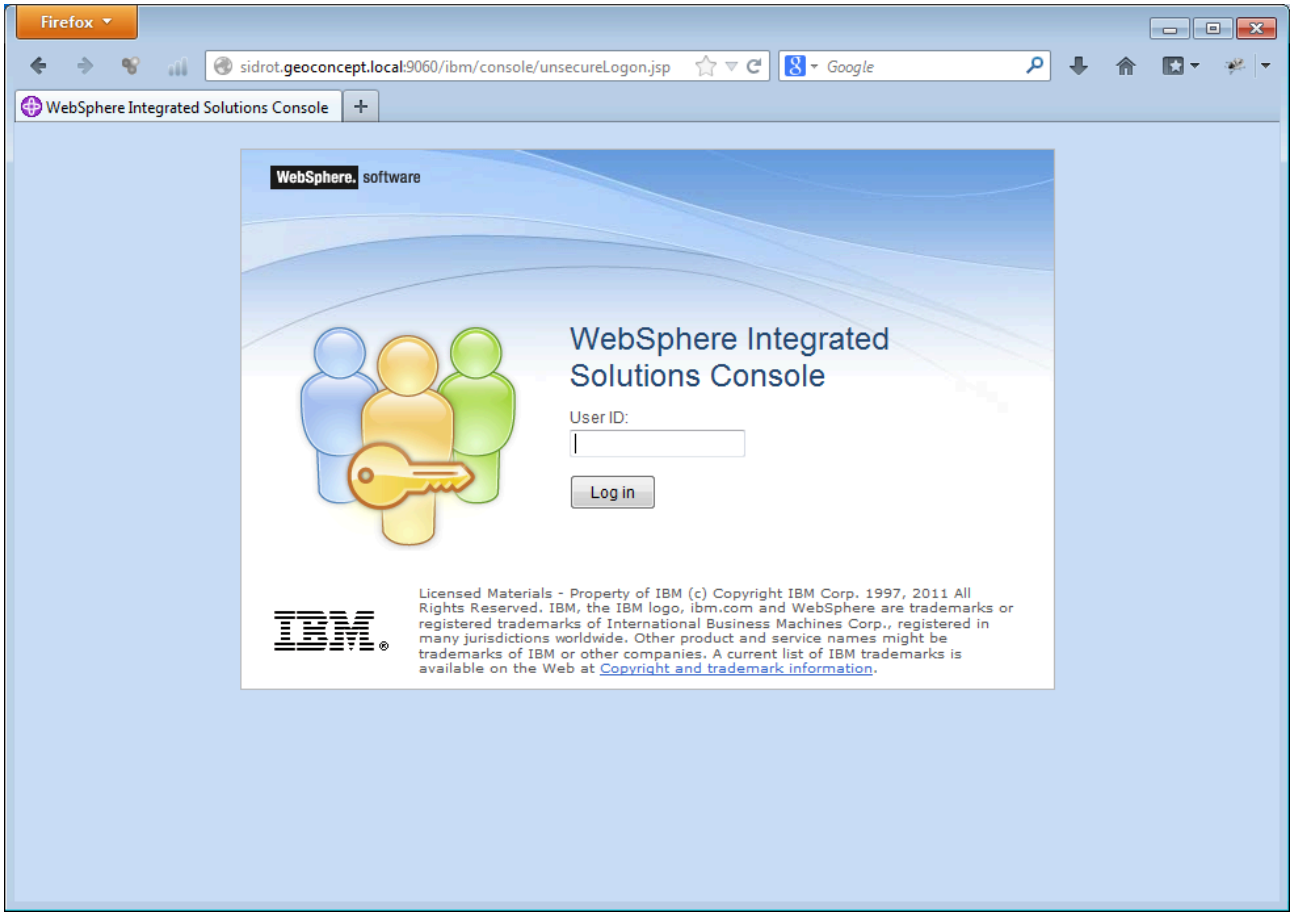
L'adaptateur de ressource (ugc.rar) et un ensemble de modules de publication optionnels (ugc-admin.war, ugc-samples.war, ugc-axis-fusion.war, etc) se déploient sous la forme de modules séparés (standalone). Les modalités de déploiement d'un module utilisateur qui consommerait les services de l'adaptateur de ressource ugc.rar sont les mêmes que ceux des modules optionnels fournis. Aussi le déploiement du module optionel ugc-admin sera décrit içi.

ugc-admin est un module optionel mais il permet en outre de valider le bon déploiement de l'adaptateur de ressource.

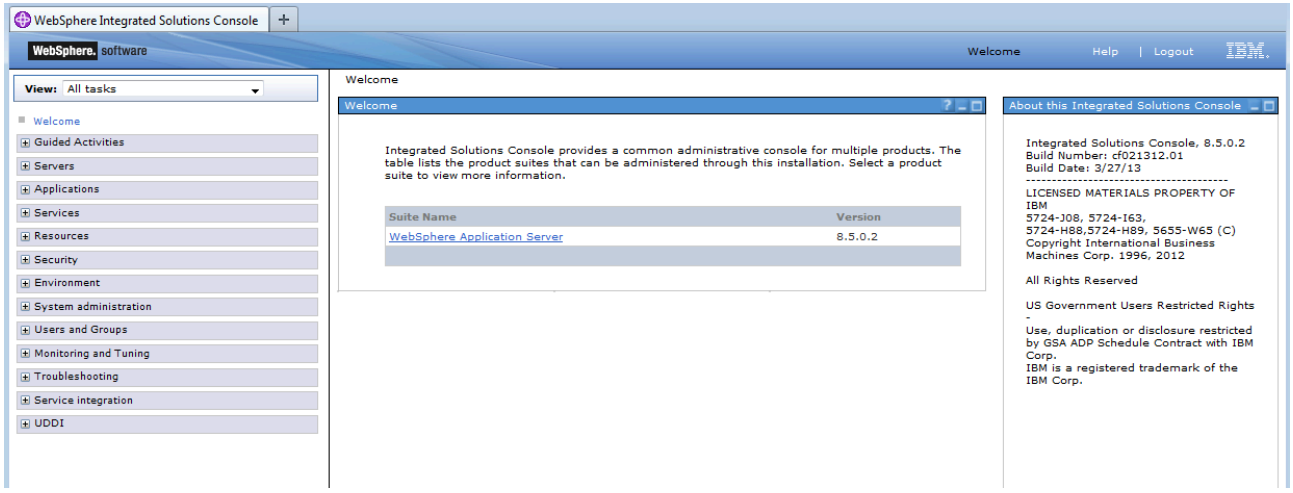
Il exite de nombreuses manieres de réaliser des déploiements dans WebSphere Application Server (par scripting, programmatiquement, par fichiers de propriétés, via wsadmin, etc), içi le processus de déploiement décrit sera celui de WebSphere Integrated Solutions Console (aussi appelé administrative console).

1. Déploiement de l'adaptateur de ressource (ugc.rar) : se logger sur WebSphere Integrated Solutions Console

Déploiement de l'adaptateur de ressource

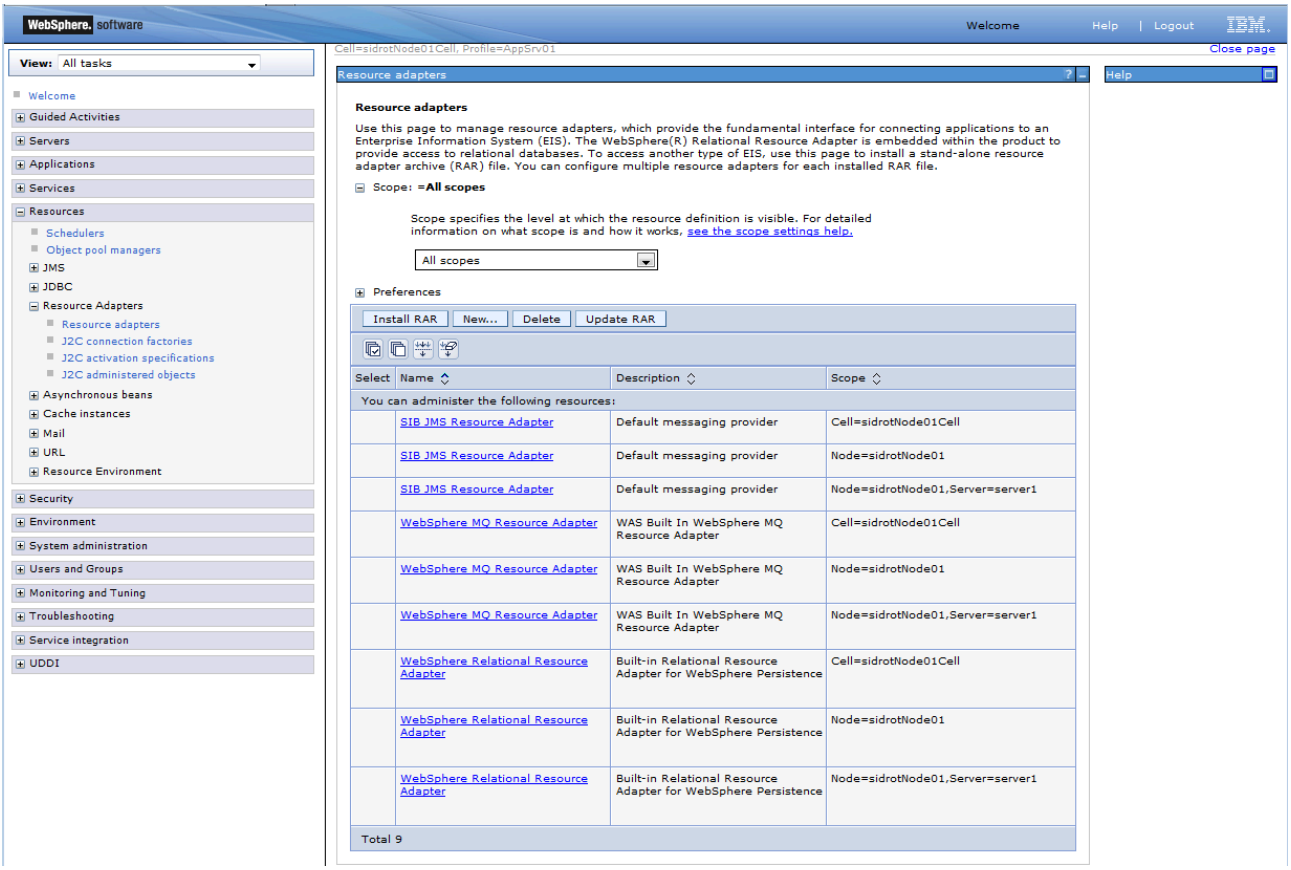


Déploiement de l'adaptateur de ressource



Aller dans Resources > Resource Adapters > Resource adapters (menu à gauche).

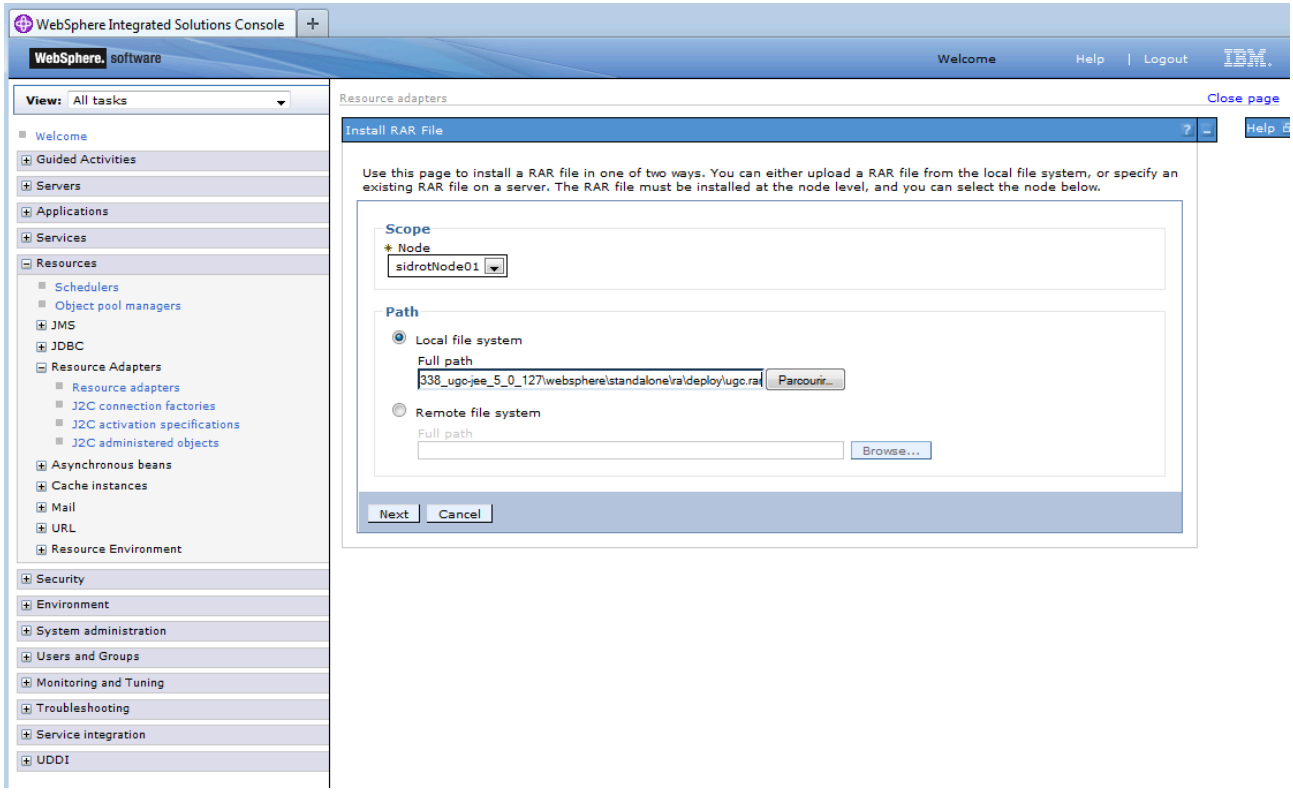
Déploiement de l'adaptateur de ressource



> Dans notre cas, Preferences/"show built-in resources" est coché (c'est pour cette raison qu'on voit les adaptateurs built-in).

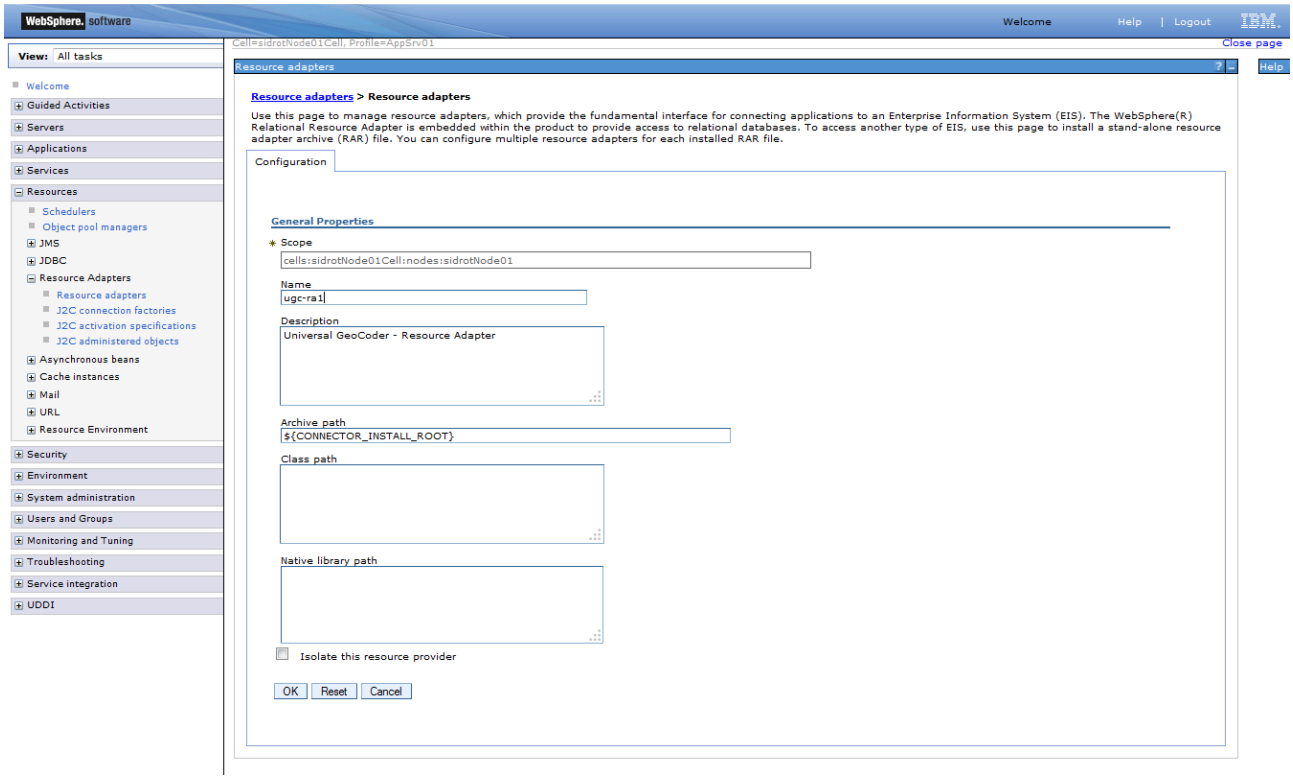
Cliquer sur "Install RAR" et sélectionner le fichier "ugc.rar" situé dans le sous répertoire websphere/standalone/ra/deploy du livrable

Installation RAR

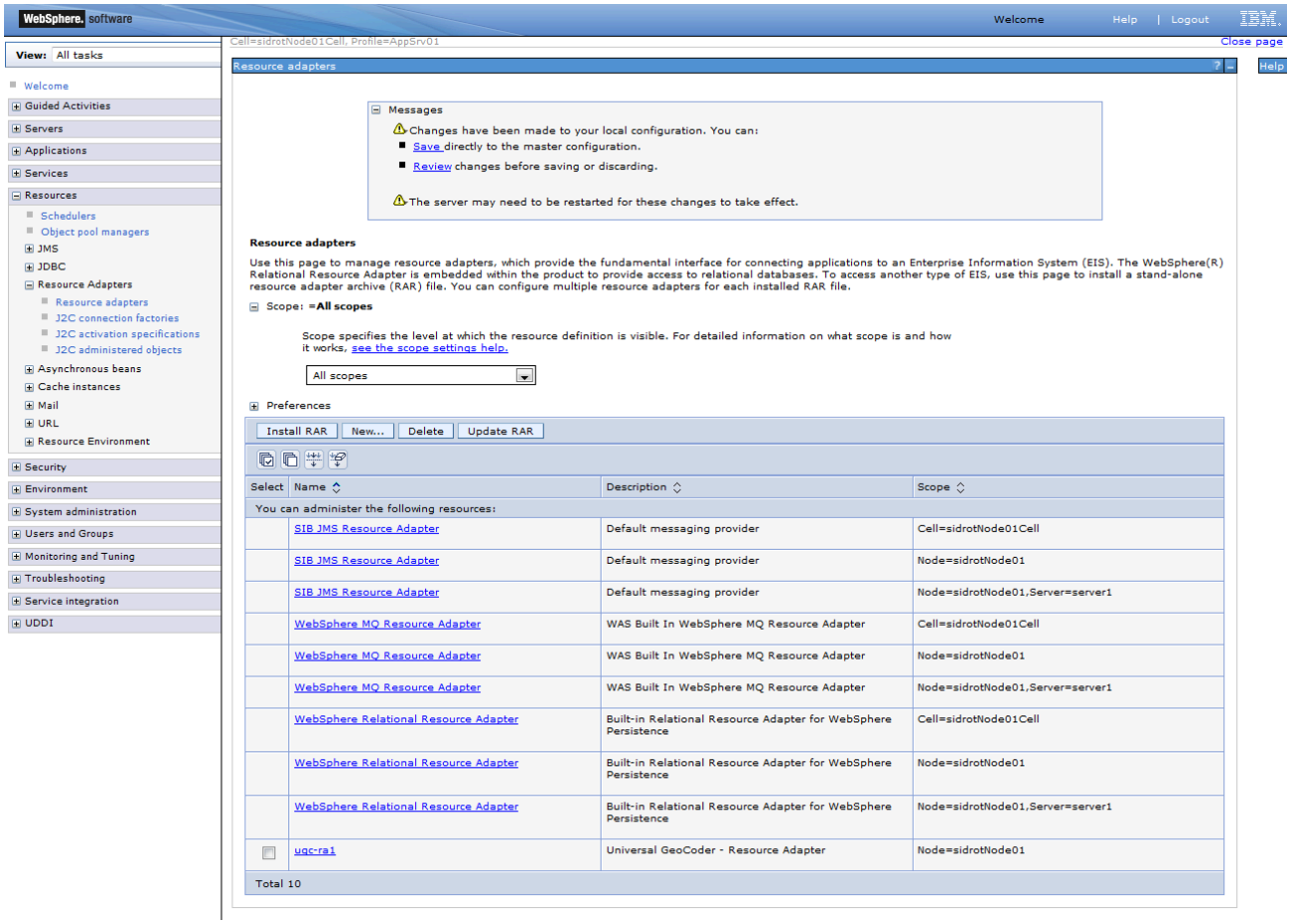


Puis « next » ; dans le champ « name », remplacer geoconcept/ugc/default par ugc-ra1 et validez.

Installation RAR

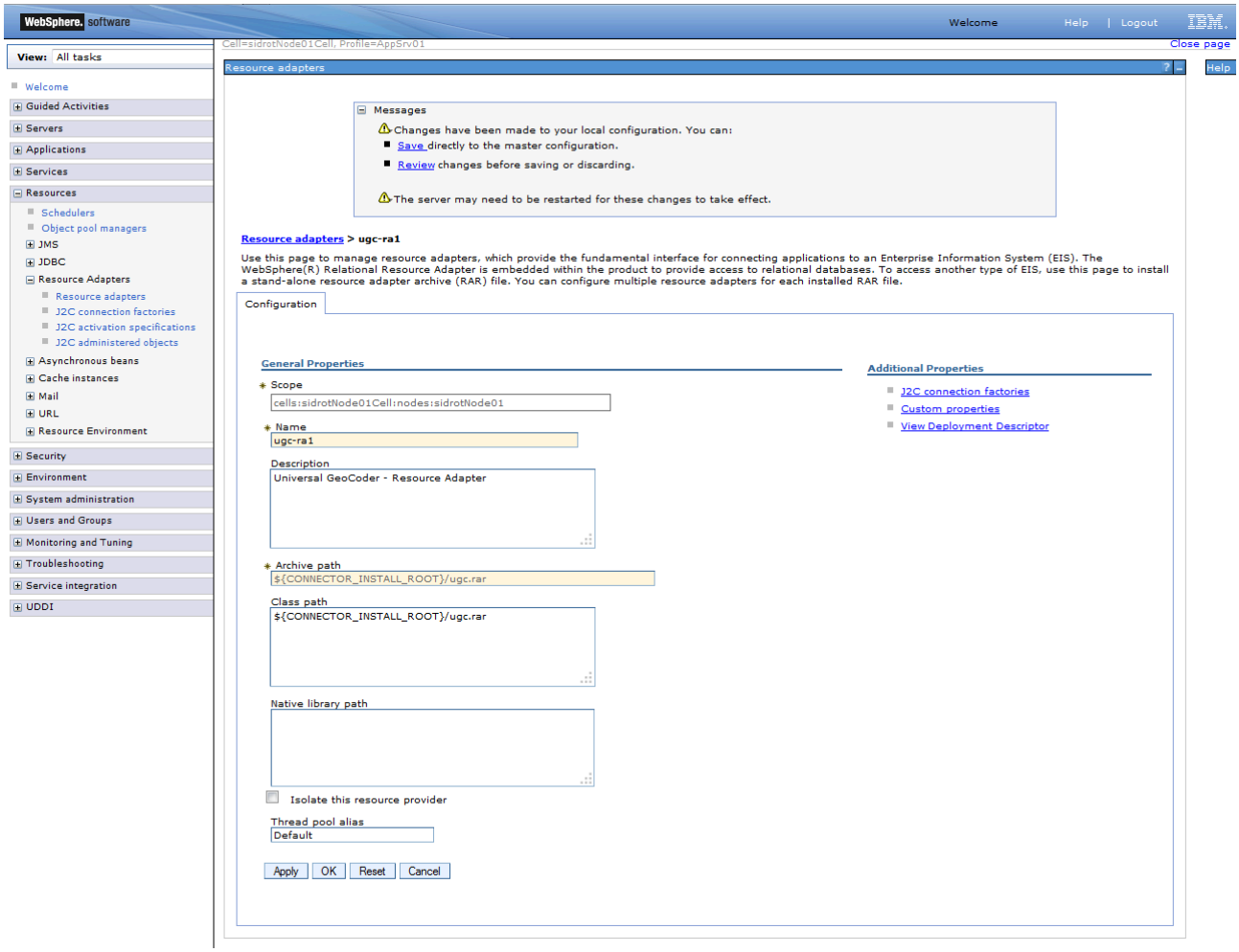


Installation RAR



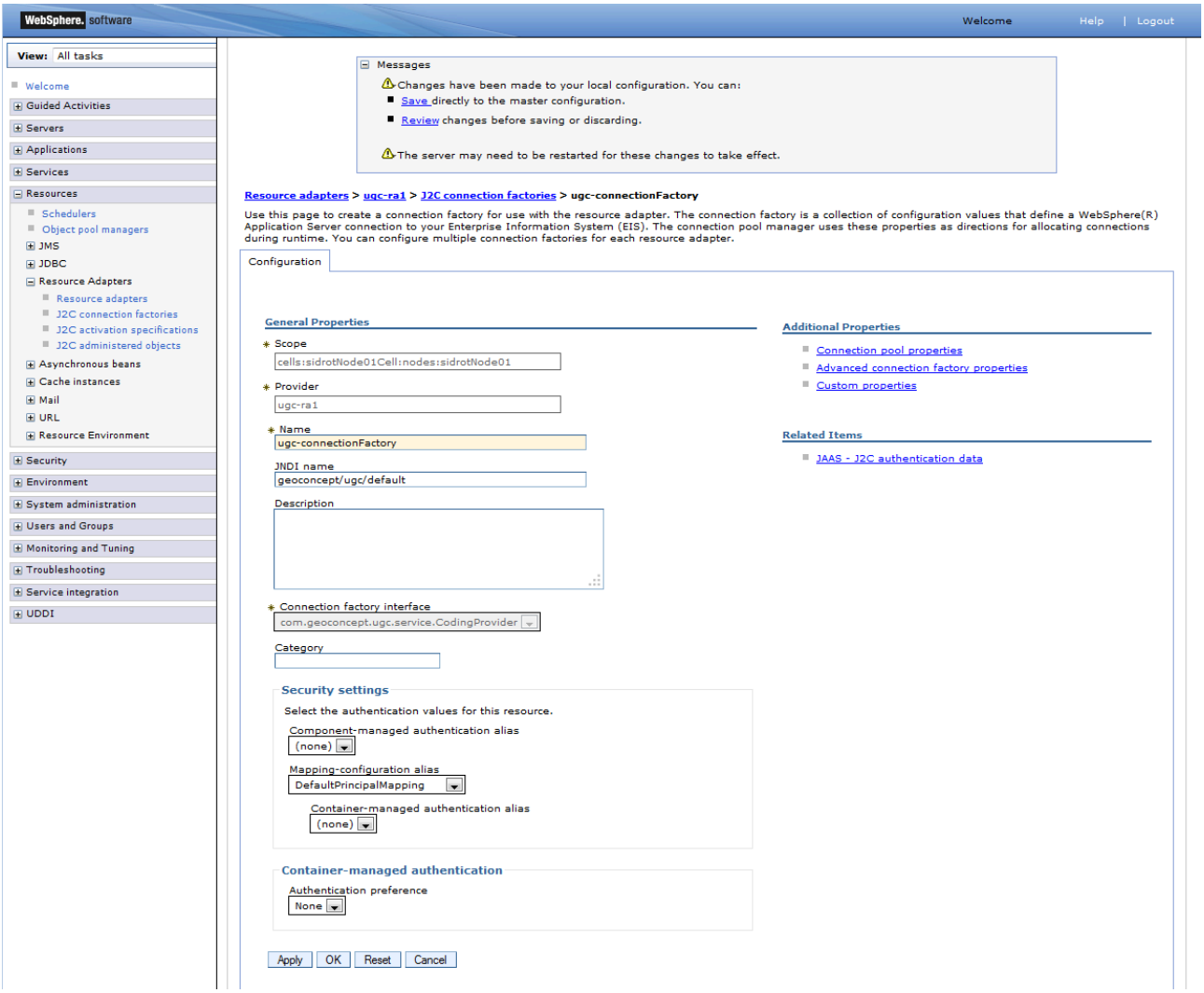
ugc-ra1 apparait désormais dans la liste des adaptateurs de ressource déployés, cliquer dessus dans la liste

Liste des adaptateurs de ressource déployés



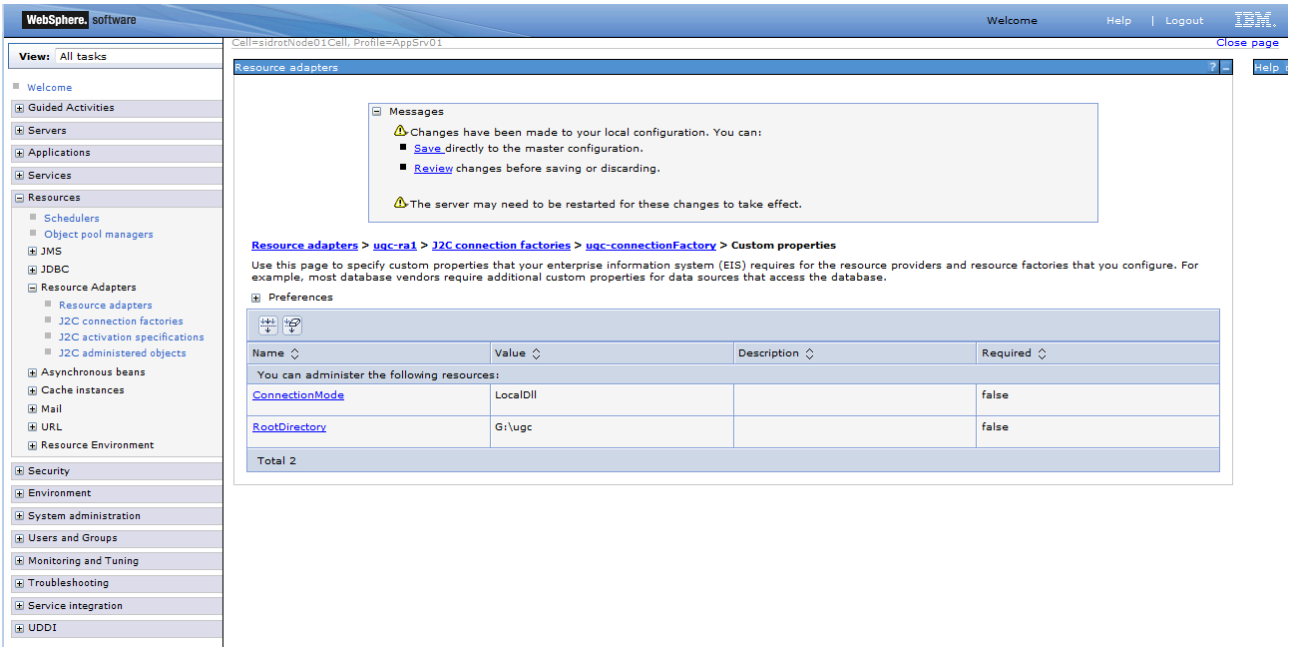
Dans la colonne Additional properties (à droite) cliquer sur " J2C connection factories"

Colonne Additional properties



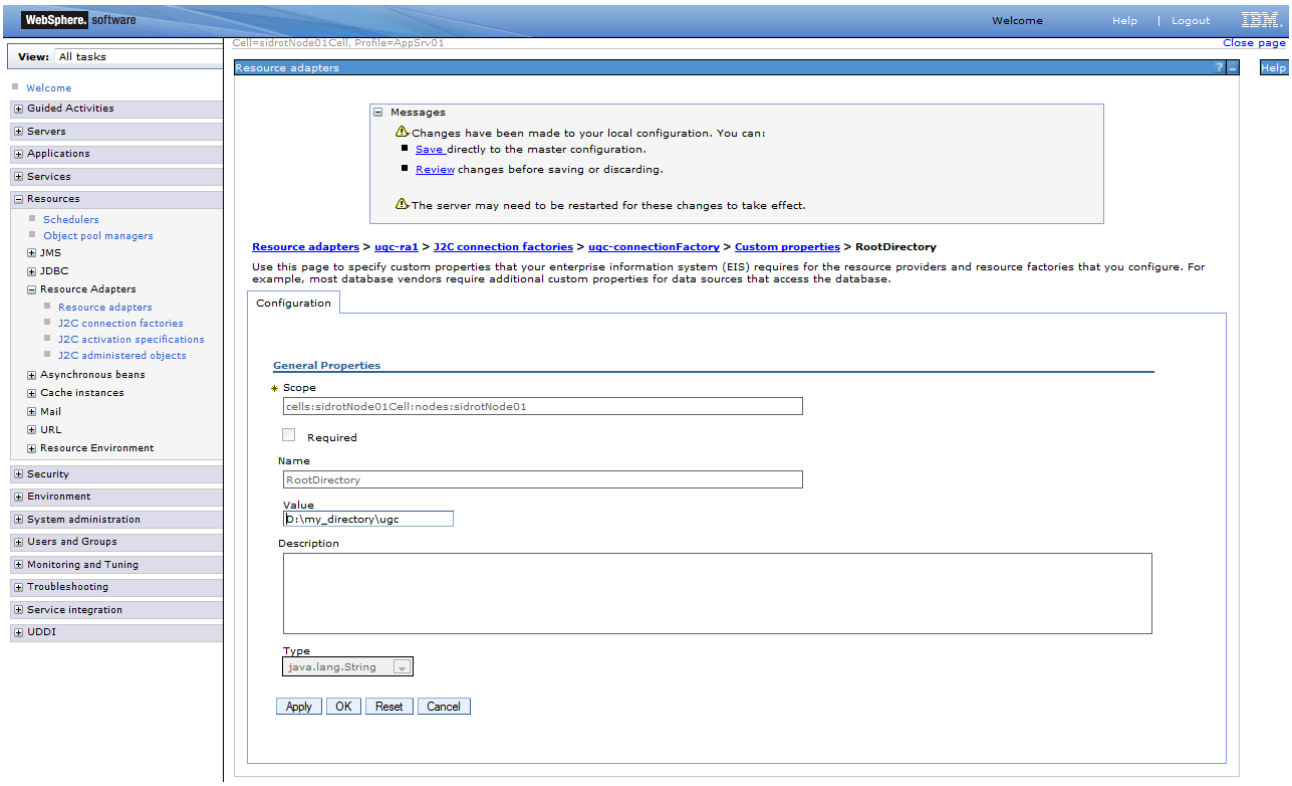
Dans le champ Name, entrez ugc-connectionFactory, et dans le champ JNDI name, entrez geoconcept/ugc/default, puis « Apply ». Dans la colonne Additional properties (à droite), cliquer sur Custom properties.

Colonne Custom properties



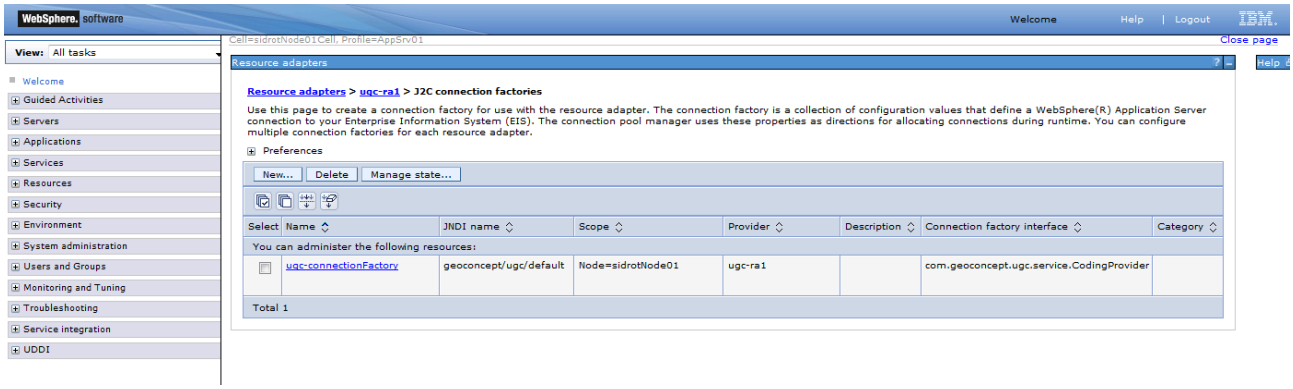
Editez la valeur de RootDirectory pour lui donner le répertoire dans lequel sont les fichiers externes ugc, puis validez.

Valeur de RootDirectory



Sauvegardez (Save directly to the master configuration)

Sauvegarde



La connection factory est désormais déployée.

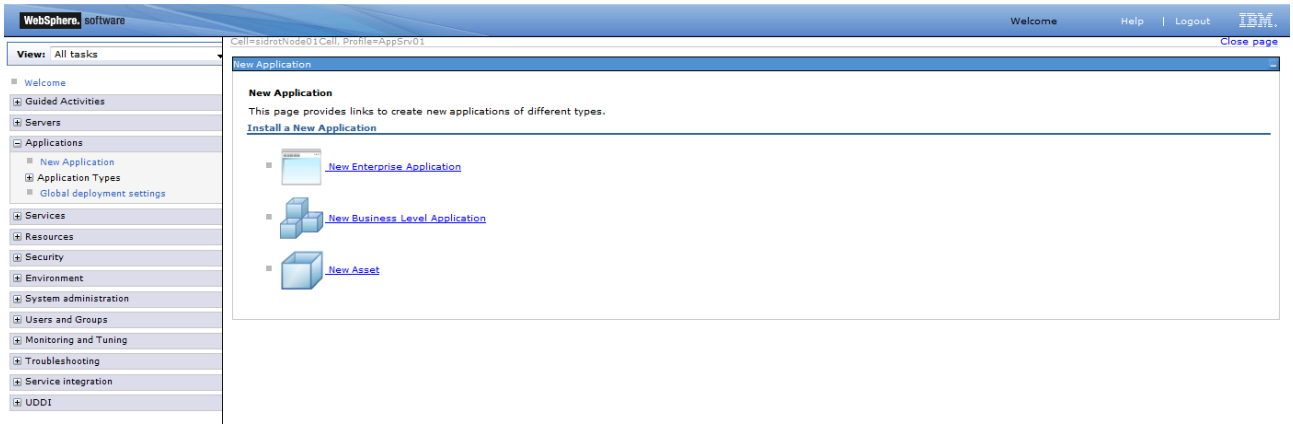
1. Déploiement d'un module optionnel : nous allons maintenant déployer un module optionnel fourni, ici la webapp ugc-admin (un war), qui va consommer les services de l'adaptateur de ressources.

Le principe est le même pour n'importe quel module (fourni ou développé spécifiquement) consommant les services de l'adaptateur de ressources.

En outre ugc-admin permet de valider le déploiement de l'adaptateur de ressources.

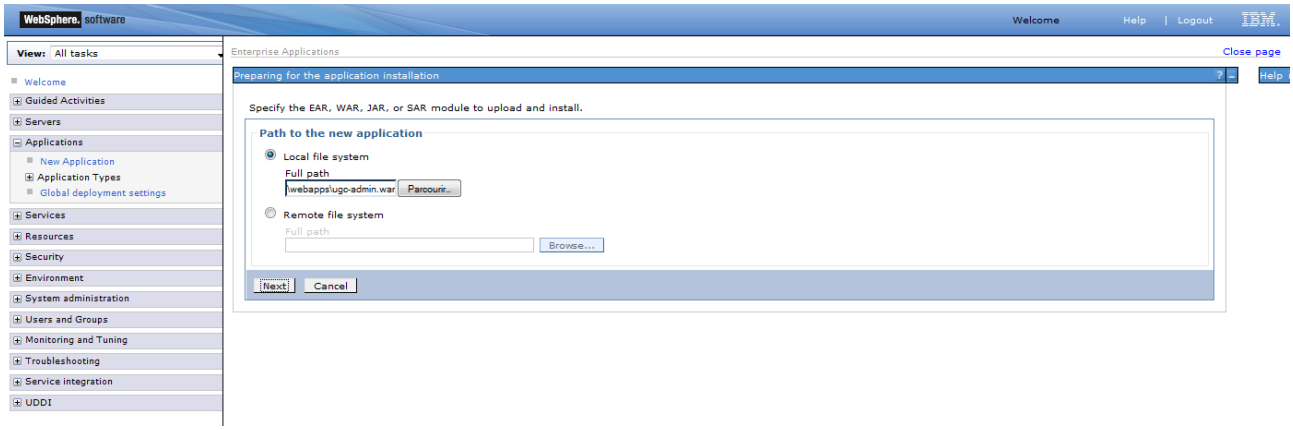
Dans le menu à gauche aller dans Applications > New Application

Nouvelle application



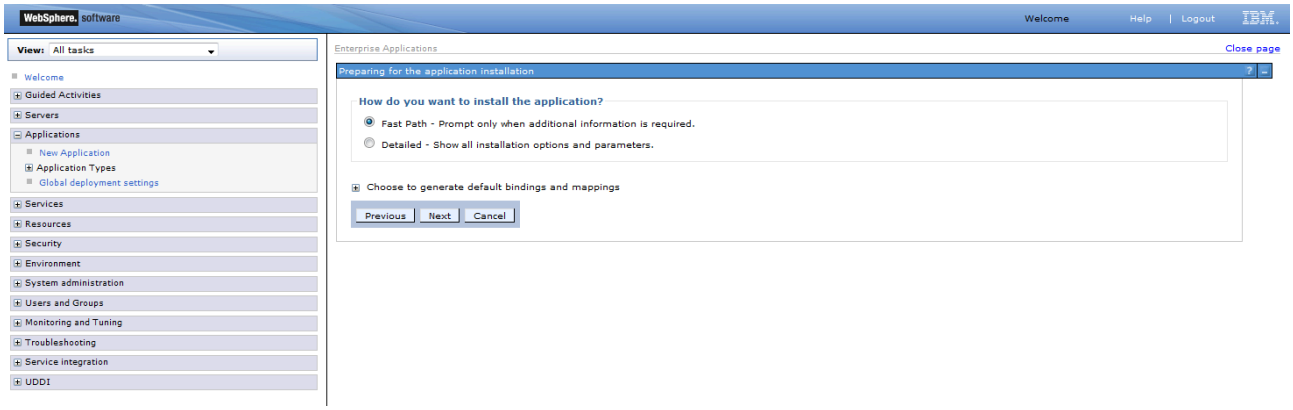
Choisir New Enterprise Application.

New Enterprise Application



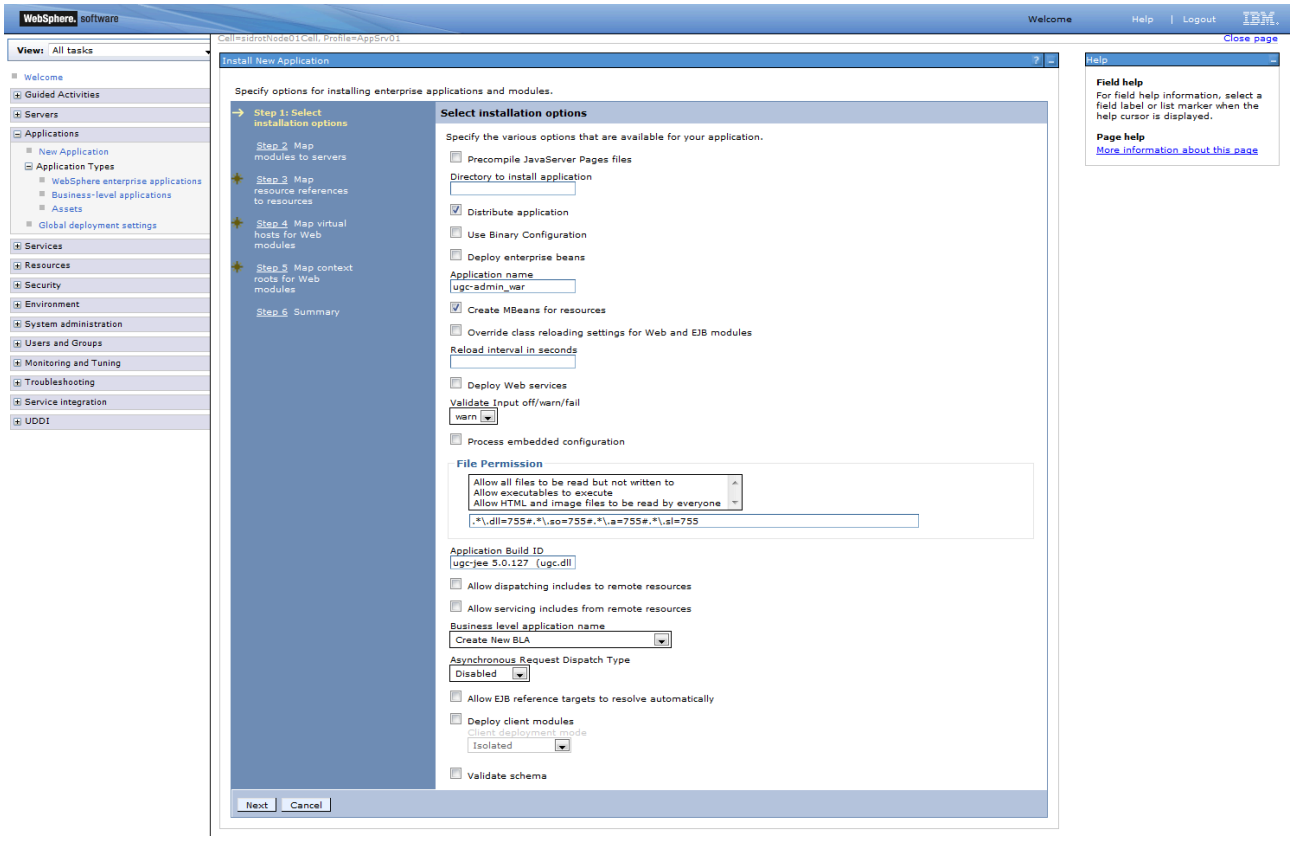
Cliquer parcourir et sélectionner le fichier ugc-admin.war situé dans le sous répertoire websphere/standalone/webapps du livrable puis cliquer sur Next.

Fichier ugc-admin.war



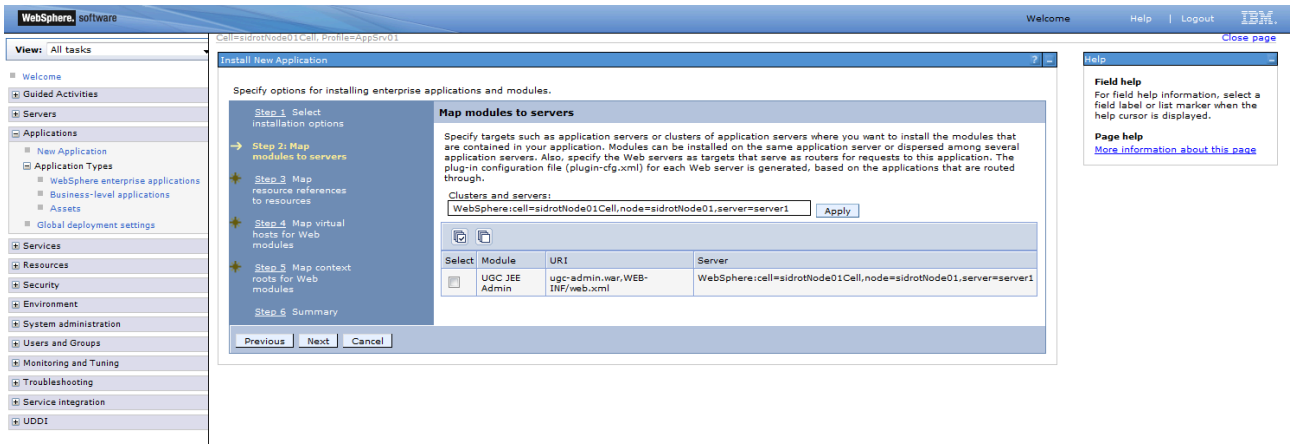
Laisser par défaut fastpath et cliquer Next.

Paramètre fastpath



Step 1 : laisser tout par défaut, puis Next.

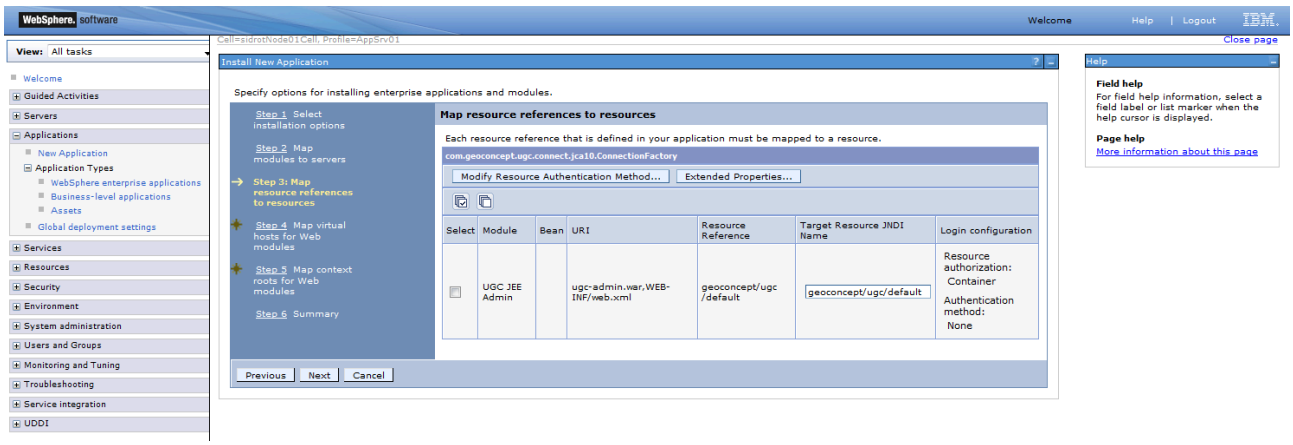
Etape 1



Step 2 : laisser tout par défaut, puis Next.

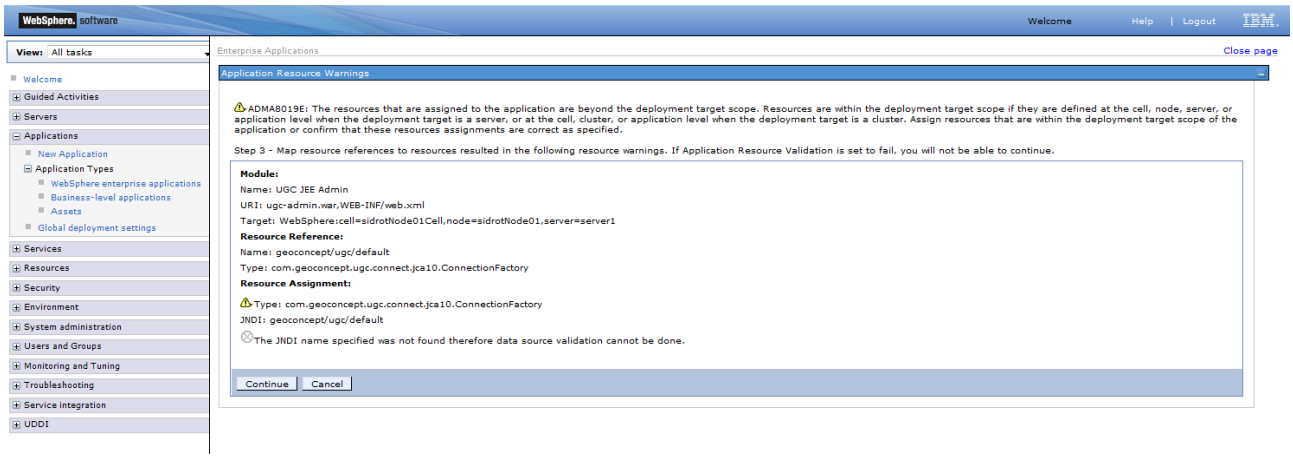
Step 3 : entrer geoconcept/ugc/default dans Target Resource JNDI Name.

Etape 3



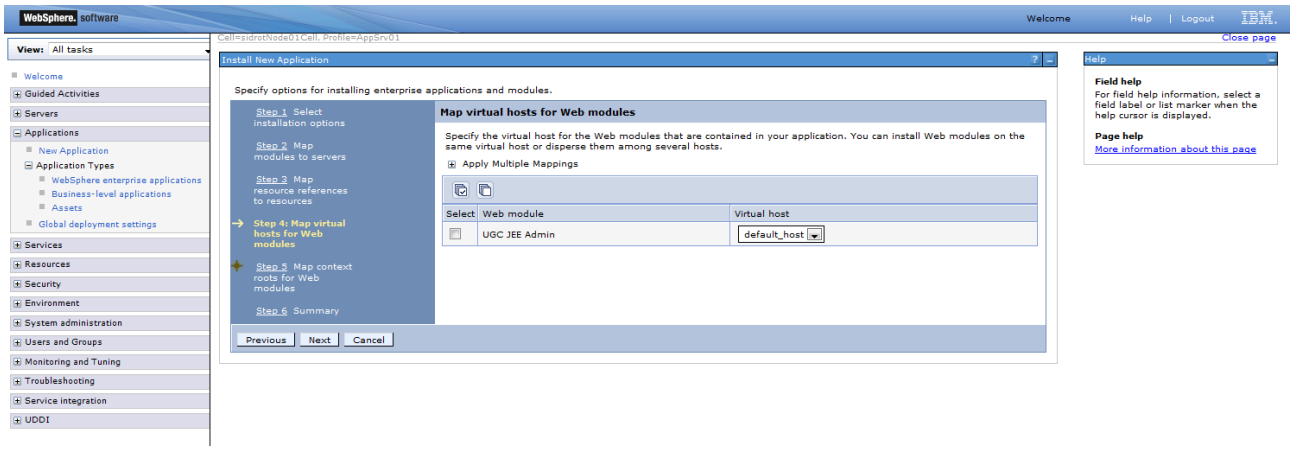
Un warning est indiqué, cliquez sur « continue ».

Etape 3



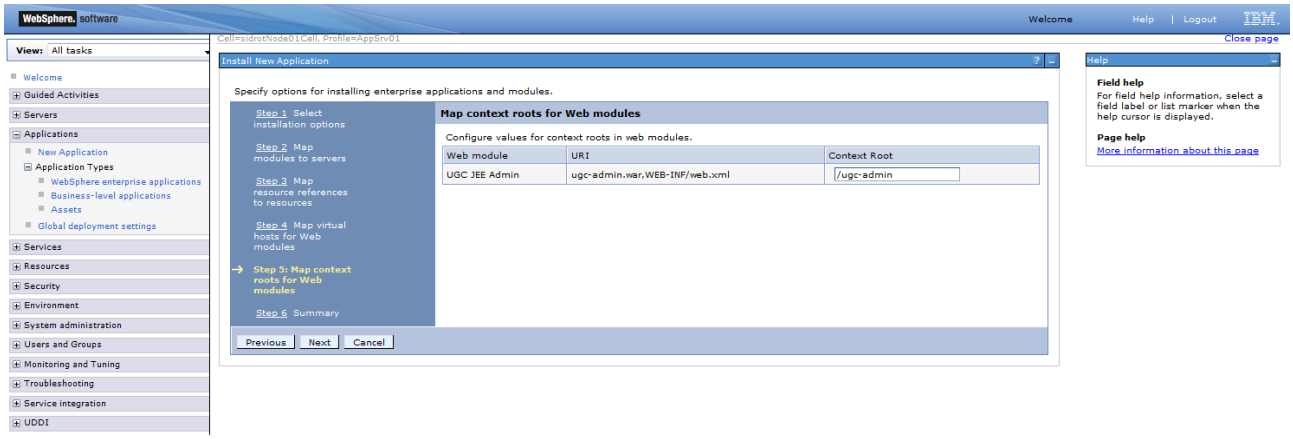
Step 4 : laisser tout par défaut, puis Next.

Etape 4



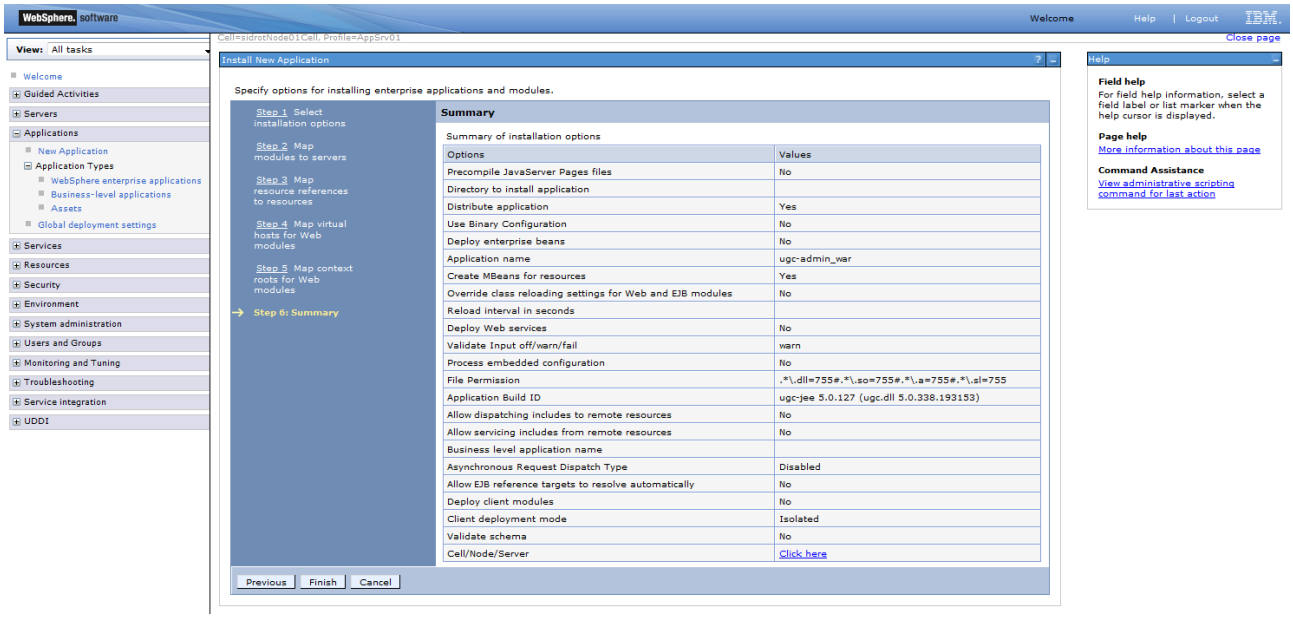
Step 5 : entrer le context root /ugc-admin (il propose / par défaut)

Etape 5



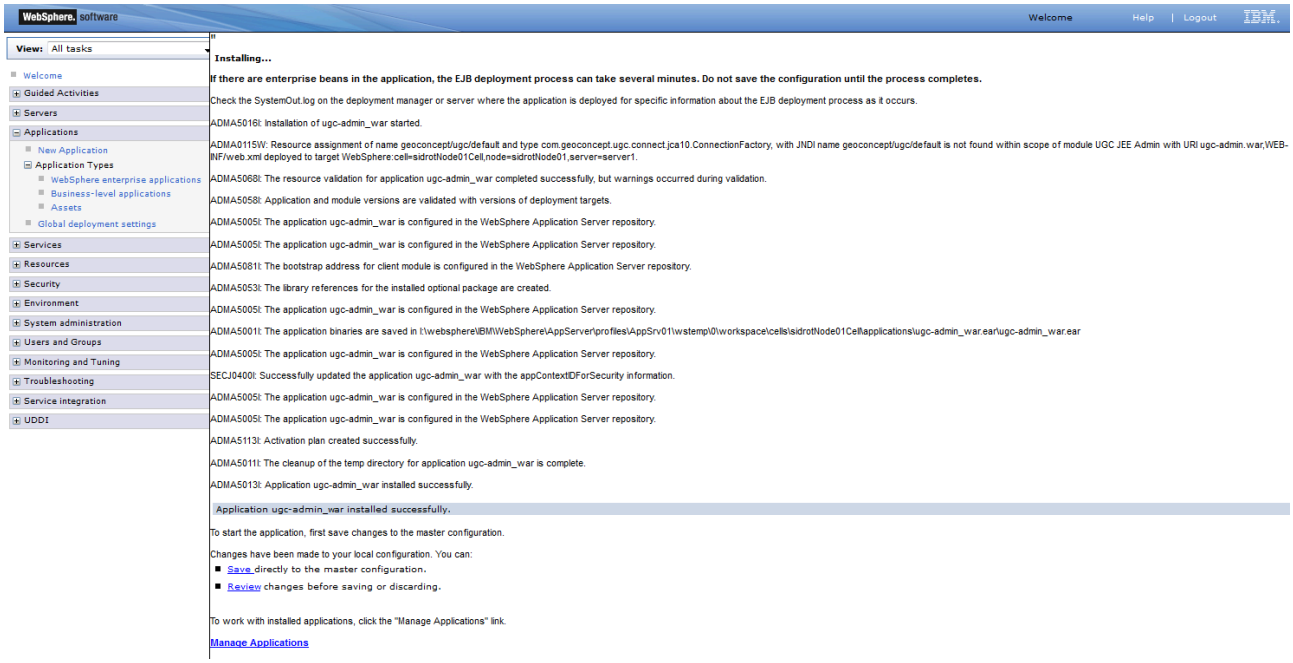
Step 6 : résumé (valider par Finish)

Etape 6



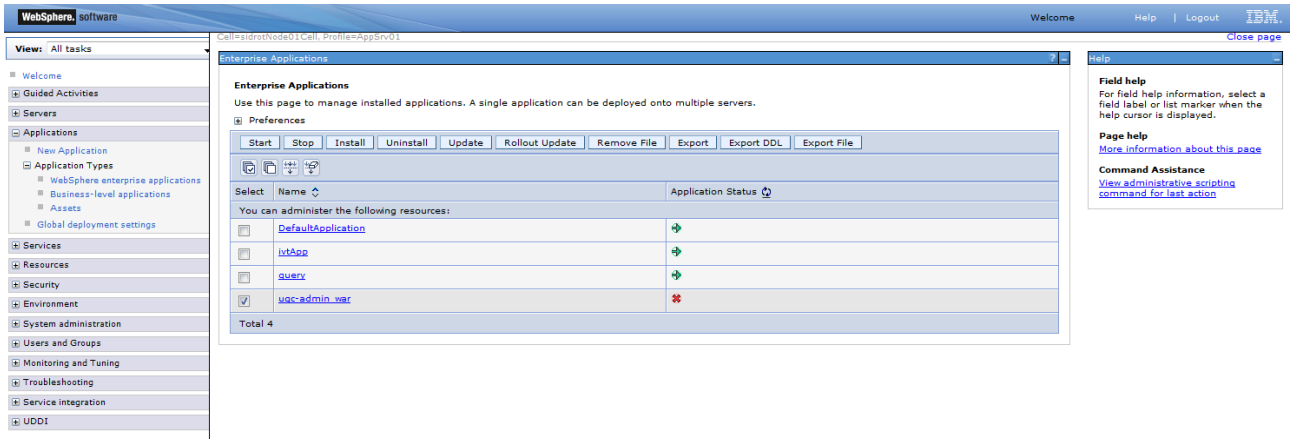
La console présente le statut de déploiement et indique Application ugc-admin_war installed successfull

Statut de déploiement



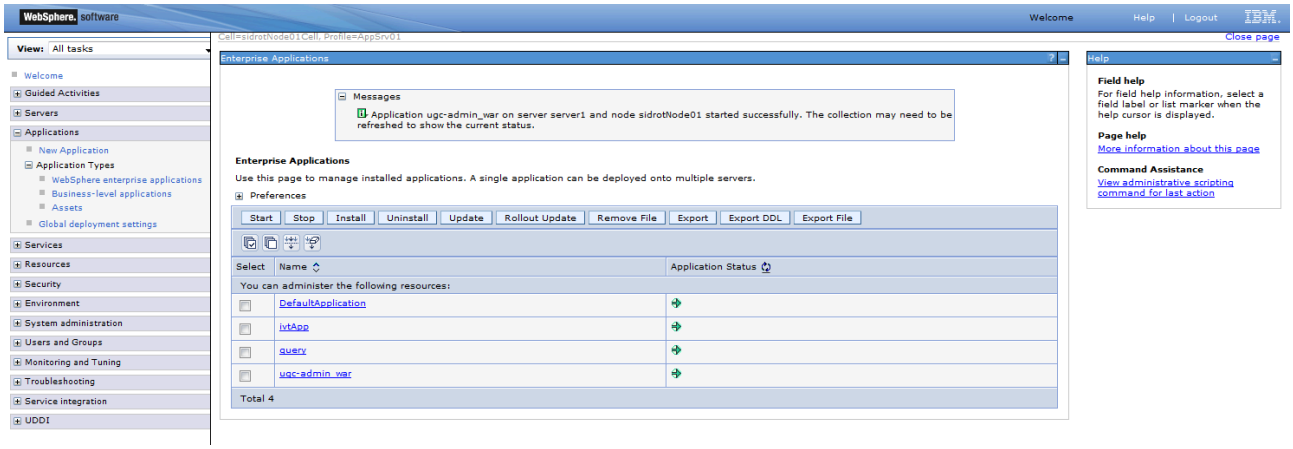
Démarrer la webapp (cocher ugc-admin.war et cliquer sur Start)

Statut de déploiement



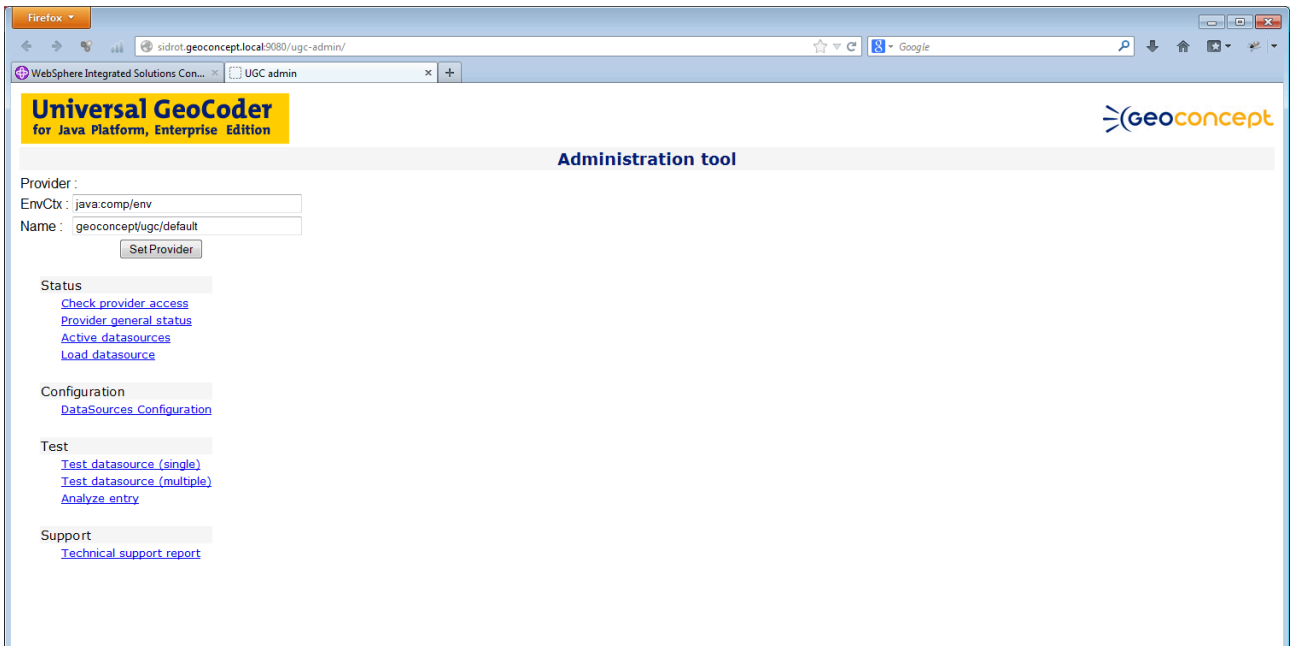
La console indique "ugc-admin_war on server server1 and node sidrotNode01 started successfully. The collection may need to be refreshed to show the current status."

Console



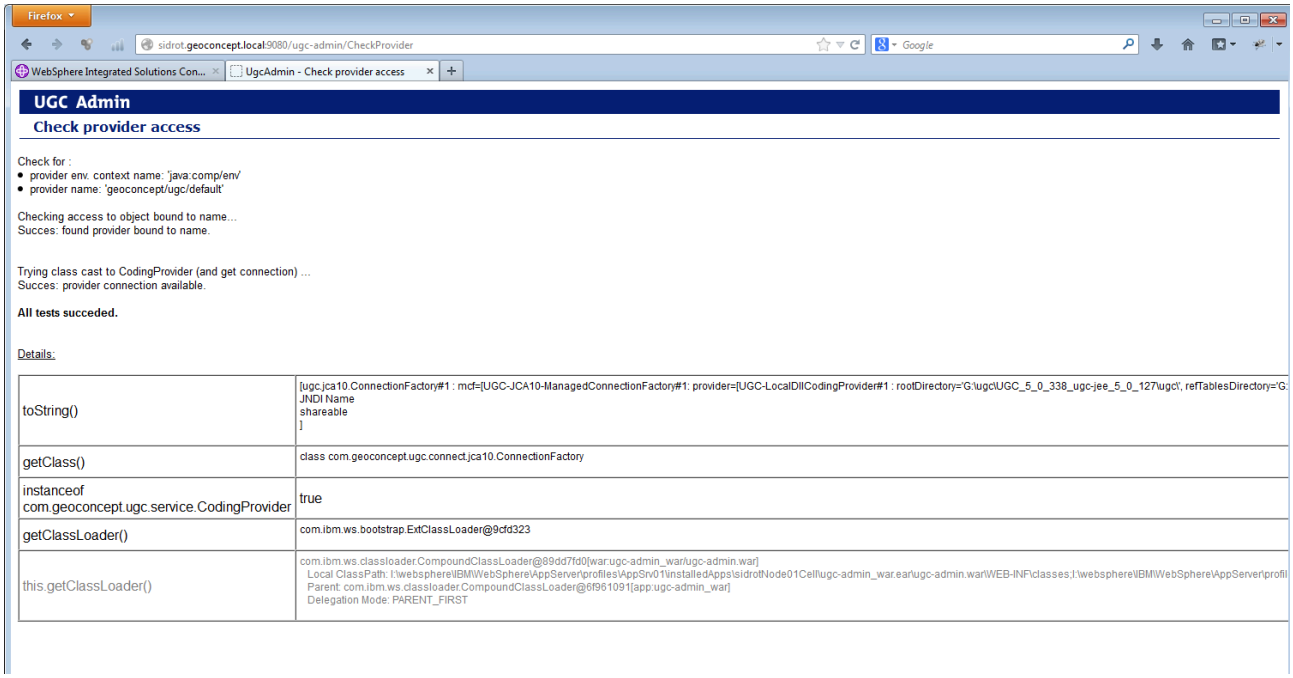
1. Vérification du déploiement avec ugc-admin

Accès à la webapp ugc-admin précédemment déployée (par défaut port 9080) :



Cliquer sur Check provider access

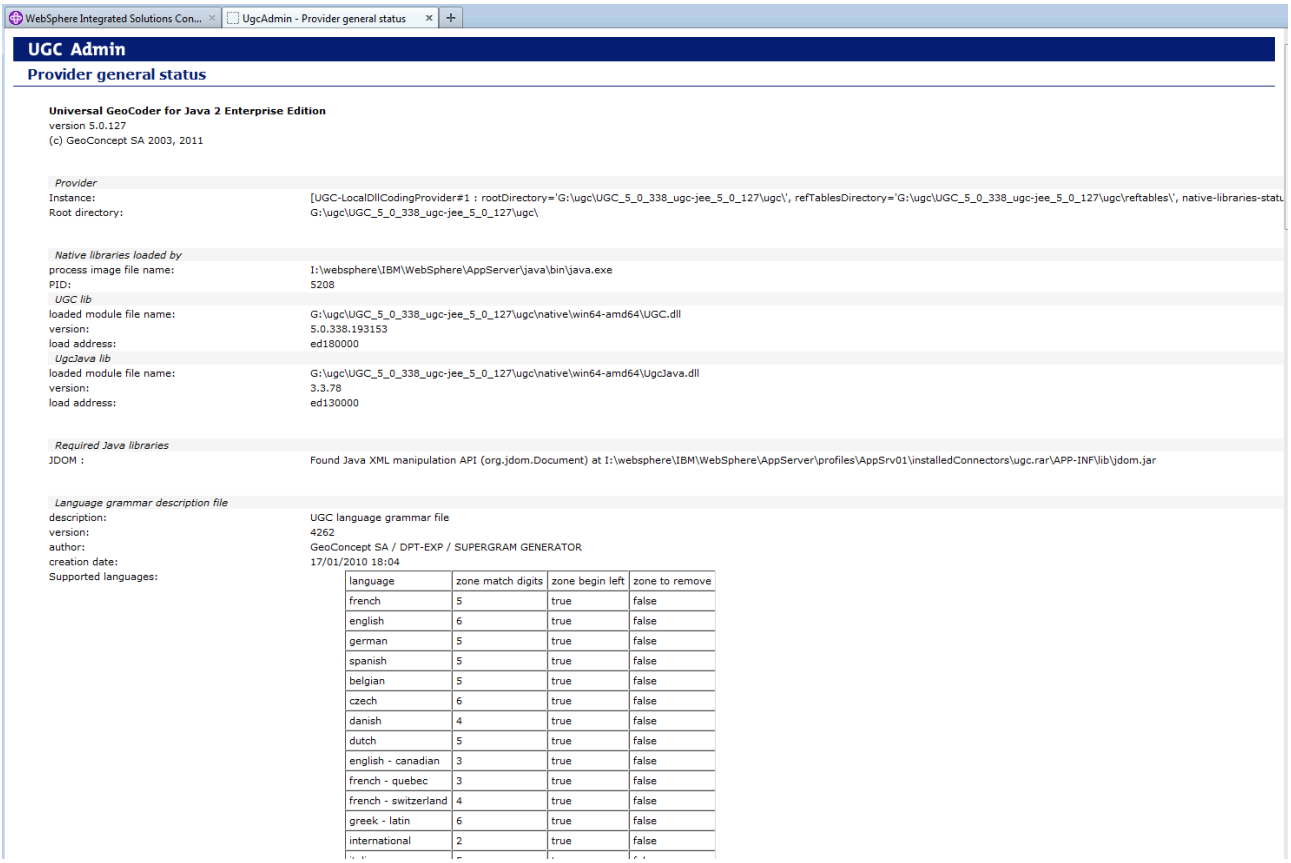
Check provider access



La page doit indiquer All tests succeeded.

Revenir à la page d’accueil de ugc-admin et cliquer sur Provider general status

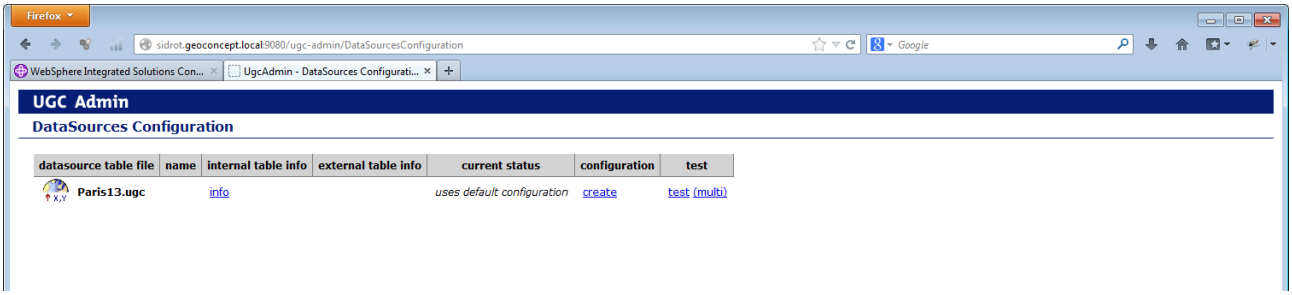
Provider general status



La page indique des informations de version et de runtime de l'instance du fournisseur et son environnement

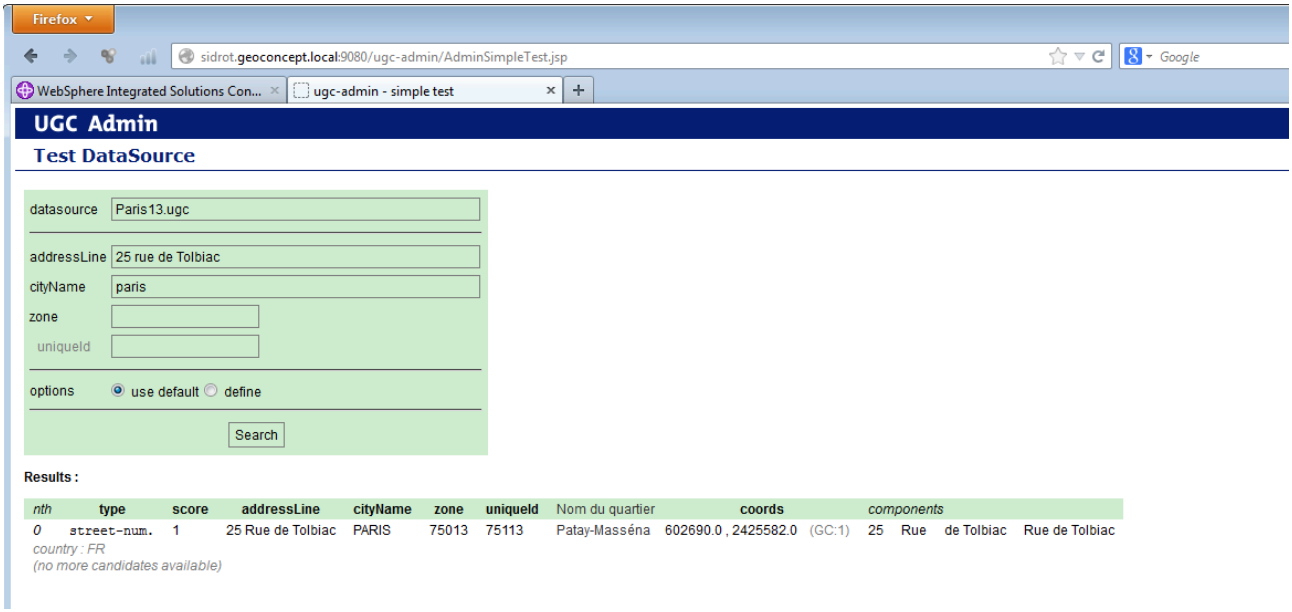
Revenir à la page d'accueil de ugc-admin et cliquer sur DataSources Configuration

Datasources configuration



La page liste les tables de référence disponibles. Cliquer sur test pour une des tables de référence listée. Cliquer sur Search

Test UGC



La page doit afficher un résultat correct, ce qui valide le déploiement.

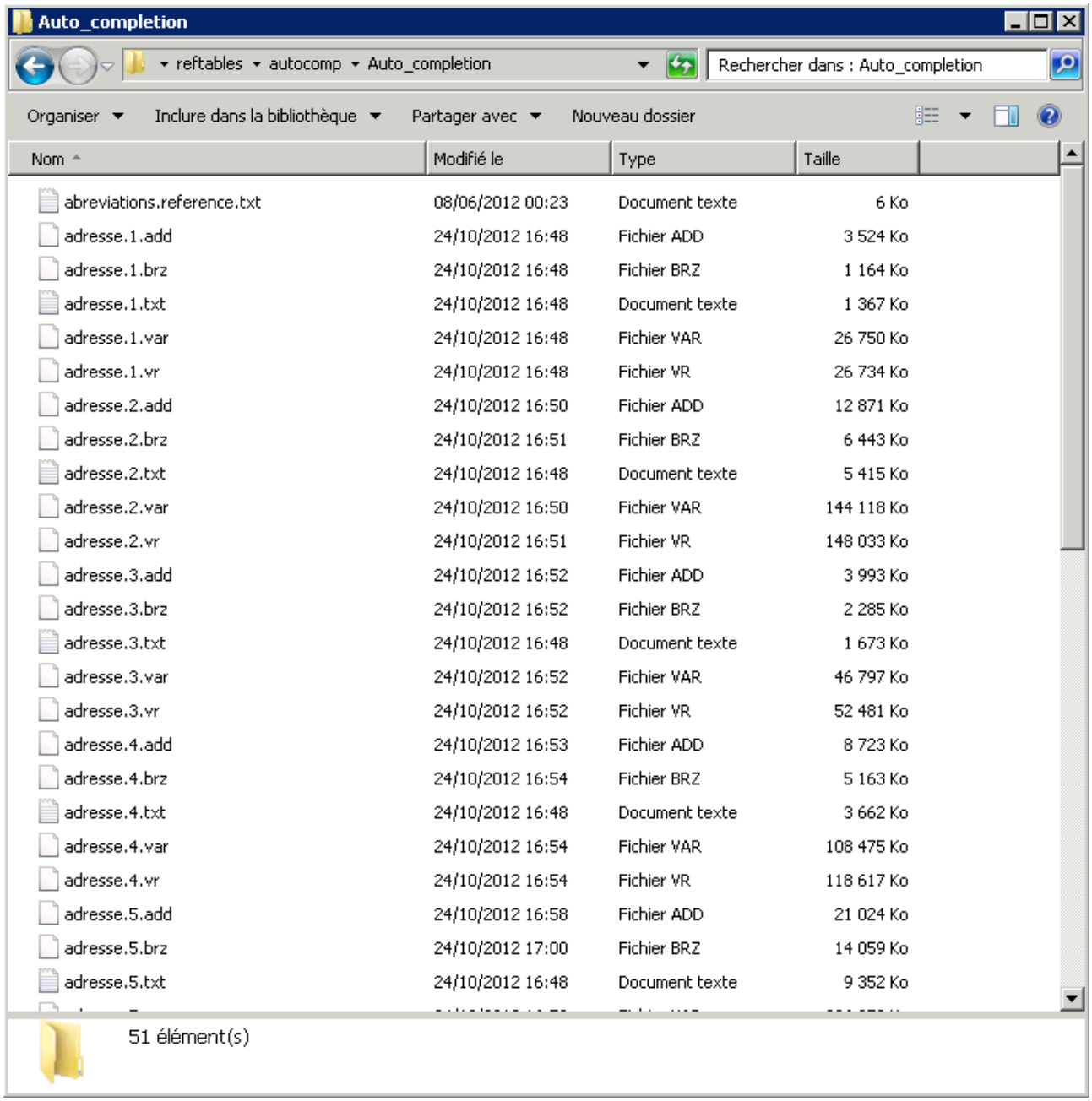
Fichiers pour l'autocomplétion

Si le web service d'autocomplétion fourni par Geoconcept Web est utilisé, il est nécessaire de disposer et d'installer les fichiers de référence correspondants (copie simple des fichiers dans un répertoire).

Il est nécessaire de suivre les étapes suivantes pour un bon fonctionnement du web service d'autocomplétion :

- Dans le répertoire `reftables` présents dans UGC, créer un répertoire autocomp,
- Copier le répertoire fourni par Geoconcept dans ce répertoire : le répertoire fourni par Geoconcept contient les fichiers de référence. Vous devez avoir l'architecture suivante :

Aperçu du répertoire contenant les fichiers de référence pour l'autocomplétion



Le nom du répertoire dans lequel se trouvent les fichiers de référence est important. Ce nom sera renseigné dans les paramètres à sauvegarder dans l'administration de Geoconcept Web (cf. Manuel de Geoconcept Web, Installation, Paramètres d'UGC). Merci de vous y reporter pour voir les paramètres à renseigner pour faire fonctionner le web service d'autocomplétion.

Il est fortement déconseillé d'activer les logs UGC lors de l'utilisation du web service d'autocomplétion. En effet, ces logs sont alimentés à chaque requête d'un utilisateur et

atteindront des volumétries considérables en quelques minutes d'une utilisation en production. De plus, les performances sont bien moindres avec les logs activés.

Guide de référence de Universal Geocoder Server

Le géocodage est l'opération d'obtenir des coordonnées à partir d'adresses. Pour plus d'informations sur la terminologie du géocodeur (types de géocodage, tolérances…) , se reporter au manuel d'utilisation général de Universal Geocoder

Cette documentation présente le composant Universal Geocoder de Geoconcept.

Elle est présentée en trois parties distinctes :

- UGC JEE : il s'agit d'un composant destiné à l'intégration du moteur de géocodage Universal GeoCoder (UGC) à la plate-forme Java Enterprise Edition (JEE) et ses sous-ensembles comme le moteur de servlet Tomcat. Par extension (déploiement de module optionnel), le produit peut être utilisé dans le but de déployer un web service de géocodage.
- UGC Command Line
- UGC .NET

Cette documentation n'est pas la documentation de Universal Geocoder Standalone de Geoconcept.

Universal Geocoder JEE

Il s'agit d'un composant destiné à l'intégration du moteur de géocodage Universal GeoCoder (UGC) à la plate-forme Java Enterprise Edition (JEE) et ses sous-ensembles comme le moteur de servlet Tomcat. Par extension (déploiement de module optionnel) le produit peut être utilisé dans le but de déployer un web service de géocodage.

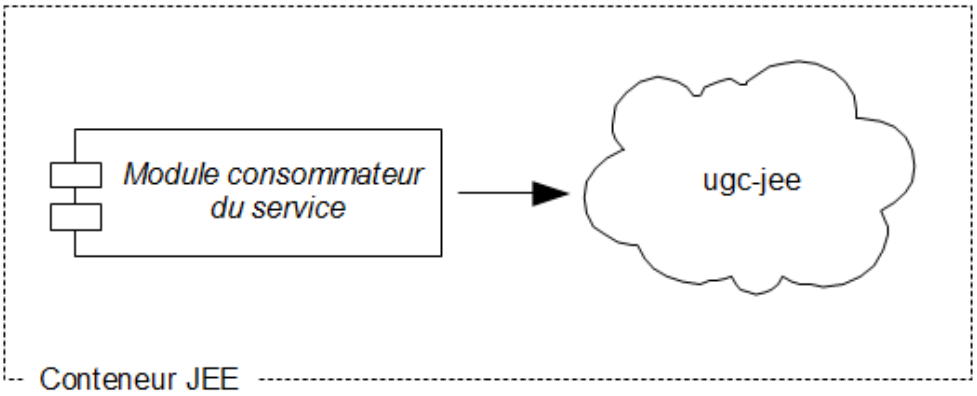
Du point de vue fonctionnel, le géocodage est l'opération d'obtenir des coordonnées à partir d'adresses. Pour plus d'informations sur la terminologie du géocodeur (types de géocodage, tolérances…) , se reporter au manuel d'utilisation général de Universal Geocoder. Ce manuel est destiné a décrire spécifiquement la mise en oeuvre d'Universal Geocodeur pour Java Enterprise Edition (ugc-jee).

Principes

Intégration JEE

ugc-jee est un composant d'intégration à la plate-forme JEE, il offre un service de géocodage aux modules JEE déployés sur une serveur d'applications (au sens large, moteur de servlet type Tomcat compris).

Intégration JEE



Le module consommateur peut être de n'importe quel type (webapp, ejb, etc). De même à l'intérieur du module JEE le consommateur peut être de n'importe quel type (servlet, jsp, pojo, etc).

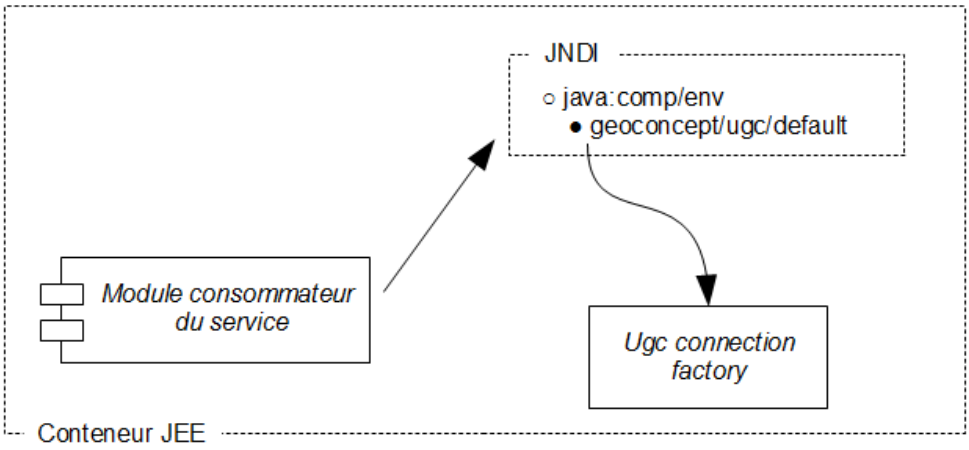
Mode d'accès

L'application utilisant le géocodeur référence le fournisseur via un nom logique de l'annuaire JNDI (Java Naming and Directory Interface) du serveur d'applications. La manière de monter le fournisseur sera décrite plus loin.

Traditionellement, le nom logique utilisé est « geoconcept/ugc/default » du contexte « java:comp/env » mais il est possible d'utiliser un autre nom, un autre contexte ou de monter plusieurs fournisseurs de types différents.

Par soucis de simplicité le cas d'utilisation le plus courant (un seul fournisseur primaire local avec nommage par défaut) sera décrit dans un premier temps.

Mode d'accès



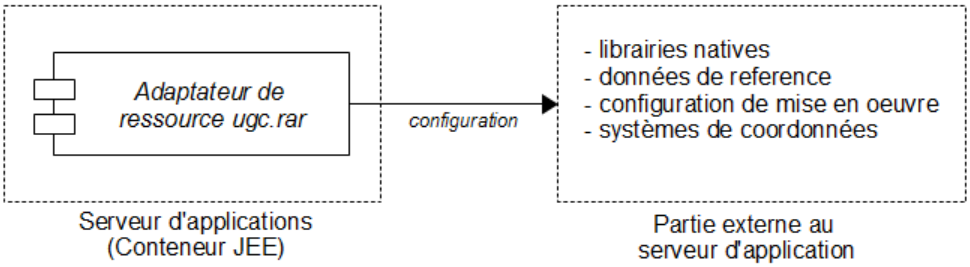
Organisation du composant

Adaptation de ressource JEE et éléments externes

ugc-jee se compose de plusieurs parties. Pour des questions de performances et de réutilisation, le moteur (aussi appelé noyau) est écrit en C++ et est donc diffusé sous forme de librairies natives. La

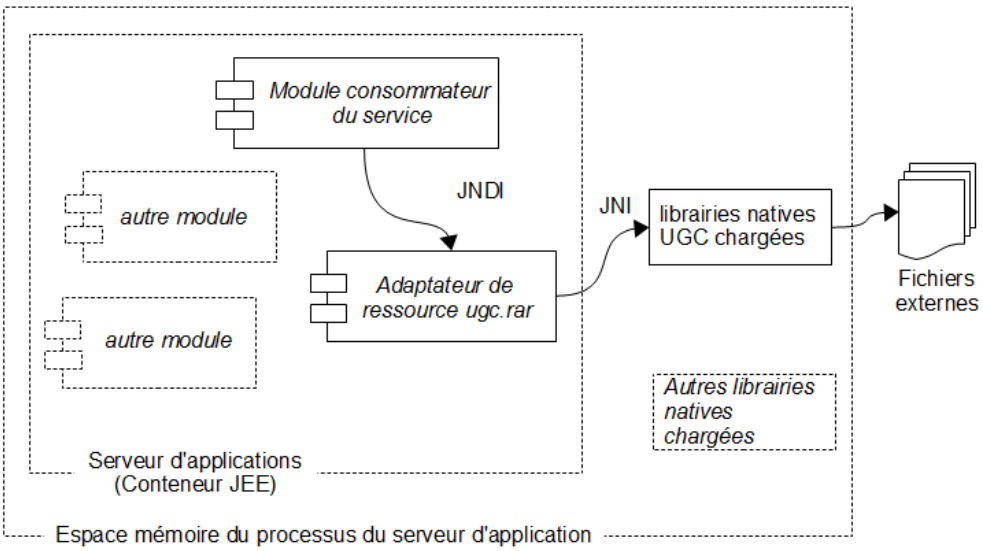
partie intégration (adaptateur de ressource) réalise le lien avec le moteur en gérant sa mise en oeuvre par chargement de ses librairies natives. En terme de déploiement de fichiers cela concerne donc deux arborescences distinctes :

Organisation du composant



En terme d'exécution les librairies natives seront chargées en mémoire dans le processus de l'instance du serveur d'application (ou sa partition éventuelle selon les modèles) au démarrage. La partie adaptateur de ressource expose une interface java et gère le moteur instancié en mémoire directement via JNI.

Organisation du composant



Autre type de fournisseur

L'adaptateur de ressource décrit précédemment correspond au type LocalDll, il s'agit du fournisseur primaire, celui qui est lié au moteur de géocodage et réalise concrètement le traitement.

Dans certaines situations (pour diverses raisons : frontal séparé, partage, architecture, isolation, etc, etc) il est souhaité que le module qui consomme le service ne soit pas déployé sur la même instance de serveur d'application que le fournisseur primaire ugc (par exemple chacun étant sur deux machines serveur différentes - remoting).

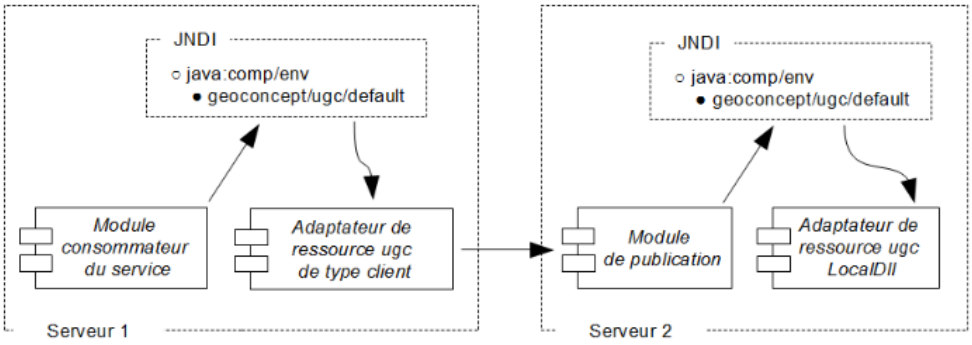
Dans ce cas il existe plusieurs possibilités :

- soit la création d'un module destiné à la publication qui accèdera au fournisseur ugc (dans ce cas le protocole utilisé et les modalités sont propres au module créé).
- soit le consommateur est implémenté comme un client d'un module fourni par ugc-jee comme par exemple un web service du module ugc-ws.
- soit un module fourni par ugc-jee est déployé sur la machine du fournisseur primaire et un adaptateur de ressource client de ce module est monté sur la machine consommatrice.

Le dernier cas est transparent pour l'application consommatrice : que le fournisseur soit local ou distant son code source ne change pas, il ne s'agit que d'une question de déploiement et de configuration (il s'agit en quelque sorte du pattern Multiple business delegate).

Le protocole de transport utilisé dépend du couple module de publication / client choisi : cela peut être web service (http/soap) ou rmi.

Autre type de fournisseur



Dans tous les cas, dans le cadre de la mise en oeuvre de ugc-jee en mode installé (c'est à dire non SAAS), il existe une instance de fournisseur de type LocalDll. Par ailleurs, dans la mesure où ce fournisseur est local (pas de transfert, de sérialisation, etc) il est le plus performant (latence, débit).

Modules additionnels

Un certain nombre de modules additionnels sont fournis dans ugc-jee. Aucun n'est indispensable au bon fonctionnement de l'adaptateur de ressource et leur déploiement est optionnel.

ugc-admin

ugc-admin est une webapp permettant la vérification de la bonne installation du produit, sa configuration et le test de bon fonctionnement.

ugc-ws

ugc-ws est une webapp publiant un web service de géocodage. Elle est basée sur spring-ws (elle publie au format document / encodage litteral).

ugc-axis-fusion

ugc-axis-fusion est une webapp publiant un web service de géocodage prédéployé sur Axis 1.4 (elle publie au format rpc / encodage soapenc).

ugc-remote

ugc-remote est une webapp publiant le service de geocodage via rmi. Elle est notamment utilisée pour le remoting en conjonction avec le fournisseur de type RmiClient.

Administration / Configuration

Configuration de mise en oeuvre

Un référentiel constitué d'un ensemble de fichiers portant des extensions .ugc.xxi, publié constitue une source de données (datasource).

Le fichier service.xml situé dans %ugc%/conf définit la configuration générale du fournisseur de service de géocodage. Celui ci contient notamment une configuration par défaut pour les sources de données (default-datasource). Il est possible de définir une configuration spécifique à une source de données.

Pour définir une configuration particulière pour une source de données il est recommandé d'utiliser la webapp ugc-admin (rubrique DataSourcesConfiguration puis configuration > create). Il est aussi possible d'éditer directement le fichier service.xml.

Si aucune configuration spécifique n'a été définie alors la configuration par défaut s'applique à la mise en oeuvre de cette source de données.

Par ailleurs il est possible de définir certains paramètres lors de l'appel (via findAddressOptions). Ainsi la valeur concrète que prend un paramètre (comme par exemple discrepancy) peut provenir (dans l'ordre) :

- de l'appel si il est affecté dans findAddressOptions (cette affectation est optionnelle)
- de la valeur indiquée au niveau de la configuration de datasource particulière (si une configuration de datasource particulière a été définie)
- de la valeur indiquée au niveau de la configuration de datasource par défaut (default-datasource).

Configuration détaillée d'une source de données

Il est possible de définir la configuration de mise en oeuvre d'un référentiel de manière très détaillée.

Cette définition peut répondre à des besoins particuliers, elle n'est pas indispensable car la configuration par défaut convient pour la grande majorité des cas.

Certains paramètres sont définissables au moment de l'appel, d'autres sont fixés à la création de l'instance de source de données et non modifiables par la suite.

Les paramètres que l'on peut définir aussi au moment de l'appel seront décrits dans la section FindAddressOptions.

Datasource identification

Cette partie permet l'identification d'une source de données dans l'administration.

File : référentiel de géocodage utilisé. Elle est contenue dans %geoconcept%/ugc/reftables ou dans un sous-répertoire.

Name : alias de nom de la source de données (optionnel).

Publication information

Cette partie permet d'ajouter des informations sur la source de données.

Title : un titre pour la source de données.

Abstract : contient une courte description de la source de données.

Online ressource : lien HTTP donnant plus d'informations sur la source de données.

Reference table settings

Version : version du référentiel.

Zone meaning : sens de l'attribut "zone" de l'adresse. Exemple : Code postal.

UniqueId meaning : sens de l'identifiant de recherche uniqueId.

SecondaryZone meaning : sens de l'attribut secondaryZone de l'adresse. Exemple : Code IRIS.

StreetSectionId meaning : sens de l'attribut streetSectionId de l'adresse. Exemple : PID navteq.

Coordinate system : identifiant du système de coordonnées de référence pour la publication (cf norme OpenGIS).

Country : pays concerné.

Bounds : rectangle d'encombrement des données du référentiel

Run time settings

Cache : activation ou non du cache. Ce cache permet de garder en mémoire des villes du référentiel.

Max Cache Mem Size est utilisé uniquement si le cache est actif. Taille de la mémoire réservé pour le cache en Ko.

Min processors : nombre initial d'instances de géocodage.

Max processors : nombre maximum d'instances de géocodage.

Finder general settings

City scoring method : méthode de calcul à utiliser pour le score de la ville :

- Standard : rapide mais moins précis. Recommandé pour une utilisation de type batch,

- Levenshtein : moins rapide mais plus précis. Recommandé pour une utilisation de type saisie d'adresse en ligne.

Street scoring method : méthode de calcul à utiliser pour le score de la rue :

- Standard : rapide mais moins précis. Utilisation non recommandé,
- Levenshtein : moins rapide mais plus précis. Recommandé pour une utilisation de type saisie d'adresse en ligne ou batch.

Min streets : nombre de rues minimum pour qu'une ville soit considérée comme couverte. Utilisé pour la tolérance à la vill (FindAddressResults.TOLERATE_TYPE_CITY).

Stratégie de recherche :

Pour plus de détail sur les stratégies de recherche, se reporter au manuel d'utilisation de Universal Geocoder

Finder request defaults settings

Max candidates : nombre maximum de résultats renvoyés par un géocodage. Voir FindAddressOptions.candidateCount.

Score threshold : score au-delà duquel les résultats du géocodage sont conservés. Voir FindAddressOptions. ScoreThreshold.

Score threshold : seuil de proposition de candidats de type « rue ». Voir FindAddressOptions. ScoreThreshold.

Find type : type de géocodage souhaité. Voir FindAddressOptions.geocodingType.

Tolerate geocoding type : tolérance de géocodage souhaitée. Voir FindAddressOptions.tolerateGeocodingType.

Max meter error : erreur de placement maximale (en mètres) pour considérer un géocodage à la rue comme un géocodage au numéro. Voir FindAddressOptions.maxMeterError. Discrepency : décalage latéral. Voir FindAddressOptions.discrepency.

Discrepancy along street : décalage longitudinal. Voir FindAddressOptions.discrepency.

Favor city match element : orienter la recherche de la ville avec un élément descriptif de la ville. Voir FindAddressOptions.favorCityMatchElement.

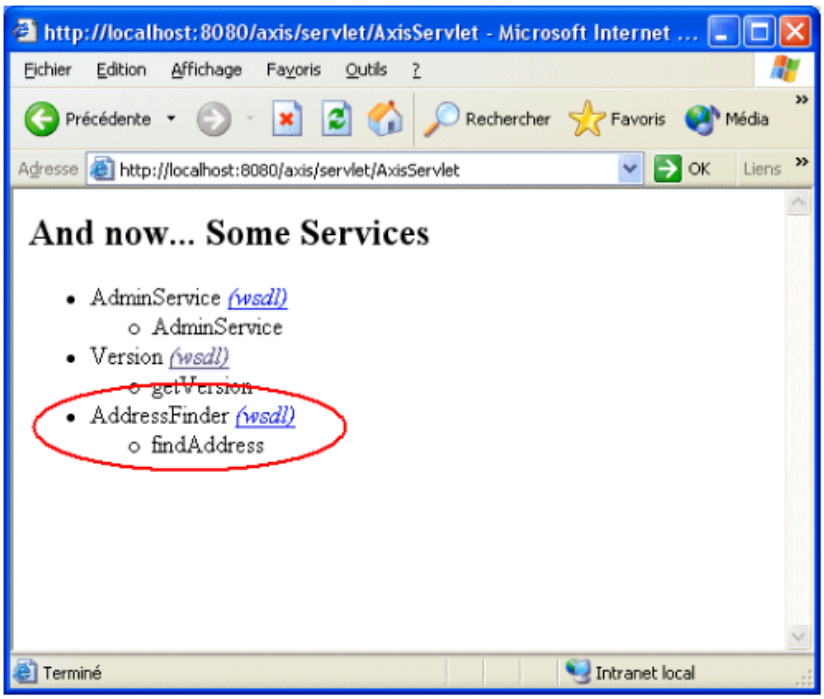
Zone match digits : prise en compte de n premiers caractères de l'attribut zone de l'adresse. Voir FindAddressOptions.zoneMatchDigits.

Tests / Troubleshooting pour AXIS 1.4

Vérifier le bon chargement de l'adaptateur de ressource au démarrage de Tomcat. Vous pouvez utiliser la webapp ugc-admin pour valider l'installation.

Dans le cas de l'utilisation du Web Service, vérifier la présence du service AddressFinder dans axis :

Tests / Troubleshooting pour AXIS 1.4



Interface JAVA

Lorsque le fournisseur est utilisé directement à l'intérieur de la plate-forme java (via des objets java simples ou « POJO »), l'appel prend la forme suivante :

- récupération d'une connection au fournisseur via JNDI
- appel de la fonction findAddress

La fonction findAddress a la forme simplifiée suivante :

```
FindAddressResults findAddress(Address, FindAddressOptions);
```

C'est à dire que l'appel prend en entrée une adresse et des options et renvoie en sortie un certain nombre de candidats.

La décomposition précise est donnée ci-après.

Package com.geoconcept.ugc.service

Class CodingProvider

Méthode getConnection :

Cette méthode permet d'ouvrir une connexion sur le fournisseur de géocodage. Une connexion doit être par la suite fermée par sa méthode Close.

Prototype :

Connection getConnection() throws ResourceException;

Valeur retournée :

Connexion sur le fournisseur de service de géocodage.

Class Connection

Méthode findAddress :

Cette méthode permet de géocoder une adresse avec d'éventuelles options de géocodage.

Prototype

FindAddressResults findAddress(Address address, FindAddressOptions options)

throws InvalidDataSourceException , InternalErrorException;

Paramètre Description

address Adresse à géocoder

options Options de géocodage

Valeur retournée

Liste résultat du géocodage.

Package com.geoconcept.common.geo

class Location

Cette classe contient les coordonnées trouvées lors d'un géocodage.

Membres

Type	Nom	Description
double	x	Coordonnées X
double	y	Coordonnées Y
String	coordinateSystem	Identifiant de système de coordonnées. Identifiant "SRS" ("Spatial Reference System") de Web Map Service. Voir la norme OpenGIS.

Package com.geoconcept.ugc

class Address

Cette classe contient la description d'une adresse.

Membres

Type	Nom	Description
String	addressLine	Concaténation du numéro, du type de voie et du nom de la voie. Exemple : 25 rue de Tolbiac
String	zone	Code de la ville(exemple : le code postal avec 75013)
String	cityName	Nom de la ville (exemple : Paris)

Type	Nom	Description
String	uniqueId	Code unique { ville, zone } de recherche (exemple : 75113000)
String	secondaryZone	Code secondaire de l'adresse (exemple : le code INSEE, le code IRIS…)
String	StreetSectionId	Identifiant de tronçon (non utilisé pour l'instant)

class FindAddressOptions

Cette classe contient le paramétrage d'un géocodage.

Tous les éléments, à part dataSourceName sont optionnels. Si ils ne sont pas affectés ils prennent leur valeur par défaut qui est optimale dans la grande majorité des cas.

Type	Nom	Description
String	dataSourceName	Nom de la source de données défini dans l'administrateur à utiliser pour les options de géocodage
short	candidateCount	Nombre maximum de résultats à renvoyer lors du géocodage
String	findType	Type de géocodage souhaité. Il existe trois types de géocodage : à la ville (FindAddressResults.FIND_TYPE_CITY), à la rue (FindAddressResults.FIND_TYPE_STREET) , au numéro (FindAddressResults.FIND_TYPE_STREET_NUMBER)
String	tolerateFindType	Cette propriété permet de fixer et d'obtenir la valeur cumulée des géocodages tolérés. Il existe trois niveaux de tolérance : - à la ville (FindAddressResults.TOLERATE_TYPE_CITY) dans le cas où la rue n'a pu être trouvée et que la ville n'est pas couverte (nombre de rues de cette ville non significatif dans la table de référence), - à la rue (FindAddressResults.TOLERATE_TYPE_STREET) dans le cas où le numéro demandé n'a pu être trouvé, - au numéro estimé(FindAddressResults.TOLERATE_TYPE_STREET_ENHANCE), si le numéro n'a pu être trouvé et que l'erreur du placement n'excède pas la valeur de "maxMeterError". Les tolérances sont des "flags" que l'on peut cumuler. Par exemple, si l'on tolère ces trois catégories, on fixe un numéro de tolérance à 7, tandis qu'une tolérance uniquement à la ville correspond à une tolérance de valeur 1. Le tableau suivant récapitule les possibilités existantes :
long	maxMeterError	Erreur de placement maximale (en mètres) pour considérer un géocodage à la rue comme un géocodage au numéro. Le type de géocodage dépend donc de la longueur d'une rue,
double	discrepency	Décalage orthogonal (en mètres) à la rue trouvée à appliquer. Ceci permet de ne pas placer l'adresse géocodée sur le milieu de la voie
double	discrepancyAlongStreet	Décalage longitudinal (en mètres) à appliquer. Ceci permet de ne pas placer l'adresse géocodée sur un carrefour
long	favorCityMatchElement	Permet d'améliorer la recherche de la ville en spécifiant l'élément de la ville (le nom de la ville, le code de la ville ou le code unique de la ville dont on est certain de la nature des données. L'affectation de cette valeur dépend donc de l'adresse à géocoder. Trois valeurs sont possibles : le nom de la ville (FindAddressResults.FAVOR_CITY_NAME), le code de la ville (FindAddressResults.FAVOR_ZONE), le code unique de la ville (FindAddressResults.FAVOR_UNIQUE_ID)
long	zoneMatchDigits	Permet de fixer le nombre de caractères à utiliser pour l'appariement entre le code de la ville stockée dans la table de référence et celui passé en paramètre de géocodage.

Type	Nom	Description
		Une valeur peut être spécifiée uniquement si le membre favorCityMatchElement est différent de (FindAddressResults.FAVOR_ZONE), il faut utiliser l'intégralité du code postal.
int	scoreThreshold	Score minimal des propositions du géocodeur à retenir
int	scoreThreshold ToTolerateStreet GeocodingType	Défini un score minimum pour que des candidats de type « rue » soient renvoyés. Si aucune rue n'atteint ce seuil, alors la commune sera retournée comme candidat
String	coordinateSystem	Défini le Système de référence pour des coordonnées à retourner

Class FindAddressResults

Classe qui contient les résultats d'un géocodage, classés par score.

Membres

Type	Nom	Description
FindAddressResult[]	results	Tableau de candidats trouvés
int	matchType	Type de géocodage effectué

Constantes

Type	Nom	Valeur	Description
int	FIND_MATCH_TYPE_CITY	1	Type de géocodage demandé : l'adresse doit être géocodée à la ville
int	FIND_MATCH_TYPE_STREET	2	Type de géocodage demandé : l'adresse doit être géocodée à la rue
int	FIND_MATCH_TYPE_STREET_NUMBER	3	Type de géocodage demandé : l'adresse doit être géocodée au numéro de rue
int	FOUND_MATCH_TYPE_CITY	1	Type de géocodage résultat : l'adresse a été géocodée à la ville
int	FOUND_MATCH_TYPE_STREET	2	Type de géocodage résultat : l'adresse a été géocodée à la rue
int	FOUND_MATCH_TYPE_STREET_ENHANCED	3	Type de géocodage résultat : l'adresse a été géocodée au numéro de rue approché
int	FOUND_MATCH_TYPE_STREET_NUMBER	4	Type de géocodage résultat : l'adresse a été géocodée au numéro de rue exact
int	TOLERATE_TYPE_CITY	1	Tolérance du géocodage à la ville
int	TOLERATE_TYPE_STREET	2	Tolérance du géocodage à la rue
int	TOLERATE_TYPE_STREET_ENHANCED	4	Tolérance du géocodage au numéro estimé
int	FAVOR_CITY_NAME	1	Oriente la recherche de la ville à géocoder avec le nom de la ville
int	FAVOR_ZONE	2	Oriente la recherche de la ville à géocoder avec le code de la ville
int	FAVOR_UNIQUE_ID	3	Oriente la recherche de la ville à géocoder avec le code unique de la ville

class FindAddressResult

Classe qui contient le résultat d'un géocodage.

Membres

Type	Nom	Description
Address	address	Adresse trouvée
Location	location	Coordonnées trouvées
double	score	Score de ressemblance attribué entre l'adresse à géocoder et l'adresse trouvée. Varie entre 0 (aucune ressemblance) et 1 (adresse exacte)
int	type	Type de géocodage trouvée. Les valeurs possibles sont : - FindAddressResults.FOUND_MATCH_TYPE_CITY - FindAddressResults.FOUND_MATCH_TYPE_STREET - FindAddressResults.FOUND_MATCH_TYPE_STREET_ENHANCED - FindAddressResults.FOUND_MATCH_TYPE_STREET_NUMBER

Utilisation

Principe d'utilisation

Pour réaliser un géocodage il faut effectuer les opérations suivantes :

- Connection au fournisseur de service de géocodage,
- Construction de l'adresse à géocoder,
- Construction des options de géocodage,
- Exécution du géocodage,
- Exploitation du résultat,
- Déconnexion du fournisseur de service de géocodage.

Exemple

Connection au fournisseur de service de géocodage

```
import com.geoconcept.ugc.service.CodingProvider;
import com.geoconcept.ugc.service.Connection;

/*
*Open a connection on the geocoding server
*/
public Connection getConnection() throws Exception
{
        Connection connection = null;
        try
        {
                // get context
                Context initCtx = new InitialContext();
                Context envCtx = (Context) initCtx.lookup("java:comp/env");

                // retrieves the geocoding server form the logical name
                CodingProvider codingProvider = (CodingProvider) envCtx.lookup("geoconcept/ugc/default");
                connection = codingProvider.getConnection();
        }
```

```
        catch (Exception e) { e.printStackTrace(); }
        return connection;
}
```

Construction de l'adresse à géocoder

```
import com.geoconcept.ugc.Address;

/*
*Construct an adress to geocode
*@param  addressLine address number + address  way type + address  way name. Sample : "25 rue de Tolbiac"
*@param   zone adress sone. Sample : "75013"
*@param  cityName city name. Sample : "Paris"
*@param  uniqueId city unique identifier. Sample : "75113000"
*
*/
public Address getAddressToGeocode(String  addressLine, String  zone,String cityName,String uniqueId)
throws Exception
{
        Address address = new Address();
        address.addressLine = addressLine;
        address.zone = zone;
        address.cityName = cityName;
        address.uniqueId = uniqueId;
        return  address;
}
```

Construction des options de géocodage

```
import com.geoconcept.ugc.FindAddressOptions;

/*
*Retrieves geocoding options from the data source defined in administration
*@param  datasource Name of a data source
*/
public FindAddressOptions  getOptions(String datasource) throws Exception
{
        FindAddressOptions options = new com.geoconcept.ugc.FindAddressOptions("myDataSource");
     return  options;
}
```

Exécution du géocodage

```
import com.geoconcept.ugc.service.Connection;
import com.geoconcept.ugc.Address;
import com.geoconcept.ugc.FindAddressOptions;
import com.geoconcept.ugc.FindAddressResults;

/*
*Launch geocode process and retrieves results
*@param  connection Connection to the geocode server
*@param  address Address to geocode
*@param   options Geocoding options
*/
public FindAddressResults  findGeocode(Connection connection, Address address , FindAddressOptions options)
 throws Exception
{
        FindAddressResults findAddressResults  = connection.findAddress(address, options);
        return  findAddressResults;
}
```

Exploitation du résultat

```
import com.geoconcept.ugc.FindAddressResults;
import com.geoconcept.ugc.FindAddressResult;
/*
*Browse geocoding results and display result
*@param  findAddressResults  Results of geocode process
*/
public void displayResult(FindAddressResults findAddressResults) throws Exception
{
        // if at least one found result
        if (findAddressResults.results.length > 0)
        {
                // Display colunm name
                system.out.println("Found results : ");
                system.out.println( "N°"+ "\t"
                                + "Geocoding Type"+ "\t"
                                + "Score"+ "\t"
                                + "Address line"+ "\t"
                                + "Zone"+ "\t"
                                + "City Name"+ "\t"
                                + "City Unique Identifier"+ "\t"
                                + "Adress Secondary Zone"+ "\t"
                                + "Coordinates (Coordinates System)");

                // For each found results
                for (int i = 0; i < findAddressResults.results.length; i++)
                {
                // get next found result
                       FindAddressResult findAddressResult = found.results[i];

                String  coordinateSystem = null;
                if (findAddressResult.location.coordinateSystem != null)
                {
                        coordinateSystem = findAddressResult.location.coordinateSystem;
                }
                else
                        coordinateSystem = "(Unknown)";

                        // Display result
                system.out.println( i + "\t"
                                + findAddressResult.address.type + "\t"
                                + findAddressResult.address.score + "\t"
                                + findAddressResult.address.addressLine + "\t"
                                + findAddressResult.address.zone + "\t"
                                + findAddressResult.address.cityName + "\t"
                                + findAddressResult.address.uniqueId + "\t"
                                + findAddressResult.address.secondaryZone + "\t"
                                + findAddressResult.location.x + ","+ findAddressResult.location.y
                                + "( + coordinateSystem + ")");
                }
        }
        else
        {
                system.out.println("No found result.");
        }
}
```

Déconnexion du fournisseur de service de géocodage

```
import com.geoconcept.ugc.FindAddressResult;
import com.geoconcept.ugc.service.Connection;
```

```
/*
*Disconnection of the geocode server
*@param  connection  Connection to the geocode server
*/
public void closeConnection(Connection connection) throws Exception
{
         connection.close();
}
```

Exemple complet

```
import com.geoconcept.ugc.service.CodingProvider;
import com.geoconcept.ugc.service.Connection;
import com.geoconcept.ugc.Address;
import com.geoconcept.ugc.FindAddressOptions;
import com.geoconcept.ugc.FindAddressResults;
public void geocodingSample()
{
        try
        {
                // Open connection
                Connection connection = getConnection();
                // Construct the address to geocode
                Address address = getAddressToGeocode("25 rue de Tolbiac","75013","Paris","");
                // retrieves geocode options
                FindAddressOptions options = getOptions("myDataSource");
                // launch geocode process
                FindAddressResults findAddressResults = findGeocode(connection, address , options);
                // print geocoding result.
                displayResult(findAddressResults);
                // disconnection
                closeConnection(connection);
        }
        catch(Exception e)
        {
                // geocoding problem
        }
}
```

SmartRouting Server

Installation du composant SmartRouting Server

L'installeur de Geoconcept Web propose d'installer les deux versions de SmartRouting Server :

- SmartRouting JEE,
- SmartRouting Command Line.

Ces versions sont indépendantes entre elles et répondent à des problématiques différentes. Par exemple, SmartRouting JEE est nécessaire pour le widget « Calcul d'itinéraires » dans les portails.

Mise en oeuvre des librairies natives

Pour Windows

Les librairies sont chargées explicitement depuis le sous-répertoire correspondant à l'environnement (32/64 bits) du répertoire externe configuré (RootDirectory) pour le fournisseur local (ConnectionMode="LocalDll").

Mise en oeuvre sous Linux

Seules les librairies linkées dynamiquement sont disponibles actuellement.

Installation pour utilisation des libraires linkées dynamiquement

Le chargement des libraires de dépendance linkées dynamiquement nécessite que la variable système LD_LIBRARY_PATH contienne le répertoire des librairies qui vont être utilisées. La manière d'affecter la variable d'environnement LD_LIBRARY_PATH dépend du shell, du mode de lancement du serveur d'application, etc. De même, la valeur affectée varie selon l'OS hôte, le répertoire de déploiement, le mode de chargement (décrit plus loin).

Par exemple, pour un lancement manuel de tomcat via sh ou compatible (bash, etc) on entrera export LD_LIBRARY_PATH=/home/smartrouting/native/linux32-x86/debian/3.1/dynamic ou export LD_LIBRARY_PATH=/deploy/smartrouting/native/linux64-amd64/red-hat/5.6/dynamic puis startup.sh du répertoire bin de tomcat (la variable est affectée pour la session).

Il est aussi possible de modifier le fichier catalina.sh afin de définir la variable LD_LIBRARY_PATH pour tous les lancements de tomcat, ou bien encore de modifier les fichiers /etc/rc.d/… pour une exécution automatique.

Le fichier service.xml situé dans le répertoire conf de l'arborescence externe au serveur d'application (%smartrouting%) défini le mode de chargement des librairies natives (sous Linux uniquement car sous Windows le mode est toujours le même) dans la section native-libraries.

```
<?xml version="1.0" encoding="UTF-8"?>
<service>
  [...]
  <native-libraries>
    <log-enabled>false</log-enabled>
    <always-detect>true</always-detect>
    <detect-method>auto</detect-method>
```

```
    <last-detect />
    <!--force loading libraries from given directory under ugc/native eg 'linux64-amd64/my-dir' : -->
    <dir></dir>
    <link-mode>dynamic</link-mode>
  </native-libraries>
    […]
```

Le répertoire depuis lequel l'adaptateur de ressource (de type ConnectionMode="LocalDll") chargera les librairies est défini soit de manière automatique (par détection) soit configuré manuellement. C'est la valeur de l'élément dir qui le détermine : si il est défini et non vide, alors c'est la valeur de l'élément qui sera utilisée comme sous-répertoire du répertoire native de l'arborescence externe au serveur d'application.

Si dir est défini à la valeur « mydir/test » et que le RootDirectory de l'adaptateur de ressource est défini à la valeur « /home/smartrouting » alors le répertoire utlisé pour le chargement sera / home/smartrouting/native/mydir/test.

Dans le cas où dir est défini (chargement forcé explicitement), les paramètres qui configurent le chargement automatique par détection n'ont pas d'effet : link-mode, detect-mode, always-detect.

Dans le cas du chargement automatique par détection (dir est indéfini ou vide), le sous-répertoire du répertoire native de l'arborescence externe au serveur d'application qui sera utilisé dépend :

- de l'environnement linux (32 / 64),
- de la distribution linux (debian, red hat),
- de la version de la distribution linux,
- du mode de link des librairies (static/dynamic) – actuellement seul le mode dynamic est disponible.

Quel que soit le mode de définition (forcé explicitement via native-libraries/dir ou détecté automatiquement) du répertoire qui sera utilisé, en mode de link dynamique (uniquement), il est nécessaire que la variable d'environnement LD_LIBRARY_PATH contienne ce répertoire.

Déploiement et configuration du serveur d'application

Merci de vous reporter au paragraphe « Configuration pour SmartRouting JEE » dans le manuel de Geoconcept Web.

Guide de référence de SmartRouting Server

SmartRouting est un moteur de calcul d'itinéraires et de recherche de proximité performant. Les calculs d'itinéraire sont basés sur un fichier décrivant le réseau routier appelé graphe. Ce fichier peut être créé à partir du SIG Geoconcept et d'une base de navigation routière, comme celle de Navteq. Il est possible de définir des profils de vitesses (piéton, poids lourds) et de prendre en considération diverses règles de ciculation (sens interdits, interdictions de tourner, …). SmartRouting Server permet d'effectuer diverses opérations :

- caclul d'un itinéraire entre deux points,
- calcul d'un itinéraire entre un point et plusieurs points,

- création d'une feuille de route,
- recherche d'entités à proximité d'un point de référence (Search Around).

SmartRouting permet d'obtenir un itinéraire complet avec la description de la route (y compris le chemin que la route suit entre deux intersections), la durée de parcours de chaque segment et la distance.

Cette documentation présente ce composant et est décomposée en deux parties distinctes :

- SmartRouting JEE : il s'agit d'un composant destiné à l'intégration du moteur de calcul d'itinéraires SmartRouting à la plate-forme Java Enterprise Edition (JEE) et ses sous-ensembles comme le moteur de servlet Tomcat.
- SmartRouting Command Line : il s'agit d'un composant destiné à calculer des itinéraires en ligne de commande.

Principes généraux du moteur de calcul d'itinéraires SmartRouting

Le graphe : toute fonction de calcul nécessitant un parcours de réseau nécessite l'utilisation d'un graphe. Le graphe constitue une vue logique du réseau. Il contient et décrit toutes les informations relatives aux connectivités entre les tronçons, aux coûts et aux contraintes. Il est construit et organisé de façon que les calculs s'effectuent de manière performante. C'est un fichier au format SITI.

Ce format SITI correspond à la seconde génération de graphes développée pour Geoconcept (à partir de Geoconcept 6.5). Il intègre potentiellement de nombreuses caractéristiques supplémentaires (multiples profils de vitesses, nombreuses règles de gestion …).

Merci de vous reporter au guide de référence de la solution Geoconcept pour les règles de création d'un graphe.

L'accrochage au graphe

Lors des calculs, les points de départs, les étapes et les arrivées sont reliés au réseau constitué par le graphe, même si les objets sont distants de celui-ci (points situés à une certaine distance du réseau) : c'est l'accrochage au graphe. Cette opération vise à rechercher autour de ces points les tronçons les plus proches. Ensuite, ces tronçons sont utilisés pour déterminer les débuts de solution possibles.

L'itinéraire calculé intègre alors une distance parcourue entre ces points et le graphe, appelée distance d'accrochage au graphe. Pour optimiser les performances, il existe une option permettant de fixer une distance maximale d'accrochage au graphe, exprimée en mètres (*max graph snap distance*). Cela évite de parcourir tout le réseau pour trouver les tronçons les plus proches.

En outre, la vitesse d'accrochage au graphe, vitesse de parcours de la distance séparant les départs, étapes ou arrivées du graphe peuvent également être personnalisées (*graph snap speed*).

Le métagraphe

Un métagraphe consiste dans la possibilité d'intégrer à la création du graphe un graphe simplifié ou allégé d'une partie de son contenu. L'utilisation du métagraphe vise essentiellement à accélérer le parcours du graphe, lors de calculs d'itinéraires s'effectuant sur de longues distances.

Si un itinéraire doit être calculé entre deux villes distantes de plusieurs centaines de kilomètres, il est préférable de privilégier un parcours du graphe sur les niveaux de routes principaux (tendant vers le niveau logique 1, considéré comme le niveau le plus important). Autour des points de départs, étapes éventuelles et arrivées, les niveaux logiques non présents dans le métagraphe sont bien empruntés, mais pour la liaison de ville à ville, les niveaux du métagraphe sont privilégiés.

Le métagraphe est un fichier accompagnant le graphe SITI. Il est constitué de l'association d'un fichier au format MG et d'un fichier MGLIST. Le fichier MG est chargé en mémoire.

Le reference-level

Il s'agit du niveau logique sur lequel le calcul d'itinéraires va se faire principalement. Deux cas sont possibles :

- en utilisant un métagraphe : il s'agit du cas « normal », le métagraphe étant spécifiquement construit pour accélérer le calcul d'itinéraires sur de longues distances (seules les routes importantes seront sélectionnées sur les longs trajets). Il est nécessaire de renseigner le niveau logique défini dans le métagraphe pour optimiser le temps de calcul,
- sans utilisation de métagraphe : le calcul d'itinéraires sera optimisé en interrogeant prioritairement les tronçons du niveau spécifié dans le graphe directement.

Autour des points de passage (départ, étapes, arrivée), toutes les routes seront prises en compte, peu importe le niveau logique choisi.

Un niveau logique trop élevé peut dégrader le calcul d'itinéraires car de nombreuses routes ne seront pas prises en compte.

Les coordonnées en entrée et en sortie

Il est possible de spécifier les coordonnées en entrée et en sortie de SmartRouting Server.

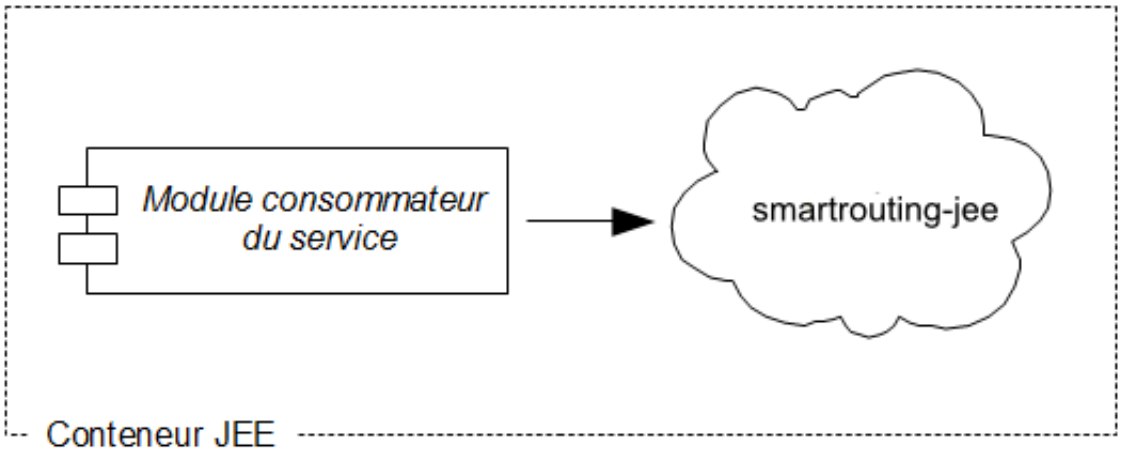
SmartRouting JEE

Principes

Intégration JEE

SmartRouting JEE est un composant d'intégration à la plate-forme JEE, il offre un service de calcul d'itinéraires aux modules JEE déployés sur un serveur d'applications (au sens large, moteur de servlet type Tomcat compris).

Intégration JEE



Le module consommateur peut être de n'importe quel type (webapp, ejb, etc). De même à l'intérieur du module JEE, le consommateur peut être de n'importe quel type (servlet, jsp, pojo, etc).

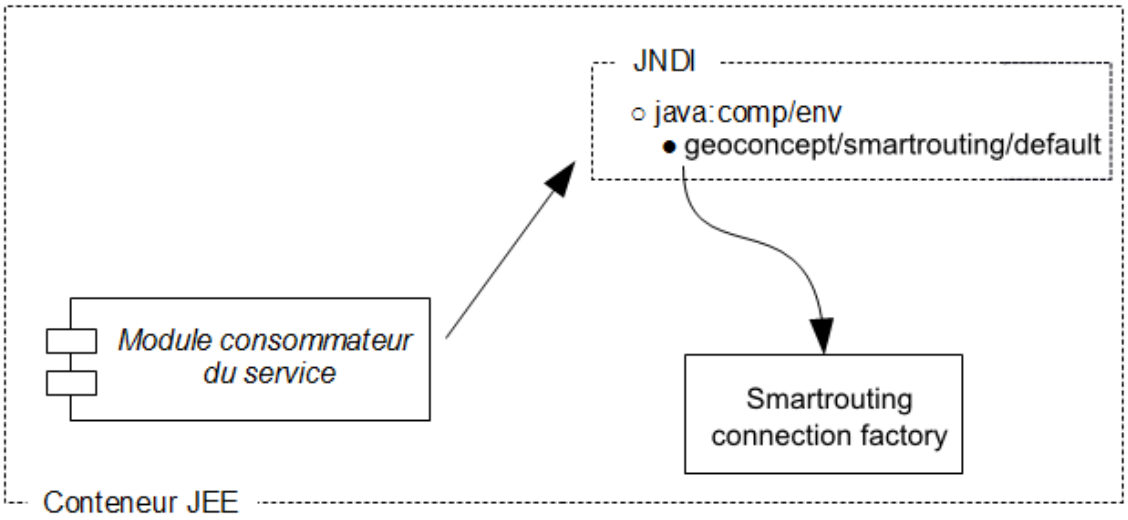
Mode d'accès

L'application utilisant le calcul d'itinéraires référence le fournisseur via un nom logique de l'annuaire JNDI (Java Naming and Directory Interface) du serveur d'applications. La manière de monter le fournisseur sera décrite plus loin.

Traditionellement, le nom logique utilisé est « geoconcept/smartrouting/default » du contexte « java:comp/env » mais il est possible d'utiliser un autre nom, un autre contexte ou de monter plusieurs fournisseurs de types différents.

Par soucis de simplicité, le cas d'utilisation le plus courant (un seul fournisseur primaire local avec nommage par défaut) sera décrit dans un premier temps.

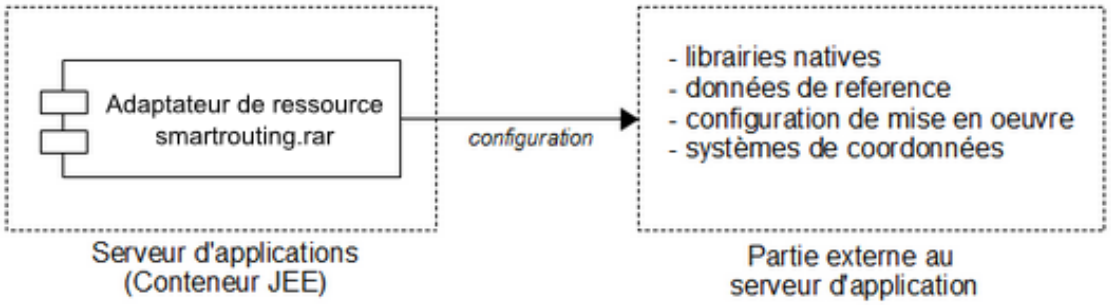
Mode d'accès



Organisation du composant : Adaptation de ressource JEE et éléments externes

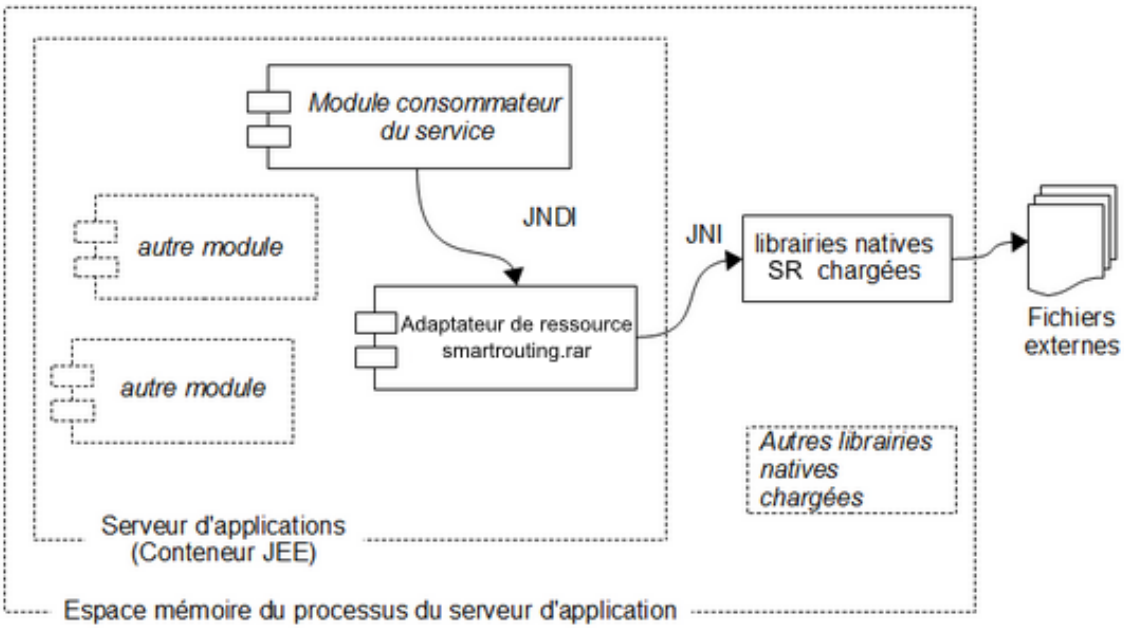
smartrouting-jee se compose de plusieurs parties. Pour des questions de performances et de réutilisation, le moteur (aussi appelé noyau) est écrit en C++ et est donc diffusé sous forme de librairies natives. La partie intégration (adaptateur de ressource) réalise le lien avec le moteur en gérant sa mise en oeuvre par chargement de ses librairies natives. En terme de déploiement de fichiers cela concerne donc deux arborescences distinctes :

Organisation du composant



En terme d'exécution, les librairies natives seront chargées en mémoire dans le processus de l'instance du serveur d'application (ou sa partition éventuelle selon les modèles) au démarrage. La partie adaptateur de ressource expose une interface java et gère le moteur instancié en mémoire directement via JNI.

Organisation du composant



Modules additionnels

Aucun module additionnel n'est indispensable au bon fonctionnement de l'adaptateur de ressource et leur déploiement est optionnel.

smartrouting-admin

smartrouting-admin est une webapp permettant la vérification de la bonne installation du produit, sa configuration et le test de bon fonctionnement.

Administration / Configuration

Configuration de mise en oeuvre

Un graphe (fichier .siti) publié constitue une source de données (datasource).

Le fichier service.xml situé dans %smartrouting%/conf définit la configuration générale du fournisseur de service de calcul d'itinéraires. Celui ci contient notamment une configuration par défaut pour les sources de données (*default#datasource*). Il est possible de définir une configuration spécifique à une source de données. Pour connaître les paramètres, vous pouvez consulter le chapitre sur les options.

Si aucune configuration spécifique n'a été définie alors la configuration par défaut (*default-datasource*) s'applique à la mise en oeuvre de cette source de données.

Par ailleurs il est possible de définir certains paramètres lors de l'appel (via CalculateRouteOptions). Ainsi la valeur concrète que prend un paramètre peut provenir (dans l'ordre) :

- de l'appel si il est affecté dans CalculateRouteOptions (cette affectation est optionnelle)
- de la valeur indiquée au niveau de la configuration de datasoure particulière (si une configuration de datasource particulière a été définie)
- de la valeur indiquée au niveau de la configuration de datasoure par défaut (`default-datasource`).

Configuration détaillée d'une source de données

Il est possible de définir la configuration de mise en oeuvre d'un graphe de manière très détaillée.

Cette définition peut répondre à des besoins particuliers, elle n'est pas indispensable car la configuration par défaut convient pour la grande majorité des cas.

Certains paramètres sont définissables au moment de l'appel, d'autres sont fixés à la création de l'instance de source de données et non modifiables par la suite.

Les paramètres que l'on peut définir aussi au moment de l'appel seront décrits dans la section CalculateRouteOptions.

Datasource identification

Cette partie permet l'identification d'une source de données dans l'administration.

File : graphe,

Name : alias de nom de la source de données (optionnel).

Options disponibles dans SmartRouting JEE

Les options disponibles sont celles décrites dans le guide de référence de SmartRouting.

Elles sont accessibles visuellement à travers l'application de test smartrouting-admin.war, qui présente chacune une entrée pour voir son comportement dans le résultat.

Coordonnées en entrée et en sortie

Si les coordonnées en entrée ne sont pas dans le même système de coordonnées que le graphe, il est nécessaire de les spécifier via le paramètre `input coordinate system` (sous la forme epsg:4326 par exemple).

Les coordonnées en sortie seront dans le même système que le système de coordonnées du graphe, sauf si le paramètre `output coordinate system` est renseigné du code EPSG (sous la forme epsg:27572 par exemple).

Les résultats en sortie

Il est possible, outre les segments composants l'itinéraire calculé, de recevoir comme résultat des attributs des-dits segments, grâce aux champs fields et language.

- fields : permet de sélectionner les champs qu'on souhaite retourner (par défaut, tous les champs sont retournés si l'option est activée. Le nom des champs doit être écrit en anglais),
- language : permet de retourner ces champs dans la langue souhaitée.

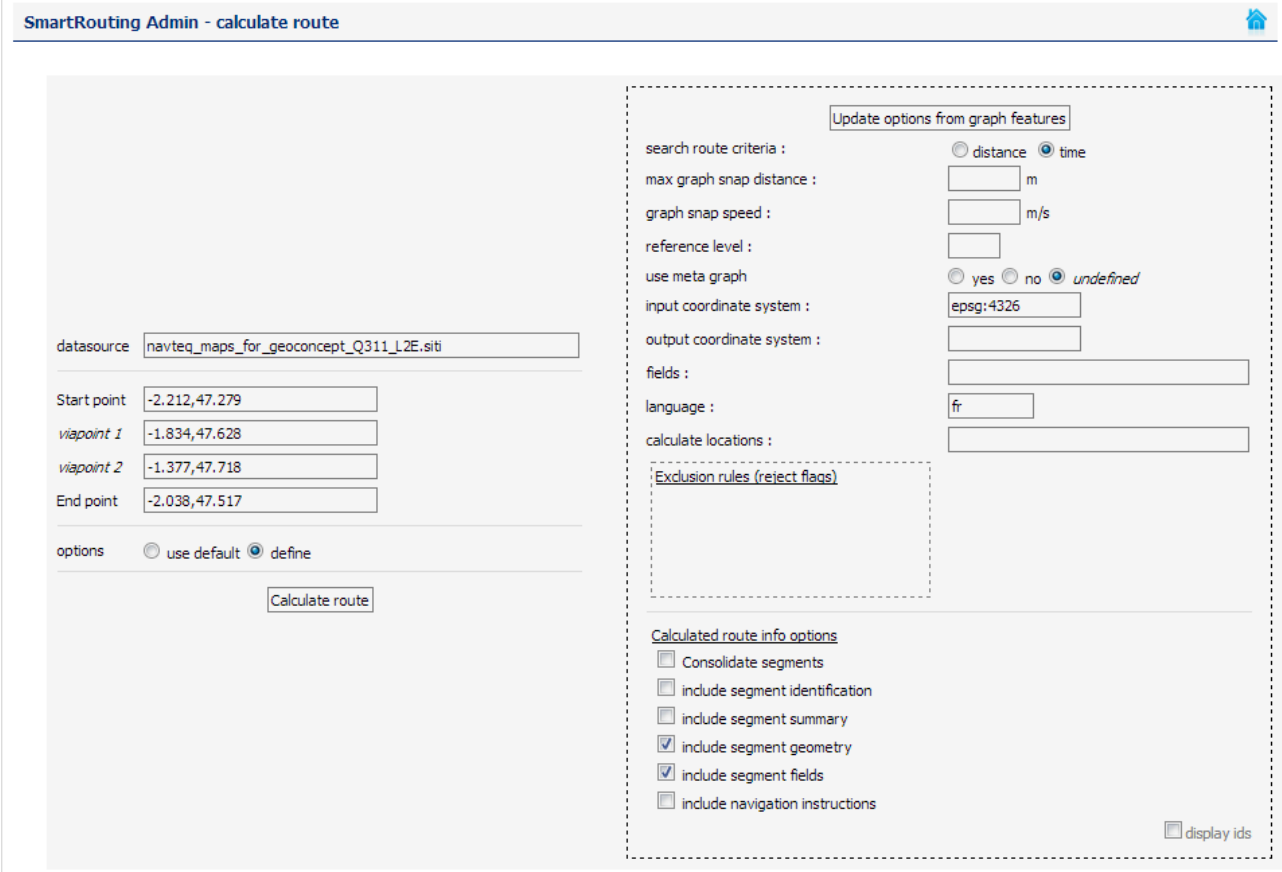
Calculate locations

Cette option permet de trouver un point sur l'itinéraire calculé à partir du point de départ. La valeur Calculate locations en secondes permet de trouver ce point à partir du début de l'itinéraire.

Consolidate segments

Cette option permet de regrouper, dans le résultat du calcul d'itinéraires, les segments qui portent le même nom et qui sont proches en un seul segment (utile pour la navigation).

Exemple d'itinéraire : coordonnées en entrée en WGS 84, coordonnées en sortie en Lambert 2 Etendu (système de projection du graphe), tous les champs sont retournés en français



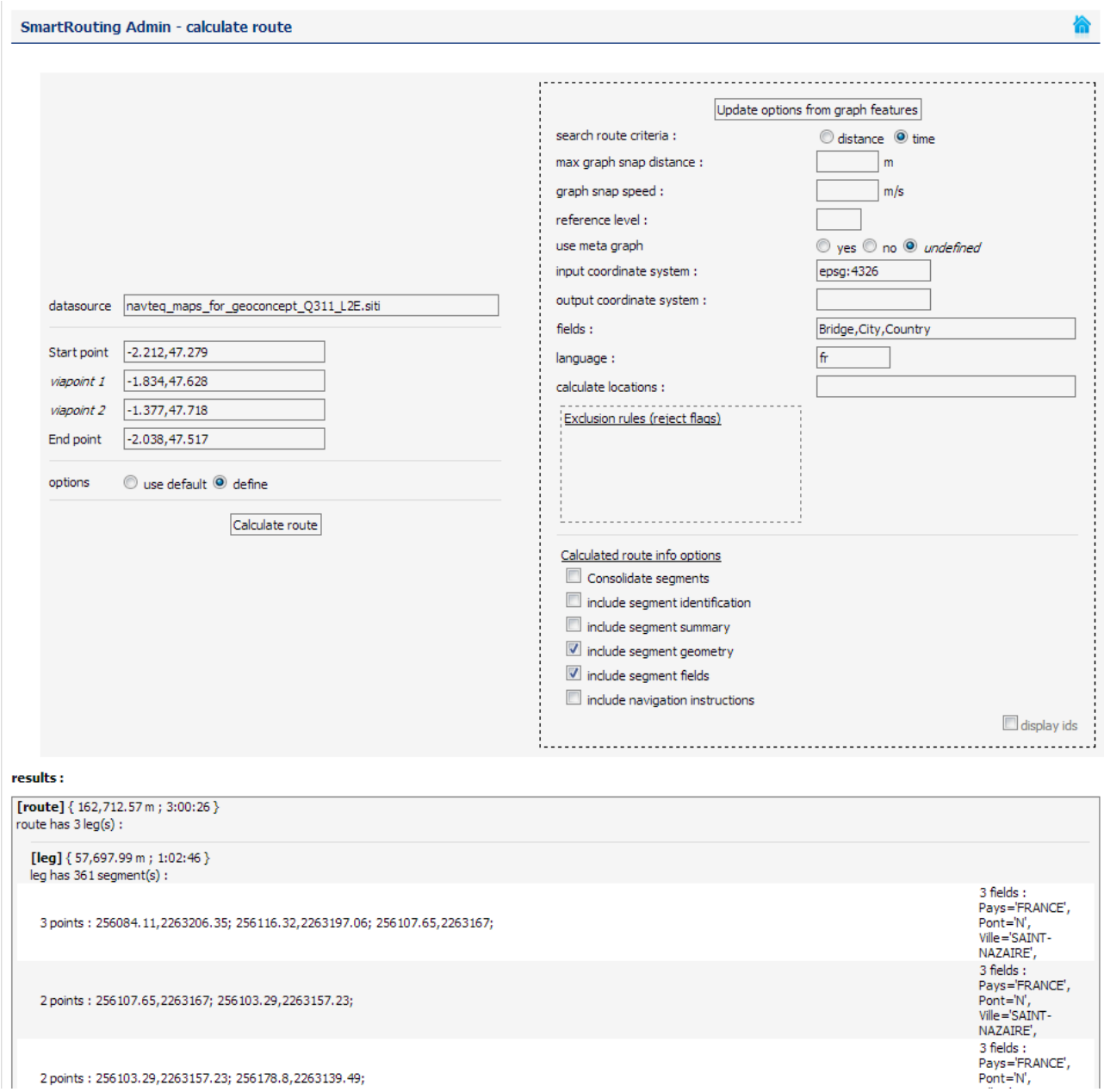
results :

[route] { 162,712.57 m ; 3:00:26 }
route has 3 leg(s) :

[leg] { 57,697.99 m ; 1:02:46 }
leg has 361 segment(s) :

3 points : 256084.11,2263206.35; 256116.32,2263197.06; 256107.65,2263167;	11 fields : Bretelle='N', Classement administratif='', Code Postal='44600', Contre-allée='N', Pays='FRANCE', Pont='N', Péage='N', Tunnel='N', Urbain='Y', Ville='SAINT-NAZAIRE', automobiles='Y',
2 points : 256107.65,2263167; 256103.29,2263157.23;	11 fields : Bretelle='N', Classement administratif='', Code Postal='44600', Contre-allée='N', Pays='FRANCE', Pont='N', Péage='N', Tunnel='N', Urbain='Y', Ville='SAINT-NAZAIRE', automobiles='Y',
2 points : 256103.29,2263157.23; 256178.8,2263139.49;	11 fields : Bretelle='N', Classement administratif='', Code Postal='44600', Contre-allée='N', Pays='FRANCE', Pont='N', Péage='N', Tunnel='N', Urbain='Y',

Exemple d'itinéraire : coordonnées en entrée en WGS 84, coordonnées en sortie en Lambert 2 Etendu (système de projection du graphe), champs retournés : Bridge, City, Country en français



Options avancées

Le fichier `%smartrouting%/smartrouting/conf/service.xml` permet de spécifier la configuration pour tous les graphes.

Vous pourrez trouver la configuration des graphes que vous possédez dans la page Datasources configuration, en cliquant sur le lien info. L'utilisation ou non d'un métagraphe dans le graphe est affichée en bas de la page.

SmartRouting Command Line

SmartRoutingCommandLine, ou SRCmdLine est un programme indépendant dont le principe est inspiré de UGCCmdLine, qui permet de calculer une série de d'itinéraires en une seule fois.

Données en entrée

Les itinéraires à calculer sont présents dans un fichier en entrée (txt) :

- Une ligne par itinéraire, avec dans l'ordre :
  - X Départ
  - Y Départ
  - X Arrivée
  - Y Arrivée
  - TypeDeCalcul(« 1 » ou « Shortest » pour un calcul au plus court chemin, « 2 » ou « Fastest » pour un calcul au plus rapide)
- Chacune de ces valeurs doit être séparée d'un ' ;'
- La dernière valeur, TypeDeCalcul, est optionnelle (si non spécifié, SRCmdLine prend la valeur spécifiée dans sa config : voir plus bas)

Fichier résultat

Les résultats des calculs d'itinéraire sont écrits dans un fichier résultat :

- Une ligne par itinéraire, avec dans l'ordre :
  - La distance en mètres de l'itinéraire,
  - Le temps en secondes de l'itinéraire,
  - Le temps en millisecondes du calcul d'itinéraire (le traitement),
  - Si aucun itinéraire n'a pu être calculé, les valeurs seront remplacées par des « N.A. »
- Chacune de ces valeurs est séparée d'un « ; »

Lancer le traitement

SRCmdLine est un outil qui se lance en ligne de commande depuis command line.

- Pour afficher l'aide : « SRCmdLine.exe –help »
- Pour afficher la version du produit : « SRCmdLine.exe –version »
- On doit ou on peut spécifier une série de paramètres (certains sont obligatoires, d'autres juste optionnels) qui seront utilisés pour configurer SmartRouting et donc l'ensemble des calculs d'itinéraires.
- Les valeurs de ces paramètres peuvent être soit définis dans un fichier de configuration au format XML, soit définis directement dans la ligne de commande.
- On spécifie le fichier de config de la manière suivante : SRCmdLine –config <CheminCompletVersLeFichier>
- Voici la liste des paramètres qui peuvent être spécifiés :
  - Fichier d'entrée (obligatoire) :

    - Ligne de commande : -in <CheminCompletVersLeFichierDEntrée>
    - XML : <smartrouting-configuration><smartrouting-input><path>
- Fichier de sortie (ou de résultats) (obligatoire) :
    - Ligne de commande : -out <CheminCompletVersLeFichierDeSortie>
    - XML : <smartrouting-configuration><smartrouting-output><path>
- Graphe (obligatoire) :
    - Ligne de commande : -graph <CheminCompletVersLeFichierGraphe>
    - XML : <smartrouting-configuration><smartrouting><connection-distance>
- Distance maximale de recherche d'accrochage (optionnel) :
    - Ligne de commande : -connectionDistance <value>
    - XML : <smartrouting-configuration><smartrouting><connection-distance>
- Vitesse d'accrochage (optionnel) :
    - Ligne de commande : -connectionSpeed <value>
    - XML : <smartrouting-configuration><smartrouting><connection-speed>
- Niveau de reference (optionnel) :
    - Ligne de commande : -refLevel <value>
    - XML : <smartrouting-configuration><smartrouting><reference-level>
- Type de calcul d'itinéraire (optionnel) :
    - Ligne de commande : -itiMode <value>
    - XML : <smartrouting-configuration><smartrouting><iti-mode>
    - Pour un calcul au plus court : value=1 ou value=Shortest
    - Pour un calcul au plus rapide : value=2 ou value=Fastest
- Chargement complet du graphe (optionnel) :
    - Ligne de commande : -fullLoadGraph <value>
    - XML : <smartrouting-configuration><smartrouting><full-load-graph>
    - Chargement complet : value=1
    - Chargement partiel : value=0
- Système de coordonnée en entrée (optionnel) :
    - Ligne de commande : -inputProj <codeEPSG>
    - XML : <smartrouting-configuration><smartrouting><input-projection>

Exemple d'appel au programme :

- SRCmdLine -graph "data/graph.siti" -in "data/input.csv" -out "output.csv" -config "config.xml"
    - Si un paramètre est défini à la fois dans le fichier de config et dans la ligne de commande, la valeur spécifiée dans la ligne de commande l'emporte.
    - Si le type de calcul d'itinéraire (plus rapide ou plus court) est défini à la fois dans le fichier d'entrée et dans le fichier de config ou la ligne de commande, la valeur spécifiée dans le fichier d'entrée l'emporte.

Exemple de fichier de configuration config.xml

```
<?xml version="1.0" encoding="utf-8"?>
<smartrouting-configuration>
  <!-- Smart Routing configuration -->
  <smartrouting>
    <!-- Graph file path-->
    <!-- <graph-path>E:\SmartRouting-cmdline\SRCmdLine\navteq_maps_for_geoconcept_Q311_france_v2.siti</
graph-path> -->
        <graph-path>E:\SmartRouting-cmdline\graphe.siti</graph-path>
    <!-- Mode of routing. 1:Distance, 2:Time -->
    <iti-mode>1</iti-mode>
    <!-- Set connection distance -->
    <connection-distance>150.</connection-distance>
        <full-load-graph>1</full-load-graph>
        <reference-level>3</reference-level>
  </smartrouting>
  <!-- Input configuration (text file format) -->
  <smartrouting-input>
    <!-- Path of the input file. -->
    <path>E:\SmartRouting-cmdline\SRCmdLine\input.csv</path>
  </smartrouting-input>
  <!-- Ouput configuration (text file format) -->
  <smartrouting-output>
    <!-- Path of the output file.  -->
    <path>E:\SmartRouting-cmdline\SRCmdLine\output.csv</path>
  </smartrouting-output>
</smartrouting-configuration>
```