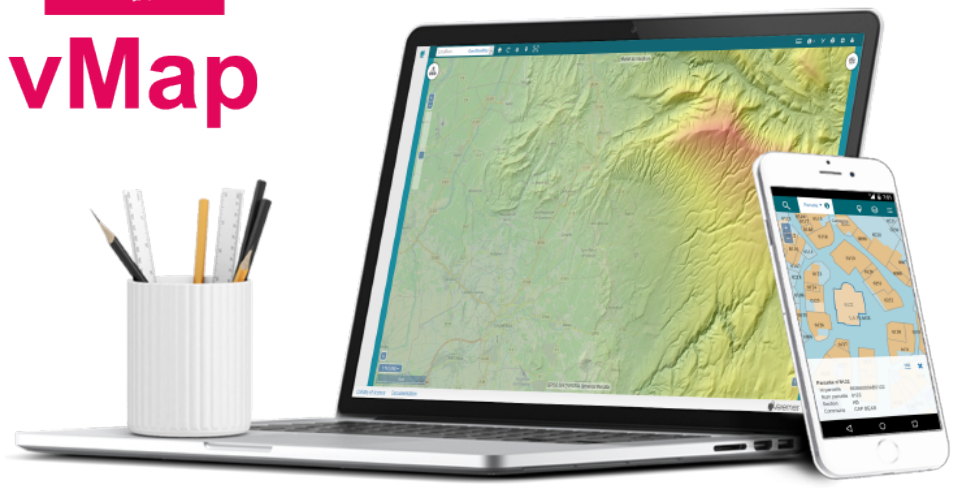
vMap Documentation

Version master

mai 26, 2021

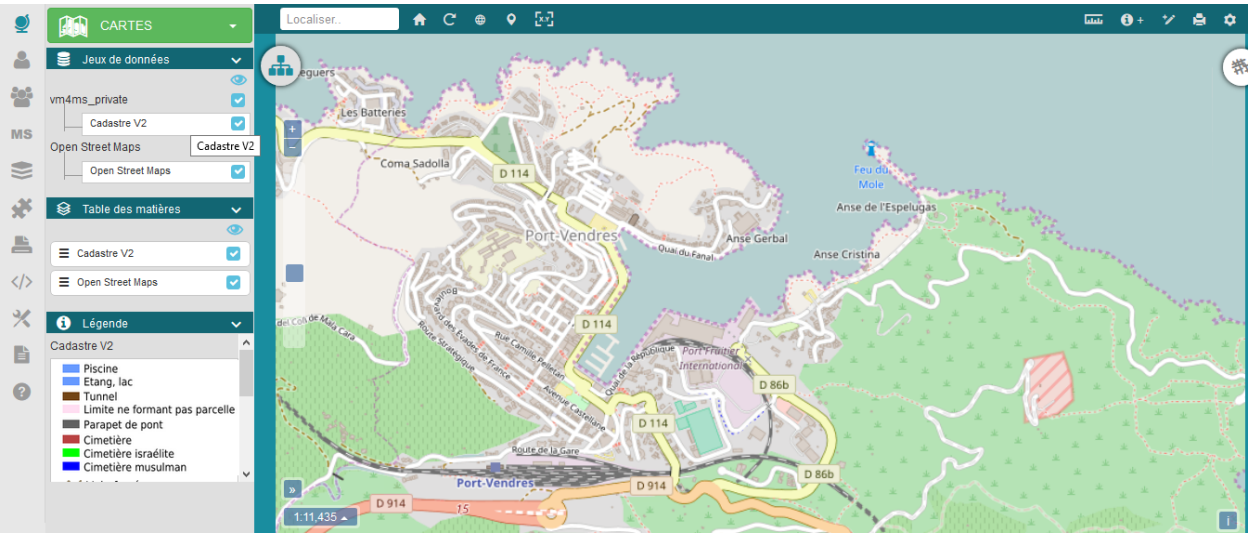
Table des matières

vMap



CHAPITRE 1

Mode cartographie



1.1 1. Définition

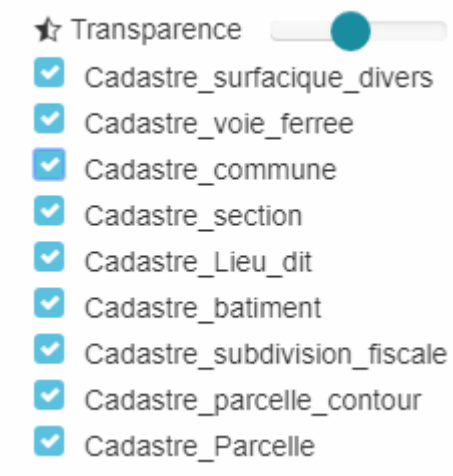
Le mode visualisation cartographique, accessible aux utilisateurs en ayant droits (vmap_user) permet l'"affichage des cartes. La liste des cartes disponibles pour l'"utilisateur connecté dépend des groupes auxquels il appartient.

1.2 2. La gestion des cartes

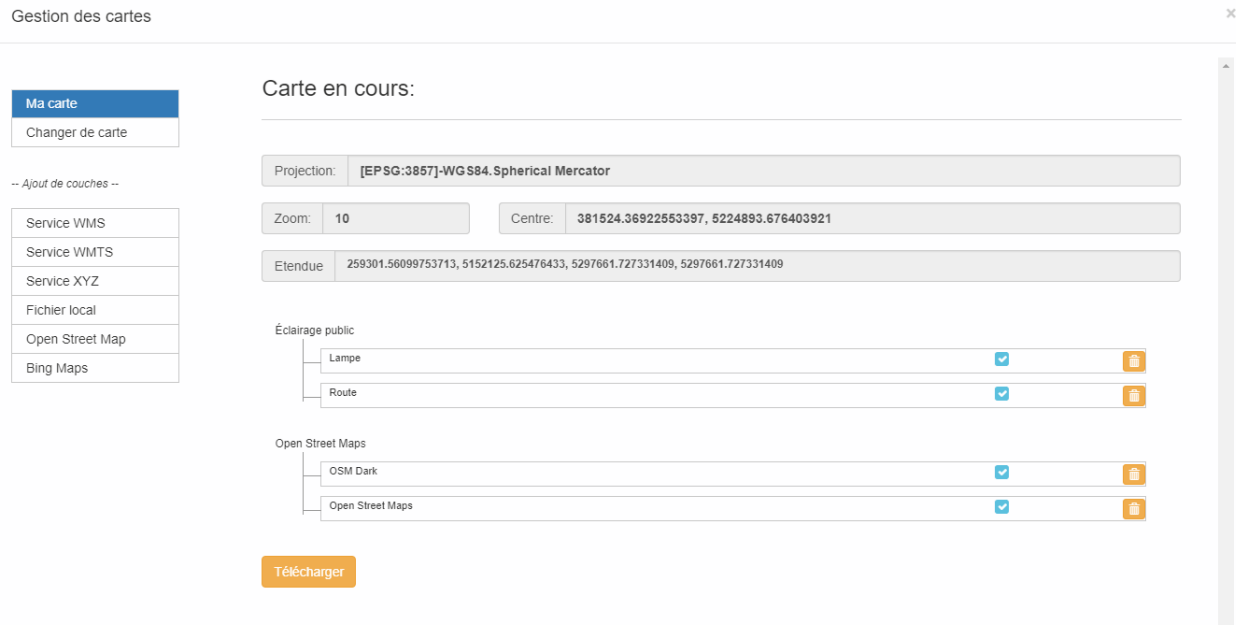
Le bouton Carte  permet de déployer :

— la table des matières

— la légende
— les jeux de données affichés sur la carte en cours. Chaque jeu de données est comosé de 1 à plusieurs calques qu"il est possible d"afficher en cliquant sur le bouton . Une boîte à cocher permet d"afficher/désafficher



les calques indépendamment les uns des autres :
— le Gestionnaire des cartes. Le gestionnaire de cartes permet de sélectionner la carte à afficher et d"y opérer des opérations d"ajout de couches à la volée. Les couches peuvent être issues de service ou de fichier local.

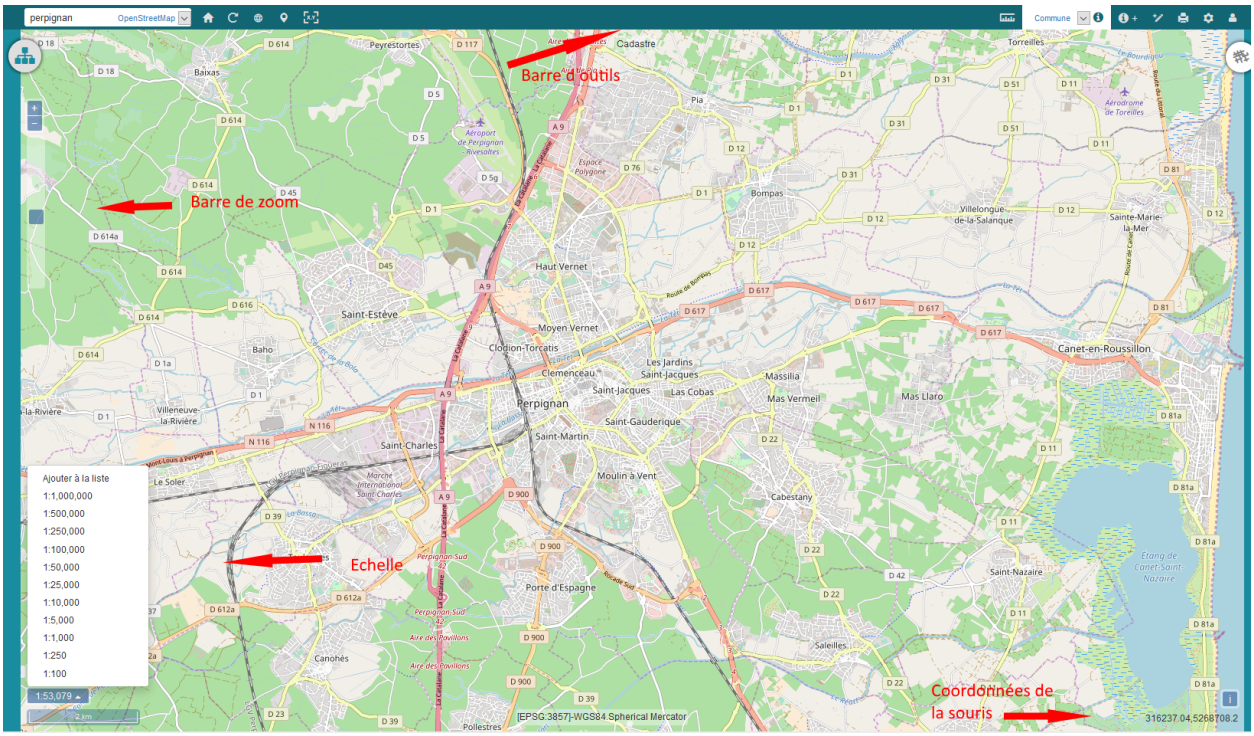


L"ajout de couches dépend des services sélectionnés. L"ajout de l"url du service permet un ajout à la volée de la couche sur la carte en cours. Il est possible d"ajouter des fichiers locaux au format :
— KML
— TopoJSON
— IGC
— GeoJSON
— GPX

Le nom de la couche est facultatif. C"est le nom affiché dans le jeu de données et la table des matières. Si aucun nom de couche n"est fourni, alors le nom du fichier et son extension sont affichés.

1.3 3. Volet carte : Outils d'affichage, de sélection, d'interrogation et de filtre

On retrouve dans la Fenêtre « Carte » l'ensemble des fonctionnalités classiques d'un web SIG mais aussi plusieurs fonctionnalités propres à vMap.



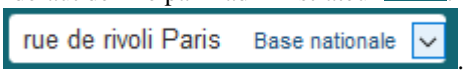
- La barre de zoom sur le côté gauche (le zoom peut aussi être effectué via la souris)
- L'échelle et l'overview en bas à gauche
- Les coordonnées de la souris en bas à droite
- Les listes des outils de contrôle en haut à droite  . Les outils de contrôle peuvent être activés ou désactivés à la volée par l'utilisateur
- La liste des modèles d'impressions disponibles pour l'utilisateur connecté  . L'ensemble des champs paramétrables pour les impressions sont définis, préalablement par l'administrateur, dans un modèle configuré.
- Un outil d'insertion d'une donnée  . Il donne accès au formulaire de création d'objet. Pour cela l'administrateur doit avoir préalablement paramétré un objet métier associé au calque de la donnée.
- Un outil de sélection multiple  qui donne accès a 2 modes de recherche :
  - Une sélection graphique à partir des outils point, ligne, polygone et cercle
  - une sélection attributaire à partir d'un requêteur. Un objet métier doit obligatoirement avoir été associé au calque de la carte.
- Un outil de sélection simple  permettant d'obtenir les informations attributaire d'un seul et unique objet sélectionné géographiquement.
- Un outil de mesure  qui permet le calcul de longueurs, superficies et mesures des géométries. Il permet l'export des mesures au format csv.
- Un outil de localisation  à partir des coordonnées X et Y et d'un système de projection.

— Un outil de géolocalisation  qui permet de centrer la carte sur la localisation de l'utilisateur en cours.

— Un outil pour centrer la carte sur l'étendue maximale  . L'étendue maximale d'une carte diffère en fonction du système de projection. Si la carte est en Lambert 93, l'étendue maximale de la carte est la France.

— Un outil pour rafraichir les couches de la carte sans avoir à recharger l'application  .

— Un outil pour recentrer la carte sur l'emprise par défaut définie par l'administrateur  .

— Un outil de localisation par la saisie d'adresse. 

. Par défaut l'outil fonctionne avec la couche Open Street Map. Si un objet métier est associé à un calque de la carte, un choix sera disponible entre plusieurs localisations.

1.3.1 L'outil de localisation

L'outil de localisation peut réaliser la recherche de plusieurs façon :

— avec la clause LIKE

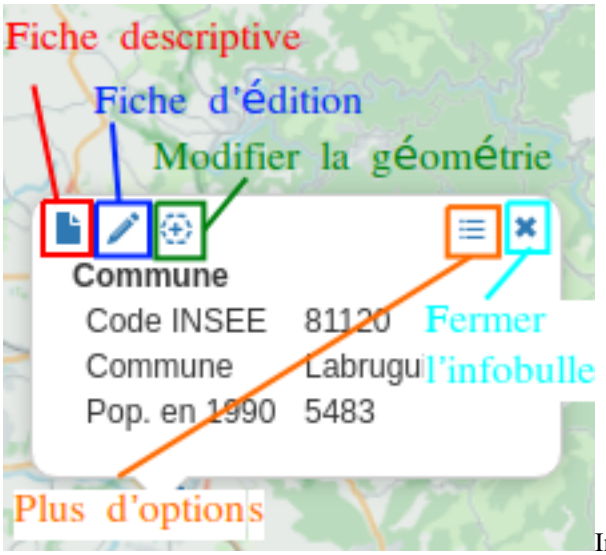
— avec la fonctionnalité plain text de postgresql

Pour configurer ce choix, il faut se rendre dans la partie administration de l'objet métier

1.3.2 3.1. Infobulles

Une infobulle est la carte d'identité d'un objet métier, pour en selectionner une ou plusieurs (si l'option est activé) il faut choisir le type d'objet métier à sélectionner 

puis cliquer sur un élément de la carte.



Infobulle

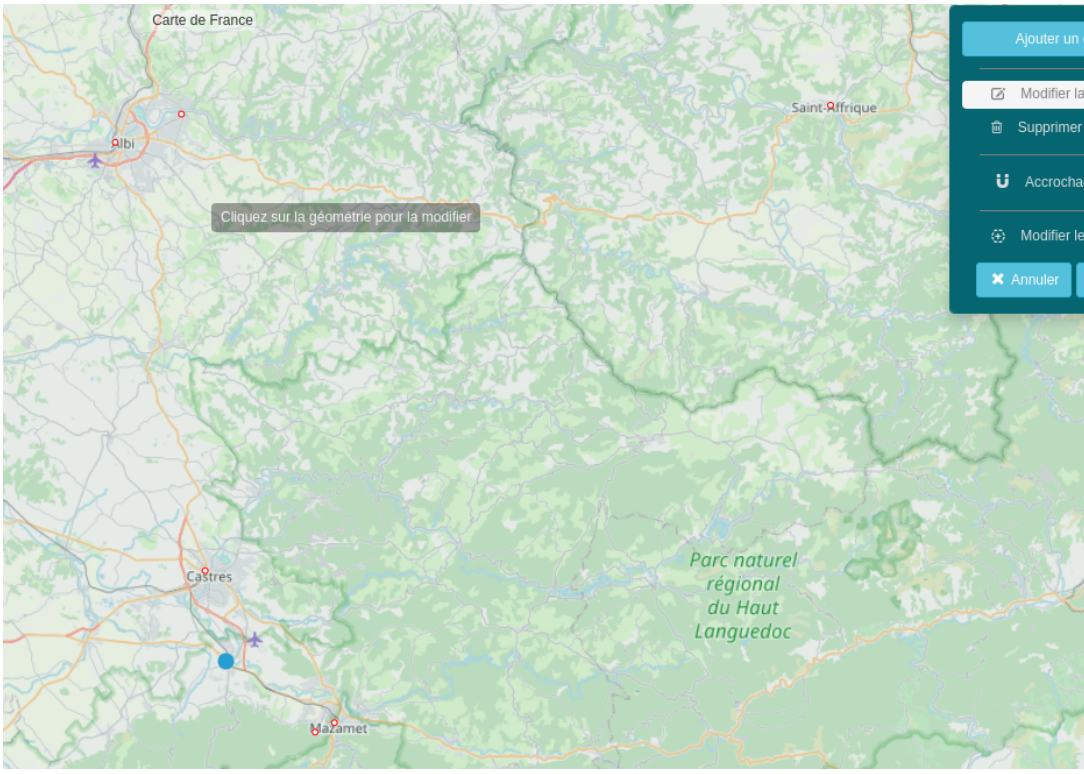
Fiche descriptive

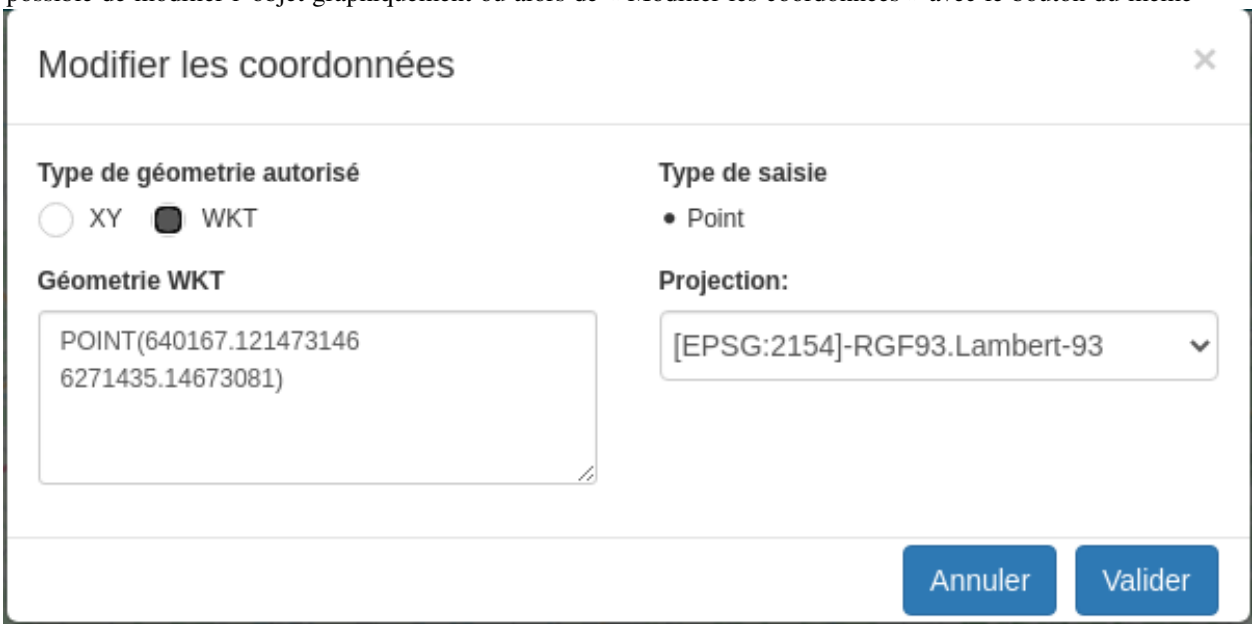
Ouvre le formulaire de description de l'objet métier

Fiche d'édition

Ouvre le formulaire d'édition de l'objet métier

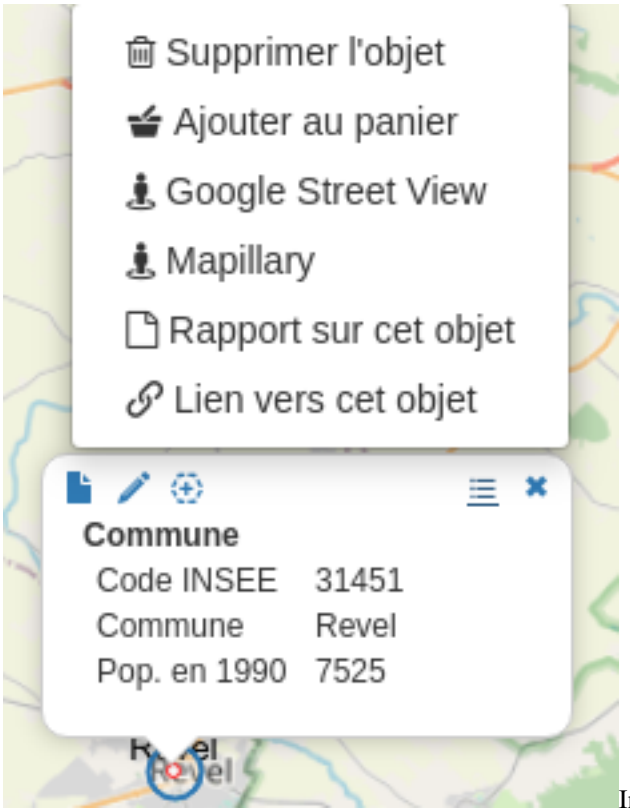
Modifier la géométrie

Ouvre l'outil d'édition graphique 

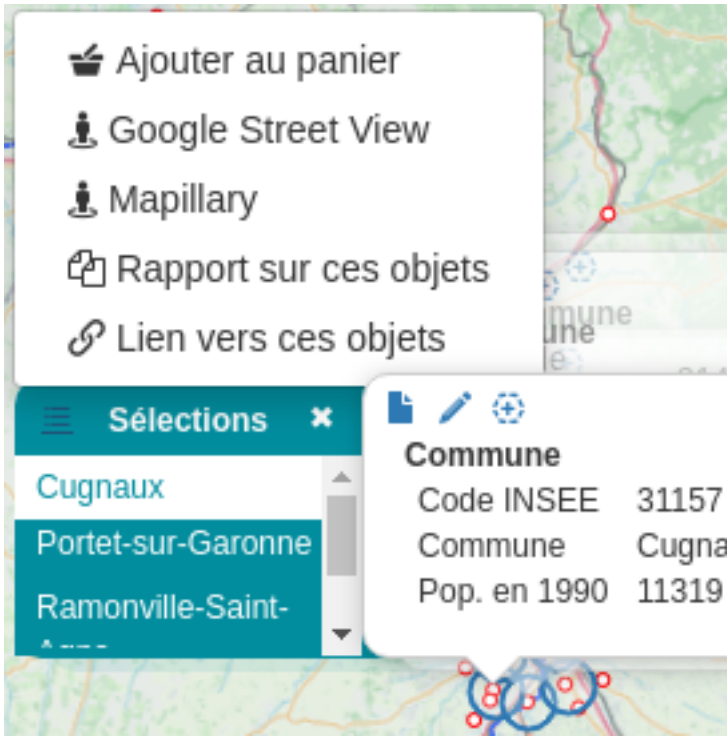
Il est possible de modifier l'objet graphiquement ou alors de « Modifier les coordonnées » avec le bouton du même nom. 

Plus d"options

Les options disponibles dans les infobulles sont paramétrables. Plus d"information sur la configuration



Infobulle



Les options suivantes sont disponibles pour un enregistrement :

- Supprimer l"objet : Supprime l"enregistrement de la base de données.
- Ajouter au panier : Ajoute le ou les enregistrement.s au panier.
- Google Street View : Ouvre un nouvel onglet de votre navigateur avec open street view positionné par rapport aux coordonnées de l"enregistrement.
- Mapillary : Ouvre un nouvel onglet de votre navigateur avec mapillary positionné par rapport aux coordonnées de l"enregistrement.
- Rapport sur cet.ces objet.s : ouvre une fenetre pour choisir le rapport à imprimer (plus de détails ici)
- Lien vers cet.ces objet.s : copie dans le presse papier un lien (un URL) pour accéder directement à cet objet.

1.4 4. Accès direct

1.4.1 4.1. Visualisation d"une zone

Il est possible de charger vMap sur une zone spécifique en renseignant l"étendu ou alors des coordonnées long/lat

- map_id : l"identifiant de la carte à afficher (récupérable dans la liste des cartes)
- extent : étendue à renseigner au format xmin|ymin|xmax|ymax
- lon : longitude (requiert lon, lat et zoom)
- lat : latitude (requiert lon, lat et zoom)
- zoom : seuil de zoom (entre 0 et 28) (requiert lon, lat et zoom)

Exemple : https ://demo.veremes.net/vmap ?mode_id=vmap&map_id=-1&extent=633212.2672198378%7C6852982.667582236%7C67

1.4.2 4.2. Visualisation d”objet spécifique

Il est possible de visualiser un objet spécifique d”une carte en indiquant dans l”url, les paramètres suivants séparés par des &. Chacun de ces paramètres est facultatif

- map_id : l”identifiant de la carte à afficher (récupérable dans la liste des cartes)
- bo_id : identifiant de l”objet métier à interroger
- ids : le ou les identifiants des entités à afficher (nécessite bo_id)

Un zoom est effectué sur l”objet défini en paramètre de l”url et ce dernier est centré sur la carte qui s”affiche.

Exemple : https ://demo.veremes.net/vmap/ ?map_id=-1&bo_id=veremes_parcelles_visualisation&ids=66366000AB0003|66366000AB

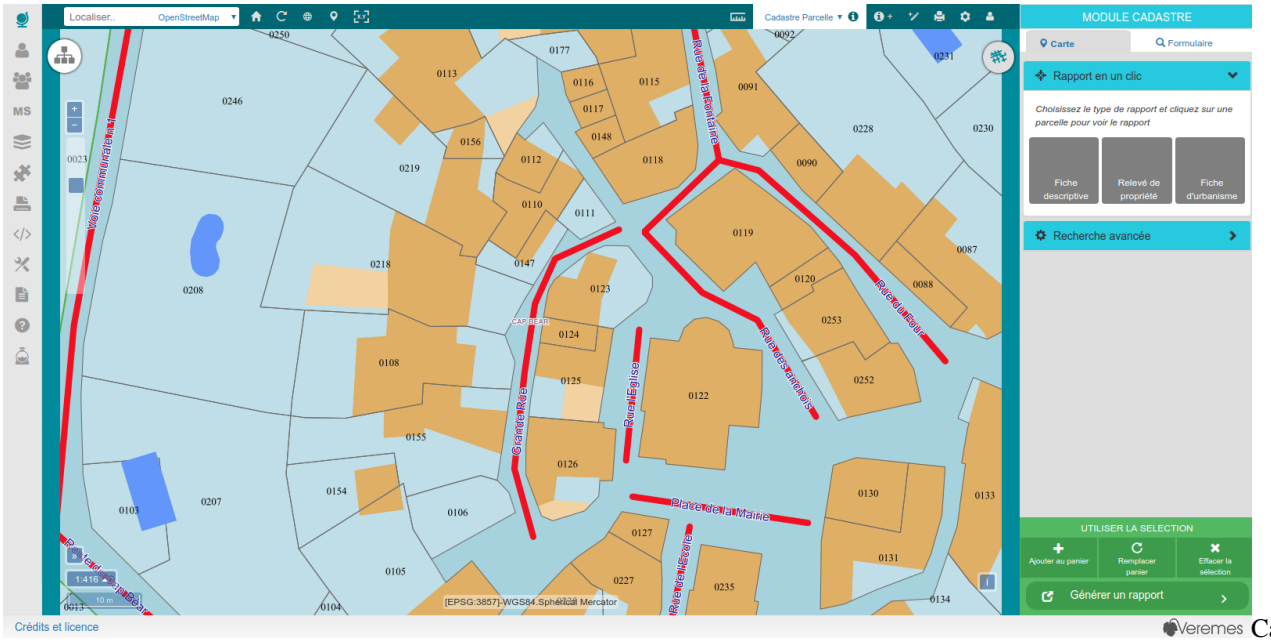
CHAPITRE 2

Module cadastre

Le module Cadastre est un module spécifique à vMap intégré dans l'interface cartographique. Le but du module Cadastre est de rechercher soit cartographiquement soit de manière attributaire des entités cadastrales (parcelles, sections, bâtis etc..) pour les visualiser, consulter leurs informations et générer des rapports spécifiques.

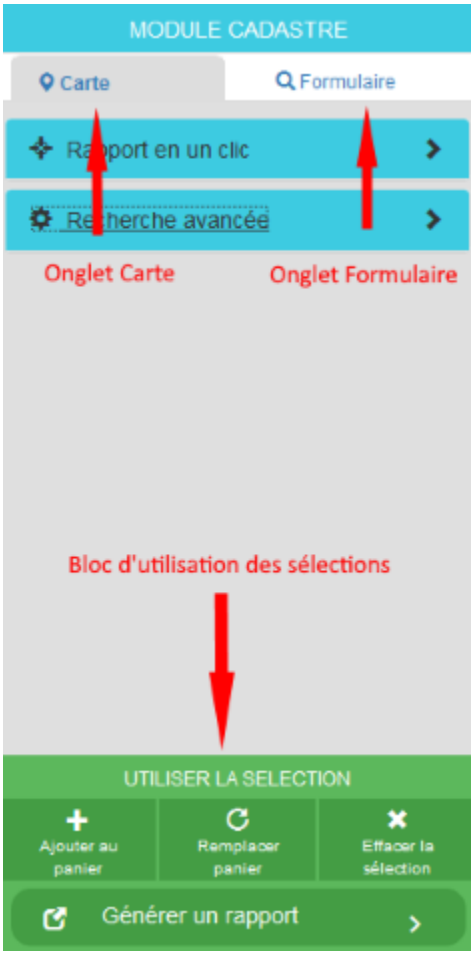
Le module Cadastre est accessible aux personnes en ayant droit, c'est à dire à tout utilisateur pour lequel l'administrateur de l'application a attribué les droits nécessaires pour accéder au module. En savoir plus sur la gestion des utilisateurs et de leurs privilèges dans vMap.

Le bouton **module cadastre**  Bouton du module cadastre permet de déployer le module Cadastre à droite de la carte. Il comporte l'intégralité des fonctionnalités spécifiques à la recherche d'informations cadastrales de la carte Cadastre.



Carte cadastre

Le module Cadastre est composé des éléments suivants :



— L'onglet Carte permet la recherche graphique des composants du cadastre.

— L”onglet Formulaire permet la recherche attributaire des composants du cadastre.
— Le bloc « Utiliser la sélection » permet l”ajout et la suppression des éléments sélectionnés dans le panier, ainsi que la génération de rapports.

2.1 1. Recherche graphique : l”onglet Carte

L”onglet Carte du module Cadastre permet de sélectionner des éléments du cadastre directement sur la carte puis d”en générer un rapport. Il existe deux façons de sélection d”objets et de génération de rapports :
— Le rapport en un clic
— La recherche avancée

2.1.1 1 - Le rapport en un clic

Relatif à la parcelle, le « rapport en un clic » permet de générer des rapports parcelle par parcelle. Une fois une parcelle sélectionnée, ses données littérales peuvent être affichées par l”intermédiaire d”un panier  .

Après avoir choisi le type de rapport à générer, l”opérateur clique dans la carte sur la parcelle dont il veut extraire les informations.

3 types de rapports relatifs à une parcelle sélectionnée peuvent être générés : la fiche descriptive de la parcelle, le relevé de propriété et la fiche d”urbanisme

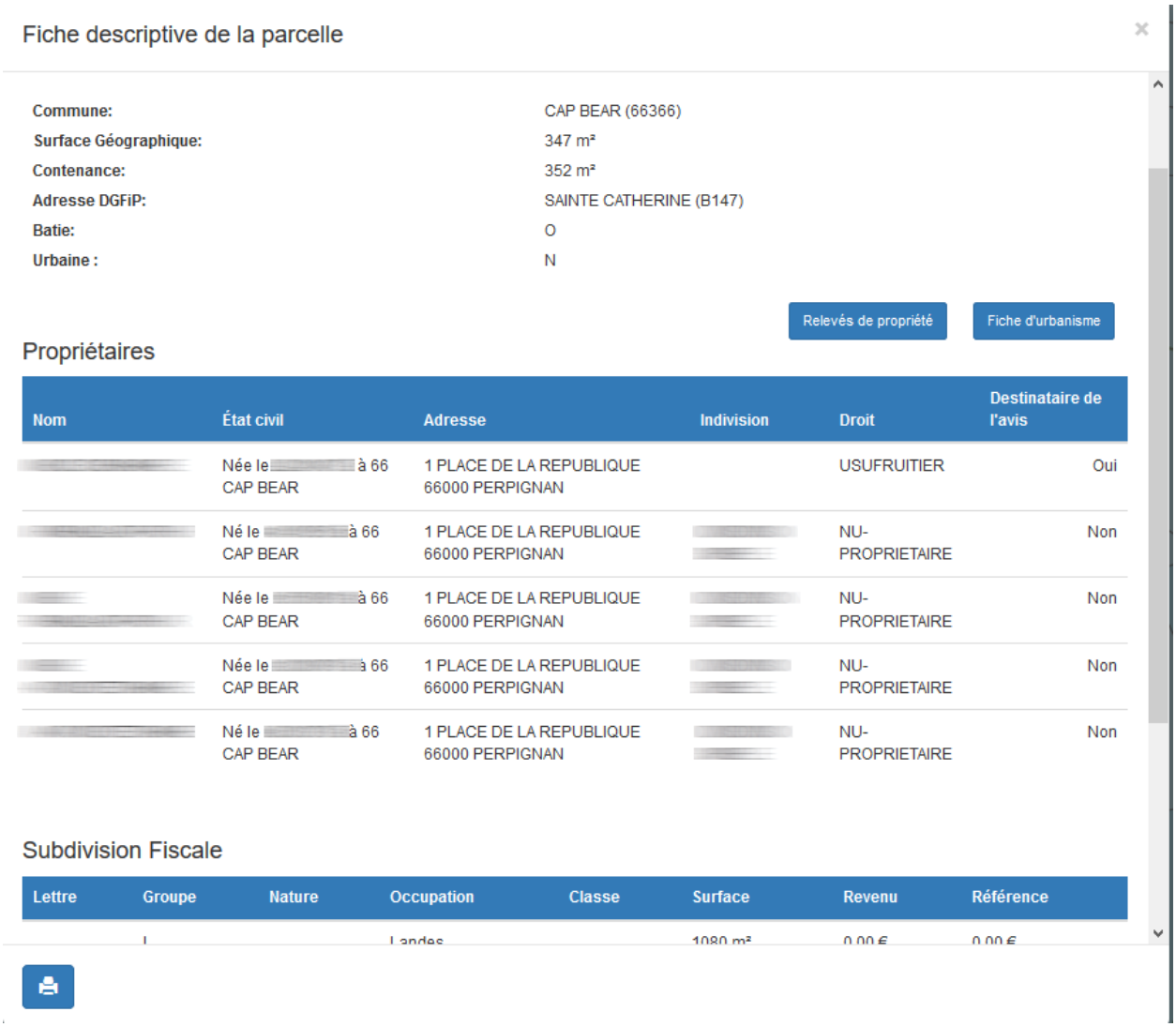
Fiche descriptive de la parcelle.

Cliquer sur le bouton Fiche descriptive 

, puis sélectionner sur la carte, la parcelle dont on souhaite extraire les informations cadastrales.

Les éléments constitutifs de la fiche descriptive d”une parcelle sont préalablement configurés par l”administrateur de l”application vMap. En savoir plus sur la configuration de la fiche descriptive d”une parcelle

La fiche descriptive fournit généralement les éléments suivants :
— le numéro, la superficie et la commune d”appartenance de la parcelle
— la liste des propriétaires de la parcelle
— la liste des subdivisions fiscales de la parcelle
— la liste des bâtis de la parcelle



Un bouton Impression en bas de la fenêtre permet d'"imprimer la fiche descriptive de la parcelle.

Les boutons Relevés de propriété et Fiche d'urbanisme en haut à droite de la fenêtre, permettent de générer directement ces deux rapports sans passer par le module Cadastre.

Le relevé de propriété

Cliquer sur le bouton Relevé de propriété Relevé de propriété, puis sélectionner la parcelle à interroger par un clic sur la carte.

3 types de relevés sont disponibles :

— Relevé de propriété standard : génération au format pdf, du relevé de propriété de la parcelle.
— Relevé de propriété tiers : génération au format pdf du relevé de propriétés tiers.
— Relevé de propriété de la parcelle : génération au format pdf du relevé de propriété de la parcelle sélectionnée.
— Relevé de propriété de la parcelle tiers : génération au format pdf du relevé de propriété de la parcelle tiers sélectionnée.

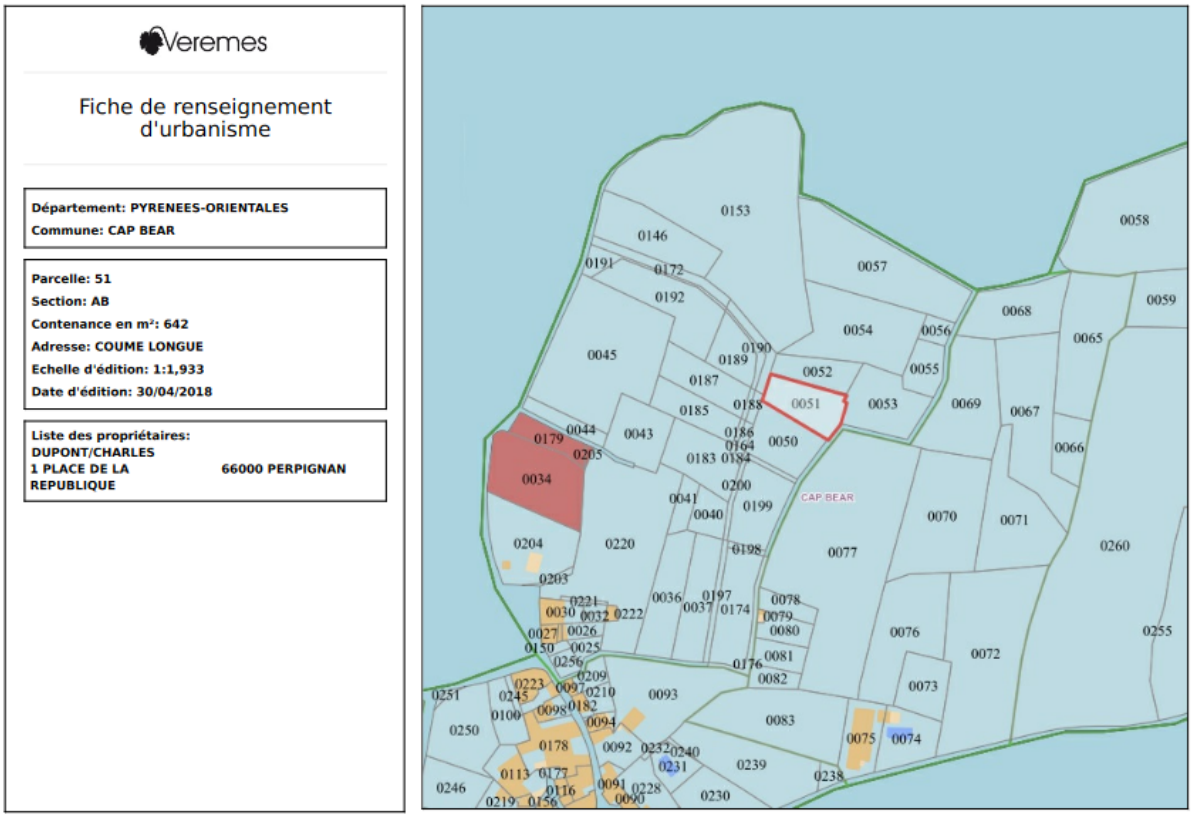
La fiche d'urbanisme

Cliquer sur le bouton Fiche d'urbanisme 

, puis cliquer sur la parcelle dont on souhaite extraire la fiche.

Les éléments constitutifs de la fiche d'urbanisme sont préalablement configurés par l'administrateur de l'application vMap. En savoir plus sur la configuration de la fiche d'urbanisme d'une parcelle.

Une fiche d'urbanisme est généralement constitutée des éléments suivants :
— Titre et en-tête
— Carte illustrant la parcelle sélectionnée dans sa commune d'appartenance
— Descriptif de la parcelle (numéro de section, numéro de parcelle. . . .)
— Liste des propriétaires de la parcelles
— . . .



2.1.2 2- La recherche avancée

La recherche avancée permet de sélectionner sur la carte un objet puis d'"en afficher ses données littérales par l'"intermédiaire d'"un panier  .

La recherche avancée s'"opère en 2 temps :

1. Sélection de l'"objet à rechercher

La première étape consiste à sélectionner le type d'"objet à requêter. Cliquer sur l'"un des objets suivants :

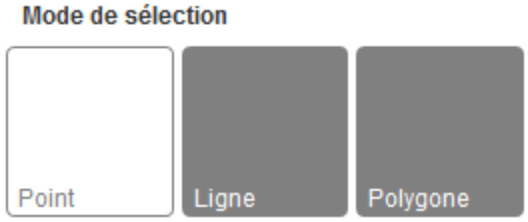
- la commune
- la section
- le lieu dit
- la parcelle
- le bâtis

Une fois sélectionné, le type d'"objet à interroger apparait en blanc :

Dans l'exemple ci-dessus, l'opérateur effectue une recherche des éléments relatifs à la commune.

2. Choix du mode de sélection graphique

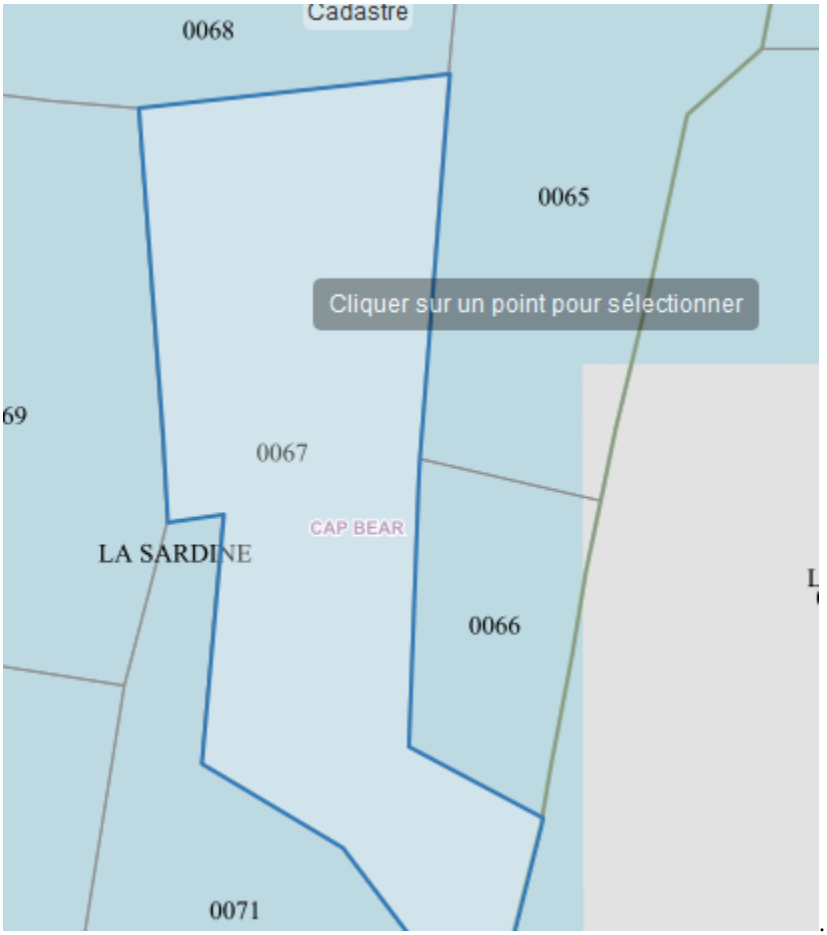
La deuxième étape consiste à choisir la façon dont les objets de la carte seront sélectionnés.



Dans l'exemple ci-dessus, l'opérateur effectue une recherche de commune par point.

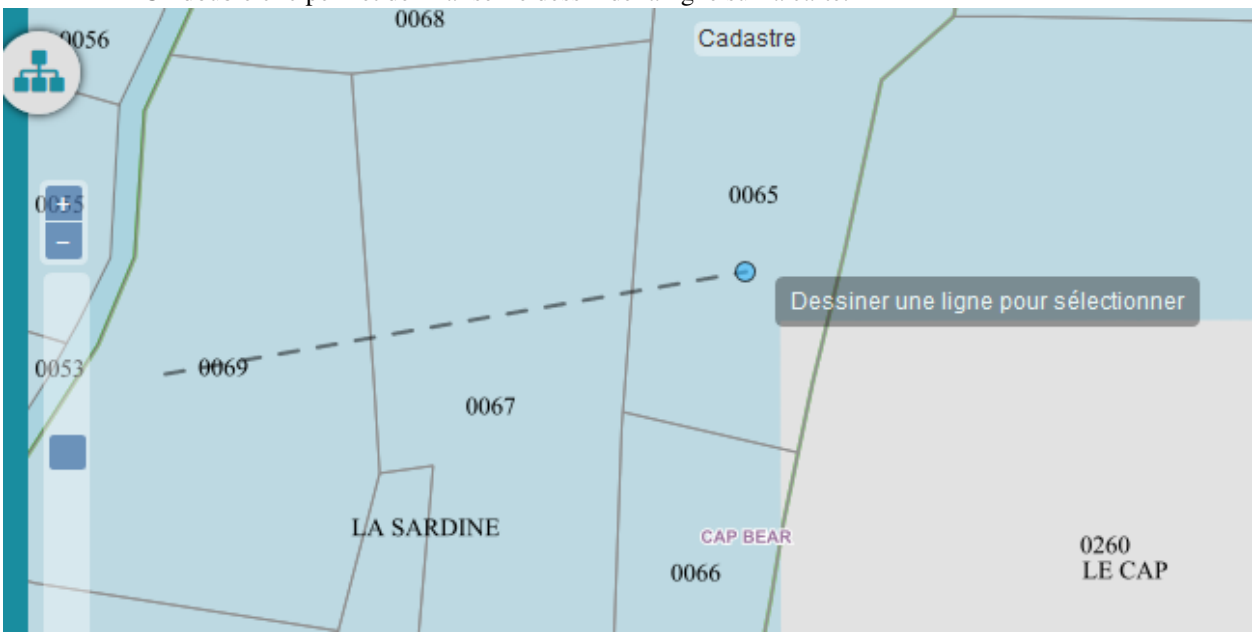
Cliquer sur l'un des modes de sélection suivants :

— Par point : il s'agit du mode de sélection le plus simple puisqu'un seul objet est retourné. L'objet intersecté par le point (emplacement du clic) est sélectionné. Il apparaît en surbrillance sur la carte. Il peut désormais être ajouté au panier.

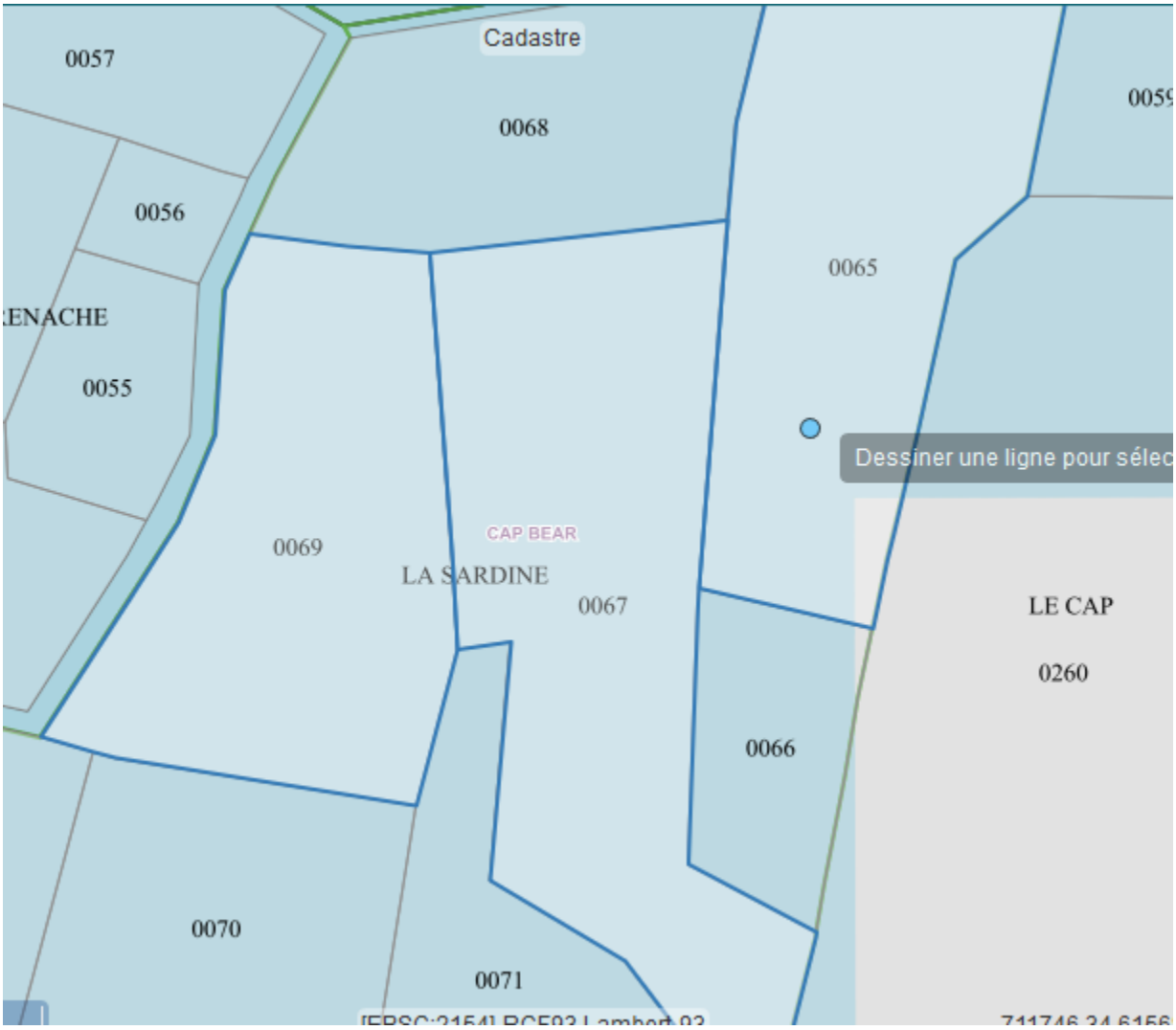


— Par ligne : tous les objets intersectant la ligne dessinée à l'écran sont retournés.

Un double clic permet de finaliser le dessin de la ligne sur la carte.



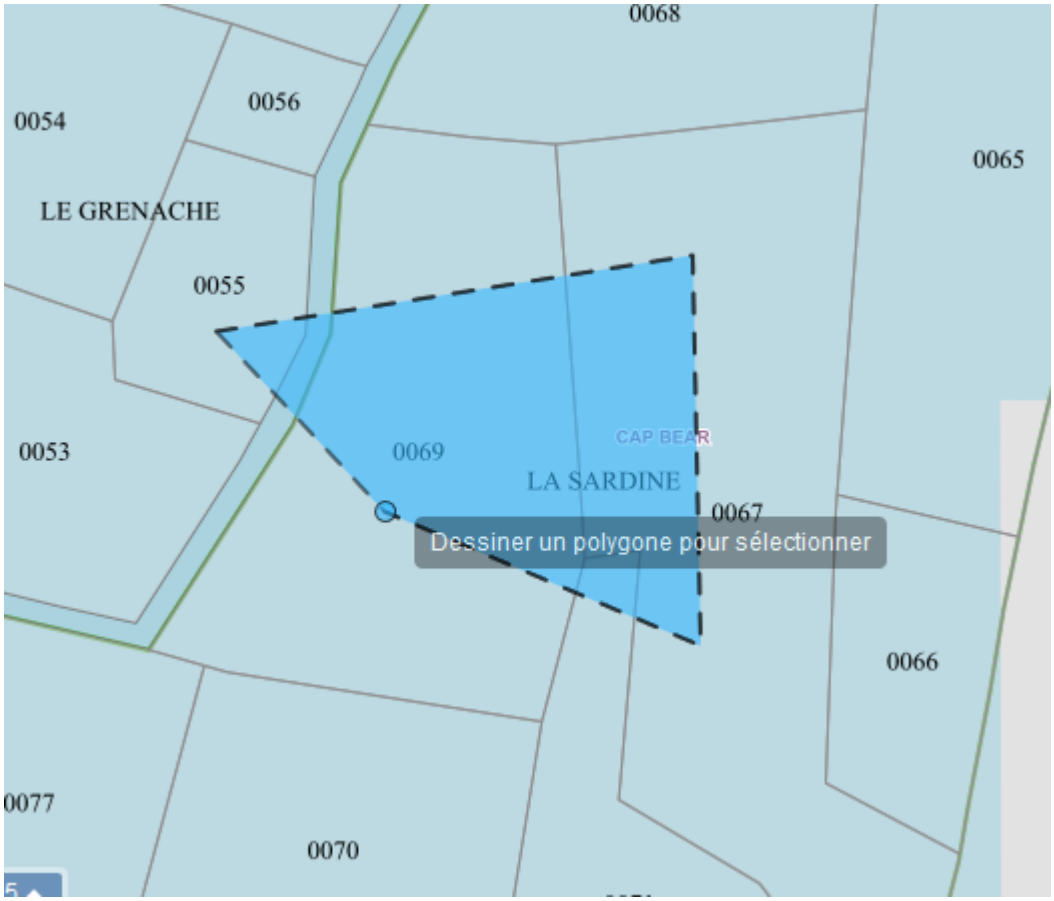
Dans l'exemple ci-dessus, la ligne déssinée intersecte 3 parcelles.



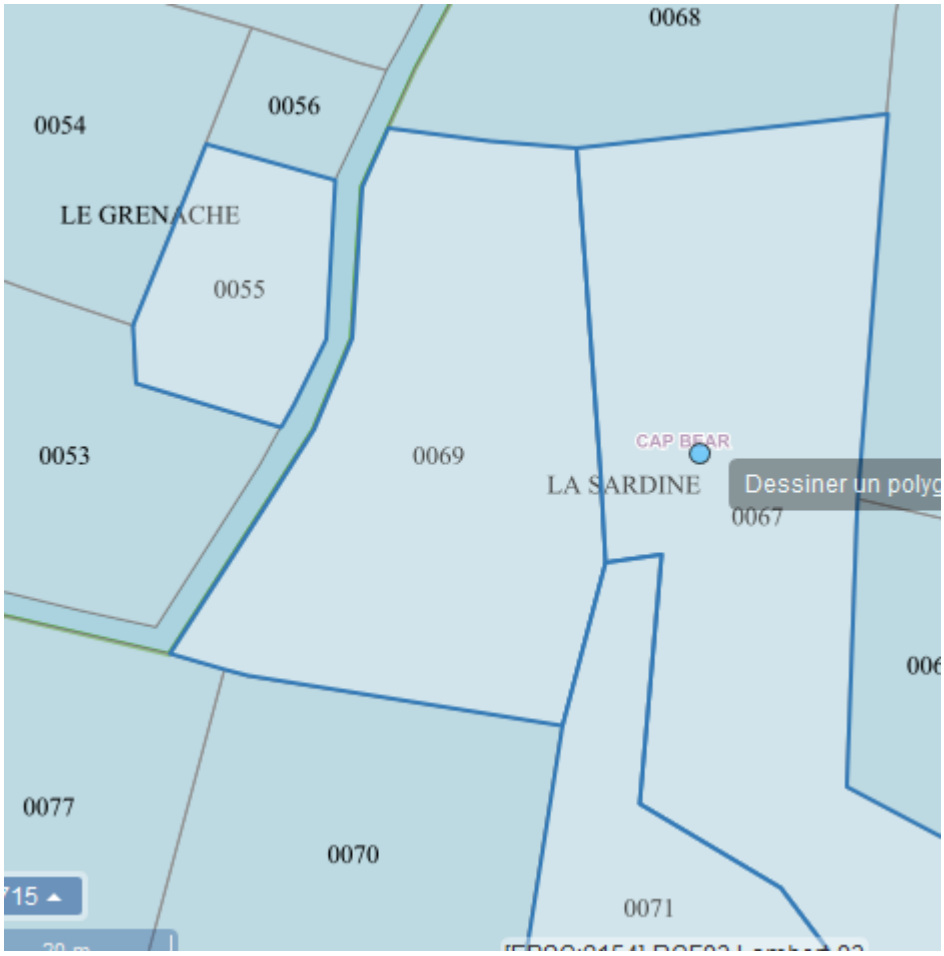
Les 3 parcelles qui intersectent la ligne dessinée à l'écran apparaissent en surbrillance. Elles peuvent désormais être ajoutées au panier.

— Par polygone : tous les objets compris et intersectés par le polygone dessiné à l'écran sont retournés.

Un double clic permet de finaliser le dessin du polygone.



Dans l'exemple ci-dessus, le polygone dessiné intersecte 4 parcelles.



Les 4 parcelles qui intersectent le polygone dessiné à l'écran apparaissent en surbrillance. Ils peuvent désormais être ajoutés au panier.

Cocher l'option Ajouter à la sélection Ajouter à la sélection pour que les éléments préalablements sélectionnés graphiquement soient toujours sélectionnés.

2.2 2. Recherche attributaire : l'onglet Formulaire

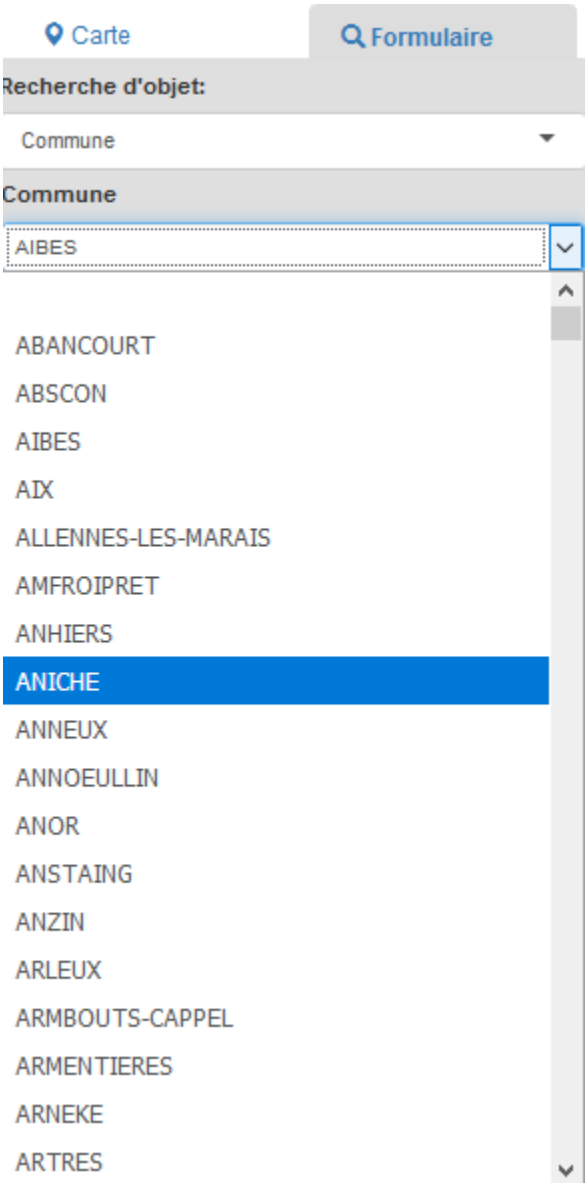
La recherche des objets du cadastre peut être effectuée à partir des données littérales sans passer par la carte et la sélection graphique. L'onglet Formulaire permet ce mode de recherche. 5 types de recherche sont disponible :

- La recherche de commune
- La recherche de section(s)
- la recherche de lieu(x) dit(s)
- la recherche de parcelle(s)
- la recherche de bâti(s)

A chaque objet sélectionné, correspondent des critères de recherche différents.

2.2.1 La recherche de commune

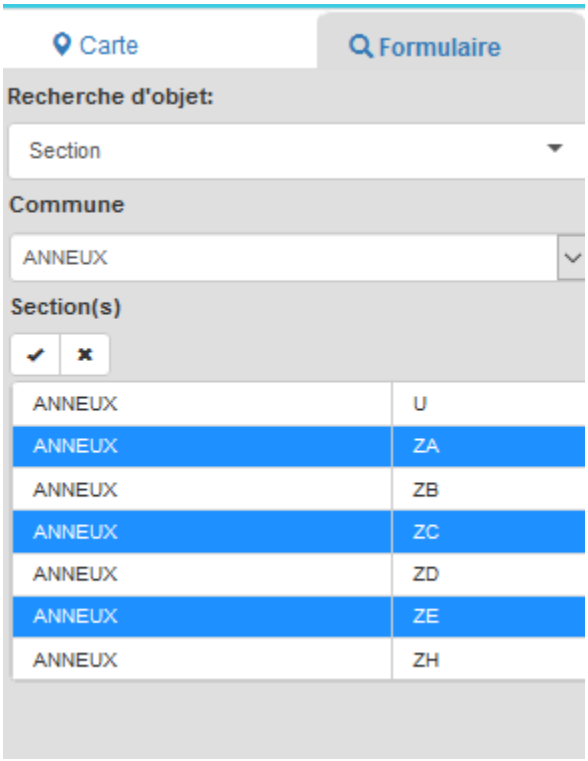
Après avoir indiqué la commune comme objet recherché, l'opérateur sélectionne dans la liste déroulante, la commune dont il veut extraire les données attributaires.



Une fois sélectionnée la commune peut être ajoutée au panier.

2.2.2 La recherche de section(s)

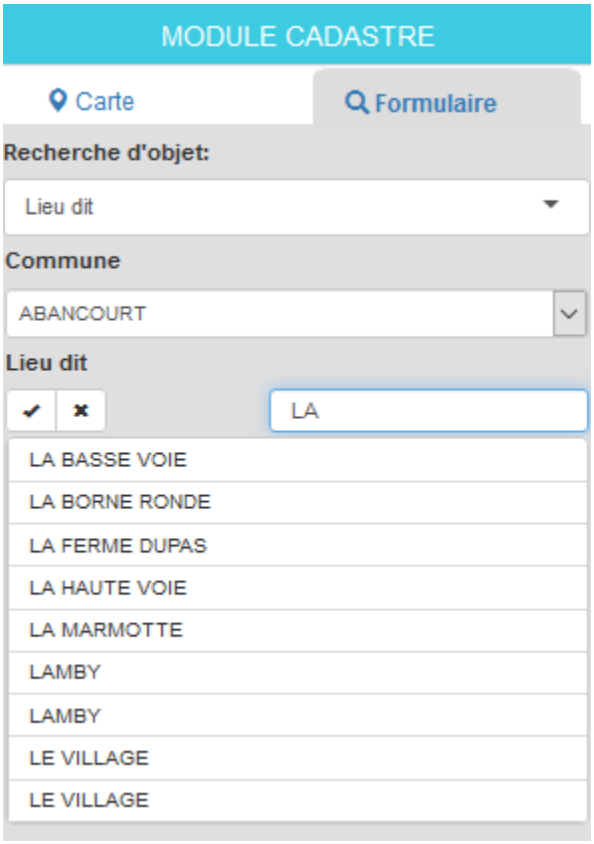
Après avoir indiqué la section comme objet recherché, l'"opérateur sélectionne dans la liste déroulante, la commune de la section recherchée. Les sections s'"affichent sous forme tabulaire. L'"opérateur sélectionne la ou les sections dont il veut extraire les informations. Le bouton ✔ permet de sélectionner toutes les sections du tableau. Le bouton ✖ permet d'"annuler la sélection des sections.



Une fois sélectionnée(s), la sélection peut être ajoutée au panier.

2.2.3 La recherche de Lieu(x) dit(s)

Après avoir indiqué le lieu dit comme objet recherché, l'opérateur sélectionne dans la liste déroulante la commune du lieu dit. Les lieux dits s'affichent sous forme de liste. L'opérateur sélectionne le ou les lieux dits dont il veut extraire les informations. Le bouton ✔ permet de sélectionner toutes les lieux dits. Le bouton ✖ permet d'annuler la sélection des lieux dits. Un formulaire Rechercher permet la saisie du nom ou d'une partie du nom d'un lieu dit pour en filtrer la liste.



Une fois sélectionné(s), les lieux-dits peuvent être ajoutés au panier.

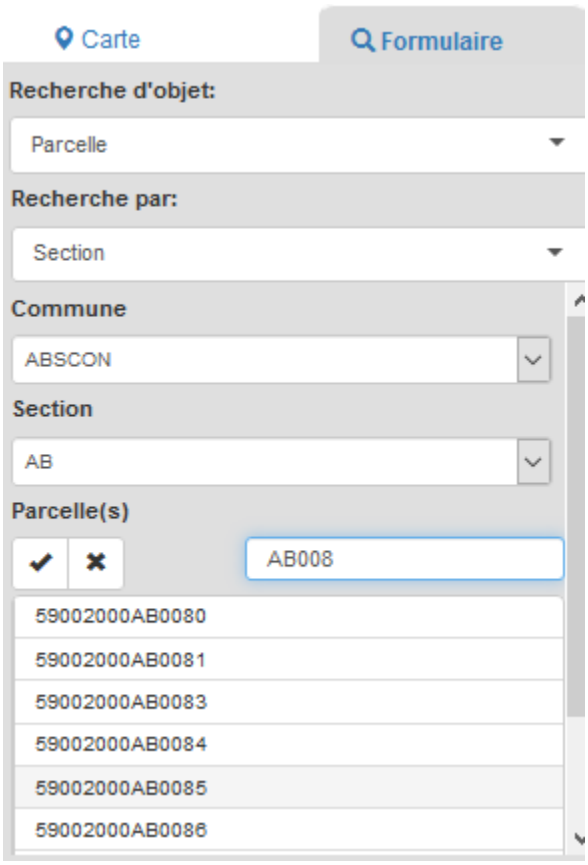
2.2.4 La recherche de parcelle(s)

Après avoir indiqué la parcelle comme objet recherché, l'opérateur sélectionne dans la liste déroulante le critère de recherche désiré :

- recherche de parcelle(s) par section
- recherche de parcelle(s) par adresse DGFIP
- recherche de parcelle(s) par propriétaire

La recherche de parcelles par section

L'opérateur sélectionne la commune puis la section d'appartenance de la parcelle. La liste des parcelles s'affichent sous forme tabulaire. L'opérateur sélectionne la ou les parcelles dont il veut extraire les informations. Le bouton ✔ permet de sélectionnes toutes les parcelles. Le bouton ✖ permet d'annuler la sélection des parcelles. Un formulaire Rechercher permet la saisie de la référence ou d'une partie de la référence de la parcelle pour en filtrer la liste.

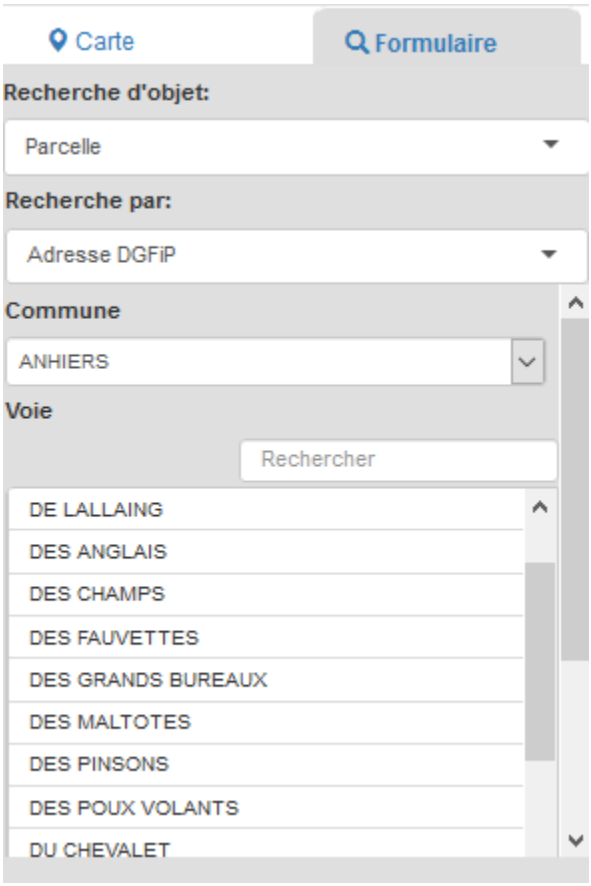


Une fois sélectionnée(s), les parcelles peuvent être ajoutées au panier.

La recherche par adresse DGFIP

L'opérateur sélectionne la commune puis la voie d'appartenance de la parcelle. La liste des parcelles s'affichent sous forme tabulaire. L'opérateur sélectionne la ou les parcelles dont il veut extraire les informations. Un formulaire Rechercher permet la saisie du nom ou d'une partie du nom de la voie pour en filtrer la liste.

La liste « Adresses » permet ensuite de sélectionner une ou toutes les adresses de la voie sélectionnée.

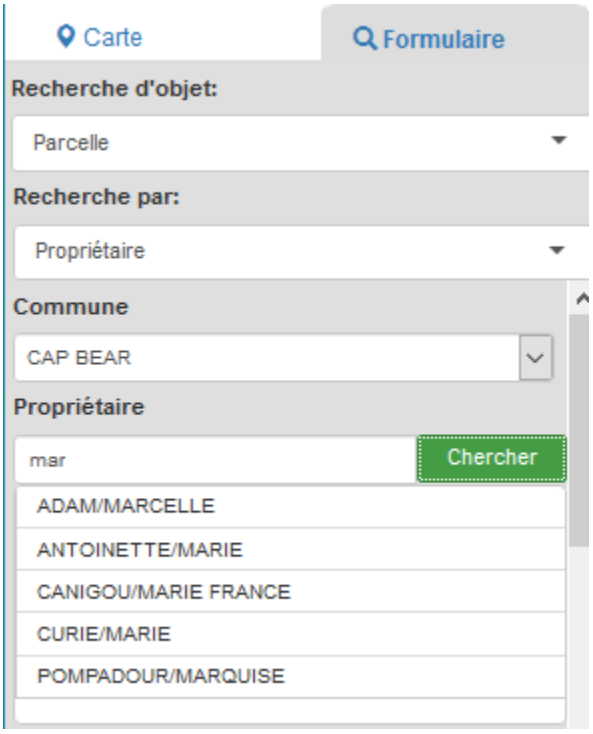


Une fois sélectionnée(s), la(es) parcelles(s) peut(vent) être ajoutée(s) au panier.

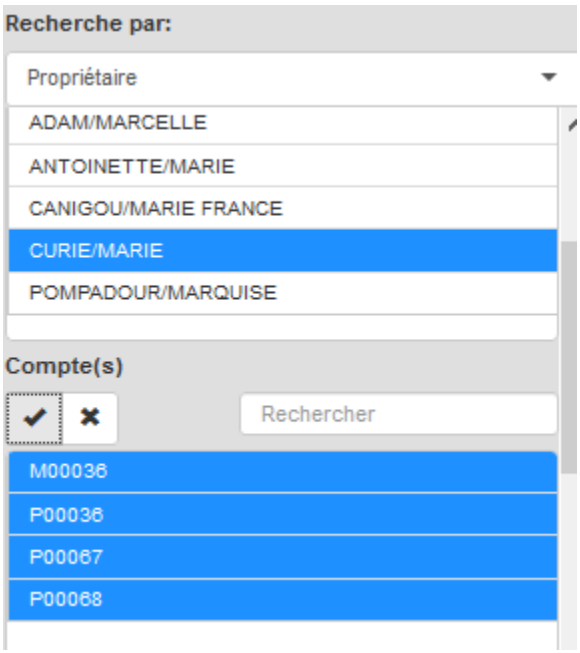
La recherche par propriétaire

L'opérateur sélectionne la commune puis le propriétaire de la parcelle. Un module de recherche permet de filtrer la lister des propriétaires en saisissant le nom ou une partie du nom du propriétaire.

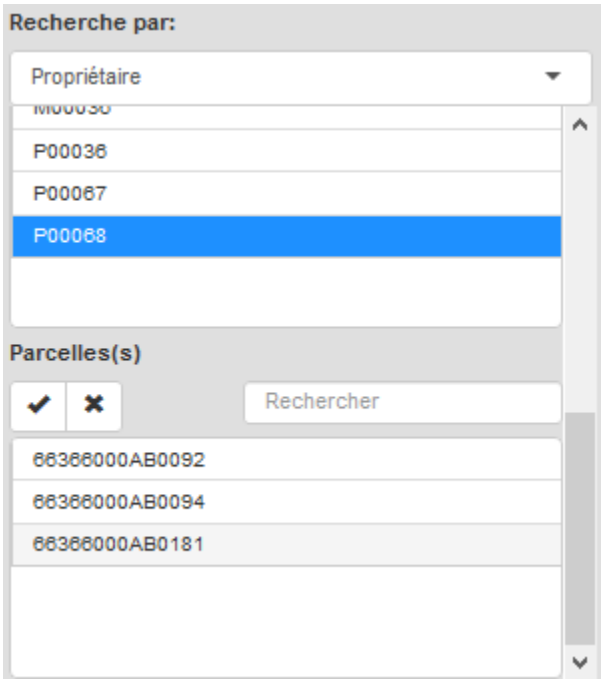
Il faut saisir un minimum de 3 caractères pour que le module de recherche puisse fonctionner.



La liste des propriétaires de la commune s"affiche. L"opérateur sélectionne ensuite le ou les comptes du propriétaire sélectionné. Le bouton ✔ permet de sélectionnes toutes les comptes. Le bouton ✖ permet d"annuler la sélection des comptes. Un formulaire Rechercher permet la saisie de la référence ou d"une partie de la référence du compte pour en filtrer la liste.



L"opérateur sélectionne ensuite la liste des parcelles du ou des comptes sélectionnés.



Le bouton ✔ permet de sélectionnes toutes les parcelles. Le bouton ✖ permet d'annuler la sélection des parcelles. Un formulaire Rechercher permet la saisie de la référence ou d'une partie de la référence de la parcelle pour en filtrer la liste.

Une fois sélectionnée(s), la(es) parcelles(s) peut(vent) être ajoutée(s) au panier.

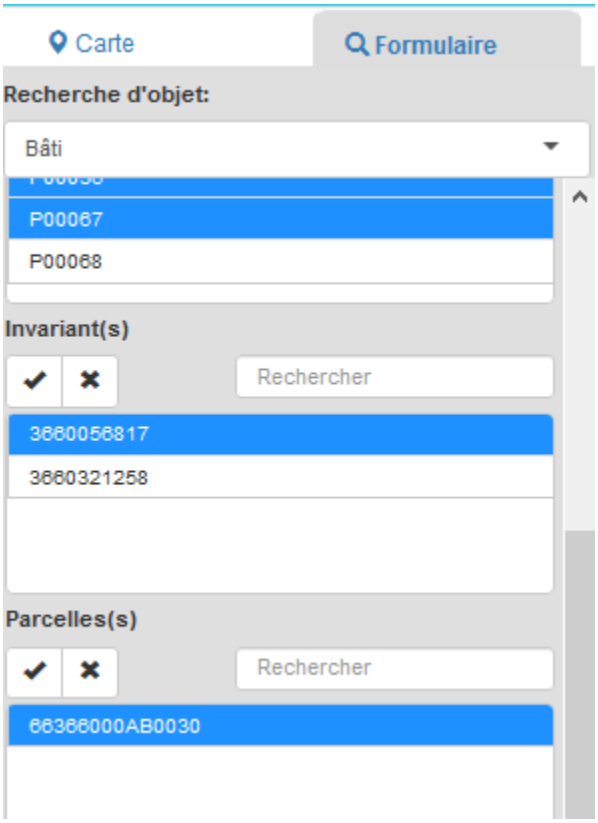
La recherche par bâti

L'opérateur sélectionne la commune puis le propriétaire de la parcelle. Un module de recherche permet de filtrer la lister des propriétaires en saisissant le nom ou une partie du nom du propriétaire.

Il faut saisir un minimum de 3 caractères pour que le module de recherche puisse fonctionner.

La liste des propriétaires de la commune s'affiche. L'opérateur sélectionne ensuite le ou les comptes du propriétaire sélectionné. Le bouton ✔ permet de sélectionnes toutes les comptes. Le bouton ✖ permet d'annuler la sélection des comptes. Un formulaire Rechercher permet la saisie de la référence ou d'une partie de la référence du compte pour en filtrer la liste.

L'opérateur sélectionne ensuite dans la liste le bâti associés au(x) compte(s) sélectionné(s). Il sélectionne ensuite les invariants, puis la parcelle dont il cherche à extraire les informations.



Une fois sélectionnée(s), la(es) parcelles(s) peut(vent) être ajoutée(s) au panier.

2.3 3. Le bloc Utilisation de la sélection et le panier

Une fois un objet sélectionné, il peut être ajouté au panier et être utilisé pour générer un rapport lui étant propre.

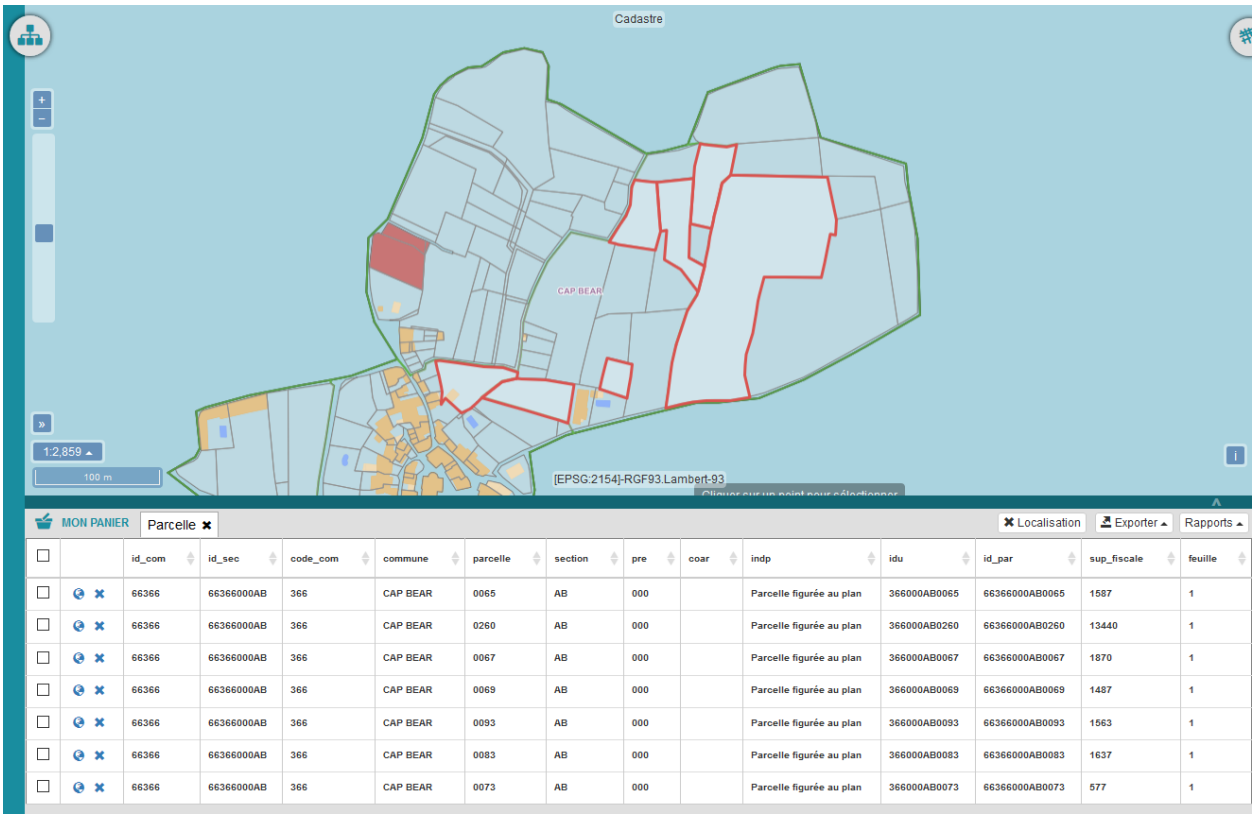
2.3.1 Ajouter une sélection au panier

Une fois les objets sélectionnés, cliquer sur le bouton 

.

Le panier permet d'"afficher, sous forme tabulaire, les objets sélectionnés graphiquement. L'"intégralité des attributs des objets sont affichés. Une fois qu'"un élément sélectionné est ajouté au panier, il apparait dynamiquement dans le tableau. A chaque ajout au panier, le tableau est enrichi d'"un enregistrement supplémentaire.

Un objet ajouté au panier apparaît en rouge et est centré sur la carte.



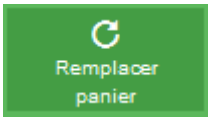
		id_com	id_sec	code_com	commune	parcelle	section	pre	coar	indp	idu	id_par	sup_fiscale	feuille
☐		66366	66366000AB	366	CAP BEAR	0065	AB	000		Parcelle figurée au plan	366000AB0065	66366000AB0065	1587	1
☐		66366	66366000AB	366	CAP BEAR	0260	AB	000		Parcelle figurée au plan	366000AB0260	66366000AB0260	13440	1
☐		66366	66366000AB	366	CAP BEAR	0067	AB	000		Parcelle figurée au plan	366000AB0067	66366000AB0067	1870	1
☐		66366	66366000AB	366	CAP BEAR	0069	AB	000		Parcelle figurée au plan	366000AB0069	66366000AB0069	1487	1
☐		66366	66366000AB	366	CAP BEAR	0093	AB	000		Parcelle figurée au plan	366000AB0093	66366000AB0093	1563	1
☐		66366	66366000AB	366	CAP BEAR	0083	AB	000		Parcelle figurée au plan	366000AB0083	66366000AB0083	1637	1
☐		66366	66366000AB	366	CAP BEAR	0073	AB	000		Parcelle figurée au plan	366000AB0073	66366000AB0073	577	1

2.3.2 Effacer la sélection

Le bouton **Effacer la sélection** 

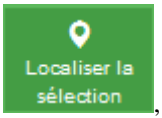
permet de désélectionner les objets préalablement sélectionnés.

2.3.3 Vider le panier

Le bouton **Remplacer le panier** 

permet de vider l'"intégralité des objets stockés dans le panier.

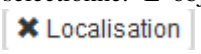
2.3.4 Localiser les éléments sélectionnés

Le bouton 

, uniquement accessible lorsque la recherche a été opérée via l'"onglet Formulaire, permet d'"afficher en surbrillance sur la carte, les éléments requêtés. Le ou les objets sont centrés sur la carte et un zoom est effectué dessus.

2.3.5 Naviguer dans le panier

— Afficher/désafficher le panier : cliquer sur le bandeau situé en haut du panier pour faire apparaître/disparaître le panier.

— Zoom sur une géométrie : le pictogramme Zoom sur la géométrie  permet de zoomer sur l'élément sélectionné. L'objet sélectionné est centré sur la carte est apparaît en surbrillance bleue. Le bouton 

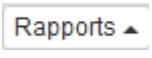
permet d'annuler la mise en surbrillance d'un objet.

— Tri des objets : un clic sur l'en-tête d'une colonne permet de trier les données par ordre croissant et décroissant.

— Onglets du panier : à chaque type d'objet sélectionné correspond un onglet dans le panier. Ainsi par exemple, si des communes et des parcelles ont été recherchées puis ajoutées au panier, alors deux onglets sont affichés dans le panier :

— Exporter les données attributaires : le bouton 

en haut à droite du panier permet l'export de l'intégralité des attributs de tous les objets de l'onglet en cours d'affichage. L'opérateur sélectionne le format destination dans lequel exporter les données attributaires :
  — EXCEL
  — JSON
  — CSV
  — XML
  — TXT

— Générer un rapport depuis le panier : le bouton 

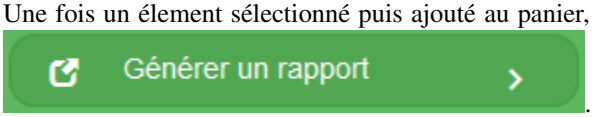
permet de générer un rapport directement à partir des éléments affichés dans le panier. En savoir plus les rapports.

Le bouton **Remplacer le panier** vide l'intégralité des objets de tous les onglets du panier.

Le panier est mis à jour à chaque fois qu'un objet est Ajouté au panier.

2.4 4. La génération de rapports

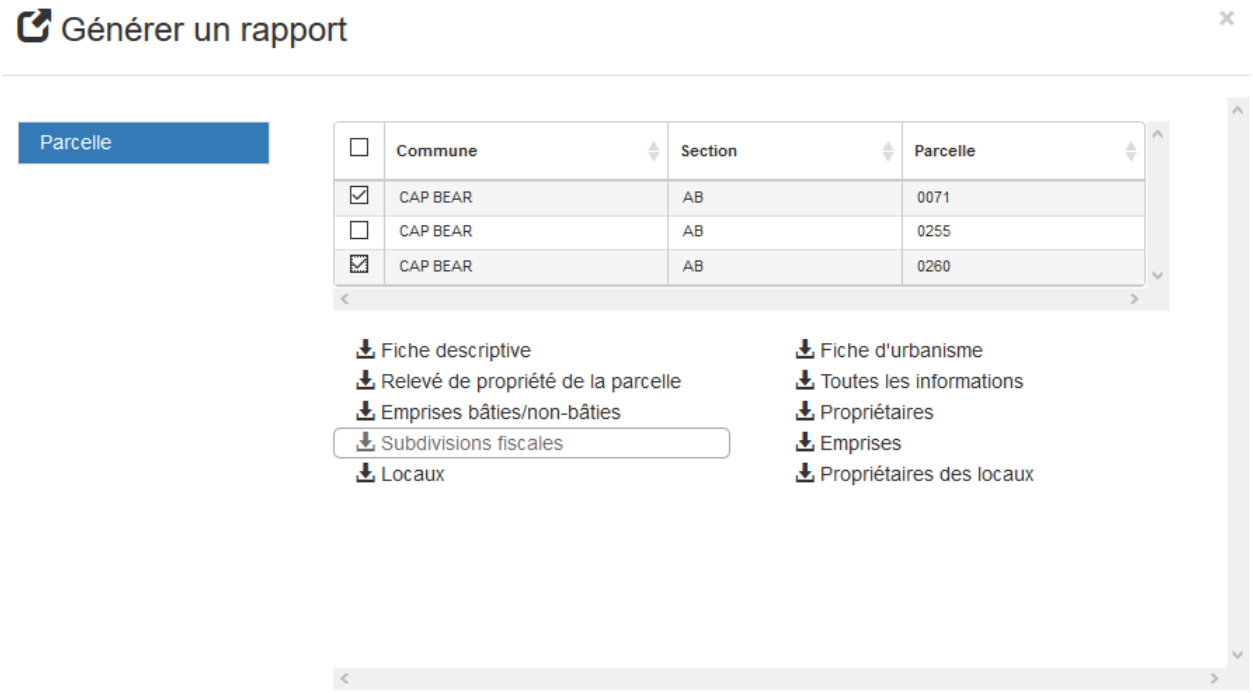
La dernière étape du processus d'interrogation des données cadastrales consiste à générer un rapport du ou des objets sélectionnés et ajoutés au panier.

Une fois un élement sélectionné puis ajouté au panier, il est possile de générer un rapport en cliquant sur le bouton 

.

Une fenêtre de génération de rapport s'ouvre. L'opérateur sélectionne dans le tableau les objets dont il souhaite extraire les données, puis il sélectionne le rapport de son choix.

Les objets qui apparaissent dans le tableau de la fenêtre sont les objets du panier.



La génération de rapport de l'élément requêté n'est possible que si l'administrateur de l'application en a préalablement conçu un, et s'il a pris soin d'associer le rapport au type d'objet sélectionné.

9 rapports relatifs aux parcelles sont livrés avec l'application vMap.

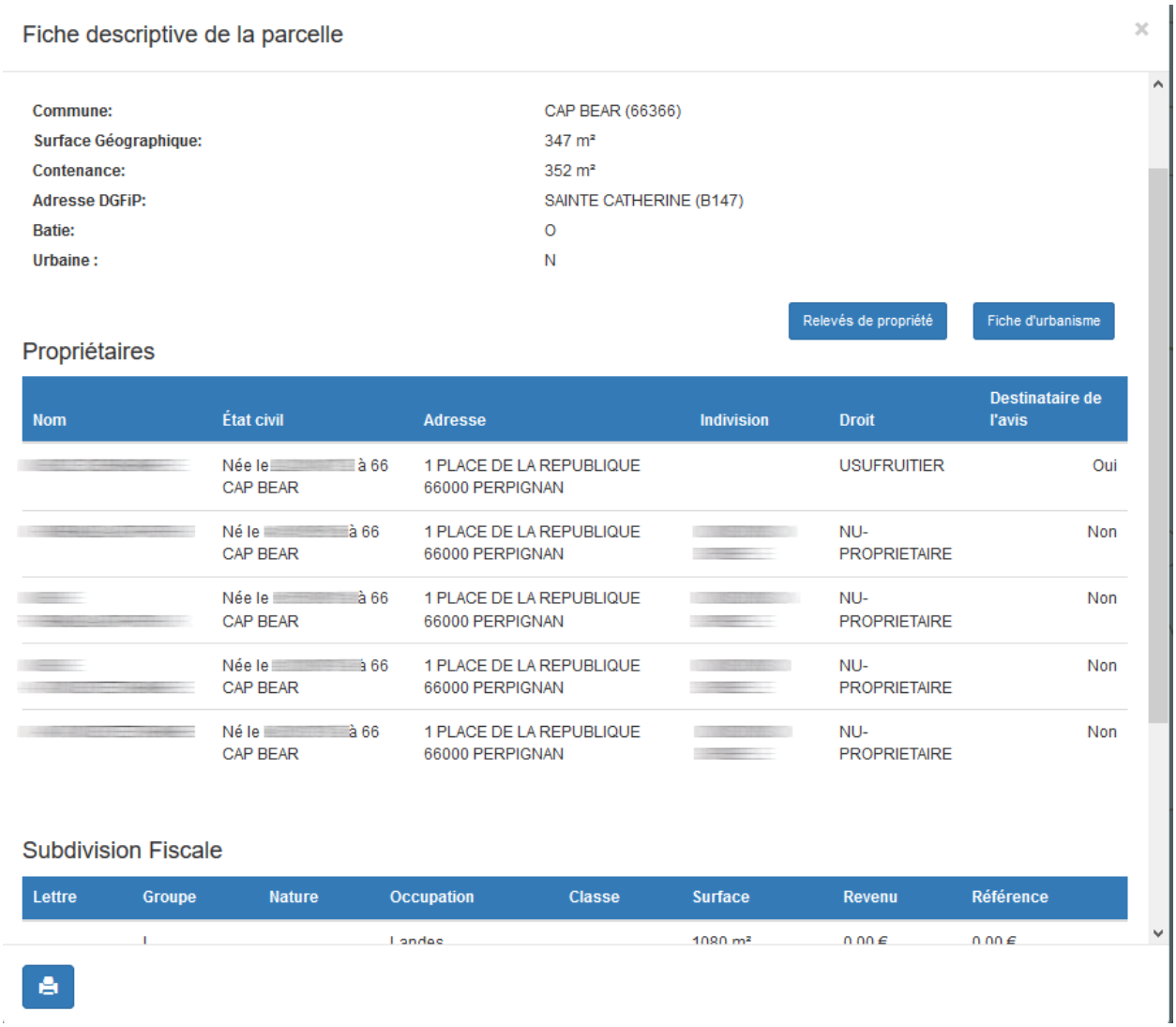
- Fiche descriptive de la parcelle
- Fiche d'urbanisme
- Relevé de propriété de la parcelle
- Emprise bâties/non bâties
- Subdivisions fiscales
- Locaux
- Toutes les informations
- Propriétaires
- Propriétaires des locaux

Certains rapports sont relatifs à une et une seule parcelle, alors que d'autres sont relatifs à plusieurs parcelles sélectionnées et ajoutées au panier.

2.4.1 La Fiche descriptive de la parcelle

La fiche descriptive de la parcelle retourne sous forme tabulaire la liste des propriétaires et des subdivisions fiscales de la parcelle sélectionnée. Elle est relative à une unique parcelle.

Il s'agit de la même fiche descriptive retournée par la recherche en un clic

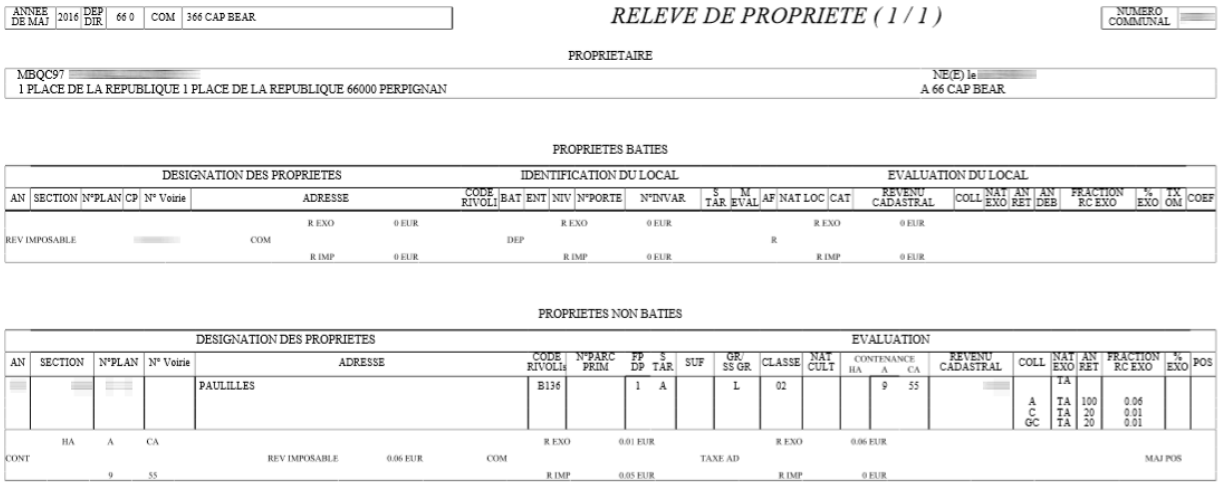


La fiche descriptive de la parcelle est configurable par l"administrateur de l"application. En savoir plus sur la configuration des rapports.

2.4.2 Le relevé de propriété de la parcelle

Ce rapport retourne au format PDF le relevé de propriété de la parcelle sélectionnée. Il est relatif à une unique parcelle. La fiche de relevé de propriété retourne le ou les propriétaires de la parcelle sélectionnée et en détaille le bâti et le non bâti.

Il s"agit du relevé de propriété stanadard généré par « la recherche en un clic »



ANNEE DE MAJ 2016 | DEP DIR 66 0 | COM 366 CAP BEAR

RELEVE DE PROPRIETE (1 / 1)

NUMERO COMMUNAL

PROPRIETAIRE

MBQC97 [redacted]
1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN

NE(E) le [redacted]
A 66 CAP BEAR

PROPRIETES BATIES

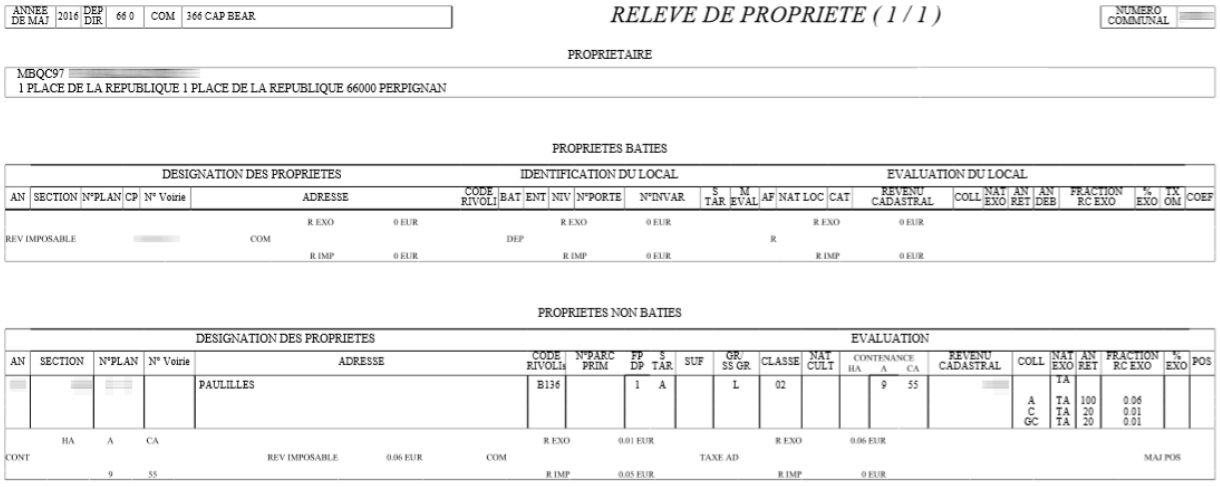
DESIGNATION DES PROPRIETES						IDENTIFICATION DU LOCAL									EVALUATION DU LOCAL									
AN	SECTION	N°PLAN	CP	N° Voirie	ADRESSE	CODE RIVOLI	BAT	ENT	NIV	N°PORTE	N°INVAR	S TAR	M EVAL	AF	NAT LOC	CAT	REVENU CADASTRAL	COLL	NAT EXO	AN RET	AN DEB	FRACTION RC EXO	% EXO	TX OM	COEF
					R EXO 0 EUR					R EXO 0 EUR						R EXO 0 EUR									
REV IMPOSABLE [redacted]					COM				DEP						R										
					R IMP 0 EUR					R IMP 0 EUR						R IMP 0 EUR									

PROPRIETES NON BATIES

DESIGNATION DES PROPRIETES											EVALUATION													
AN	SECTION	N°PLAN	N° Voirie	ADRESSE	CODE RIVOLI	N°PARC PRIM	FP DP	S TAR	SUF	GR/ SS GR	CLASSE	NAT CULT	CONTENANCE HA	A	CA	REVENU CADASTRAL	COLL	NAT EXO	AN RET	FRACTION RC EXO	% EXO	POS			
[redacted]	[redacted]	[redacted]		PAULILLES	B136		1	A		L	02			9	55	[redacted]		TA							
																	A	TA	100	0.06				
																	C	TA	20	0.01				
																	GC	TA	20	0.01				
	HA	A	CA		R EXO 0.01 EUR						R EXO 0.06 EUR													
CONT				REV IMPOSABLE 0.06 EUR	COM					TAXE AD												MAJ POS		
		9	55		R IMP 0.05 EUR						R IMP 0 EUR													

2.4.3 Le relevé de propriété de la parcelle tiers

Ce rapport retourne au format PDF le relevé de propriété de la parcelle tiers. Le contenu est le même que le relevé de propriété de la parcelle sans les informations de naissance du propriétaire



ANNEE DE MAJ 2016 | DEP DIR 66 0 | COM 366 CAP BEAR

RELEVE DE PROPRIETE (1 / 1)

NUMERO COMMUNAL

PROPRIETAIRE

MBQC97 [redacted]
1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN

PROPRIETES BATIES

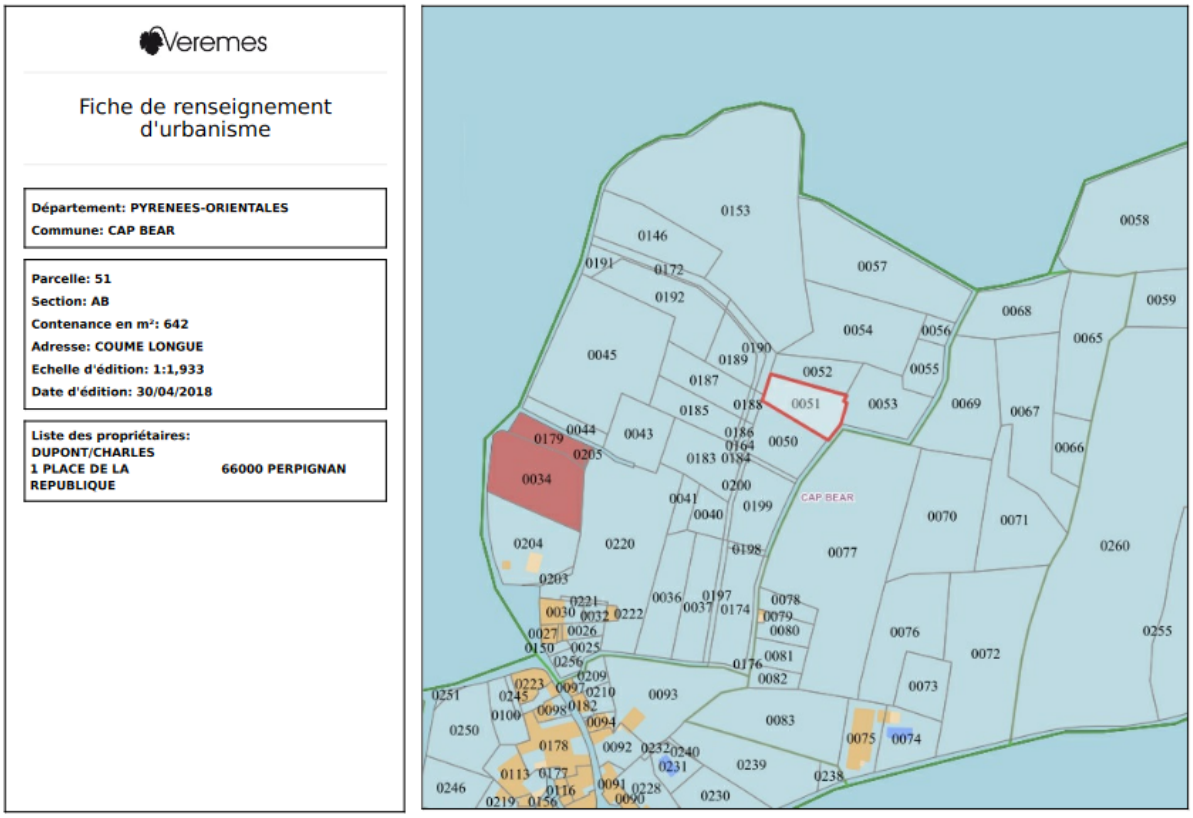
DESIGNATION DES PROPRIETES						IDENTIFICATION DU LOCAL									EVALUATION DU LOCAL									
AN	SECTION	N°PLAN	CP	N° Voirie	ADRESSE	CODE RIVOLI	BAT	ENT	NIV	N°PORTE	N°INVAR	S TAR	M EVAL	AF	NAT LOC	CAT	REVENU CADASTRAL	COLL	NAT EXO	AN RET	AN DEB	FRACTION RC EXO	% EXO	TX OM	COEF
					R EXO 0 EUR					R EXO 0 EUR						R EXO 0 EUR									
REV IMPOSABLE [redacted]					COM				DEP						R										
					R IMP 0 EUR					R IMP 0 EUR						R IMP 0 EUR									

PROPRIETES NON BATIES

DESIGNATION DES PROPRIETES											EVALUATION													
AN	SECTION	N°PLAN	N° Voirie	ADRESSE	CODE RIVOLI	N°PARC PRIM	FP DP	S TAR	SUF	GR/ SS GR	CLASSE	NAT CULT	CONTENANCE HA	A	CA	REVENU CADASTRAL	COLL	NAT EXO	AN RET	FRACTION RC EXO	% EXO	POS			
[redacted]	[redacted]	[redacted]		PAULILLES	B136		1	A		L	02			9	55	[redacted]		TA							
																	A	TA	100	0.06				
																	C	TA	20	0.01				
																	GC	TA	20	0.01				
	HA	A	CA		R EXO 0.01 EUR						R EXO 0.06 EUR													
CONT				REV IMPOSABLE 0.06 EUR	COM					TAXE AD												MAJ POS		
		9	55		R IMP 0.05 EUR						R IMP 0 EUR													

2.4.4 La fiche d''urbanisme

Relative à une unique parcelle, la fiche d''urbanisme est associée à un modèle d''impression dont les éléments sont configurables par l''administrateur de vMap. Il s''agit de la même fiche d''urbanisme générée par « le rapport en un clic »



2.4.5 Emprises bâties/non bâties

Le rapport Emprises bâties/non bâties fournit un fichier Excel listant, pour toutes les parcelles sélectionnées, la référence cadastrale (section et numéro), l'adresse, la superficie cadastrale en m^2 et la proportion du bâti / du non bâti les composant.

A1 | PARCELLES

	A	B	C	D	E	F	G
1	PARCELLES			EMPRISES			
2	Parcelle	Adresse	Surface cadastrale (m²)	Emprise bâtie (m²)	Emprise bâtie (%)	Emprise non bâtie (m²)	Emprise non bâtie (%)
3	66366000	PAULILLES	955	0	0	955	100
4	66366000	SAINTE CATHERINE	1238	259,8	74	978,2	26
5	66366000	SAINTE CATHERINE	21940	110,6	200	21829,4	-100
6	66366000	SAINTE CATHERINE	9180	237,8	154	8942,2	-54
7	66366000	SAINTE CATHERINE	5244	0	0	5244	100
8	66366000	SAINTE CATHERINE	17625	416,8	100	17208,2	0
9	66366000	RUE DE LA FONTAINE	8740	240,8	172	8499,2	-72
10	66366000	SAINTE CATHERINE	2060	0	0	2060	100
11	66366000	RUE DE LA FONTAINE	352	927,8	140	-575,8	-40
12	66366000	SAINTE CATHERINE	2320	228,8	44	2091,2	56
13	66366000	SAINTE CATHERINE	7685	0	0	7685	100
14	66366000	SAINTE CATHERINE	7010	0	0	7010	100
15	TOTAUX		84349	2422,4		81926.6	
16							

2.4.6 Emprises

Le rapport Emprises fournit un fichier Excel listant, pour toutes les parcelles sélectionnées, la référence cadastrale (section et numéro), l'adresse, la superficie cadastrale en m^2 et la proportion des emprises diverses les composant (ex :

POS / PLU).

PARCELLES			EMPRISES				
Parcelle	Adresse	Surface cadastrale (m²)					
[redacted]	IMP[redacted]	428	POS-PLU :	Pourcentage de la parcelle intersectée : 100 % Surface intersectze : 871.4 m² Id : 3, Nom : Zonage du nord-ouest	Emprise Bati :	Pourcentage de la parcelle intersectée : 4 %,	Pourcentage de la parcelle intersectée : 41 %,
[redacted]	IMP[redacted]	152	POS-PLU :	Pourcentage de la parcelle intersectée : 3 % Surface intersectze : 4.6 m² Id : 3, Nom : Zonage du nord-ouest	Emprise Bati :	Pourcentage de la parcelle intersectée : 12 %,	Pourcentage de la parcelle intersectée : 17 %,
[redacted]	IMP[redacted]	927	POS-PLU :	Pourcentage de la parcelle intersectée : 27 % Surface intersectze : 250.3m² Id : 27, Nom : Zonage 27 du centre	Emprise Bati :	Pourcentage de la parcelle intersectée : 32 %,	Pourcentage de la parcelle intersectée : 6 %,
TOTAUX		2204					

2.4.7 Subdivisions fiscales

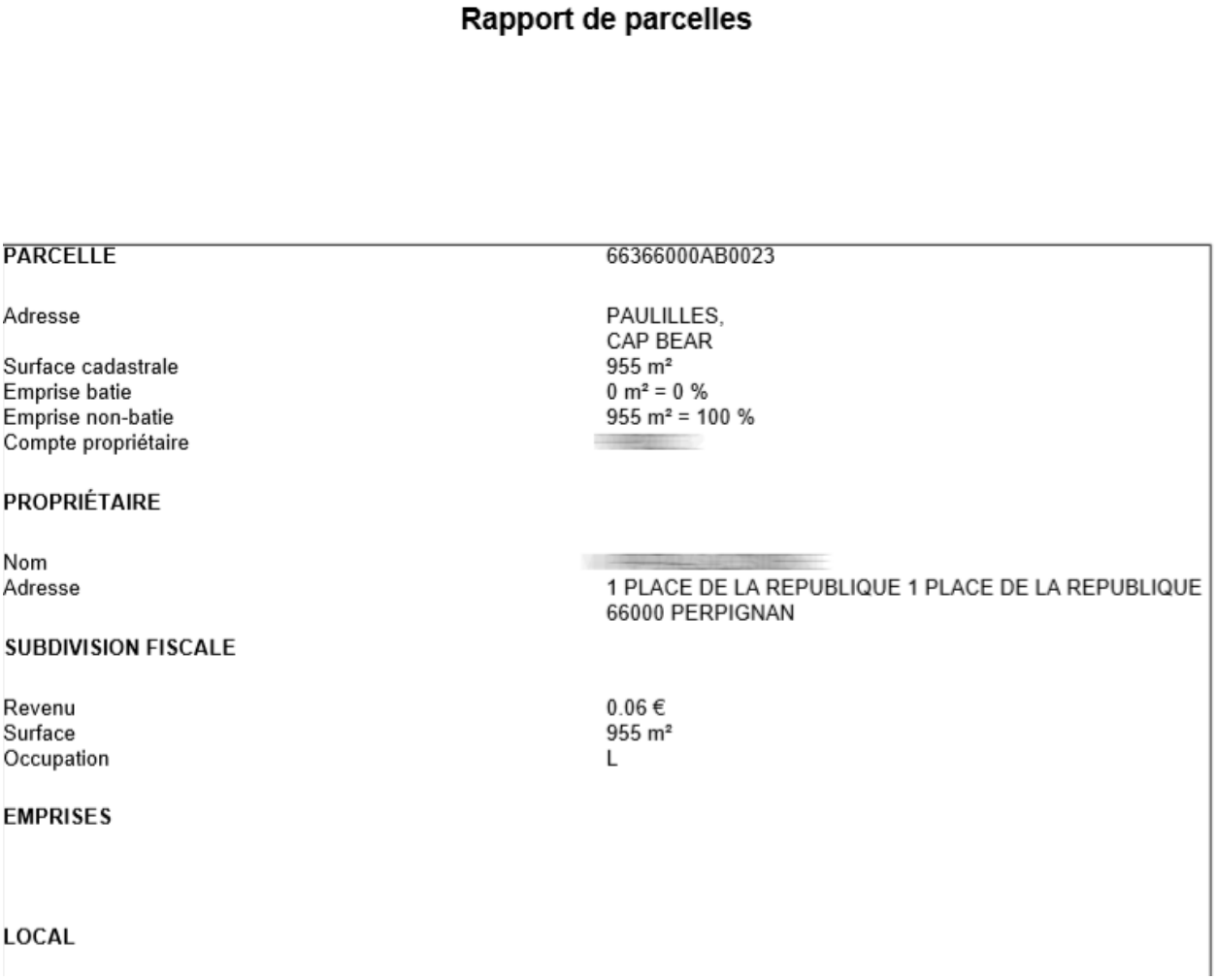
Le rapport Emprises bâties/non bâties fournit un fichier Excel listant pour toutes les parcelles sélectionnées le revenu fiscal, la surface et son occupation.

A1 fx PARCELLES

	A	B	C	D	E	F
1	PARCELLES			SUBDIVISION FISCALE		
2	Parcelle	Adresse	Surface cadastrale (m²)	Revenu (Euros)	Surface (m²)	Occupation
3	66366000[redacted]	PAULILLES	955	0,06	2750	Landes
4	66366000[redacted]	SAINTE CATHERINE	1238	0,09	350	Landes
5	66366000[redacted]	SAINTE CATHERINE	21940	3,4	58	Landes
6	66366000[redacted]	SAINTE CATHERINE	9180	0	150	Landes
7	66366000[redacted]	SAINTE CATHERINE	5244	0,41	341	Landes
8	66366000[redacted]	SAINTE CATHERINE	17625	0,95	422	Taillis simples
9	66366000[redacted]	RUE DE LA FONTAINE	8740	0,67	138	Landes
10	66366000[redacted]	SAINTE CATHERINE	2060	0,15	44	Landes
11	66366000[redacted]	RUE DE LA FONTAINE	352	0,02	663	Landes
12	66366000[redacted]	SAINTE CATHERINE	2320	0,17	531	Landes
13	66366000[redacted]	SAINTE CATHERINE	7685	0,6	952	Landes
14	66366000[redacted]	SAINTE CATHERINE	7010	0,54	30	Landes
15	TOTAUX		84349	7,06	84349	
16						

2.4.8 Toutes les informations

Un rapport des parcelles sélectionnées est généré au format PDF. Il contient pour chaque parcelle l'"intégralité des composants cadastraux : références de parcelle, propriétaires, subdivisons fiscales, emprise bâti/non bâti et locaux.

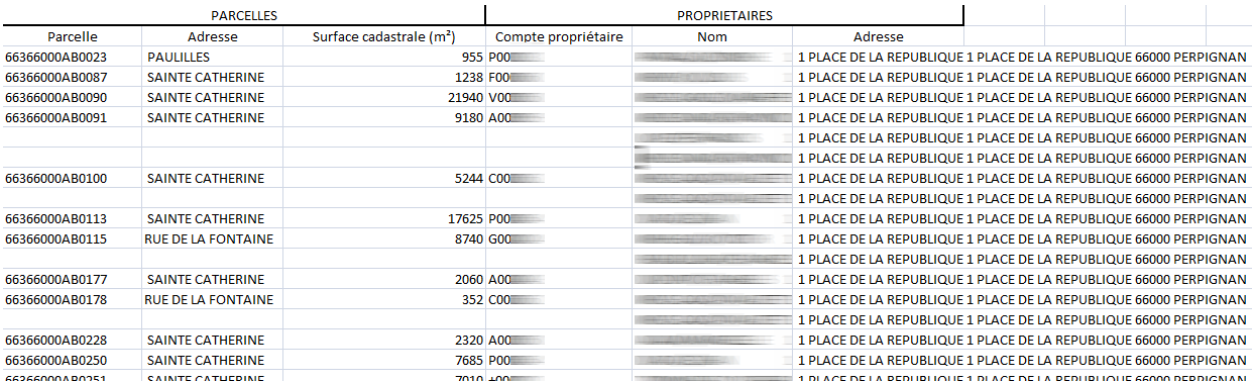


Rapport de parcelles

PARCELLE	66366000AB0023
Adresse	PAULILLES, CAP BEAR
Surface cadastrale	955 m²
Emprise batie	0 m² = 0 %
Emprise non-batie	955 m² = 100 %
Compte propriétaire	[redacted]
PROPRIÉTAIRE	
Nom	[redacted]
Adresse	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
SUBDIVISION FISCALE	
Revenu	0.06 €
Surface	955 m²
Occupation	L
EMPRISES	
LOCAL	

2.4.9 Propriétaires

Un rapport au format Excel est généré. Il liste pour chaque parcelle sélectionnée, le compte propriétaire, son nom et adresse.



PARCELLES			PROPRIETAIRES		
Parcelle	Adresse	Surface cadastrale (m²)	Compte propriétaire	Nom	Adresse
66366000AB0023	PAULILLES	955	P00[redacted]	[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
66366000AB0087	SAINTE CATHERINE	1238	F00[redacted]	[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
66366000AB0090	SAINTE CATHERINE	21940	V00[redacted]	[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
66366000AB0091	SAINTE CATHERINE	9180	A00[redacted]	[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
				[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
				[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
66366000AB0100	SAINTE CATHERINE	5244	C00[redacted]	[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
				[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
66366000AB0113	SAINTE CATHERINE	17625	P00[redacted]	[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
66366000AB0115	RUE DE LA FONTAINE	8740	G00[redacted]	[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
				[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
66366000AB0177	SAINTE CATHERINE	2060	A00[redacted]	[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
66366000AB0178	RUE DE LA FONTAINE	352	C00[redacted]	[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
				[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
66366000AB0228	SAINTE CATHERINE	2320	A00[redacted]	[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
66366000AB0250	SAINTE CATHERINE	7685	P00[redacted]	[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN
66366000AB0251	SAINTE CATHERINE	7010	[redacted]	[redacted]	1 PLACE DE LA REPUBLIQUE 1 PLACE DE LA REPUBLIQUE 66000 PERPIGNAN

2.4.10 Locaux

Un rapport au format Excel est généré. Il liste pour chaque parcelle sélectionnée, les informations fiscales des locaux les composants : adresse, type et valeur locative.

PARCELLES			LOCAUX			
Parcelle	Adresse	Surface cadastrale (m²)	Adresse local	Type de local	Valeur locative (Euros)	Millièmes et lots
66366000[redacted]	PAULILLES	2750				
66366000[redacted]	SAINTE CATHERINE	350	5025 SAINTE CATHERINE	1	1839	
66366000[redacted]	SAINTE CATHERINE	58	5096 SAINTE CATHERINE	1	1571	
66366000[redacted]	SAINTE CATHERINE	150	5019 SAINTE CATHERINE	1	2347	
66366000[redacted]	SAINTE CATHERINE	341				
66366000[redacted]	SAINTE CATHERINE	422	5027 SAINTE CATHERINE	1	2181	
66366000[redacted]	5217 RUE DE LA FONTAINE	138	5217 RUE DE LA FONTAINE	1	1026	
66366000[redacted]	SAINTE CATHERINE	44	5178 SAINTE CATHERINE	1	979	
66366000[redacted]	5124 RUE DE LA FONTAINE	663	5124 RUE DE LA FONTAINE	1	4260	
66366000[redacted]	SAINTE CATHERINE	531	5137 SAINTE CATHERINE	1	524	
66366000[redacted]	SAINTE CATHERINE	952				
66366000[redacted]	SAINTE CATHERINE	30				
TOTAUX		6429			14727	

2.4.11 Propriétaires des locaux

Un rapport Excel est généré. Il liste pour chaque parcelle sélectionnée, l'intégralité des informations fiscale des dont les comptes propriétaires, les descriptifs fiscaux des locaux, les types, natures et occupations…

								LOCAUX			
Superficie parcelle(m	Invariant	Superficie local(m²)	Type	Nature	Occupation	Année construction	Étage	Adresse local	Compte propriétaire	Propriétaire	Ad
2750											
350	3660056820	89	1		Proprietaire ou Us	1850	00	Batiment 0401, Etage 00, Porte 01001,, SAI[redacted]		[redacted]	1 PLACE DE L
58	3660056821	50	1		Vacant	1880	00	Batiment 0101, Etage 00, Porte 01001,, SAI[redacted]		[redacted]	1 PLACE DE L
150	3660056822	143	1		Proprietaire ou Us	1800	00	Batiment 0101, Etage 00, Porte 01001,, SAI[redacted]		[redacted]	1 PLACE DE L
	3660056822	143	1		Proprietaire ou Us	1800	00	Batiment 0101, Etage 00, Porte 01001,, SAI[redacted]		[redacted]	1 PLACE DE L
	3660056822	143	1		Proprietaire ou Us	1800	00	Batiment 0101, Etage 00, Porte 01001,, SAI[redacted]		[redacted]	1 PLACE DE L
341											
422	3660056828	74	1		Proprietaire ou Us	1848	00	Batiment 0101, Etage 00, Porte 01001,, SAI[redacted]		[redacted]	1 PLACE DE L
138	3660056829	50	1		Proprietaire ou Us	1800	00	Batiment 0101, Etage 00, Porte 01001,, RUE[redacted]		[redacted]	1 PLACE DE L
	3660056829	50	1		Proprietaire ou Us	1800	00	Batiment 0101, Etage 00, Porte 01001,, RUE[redacted]		[redacted]	1 PLACE DE L
44	3660347074	50	1		Proprietaire ou Us	2006	00	Batiment 0101, Etage 00, Porte 01001,, SAI[redacted]		[redacted]	1 PLACE DE L
663	3660196198	160	1		Proprietaire ou Us	1600	00	Batiment 0101, Etage 00, Porte 01001,, RUE[redacted]		[redacted]	1 PLACE DE L
	3660196198	160	1		Proprietaire ou Us	1600	00	Batiment 0101, Etage 00, Porte 01001,, RUE[redacted]		[redacted]	1 PLACE DE L
531	3660297174	95	1		Proprietaire ou Us	1991	00	Batiment 0101, Etage 00, Porte 01001,, SAI[redacted]		[redacted]	1 PLACE DE L

2.5 4. Les privilèges du module cadastre

2.5.1 4.1 vmap_cadastre_user

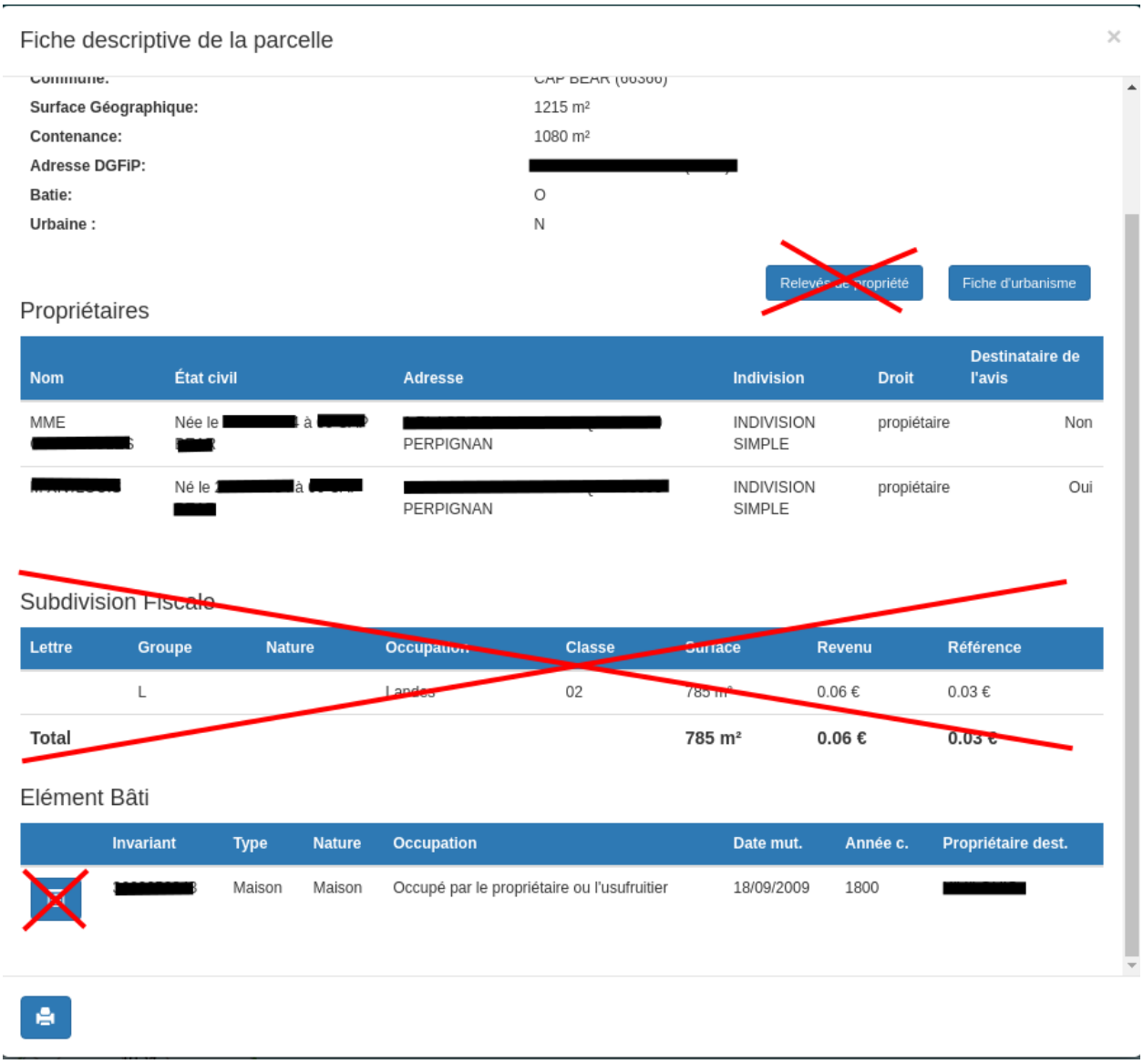
L'utilisateur qui a ce privilège a accès à toutes les fonctionnalités du module Cadastre.

2.5.2 4.2 vmap_cadastre_medium_user

L'utilisateur qui a ce privilège a accès au module cadastre Les éléments suivants ne sont pas disponibles pour ce privilège :

- La consultation du relevé de propriété
- La subdivision fiscale ainsi que la fiche d'un invariant : dans la fiche descriptive de parcelle


MODULE CADASTRE
Carte
Formulaire
Rapport en un clic
Choisissez le type de rapport et cliquez sur une parcelle pour voir le rapport
Fiche descriptive
Relevé de propriété
Fiche d'urbanisme
Recherche avancée

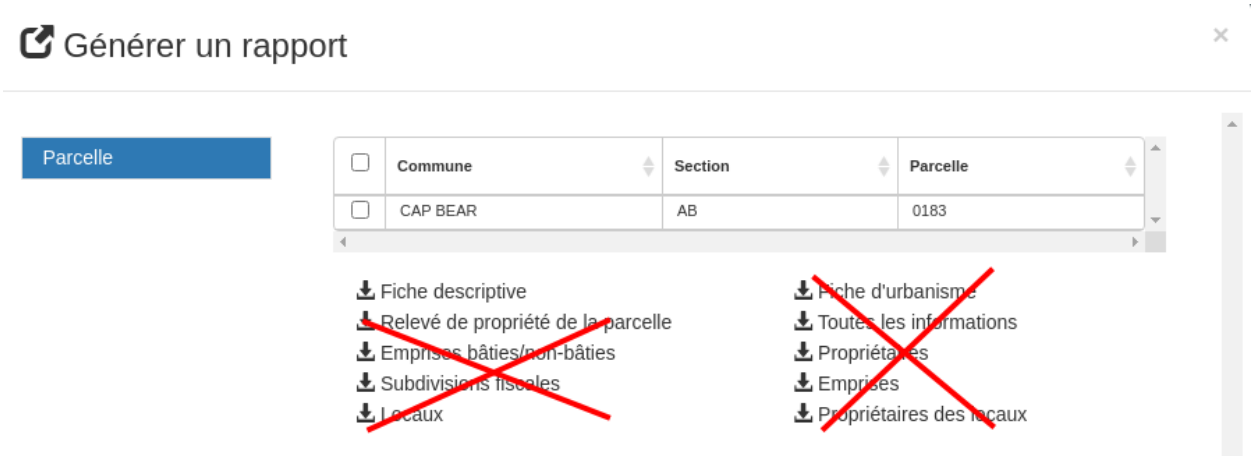


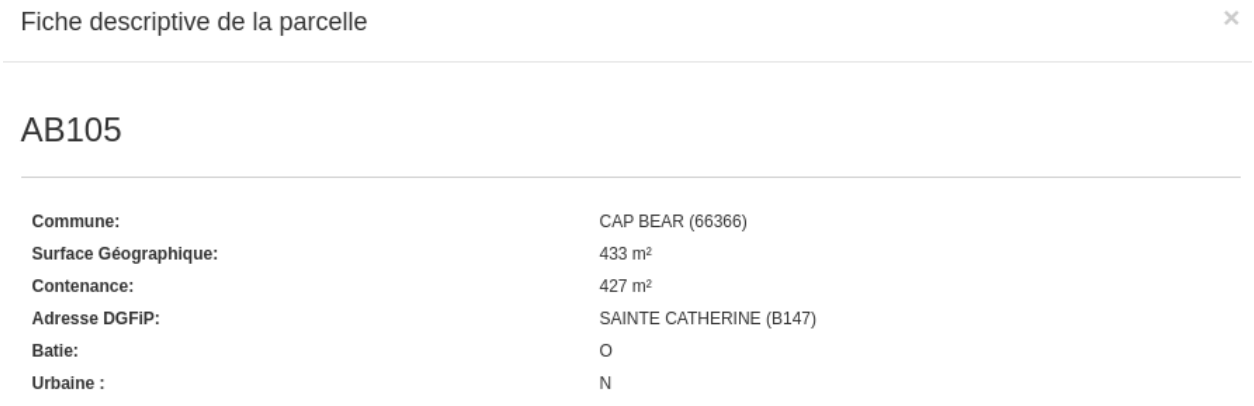
2.5.3 4.3 vmap_cadastre_light_user

L'utilisateur qui a ce privilège a accès au module cadastre Les rapports ne sont pas disponible pour ce privilège, et la fiche descriptive est simplifiée.

Aucune information liée aux propriétaires n'est disponible avec ce privilège.

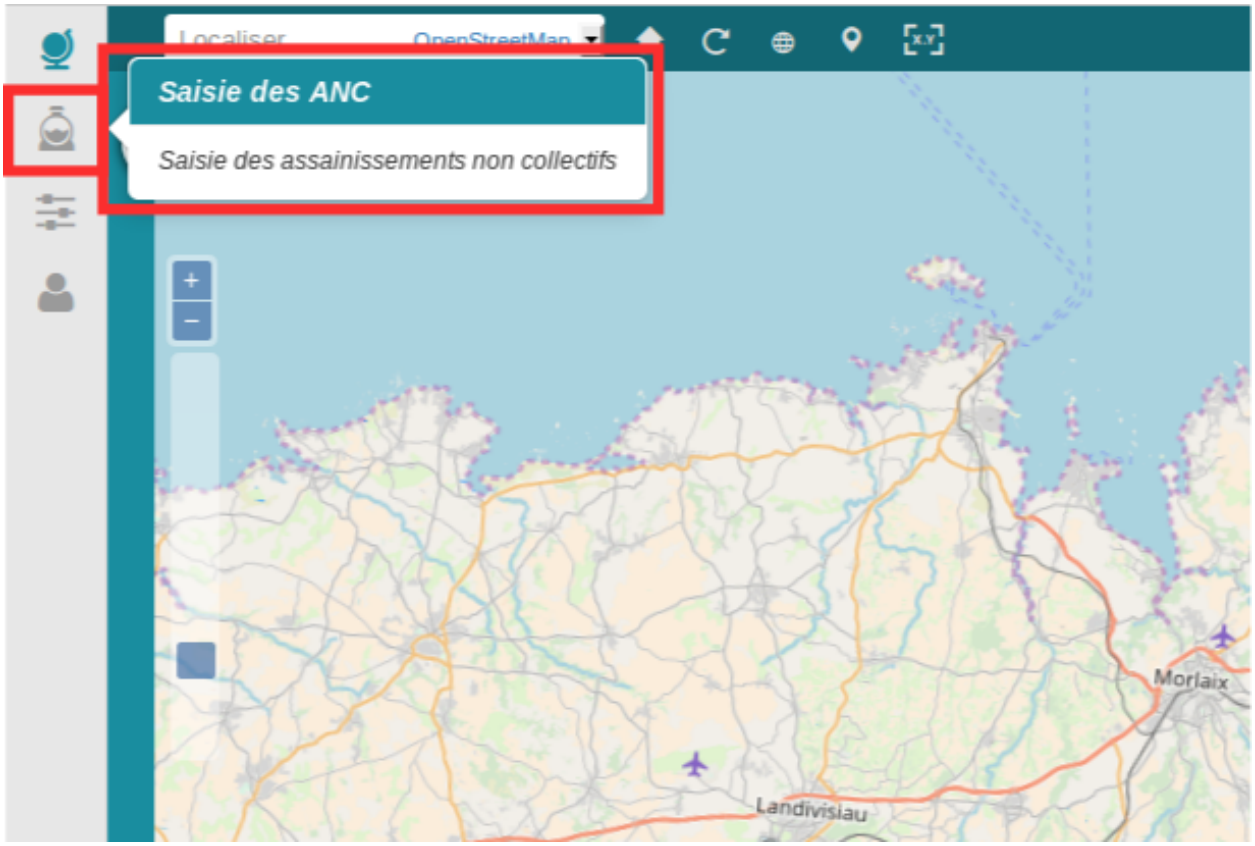

MODULE CADASTRE
Carte
Formulaire
Rapport en un clic
Choisissez le type de rapport et cliquez sur une parcelle pour voir le rapport
Fiche descriptive
Relevé de propriété
Fiche d'urbanisme


Générer un rapport
Parcelle
Commune
Section
Parcelle
CAP BEAR
AB
0183
Fiche descriptive
Relevé de propriété de la parcelle
Emprises bâties/non-bâties
Subdivisions fiscales
Locaux
Fiche d'urbanisme
Toutes les informations
Propriétaires
Emprises
Propriétaires des locaux


Fiche descriptive de la parcelle
AB105
Commune: CAP BEAR (66366)
Surface Géographique: 433 m²
Contenance: 427 m²
Adresse DGFiP: SAINTE CATHERINE (B147)
Batie: O
Urbaine : N

CHAPITRE 3

Mode saisie ANC



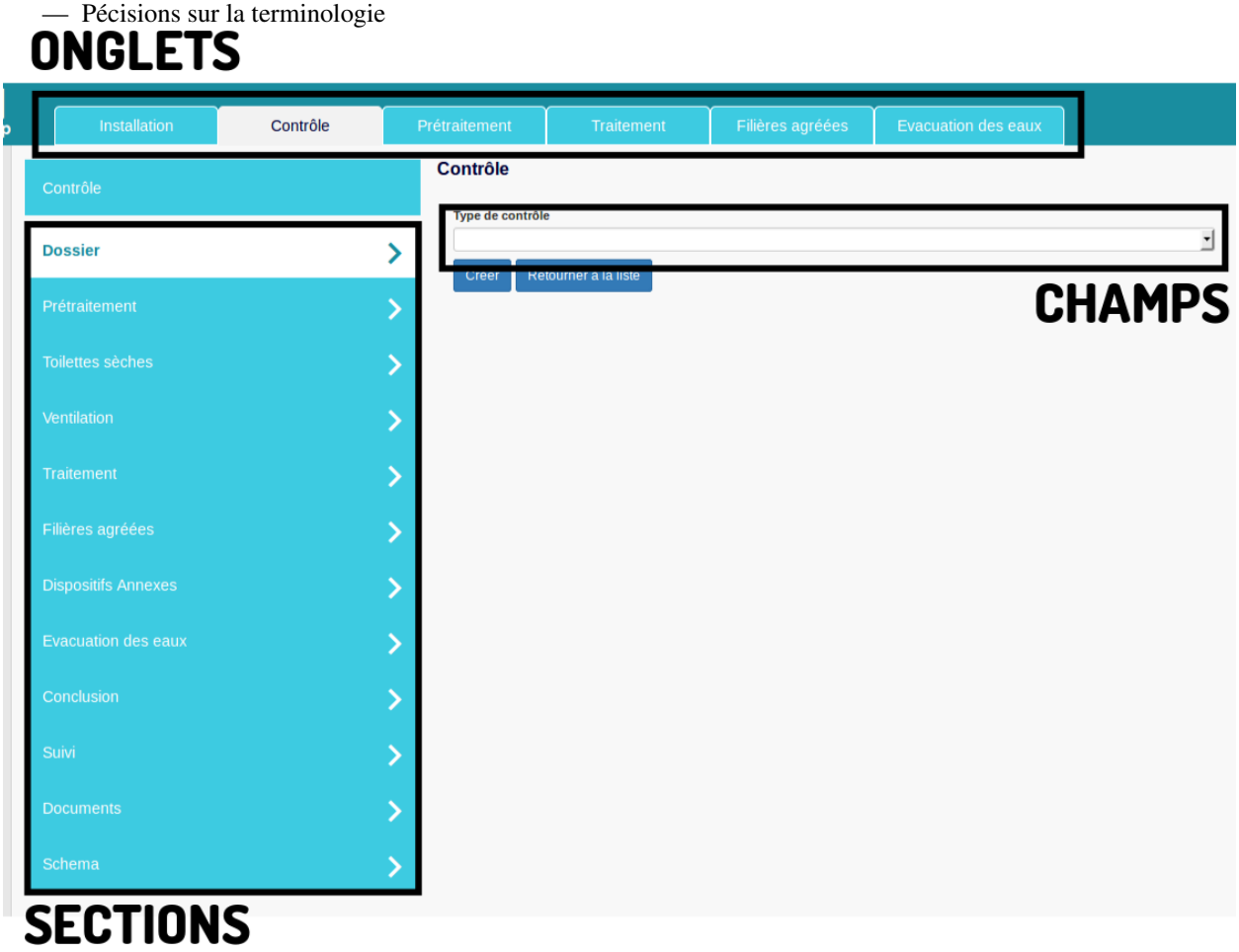
3.1 1. Définition

Le mode ANC, accessible aux utilisateurs en ayant droits (anc_user) permet la saisie des données « métiers » liées au services publiques d'Assainissement Non collectif (SPANC).

Le module métier s'organise autour de la saisie des contrôles. Ces derniers sont liés à des installations localisées sur le territoire. Une installation peut avoir été contrôlée plusieurs fois (relation 1 à plusieurs).

Cette documentation ne passera pas en revue l'intégralité des informations qu'il est possible de renseigner car elles sont trop nombreuses. Nous mettrons en exergue les ***informations obligatoires (Cadres verts)*** *pour l'enregistrement des formulaires.*

— Pécisions sur la terminologie

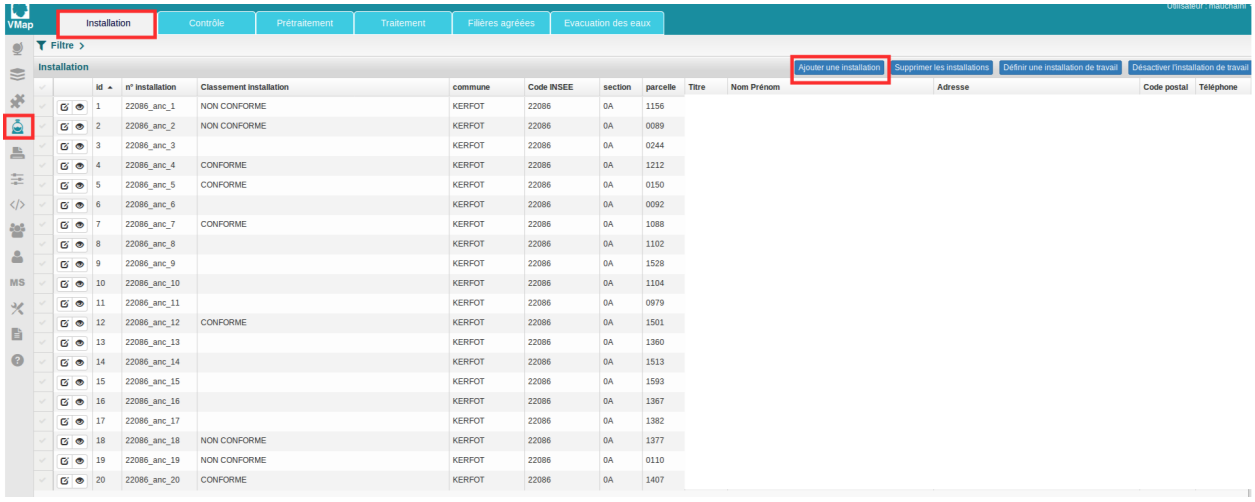


3.2 2. Les installations

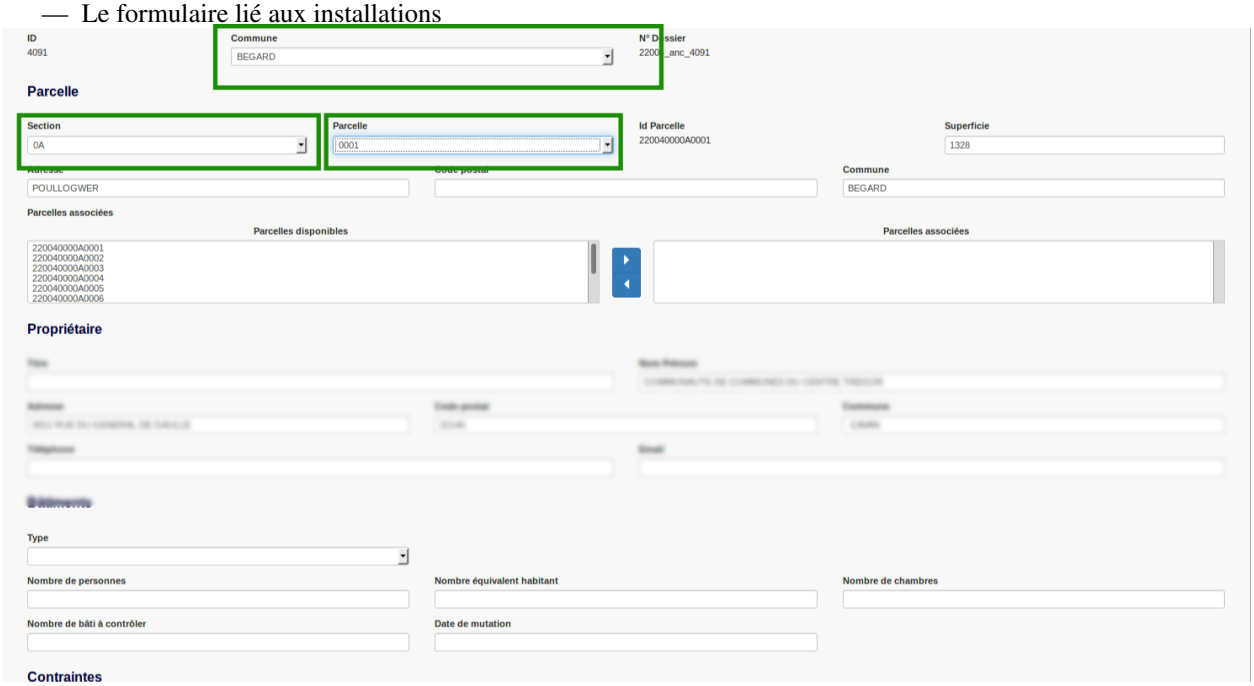
Les contrôles sont donc liés aux installations, il faut donc en premier lieu saisir les informations sur les installations et les localiser géographiquement.

3.2.1 a. Ajout d'une installation

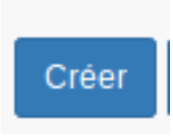
— Dans l'onglet installation –> Ajouter une installation



— Le formulaire lié aux installations



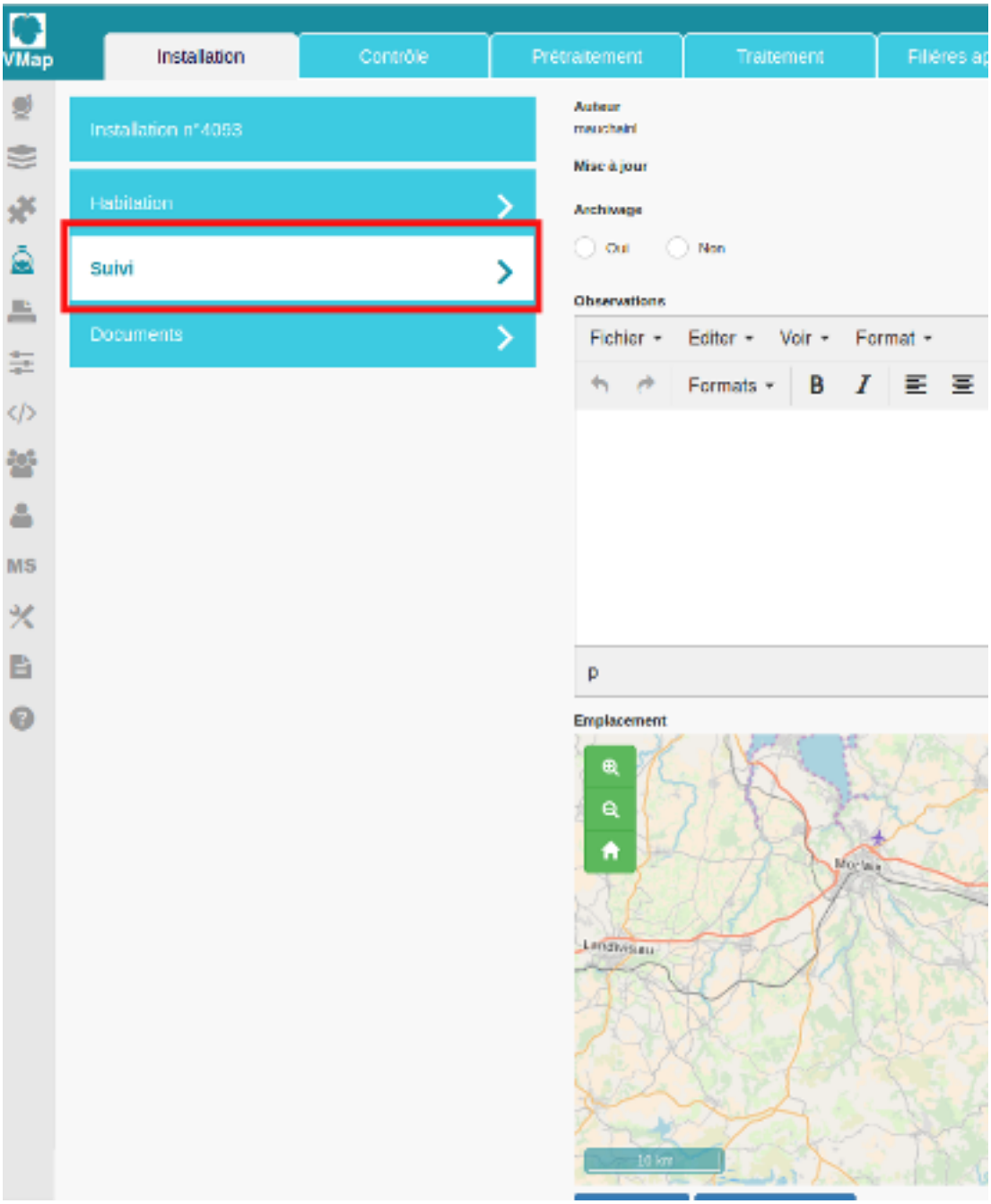
Après identification de la parcelle, les informations cadastrales seront remontées automatiquement.

— Une fois ce premier formulaire complété –> créer l"installation avec 

en bas de la fiche

A partir de là, nous avons crée une installation mais elle n"a pas encore de référencement géographique

— Pour la localisation de l"installation –> passez sur la section suivi

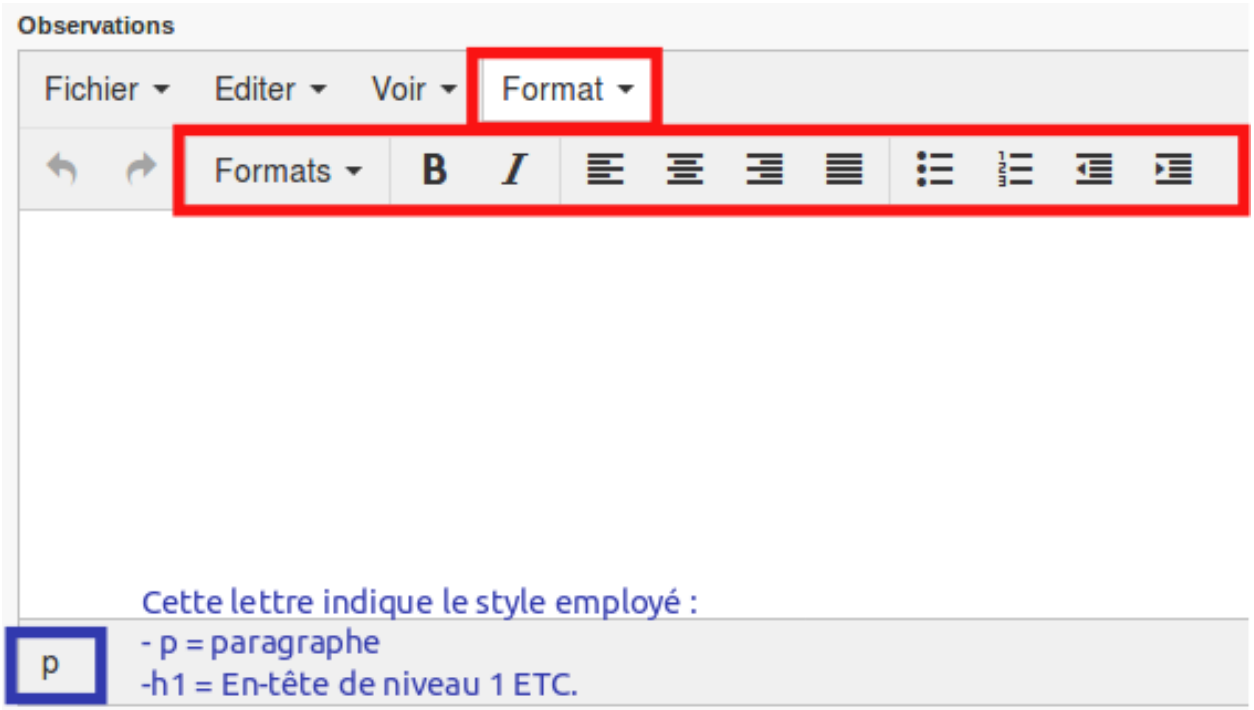


— Placement de l'"installation sur la carte

Afin de faciliter la saisie vous pouvez passez en mode **plein ecran** avec le bouton  . Vous pouvez zoomer sur la carte avec les boutons  ou alors avec la molette de la souris.

Vous pouvez déplacer le point avec  et l'"effacer avec  .

Dans cet section vous avez également la possibilité d'"ajouter des observations en texte enrichi (c'"est à dire que vous pouvez le mettre en forme comme dans un logiciel de traitement de texte)



— Pour attacher un document ou une photo à l'"installation –> Rendez vous dans la section Documents

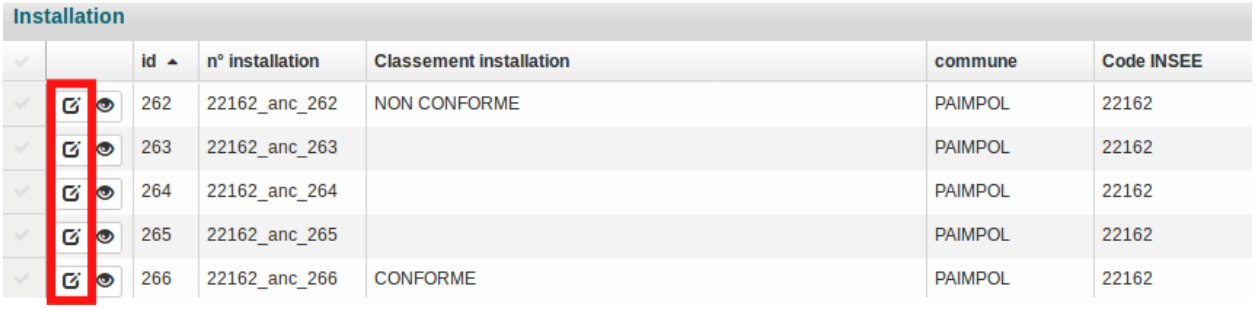
3.2.2 b. Recherche d'"une ou des installations

— Déployez le panneau de filtre

Dans le cas d'"une recherche par numéro d'"installation, il faut savoir que le numéro d'"installation se compose de la manière suivante : **code insee_anc_id de l'"installation**. Donc pour rechercher le numéro d'"installation 22086_anc_50, il faudra saisir « 50 » dans le champ ID.

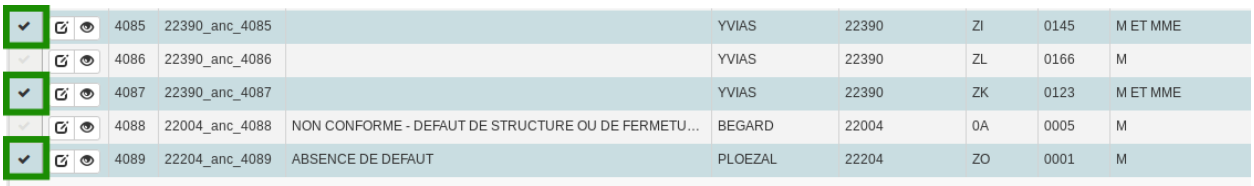
3.2.3 c. Mise à jour d'"une installation

— Dans le tableau des installations appuyer sur 



3.2.4 d. Effacer une ou plusieurs installations

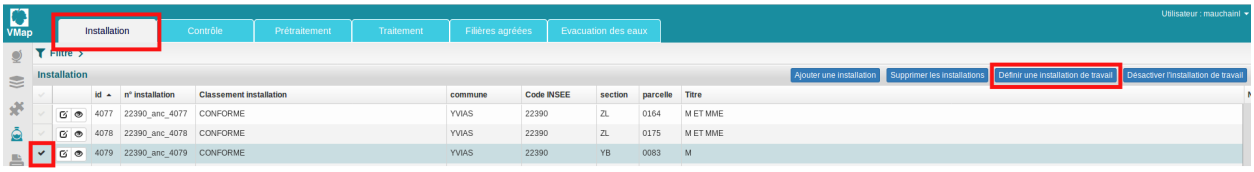
— Sélectionnez les lignes des installations dans le tableau



— Appuyer sur le bouton 

3.3 2. Les contrôles

Une fois l'"installation saisie, plusieurs contrôles peuvent lui être affectés. Afin de cibler l'"installation à laquelle vous allez ajouter un ou des contrôles il faut la définir comme installation de « travail ».

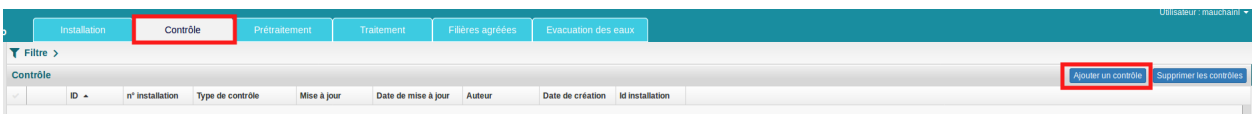


Le bouton se grise lorsque qu'"une installation est défini comme une installation de travail.

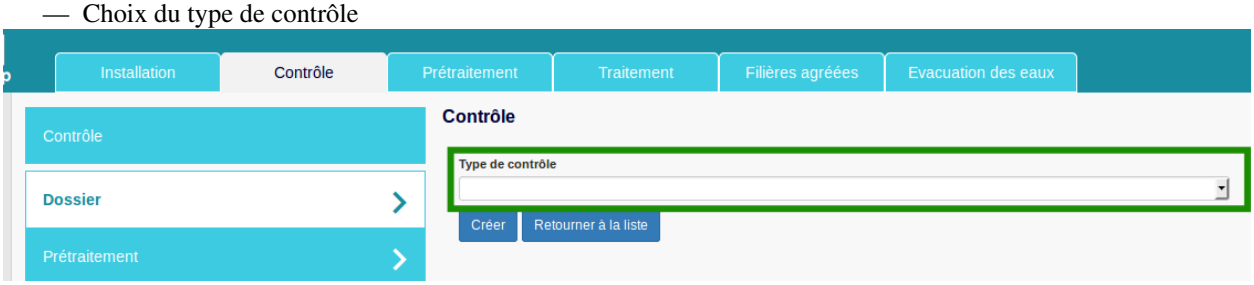
Lorsque que vous passez dans l'"onglet « Contrôle », seuls les contrôles liés à cette installation seront affichés. De même lorsque vous saisissez un nouveau contrôle la liaison avec l'"installation de travail est automatique

3.3.1 a. Ajout d'"un contrôle

— Dans l'"onglet « Contrôles » –> Ajout d'"un contrôle



— Choix du type de contrôle



3 types de contrôles possible : *Bon Fonctionnement, Conception, Execution*

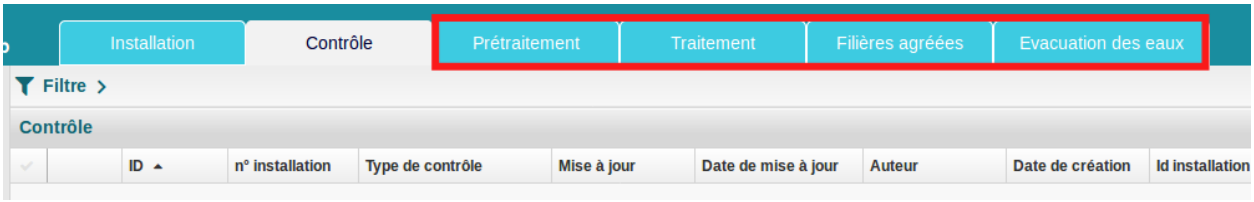
–> des champs peuvent apparaître ou disparaître en fonction du type de controle

— les champs obligatoires

— Les différentes sections liées aux contrôles

Les formulaires de contrôle contiennent un nombre de champs conséquents organisés selon des sections.

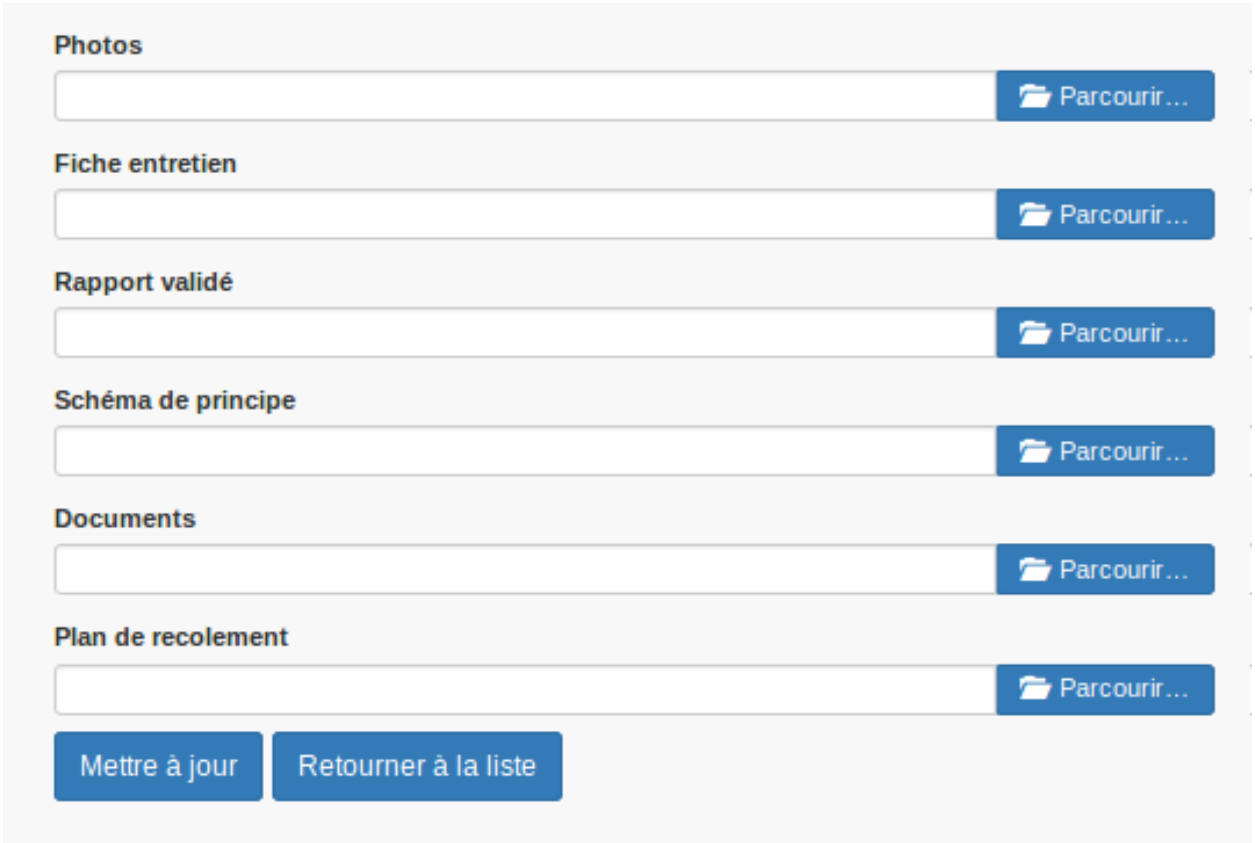

Contrôle
Dossier
Prétraitement
Toilettes sèches
Ventilation
Traitement
Filières agréées
Dispositifs Annexes
Evacuation des eaux
Conclusion
Suivi
Documents
Schema



A l"image des installations et des controles, ces sections s"organisent autour d"une grille/table listant les éléments.

— Joindre des documents au contrôle

–> Section Doucments



— Saisie du schéma de l"installation –> Section Schéma

*Localiser la parcelle :

Malheureusement, à ce stade, le module ANC ne permet pas une recherche aisée de la parcelle. Il faut localiser la parcelle sur un fond Openstreetmap (initialement à l"échelle de la France).*Conseil : Utilisez le zoom molette de la souris*

Placez-vous à une échelle de « 20 m » ou « 10 m » 

*Ajouts des Pictogrammes : Une fois la localisation effectuée, cliquez sur le bouton 

*Modification des éléments du schéma :

Avec le bouton  vous pouvez dépacer ou modifier la géométrie des éléments du schéma
*Sélection et suppression d"éléments du schéma :

Deux possibilités de sélectionner des éléments : Soit avec  , Soit directement avec la grille

3.3.2 b. Recherche d"un ou des contrôles

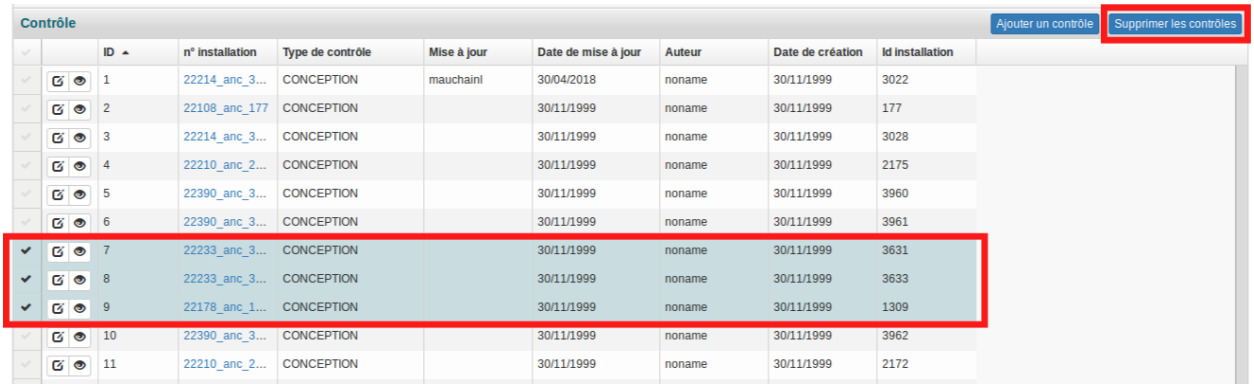
A l"instar des installations, il faut passer par le panneau de filtre déroulant 

. Il faudra dans un premier temps sélectionner le type de contrôle sur lequel s"effectue la recherche puis des champs de filtre apparaîtront.

3.3.3 c. Mise à jour d"un contrôle

Avec le bouton  dans la grille des contrôles

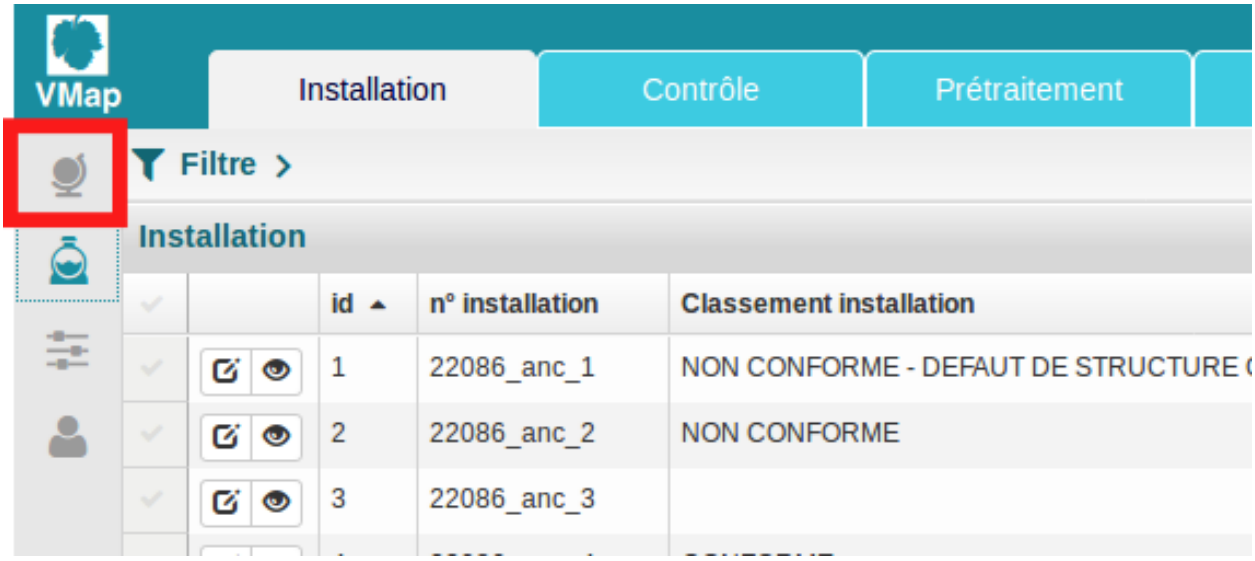
3.3.4 d. Suppression d"un ou des contrôles



3.3.5 e. Génération d"un rapport lié à un contrôle

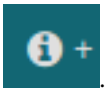
La génération des rapports n"est pas encore permise via le module SPANC. Il faut donc passer par la carte + le panier pour générer les rapports.

— Sortir du module SPANC



— Aller sur la carte « ANC »

— Localiser l'"installation

Vous pouvez localiser via l'"outil cadastre  si vous connaissez la référence cadastrale de l'"installation. Ou alors avec l'"outil requêteur  .

Si vous ne savez pas utiliser ces outils, vous pouvez vous référer à la documentation générale de Vmap.

— Ajouter l'"installation au « Panier » pour générer les rapports

A la fin de cette étape, sélectionner le rapport qui vous intéresse. Cela devrait générer un fichier au format propriétaire Microsoft WORD éditable.

CHAPITRE 4

Administration

4.1 Gestion des utilisateurs

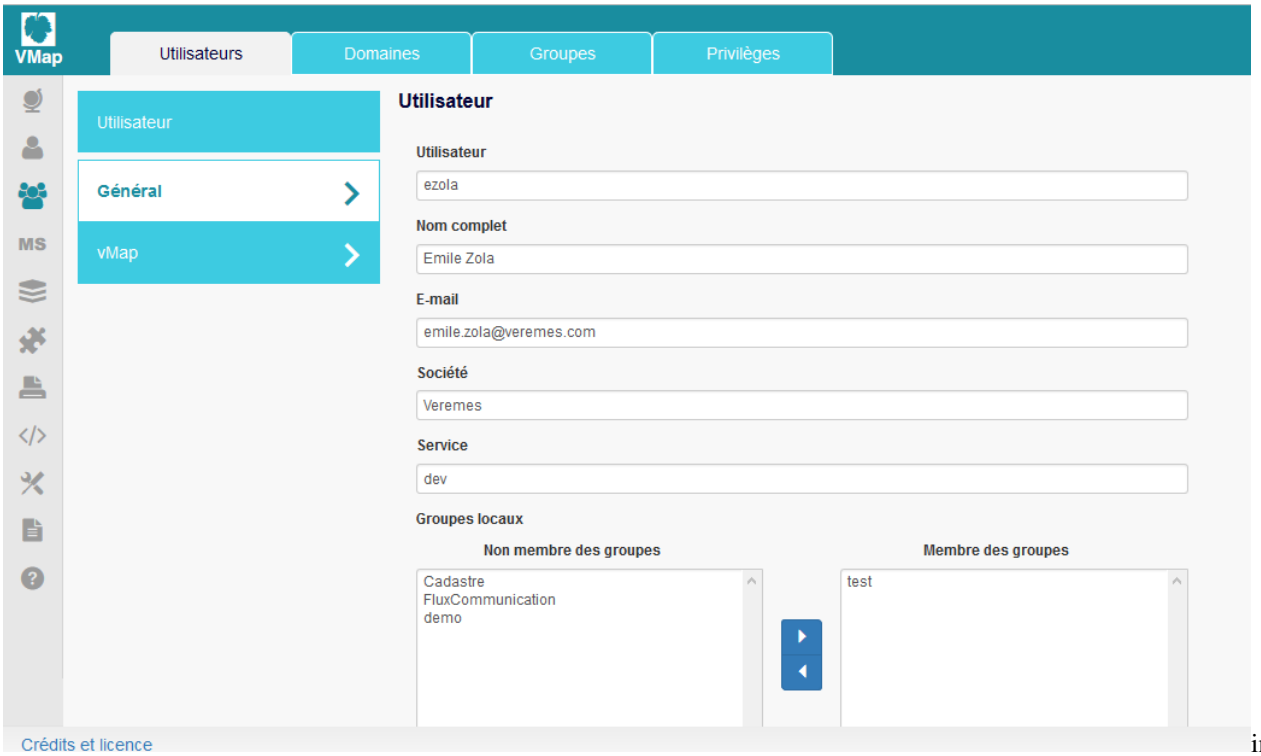
Un utilisateur vMap est un compte connu par l'application vMap qui peut se connecter et utiliser ses services.

Deux profils d'utilisateurs sont à distinguer :

- Utilisateurs PostgreSQL : utilisateurs authentifiés par la base de données interne à vmap, PostgreSQL, créés directement dans vMap.
- Utilisateurs Active Directory (AD) : utilisateurs d'un domaine et authentifiés par un annuaire Active Directory, importés dans vMap.

4.1.1 1. Création d"utilisateurs et de groupes PostgreSQL

Le mode Utilisateurs > Onglet Utilisateurs liste l'ensemble des utilisateurs. Il permet l'ajout de nouveaux utilisateurs, leur édition et suppression. Après avoir cliqué sur 'Ajouter un utilisateur', le formulaire suivant s'affiche :



image

4.1.2 2. Création d"utilisateurs et de groupes d"un annuaire Active Directory

Il est possible de gérer plusieurs domaines et d'exploiter des groupes de sécurité définis directement dans un annuaire Active Directory. L'administrateur a la possibilité d'importer des utilisateurs et des groupes depuis Active Directory. Cette méthode permet d'hériter des droits issus de la gestion centralisée AD des utilisateurs au sein d'un organisme. L'administrateur crée le nouveau domaine, puis importe les utilisateurs et les groupes. L'attribution des groupes ainsi que les mots de passe des utilisateurs ne pourront pas être changés.

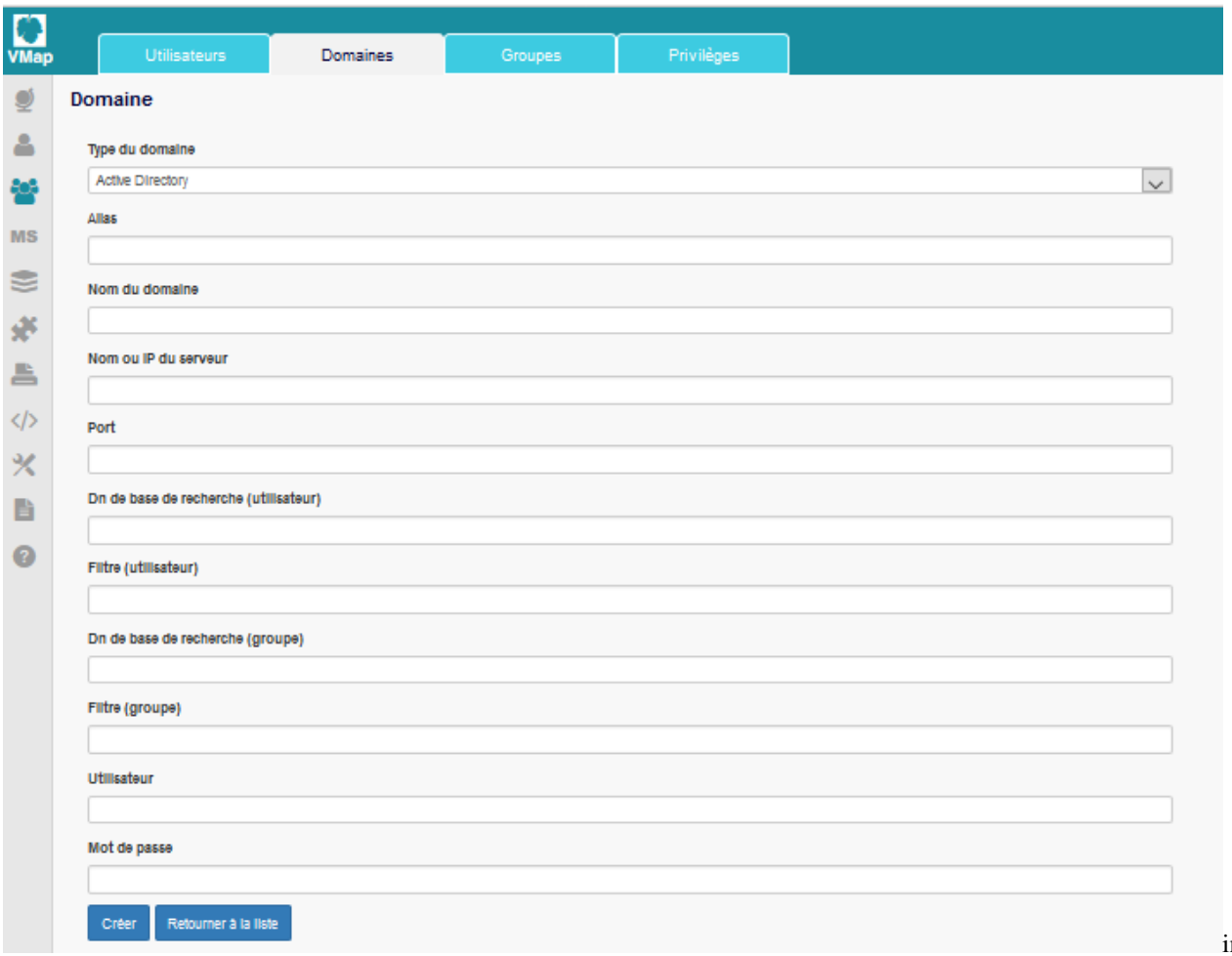
2.1. Ajout de domaines Active Directory

Le mode Utilisateurs > Onglet Domaines liste les domaines Active Directory. Il permet de créer, modifier et supprimer des domaines. Le bouton 'Ajouter un Domaine' affiche le formulaire suivant.

L'administrateur saisit les informations suivantes :

— Type, Nom et Alias du domaine : le nom de domaine utilisé pour la connexion. Par exemple, 'siege.entreprise.com'.
— IP ou nom de serveur : adresse IP ou nom du serveur assurant le rôle de serveur Active Directory.
— Dn de base de recherche (utilisateur) : point d'entrée pour la recherche des utilisateurs.
— Dn de base de recherche (groupe) : point d'entrée pour la recherche des groupes.
— Filtre (utilisateur) : permet de définir des filtres pour la recherche des utilisateurs.
— Filtre (groupe) : permet de définir des filtres pour la recherche des groupes.
— Vérifier les droits lors du lancement du robot : permet au robot de vérifier si un utilisateur du domaine possède toujours les droits lors de l'exécution du traitement. Pour cela le robot a besoin de connaître le login et le mot de passe d'un utilisateur du domaine.
— Login : login d'un utilisateur du domaine.

* Mot de passe : mot de passe de l'utilisateur du domaine. Le login et le mot de passe saisis ici permettent de vérifier les droits de l'utilisateur Active Directory lors de l'exécution du robot. En cliquant sur « Créer » la procédure de création de domaine Active Directory est finalisée.



image

L'administrateur doit ensuite modifier manuellement le fichier de configuration de la base de données PostgreSQL pour autoriser la connexion des utilisateurs du domaine. Dans le répertoire d'installation de PostgreSQL, modifier à l'aide d'un éditeur de texte le fichier pg_hba.conf situé dans le dossier data.

Avant modification vous devriez avoir la configuration suivante :

```
host    all     u_scheduler         127.0.0.1/32    trust
host    all     +superusers         127.0.0.1/32    md5
host    all     all                 127.0.0.1/32    md5
# IPv6 local connections:
host    all     u_scheduler         ::1/128         trust
host    all     +superusers         ::1/128         md5
host    all     all                 ::1/128         md5
```

Vous devez rajouter les deux lignes suivantes :

```
host    all     +gtf_nomdomaine     127.0.0.1/32    ldap ldapserver=nomduserveur␣
↪ldapprefix=""
host    all     +gtf_nomdomaine     ::1/128         ldap ldapserver=nomduserveur␣
↪ldapprefix=""
```

Pour obtenir :

```
host    all      u_scheduler        127.0.0.1/32    trust
host    all      +superusers        127.0.0.1/32    md5
host    all      +gtf_nomdomaine    127.0.0.1/32    ldap ldapserver=nomduserveur␣
↪ldapprefix=""
host    all      all                127.0.0.1/32    md5
# IPv6 local connections:
host    all      u_scheduler        ::1/128         trust
host    all      +superusers        ::1/128         md5
host    all      +gtf_nomdomaine    ::1/128         ldap ldapserver=nomduserveur␣
↪ldapprefix=""
```

2.2. Import d"utilisateurs d"Active Directory

Dans l"onglet Utilisateurs, le bouton « Importer de l'AD » permet d'importer des utilisateurs à partir d'un serveur Active Directory. Une interface de connexion apparaît à l'écran. L'administrateur saisit son login et mot de passe Active Directory afin de se connecter au domaine précédemment créé. Une fois la connexion effectuée, l'administrateur peut soit naviguer dans l'arborescence de l'annuaire du domaine, soit effectuer une recherche à partir de critères.

4.1.3 3. Gestion des privilèges

Les privilèges préfixés par « vitis_ » correspondent aux droits propres au socle de développement Vitis :

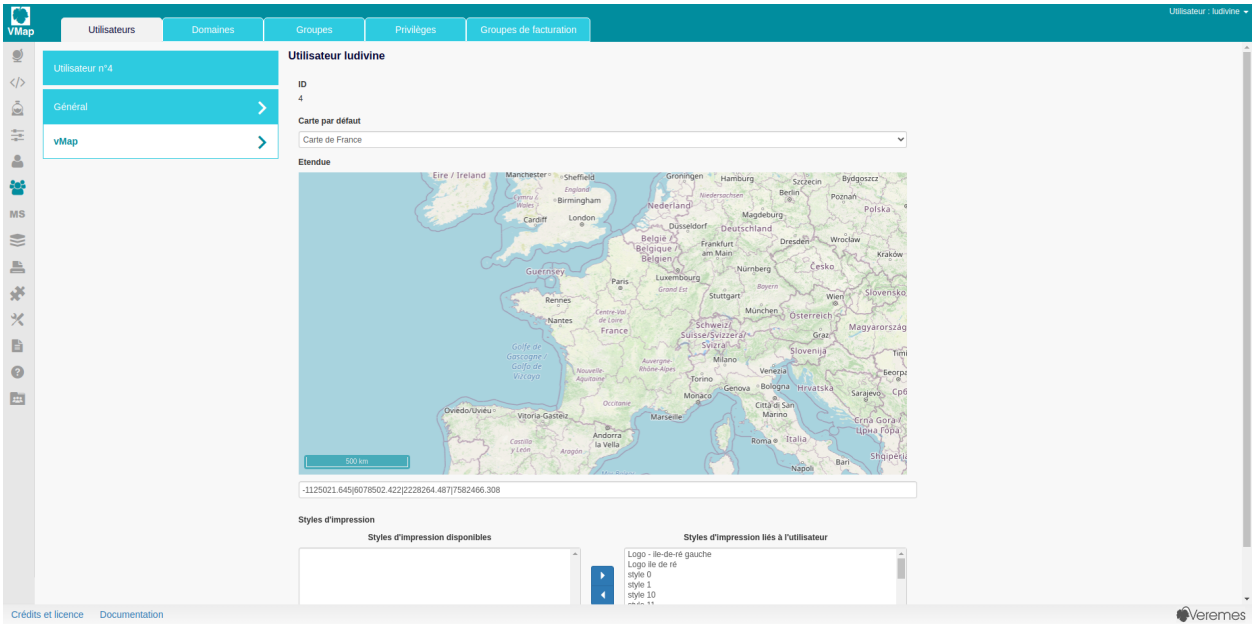
- Le privilège vitis_user permet de se connecter à l"application et d"accéder au mode Utilisateur.
- Le privilège vitis_admin permet l"accès aux modes Utilisateurs, Configuration, Logs. Il a en charge la gestion des paramètres système et de la configuration de GTF. Il accède également dans le mode Aide à la documentation relative à l"API Vitis.

Des profils d"utilisateurs propres à vMap et GTF sont disponibles, chacun ayant des privilèges spécifiques. L"accès aux modes dépend des privilèges attribués à l"utilisateur.

- vmap_user = utilisateur de vMap ayant accès aux modes Visualisation cartographique.
- vmap_admin = l"administrateur de l"application vmap a tous les droits.
- gtf_admin = dans GTF, en plus des privilèges vitis_admin, il accède au mode Moteurs dans lequel il configure les moteurs FME et GTF. Il a en charge la gestion des utilisateurs de GTF et de la base de données.
- gtf_author = l"auteur publie des projet FME sur GTF et les met à la dispsotion des utilisateurs en ayant droit.
- gtf_user = utilisateur de GTF qui éxécute des projets FME auxquels il a accès.
- vmap_cadatsre_user = utlisateur du module Cadastre de vMap
- vm4ms_admin = administrateur de vMap4MapServer

4.2 Configuration avancée des utilisateurs vMap

L"édition d"un utilisateur vMap rend disponible une section « vMap » dans laquelle plusieurs paramètres sont paramétrables :



4.2.1 1 Carte par défaut

La « carte par défaut » permet de choisir la carte qui s''ouvre lorsque l''utilisateur lance l''application.

Si aucune carte n'est sélectionnée pour l'utilisateur, l'application s'ouvrira sur la carte ayant « l'ordre » le plus petit, parmis l'ensemble des cartes disponibles pour l'utilisateur.

4.2.2 2 Etendue par défaut

« L'étendue » permet de définir l'étendue sur laquelle s'ouvre la carte quand l'utilisateur lance l'application.

Si l'étendue par défaut n'est pas définie pour l'utilisateur, la carte s'ouvrira sur l'étendue par défaut de la carte.

Ce paramétrage peut être très utile pour définir, par exemple, une étendue à certains utilisateurs ayant une restriction communale.

4.2.3 3 Styles d'impression

Le « style d'impression » permet d'attribuer à chaque utilisateur un style personnalisé. Ce paramètre est utilisé dans les impressions et les rapports.

Si aucun style n'est définit pour l'utilisateur, ce dernier ne pourra en sélectionner au moment de la génération d'une impression ou d'un rapport.

4.3 Mode calques et cartes

Le mode Calques et cartes permet la création, la configuration et la suppression des tous les éléments consitutifs des cartes affichées dans vMap.

Le mode Calques et cartes est organisé en 5 onglets :

- Onglet Services
- Onglet Calques

— Onglet Cartes
— Onglet Thèmes des calques
— Onglet Thèmes des cartes

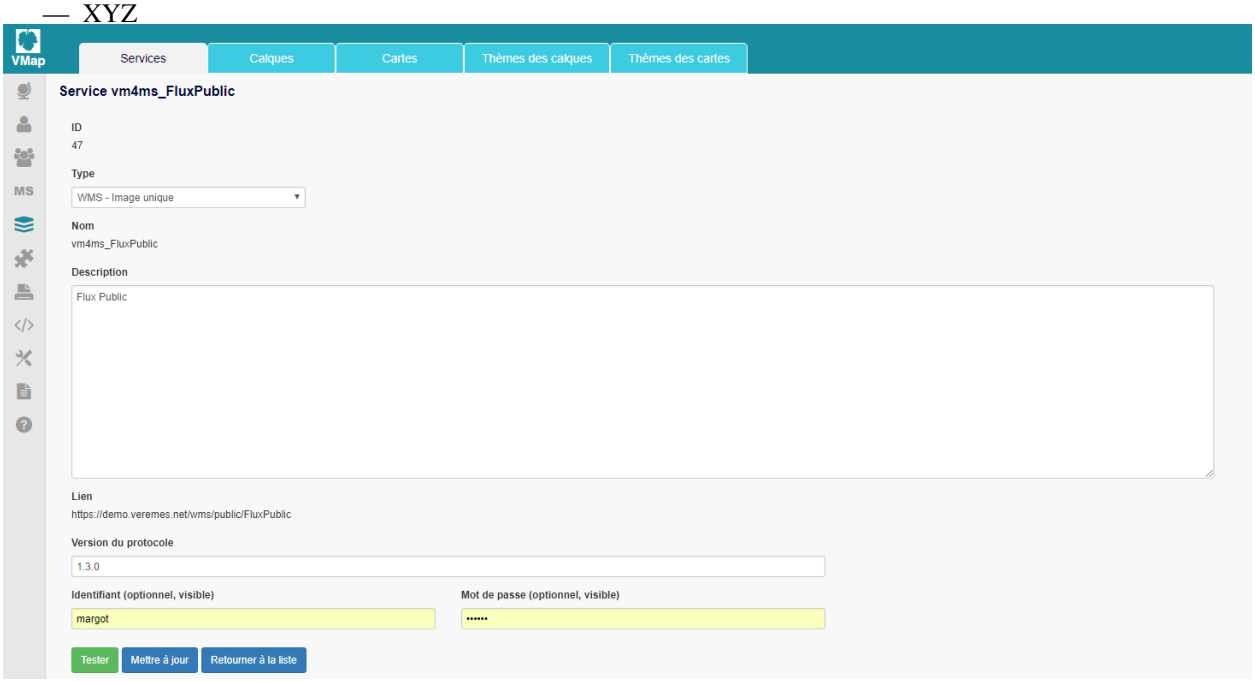
4.3.1 Onglet service : exploitation de flux

L'onglet Services permet la création, l'édition et la suppression de service.

Un service vMap est le nom donné pour représenter l'URL d'un flux. Un flux peut être privé ou public.

L'utilisation d'un service dans vMap correspond à l'utilisation d'un flux public ou d'un flux WMS prélablement ajouté dans le mode MapServer.

Type de services disponibles :
— WMS - image unique
— WMS - mutli tuilage
— Bing Maps
— OSM
— WMTS
— XYZ

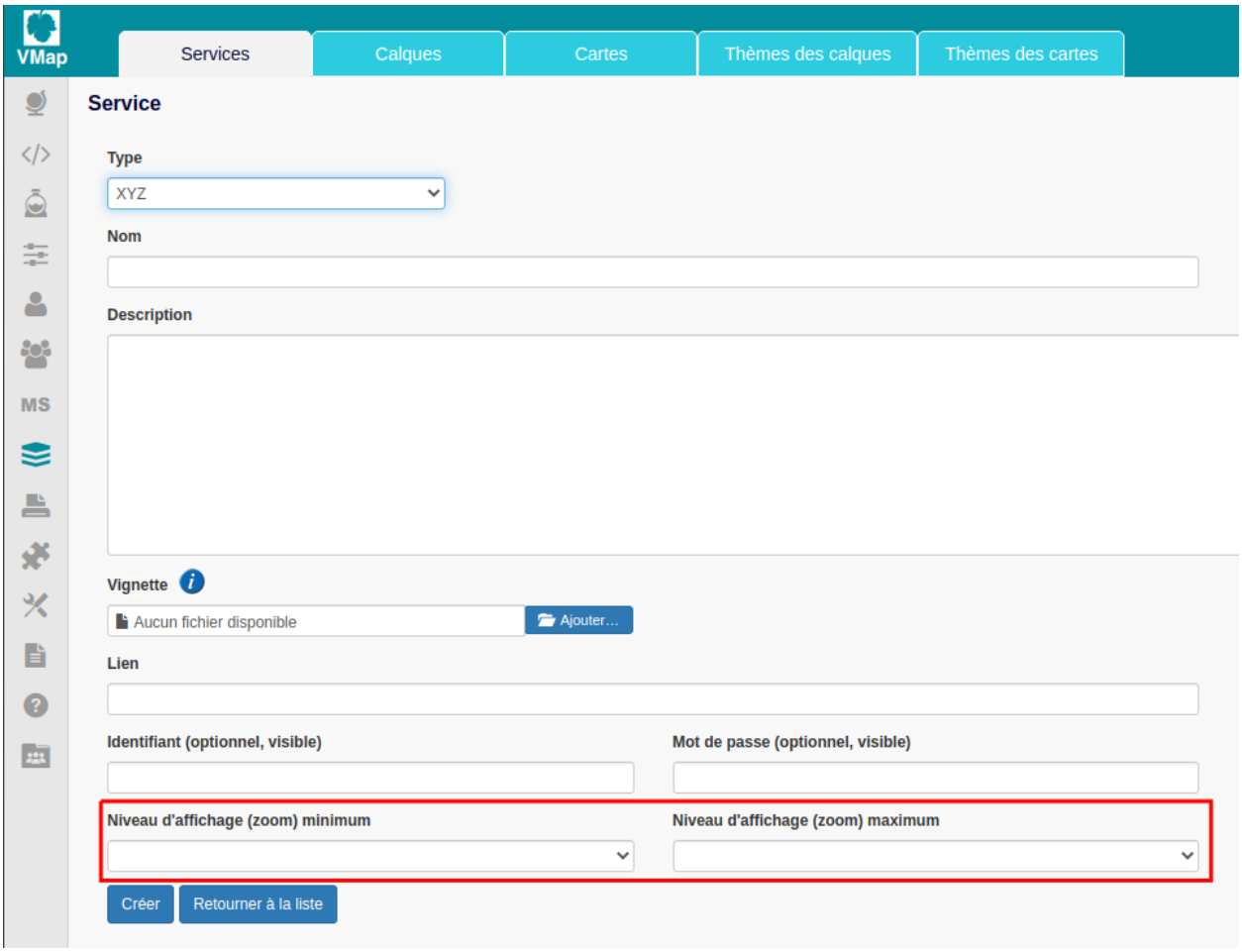


Après avoir sélectionné le type de flux à exploiter, nommer le service, le décrire et lui associer le lien du flux . Indiquer le version du protocole ainsi que les identifiants de connexion au service.

Une fois testé, cliquer sur Créer pour confirmer la création du service.

Le service xyz

Depuis la version 2020.02 deux paramètres ont été rajoutés pour les services XYZ. Un niveau de zoom minimum et un niveau de zoom maximum mettre de définir les niveaux à partir desquels les images ne seront plus chargées.



4.3.2 Gestion des calques

Dans vMap, un calque est un ensemble de couches provenant d'un seul et même service vMap.

4.3.3 Gestion des cartes

Dans vMap une carte est un ensemble de calques issues de un ou plusieurs services WMS.

Clonage de carte

Le bouton cloner permet de cloner l'ensemble des paramètres d'une carte. Il reprend notemment l'ensemble des informations attributaires de la carte clonée mais aussi l'ensemble des calques et leurs caractéristiques (opacité, visibilité par défaut, superposition).

4.3.4 Thèmes des calques

Las calques sont organisés par thèmatique. L'onglet Thèmes des calques permet la création, l'édition et la suppression de thèmes.

4.3.5 Thèmes des cartes

Las cartes sont organisées par thèmatique. L'onglet Thèmes des cartes permet la création, l'édition et la suppression de thèmes.

4.4 Mode MapServer

Le mode MapServer permet la publication de flux WMS avec MapServer.

4.4.1 Couches

Documentation en cours de rédaction..

4.4.2 Connexions

Documentation en cours de rédaction..

4.4.3 Flux public

Documentation en cours de rédaction..

4.4.4 Flux privés

Documentation en cours de rédaction..

Utiliser des mots de passe cryptés dans les mapfile du flux privé

Pour toute application ouverte à un certain nombre d'utilisateurs, il est utile voir indispensable de crypter les mots de passes situés à l'intérieur des mapfiles générés lors de l'utilisation de flux privés.

Pour faire ceci, Mapserver propose la librairie **msencrypt** qui va crypter le mot de passe pour qu'il ne soit accessible uniquement par Mapserver lui même.

Générer un fichier la clé de cryptage

Tout type de cryptage nécessite une clé de cryptage qu'il va falloir générer et copier dans le dossier « /var/www/vmap/vas/ws_data/vm4ms/map/msencrypt/ ».

Etape 1 : Vérifier si ce dossier existe.

Etape 2 : Si ce dossier n'existe pas, le générer à partir de la commande ci-après.

```
sudo mkdir /var/www/vmap/vas/ws\_data/vm4ms/map/msencrypt/
```

Etape 3 : Générer la clef de cryptage dans le dossier créé ci-dessus. Puor ce faire, exécuter la commande suivante :

```
sudo /var/www/vmap/vas/server/mapserver/bin/msencrypt -keygen /var/www/vmap/vas/ws_
↪data/vm4ms/map/msencrypt/vm4ms_key.txt
```

Modifier la définition du flux privé

Pour dire à Mapserver d'utiliser la clé de cryptage il faudra indiquer le fichier précédemment généré dans la définition, pour cela écrire la ligne ci-dessous.

```
CONFIG "MS_ENCRYPTION_KEY" "/var/www/vmap/vas/ws_data/vm4ms/map/msencrypt/vm4ms_key.
↪txt"
```

Modifier les properties du module Mapserver

Pour activer le cryptage des mots de passe par le module Mapserver, il faudra modifier le fichier de properties vmap/vas/rest/conf/vm4ms/properties_post.inc et mettre la propertie **use_msencrypt** à true

4.4.5 Flux WFS

Bien que l'interface ne soit pas prévu pour, vMap est en mesure de publier des flux au format WFS sans problème **pour les serveurs Linux**.

1. Objet Web

La première des choses à faire c'est de créer un objet web pour l'utilisation du WFS, pour ceci saisir un nom puis la définition suivante.

```
WEB
        METADATA
                "wfs_title"           "{WMSSERVICE_ID}"
              "wfs_onlineresource" "{MS_CGI_URL}public/{WMSSERVICE_ID}"
              "wfs_srs"            "EPSG:4326 EPSG:2154 EPSG:3857"
              "wfs_abstract"       "This text describes my WFS service."
              "wfs_enable_request" "*"
        END
END
```

2. Flux public

Ensuite il faudra créer un flux, pour ceci il faudra se rendre sur l'onglet **Flux WMS publics** puis **Ajouter un flux wms public** et y ajouter la définition suivante.

```
MAP
        NAME Flux_WFS_public
        STATUS ON
          SIZE 400 300
        SYMBOLSET "../symbols/symbols.sym"
        EXTENT -180 -90 180 90
          UNITS DD
        FONTSET "../fonts/fonts.list"

        CONFIG "MS_ERRORFILE" "{MS_LOG_FILE}"
        CONFIG "PROJ_LIB" "{MS_PROJ_DIR}"
        DEBUG {MS_DEBUG_MODE}
```

(suite sur la page suivante)

(suite de la page précédente)

```
        PROJECTION
                "+init=epsg:4326"
        END

        {WEB}
        {LAYERS}
END
```

Il faudra bien entendu utiliser l"objet web décrit précédemment

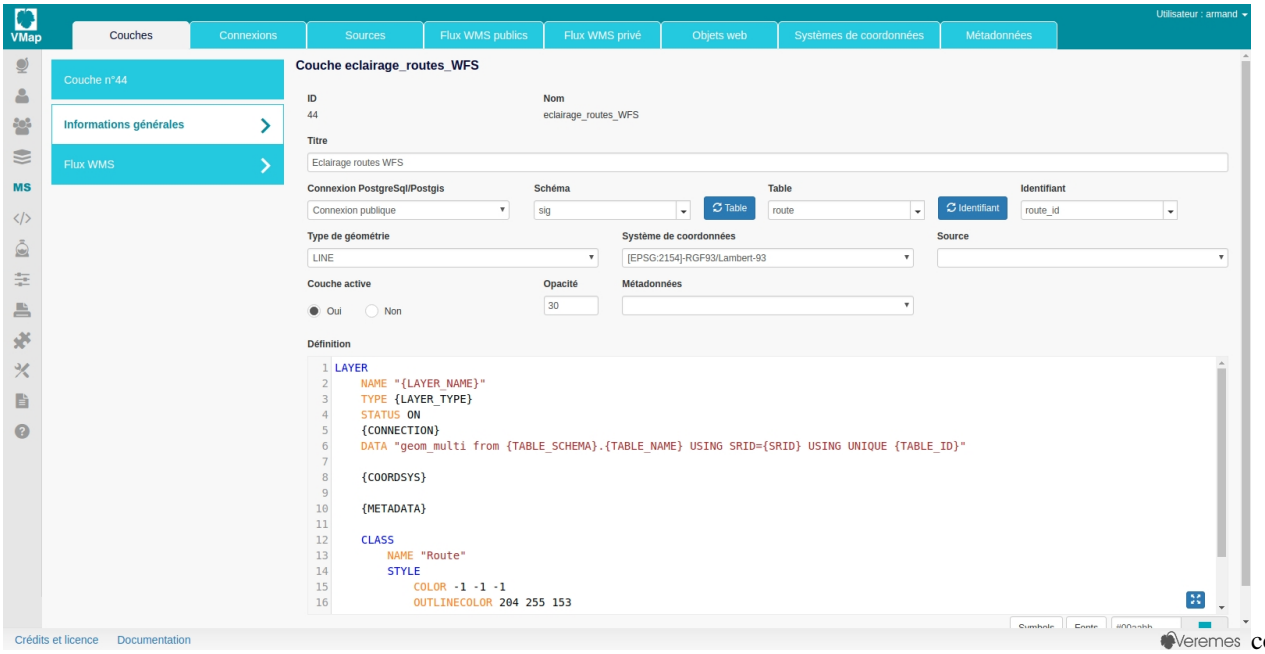
3. Métadonnées

La troisième étape consiste à créer une métadonnée qui sera utilisée dans les couches, pour cela utilisez la définition ci-dessous.

```
METADATA
        "wfs_title"          "{LAYER_TITLE}"
        "wfs_srs"            "EPSG:2154 EPSG:3857 EPSG:4326"
        "gml_include_items" "all"
        "gml_featureid"      "{TABLE_ID}"
        "wfs_enable_request" "*"
END
```

4. Couche

La dernière étape est de créer la/les couche(s) que l"on souhaite publier, pour cela on choisira obligatoirement une connexion publique, pour le reste il faudra la configurer comme on le fait d"habitude avec les couches WMS de type vecteur.



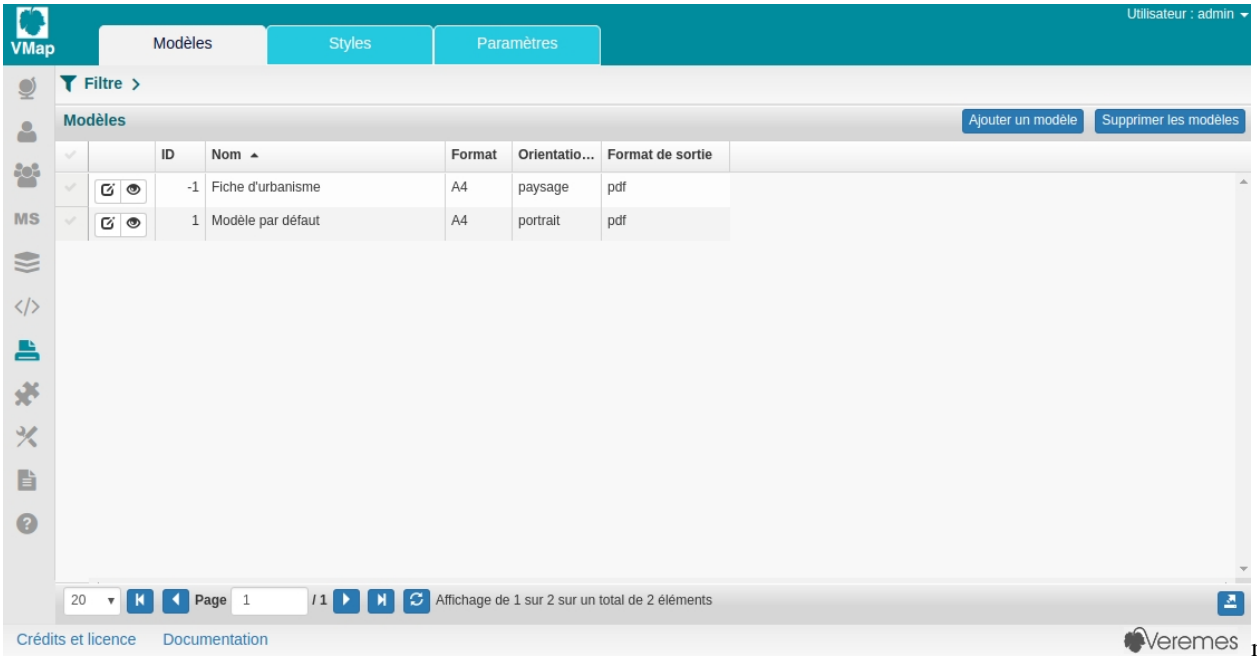
couche wfs

5. Publier le flux

Enfin, retournez sur le flux précédemment crée, associez la couche puis sauvegardez le flux. Votre flux est désormais disponible sur `https://[votre serveur]/wms/public/[nom du flux]?service=wfs&version=1.1.0&request=GetCapabilities`

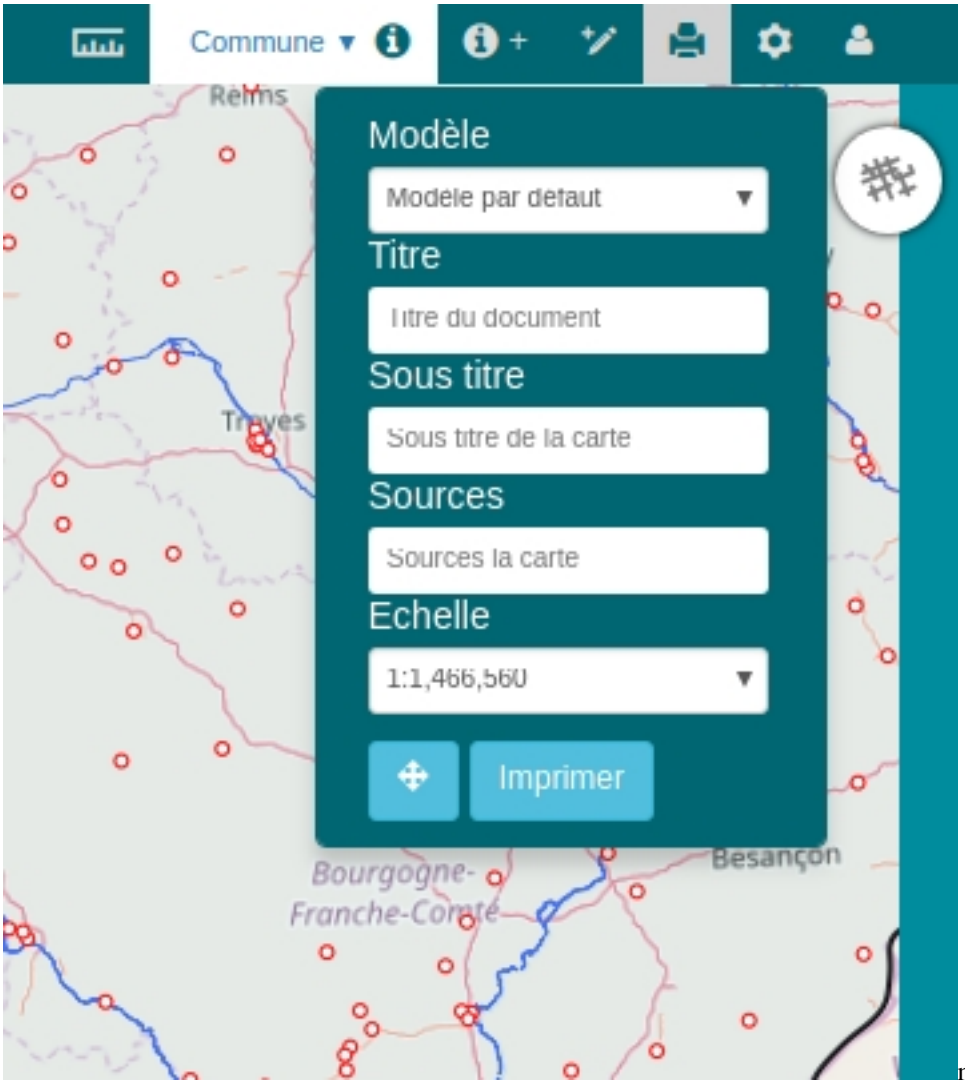
4.5 Mode impressions

Le mode impressions fait partie des modes d'"administration de vMap, il se compose de trois objets permettant de créer et modifier des modèles, des styles et des paramètres d'"impressions.



mode impressions

Une fois ces trois modes rensignés, les utilisateurs pourront utiliser depuis le mode vMap l'"outil d'"impression afin de générer des fichiers PDF avec des cartes, des paramètres et autres types d'"entités.



mode impressions

4.5.1 Objet Modèles

La première des choses à faire est de créer un modèle d'impression, pour cela il faudra cliquer sur **Ajouter un modèle** et remplir le formulaire en associant un nom, un format d'impression, une orientation, les groupes d'utilisateurs qui pourront utiliser ce modèle ainsi qu'une définition HTML.

Structure de la définition

Une définition est écrite en langage HTML et sera composée d'un style CSS ainsi que d'un corps, nous utiliserons par défaut la définition ci-dessous.

Il s'agit d'un modèle format A4 portrait, pour cela la première qui prend pour id #A4_print_template voit son style définir une hauteur de 21cm c'est à dire la hauteur d'une feuille A4. Les identifiants map_legend, map_image et map_overview sont des mots clés permettant d'afficher les différents éléments cartographiques. D'autres éléments comme les classes logo, head_element etc.. affichent des éléments de décoration, vous remarquerez qu'on utilise des logos au format base64 pour des soucis de performance et que par défaut les éléments sont en position absolue ce qui permet de les placer facilement dans la page. Pour que les couleurs s'affichent correctement il faudra utiliser la syntaxe « !important ».

```
    <style>
        #A4_print_template {
            width: 21cm;
        }
        #map_legend {

        }
        #map_image {
            background-color: #DFDFDF;
            height: 19cm;
            width: 14cm;
            border: 1px solid black;
        }
        #map_overview {
            background-color: #DFDFDF;
            height: 4cm;
            width: 4cm;
            border: 1px solid black;
        }
        .logo {
            height: 1cm;
        }
        .head_element, .body_element, .footer_element {
            text-align: center;
            position: absolute;
        }

    </style>
    <div id="A4_print_template">

        <!-- Entête -->
        <div class="head_element" style="top: 1.5cm; left: 2cm;">
            <img class="logo" src="
↪iVBORw0KGgoAAAANSUhEUgAAAM8AAAA+CAIAAAADLdi0AAAAFXRFWHRDcmVhdGlvbiBUaW1lAAfgARUXDAt3OP8+AAAAB3RJTUU
↪x3q6ugBkERQXJJsyYwZwWs+IqYFofAq6rYiAYwAwrAkqWDIqouwbAAAiSEURgFQTFCCrq+5iLtb09Mz0zCL53PHVPHc7QU9VVfe
↪7VhbV+ffVXr3QoLCzkfOdo6wB6+/
↪bt3t17e2j3FIJM1HS0uvuf8G9paeHc52hrP71582bp4mUMVSOGjfTY6ZGclJx+I/3K5StH/
↪I5MmjCpy7+74iv89fL0/vjxIxcAR1t7CLrK2XE94czQYGBUZFRzc7Ooz7t370LPhJLmg5FNiE/gAuBoaw/
↪Fx8WT3rIwtawor5DQW+dTzmt11UbPAfqGtbW1XAYcbZoRlNYwq+EAUHetHnfv3pXuDCDu3bOPtOAvm7Zwe8rRphnBLSP0uGxxUa
↪y5cvh1uNQH9Y1bKyMi4GjjYxPW14eu/ePdjBjPSMyPDIDx8+QC35+R5+8/qNs1Orx9b1+x9LS0qp8/
↪v37wNPnc7KylZ2t5wJOYuA4OnTp0OMTHB97OhxiAwyMzILCwsXLfgJV7a6uAb4BwQHBadeSBVBCnC0NLNEH93ees+fP9do0biVy5at
↪UNOxnP6v0dZQ31BZWYmQs7ioWJStJazk5ubCXcPnnx3WOjs6A4jaP+rQt/
↪hMH0aPHP3mzRvRnY8dPU7fhoWGa7ToBw8eUGA7b/
↪Y8qFguxW8Hbfv2ekK0C+YuMDe16NZVSx5tyUnJ9HnWjNnKMroIV8eNHl9dXS288+NHjwmmdtNnaGRM/
↪Xz96Lbnos5xEX47aIPmIJuorPXtpet/
↪wp8+T5wwSaInoUqYhAPCli1pzQbrdOsO9anmit++fTtqxGiMMujXn5vRbwptoWfDSP0oazCarlvdPpvLMdJoQ9uxfYfwzIrpxUN
↪kcvv20FbfX2DXt9+KgEEb12Waes+sP8glZ0BFB/v/
↪cJUiNFA41akWo959+6dOitGfEC3upl5k8vv20Hb+ZTzKtGD1qdnX8rr9tTppbLz4p8WIxoVzuLutq0Vhd93vX37tsrlwhDr6xmq
↪Pz8yIhI34O+3p7eRw4fTYhPqHxQKVEEAIMO64w7M7jjQ42MFPZved9S9bAqOSnl+LHjmOLg/
↪oPhoeF5uXlNb5okVo6nbpYR81Ax6f2y+xdT02LOxaQkp+TnF7x8+UqZ//
↪rixYu8vLykxCR0vph6sbSktKmpSX0xY66y0jKM9fM9jDX7HfI7F32upKQE19UZDqVw7dr104FBgadOX7l8paGhoX1JKym07fbYo
↪fh3g/7v1iFGiueioeuyocWdCzsifaP+9tVpazoScNR5kLKoc8/
↪HykRjVHrQtW7K8w9FGbYbtzIqKtrNUqCiYUTo2ld6v6L/
↪EfglFFRguvH7p0mWjf3JEvoHdSxcvkw8sMKmJsSk6zJk197Dv4W5dtD6fqv0DbTXVNTPtZomgLI/
↪soZbDcrJz5LcNcwAy0jPGjRkvsc5+fftFRkR9lFFYaDiZDmVt4viJ9U/qlXEMGtd6uLV41L/
↪FN5k+dboI4ixG9NjpIQ9TauudNmhaNvadhGjtFy3uJLSR5fU91BYZnDh+gi4iaJBYK0xbr+69Kc0mFCdEQmnn1vKT/
↪gOhO1MvpBbkFxQVFv1x5Q/
↪oG0iXwWLMqLF1dXUK0abbW48lFPv21t3jsYf1wd2MBw9hK8c+jAiPyM7KLiu7n5l5MyQ4BLzCHmCuxdU/
↪r4oAx9Cmr2tAH4YYmfzqvh33gWUMDgoBuLHfGIjDwyIC/
↪ANI0yq7YZyNt5dBfFxCUmLyyeMn7RfZs+lmz5yj0KlIu5jWu2cfljpY77Q+Pi4e+IMVvnH9xskTJ+2m2bGtZTxoSHl5hegOMNzN
↪orDG3URg6ZhqsndE3qaussTC3alj3OBm6AvOrCFfhVNuMnUjcDvf7l5eUK0UaHMYCyvBGsr6//
↪aaE99QFDKBuAPYDFw1kUdYabC8srk3qXS2mX5LUai/
↪CwEx5WKlZd2BWmxmZtDz7curGxUdiBPc5hvyPskcFMZ0dnur5i2QqNHDgp3Qb4M0F2eEu/
↪kS6ci7gM1VX7uFaZlz3JZnKrodHVZ0x58uTJ4IFGdEPPfV7S5wpwEPX6tAngVMAphWjD7oLIRQNXrVhN306ZNFUkD7Ev/
```

(suite sur la page suivante)

(suite de la page précédente)

```
        </div>
        <div class="head_element" style="top: 1cm; left: 0cm; width: 21cm">
            <h1 style="margin-bottom: 0px; padding-bottom: 0px;">{{title}}</h1>
            <i>{{headline}}</i>
        </div>
        <div class="head_element" style="top: 1.5cm; left: 16cm;">
            <img class="logo" src="
↪iVBORw0KGgoAAAANSUhEUgAAAM8AAAA+CAIAAAADLdi0AAAAFXRFWHRDcmVhdGlvbiBUaW1lAAfgARUXDAt3OP8+AAAAB3RJTUU
↪x3q6ugBkERQXJJsyYwZwWs+IqYFofAq6rYiAYwAwrAkqWDIqouwbAAAiSEURgFQTFCCrq+5iLtb09Mz0zCL53PHVPHc7QU9VVfe
↪7VhbV+ffVXr3QoLCzkfOdo6wB6+/
↪bt3t17e2j3FIJM1HS0uvuf8G9paeHc52hrP71582bp4mUMVSOGjfTY6ZGclJx+I/3K5StH/
↪I5MmjCpy7+74iv89fL0/vjxIxcAR1t7CLrK2XE94czQYGBUZFRzc7Ooz7t370LPhJLmg5FNiE/gAuBoaw/
↪Fx8WT3rIwtawor5DQW+dTzmt11UbPAfqGtbW1XAYcbZoRlNYwq+EAUHetHnfv3pXuDCDu3bOPtOAvm7Zwe8rRphnBLSP0uGxxUa
↪y5cvh1uNQH9Y1bKyMi4GjjYxPW14eu/ePdjBjPSMyPDIDx8+QC35+R5+8/qNs1Orx9b1+x9LS0qp8/
↪v37wNPnc7KylZ2t5joGAKo21Z3rt442sTU+KyR/
↪K1uXbQArC2bt5wJOYuA4OnTp0OMTHB97OhxiAwyMzILCwsXLfgJV7a6uAb4BwQHBadeSBVBCnC0NLNEH93ees+fP9do0biVy5at
↪UNOxnP6v0dZQ31BZWYmQs7ioWJStJazk5ubCXcPnnx3WOjs6A4jaP+rQt/
↪hMH0aPHP3mzRvRnY8dPU7fhoWGa7ToBw8eUGA7b/
↪Y8qFguxW8Hbfv2ekK0C+YuMDe16NZVSx5tyUnJ9HnWjNnKMroIV8eNHl9dXS288+NHjwmmdtNnaGRM/
↪Xz96Lbnos5xEX47aIPmIJuorPXtpet/
↪wp8+T5wwSaInoUqYhAPCli1pzQbrdOsO9anmit++fTtqxGiMMujXn5vRbwptoWfDSP0oazCarlvdPpvLMdJoQ9uxfYfwzIrpxUN
↪kcvv20FbfX2DXt9+KgEEb12Waes+sP8glZ0BFB/v/
↪cJUiNFA41akWo959+6dOitGfEC3upl5k8vv20Hb+ZTzKtGD1qdnX8rr9tTppbLz4p8WIxoVzuLutq0Vhd93vX37tsrlwhDr6xmq
↪Pz8yIhI34O+3p7eRw4fTYhPqHxQKVEEAIMO64w7M7jjQ42MFPZved9S9bAqOSnl+LHjmOLg/
↪oPhoeF5uXlNb5okVo6nbpYR81Ax6f2y+xdT02LOxaQkp+TnF7x8+UqZ//
↪rixYu8vLykxCR0vph6sbSktKmpSX0xY66y0jKM9fM9jDX7HfI7F32upKQE19UZDqVw7dr104FBgadOX7l8paGhoX1JKym07fbYo
↪fh3g/7v1iFGiueioeuyocWdCzsifaP+9tVpazoScNR5kLKoc8/
↪HykRjVHrQtW7K8w9FGbYbtzIqKtrNUqCiYUTo2ld6v6L/
↪EfglFFRguvH7p0mWjf3JEvoHdSxcvkw8sMKmJsSk6zJk197Dv4W5dtD6fqv0DbTXVNTPtZomgLI/
↪soZbDcrJz5LcNcwAy0jPGjRkvsc5+fftFRkR9lFFYaDiZDmVt4viJ9U/qlXEMGtd6uLV41L/
↪FN5k+dboI4ixG9NjpIQ9TauudNmhaNvadhGjtFy3uJLSR5fU91BYZnDh+gi4iaJBYK0xbr+69Kc0mFCdEQmnn1vKT/
↪gOhO1MvpBbkFxQVFv1x5Q/
↪oG0iXwWLMqLF1dXUK0abbW48lFPv21t3jsYf1wd2MBw9hK8c+jAiPyM7KLiu7n5l5MyQ4BLzCHmCuxdU/
↪r4oAx9Cmr2tAH4YYmfzqvh33gWUMDgoBuLHfGIjDwyIC/
↪ANI0vg7YZyNt5dPfFxCUmLyyeMn7RfZs+lmz5yj0KlIu5jWu2cfljpY77Q+Pi4e+IMVvnH9xskTJ+2m2bGtZTxoSHl5hegOMNzN
↪orDG3URg6zhqsndE3qaussTC3alj3OBm6AvOrCFfhVNuMnUjcDvf7l5eUK0UaHMYCyvBGsr6//
↪aaE99QFDKBuAPYDFw1kUdYabC8srk3qXS2mX5LUai/
↪CwEx5WKlZd2BWmxmZtDz7curGxUdiBPc5hvyPskcFMZ0dnur5i2QqNHDgp3Qb4M0F2eEu/
↪kS6ci7gM1VX7uFaZlz3JZnKrodHVZ0x58uTJ4IFGdEPPfV7S5wpwEPX6tAngVMAphWjD7oLIRQNXrVhN306ZNFUkD7Ev/
↪/wF+lBn2OWPHz4qRJvnXk9lQoI/Du0rtNTQfMqmg+YmZWy/
↪0F54HVEOu8nPDmubm5qlD2ZMPz++21Z3dh1PSuoTqBWdAxUXF5PSNTex0MiYKkUbNjf2cedZ0tEjx4Cz8ok3mA+F68m/
↪l9+WZtuwiV302LWbRkEpikJdhQQlwXQP3Gd5tKGDaMitm7dIogMMDB8/
↪fqxyCrh3+rr6ZP4uX74ijzYri6HSuZ74+ATGpdUrHSR6QtJTZeCGdX7e+Lc/
↪Cr++zW2wHiMNNaKbmTcJWD11epXfb1PJCOTJzkLniSCFf48dPY54H/
↪GHRieHStEWHXWu86BGaeEjfkfkE29QYAq3y47tO2hg5uc0GywRmRJwSs03a3Dn8WMn0H0iIyJFaIPxqqqqEg1xWufUZk18D6vJ0
↪4rFGEtgvnL6iU/
↪fChI4gVKrUpswmIHugKoktSVDBz6icprl27TqYDu5FxeIRsGWiIil69fNVZaMMDO6xy6FS06Wh1F5V+wPsmb2DXjl0i9MALplH7
↪mYcTejaUxfImRiZgOKD8JS/
↪LKdVtRw8fZfq8s8LScTbCsnLAaPTI1uN2yF7k2axeuZoAKgz06BCCsh7w7tVsluZWBCA8HYIMabRlpGfQFNBV6k+BxpJkcbFxIr
↪IF059fsploPVaM0spVJUVMyCVm8vH+1uOsI8+eSJU06e8JeOljSPEuobpJOQHdKwXf4B8SNH6fq1q9eEfgzFkqLEkvM65y9Srt2
↪gK7Aef/CNV+7el2oArLuZC2cv0iU4zU3tbh+/
↪UaHoQ0KU52aji9sUDAOq9awcPLRo0eUZFq7Zu3fnlZQiMKAcet/2uS3eNHiX923a9p2/
↪rYTjrA02gD6tndjh45sxxRomRmZXxltLCtpO822fWu+X3Zf3rNCrLp3zz6W4iZf9s7tOx2DNmgRlmmUbyPlz0M0bxvXb8zOyhYP
↪keaqupDAkO+RLXVQJtpaVltKfnzZn/
↪hQ7yV0MbwgWaaPuvv3V4EQeiKMTmdKLTmoCcOKVjMiCgmHMxymwQnWyq32AEMQpWn4RnaWYFKCsMuBI+Z5siwiLw7/
↪375cRNZ8f1op5//nmVejqtc+oktEHYVEY12NBI5fn3/
↪wna4NpS5mLq5GmdVEYfHxvPogeFB6ztQRs8UIXQmTVjVpfvu2qEtiFGJqcDT0dHncvOztm5Y5dEfhz8hWhby+bsZoFZnvu82g6z
↪0BlGh49uyZmoml04FBHrt2o7EElQTa0H/N6p9pAedTLqjJUyyVpogWFLJ/
↪NbRhzXSQgOulpaVqrjkxIZHWfPXPq6TDtrlt27RhM/
↪iv8JiOHWlKiFIztFVXV8vjBu4h8ULT1qdX3wq5c1+F5OriRhq0rLTMzMScdKHCo4Jft22nm+/
↪x2KNOyu3WrdssIGUZSwm0geB40ZDxYydIV7CxPUCSwO6Pj4v/
↪+mgDBQUG0VyrVqxShy0VFRUUh+l008nLzaM6JVLqgwwHK1TqzMsCTDsGbbB0un30hBU7RoOMD+w/
↪2L7D0x7aPdU0RjnZOV1lupOVG3h7Ks6/I6qgugnEFpfSVOTNa2pqLM2t5MvjpNEmrIVR+epry/
↪sWx7VO7C0NYR7na6INT0SONbgHXa7CFXv5arrMM0ZzXOfEjO9kmykkdIVVrqNkuSo0jSJTKbSBWeGh4YMGDJYdNGn7+fohWlmye
```

(suite sur la page suivante)

(suite de la page précédente)

```
        </div>

        <!-- Corps -->
        <div class="body_element" style="top: 4cm; left: 1cm;width: 4cm;">
            <!--Pour afficher la légende, utiliser id="map_legend"-->
            <div id="map_legend"></div>
        </div>
        <div class="body_element" style="top: 4cm; left: 6cm;">
            <!--Pour afficher la carte, utiliser id="map_image"-->
            <img id="map_image" src="images/transparent.png">
        </div>

        <!-- Pied de page -->
        <div class="footer_element" style="top: 24cm; left: 1cm;">
            <!-- Pour afficher l'overview utiliser id="map_overview-->
            <img id="map_overview">
        </div>
        <div class="footer_element" style="top: 27.5cm; left: 0cm; width: 21cm">
            <label class="ng-binding">{{footer}}</label>
        </div>
        <div class="footer_element" style="top: 27.5cm; left: 17cm;">
            <!-- Pour afficher l'échelle actuelle utiliser {{map_scale}}-->
            <label class="ng-binding">Echelle: {{map_scale}}</label>
        </div>


    </div>
```

Identifiants clés

Comme je l"ai stipulé précédemment il y a plusieurs identifiants clés permettant d"afficher des différents éléments cartgraphiques.

- **map_image** : si vous donnez cet identifiants à une balise **<img>** alors la carte résultante de l"impression viendra s"y placer.
- **map_overview** : si vous donnez cet identifiants à une balise **<img>** alors la carte de supervision y sera inscrite.
- **map_legend** : si vous donnez cet identifiants à une balise ** ** alors le contenu de la légende sera copié dedans.

Cartes en comparaison

Lors de l"utilisation du mode comparaison seule la carte principale (située à gauche) est imprimée par défaut, pour imprimer la carte de droite il faudra utiliser les balises **map_image_compare** et **map_legend_compare**.

Une fois ceci fait on retrouvera la carte secondaire sur toutes les impressions qu"on soit ou pas en mode comparaison. Pour rendre ceci variable et afficher la carte secondaire uniquement si le mode comparaison est actif la variable scope **compare_mode** utilisée avec **ng-if** permettra de conditionner l"affichage.

```
<style>
    #A4_print_template {
        width: 21cm;
        font-family: arial;
        position: absolute;
    }
```

(suite sur la page suivante)

(suite de la page précédente)

```
    #map_overview {
        background-color: #D8D8D8 !important;
        height: 4cm;
        width: 4cm;
        border: 1px solid black;
    }
    .map_image {
        background-color: #D8D8D8 !important;
        height: 22cm;
        width: 14cm;
        border: 1px solid black;
    }
    .map_image_compare_mode {
        background-color: #D8D8D8 !important;
        height: 11cm;
        width: 14cm;
        border: 1px solid black;
    }
    .map_legend {
        width: 3cm;
        margin-top: 16px;
    }
    .map_legend_compare_mode {
        width: 3cm;
        margin-top: 16px;
    }
    .color_blue{
        color: #424A96 !important;
    }
    .logo {
        height: 2cm;
    }
    .container {
        position: absolute;
    }
    #header_container{
        width: 18cm;
        height: 2.5cm;
        background-color: #D8D8D8 !important;
    }
    #footer_container{
        width: 18cm;
        height: 1cm;
        background-color: #D8D8D8 !important;
    }
    .header_content{
        margin-left: 10px;
    }
    .title{
        display: block;
        font-size: 24pt;
        font-weight: bold;
        margin-top: 0.8em;
    }
    .headline{
        font-weight: bold;
        font-size: 14pt;
```

(suite sur la page suivante)

(suite de la page précédente)

```
    }
    .legend_headline{
        font-weight: bold;
        font-size: 14pt;
        color: black;
    }
    .footer_text{
        font-weight: bold;
        font-size: 10pt;
        margin: 0.3cm;
    }

</style>
<div id="A4_print_template">

    <div id="header_container" class="container" style="top: 1.2cm; left: 1.5cm;">
        <div class="container" style="top: -0.5cm; left: 0cm;">
            <div class="header_content title color_blue">{{title}}</div>
            <div class="header_content headline color_blue">{{headline}}</div>
        </div>
        <div class="container" style="top: 0.2cm; right: 10px;">
            <img class="logo" src="
↪iVBORw0KGgoAAAANSUhEUgAAAFsAAABvCAMAAAC0A0wQAAAACXBIWXMAAAsSAAALEgHS3X78AAAAB3RJTUUH4AoaDB4SgWapiwA
↪A9a/L97/V+c/g+9/q/e/1////97/V9a/L97/V9a/L8Y+29a/L97/V9a/
↪L3wBY3wFZ3wJZ4Ahe4Ale4Q9i4RNl4hVm4x9s4yBt4yNv5CZx5Clz5S935TN65z+B50CC6U+M6VOO6lmS61+W7Gic7W+h73+r8Y
↪AbLUkSgAAABR0Uk5TAAAAAAAAAAAAAAAAAAAAEBgrYIaUqrbZfjuKAAAE00lEQVRo3u2abZOcKBCAzW02y4vnxR1390jM5Tzihgj+/
↪78XeRXUUUSn6j4MVVtxBB+bpulu2mTYaw/VRuvkKLY+ZsRle9gEDYNEdRM2H8bA6jZsKTa9DZvFiL2HTapa/
↪dNWVTkM6cxddpjd5zlmpVw9zrVKiJ4BvK6bWLbQ93tOAa/
↪ocAWqYRoUSL23nHN6RO6xjyvjFlWNkPoNsX5VApt0cvalz5aTYNwfj/
↪s6hc0x4sLv41UuNYFw2HgCm+EZRGq5xngLvs2GU0apHqIz9sxgNtlz+XL5EOzmbEj2sfkckUuR84UODMguNpoTkGKXC2wM6x1su
↪6S5953hPY5TsYCO8rwWmssHGexuksWyK97feKWadXZcJ8DzWvvfDR0PcYne72XXsWop0jWyxSb6bjWgkmyUsZaS+a4SPwNflTmD
↪PRyq8/jUJIZd+0lTdDCLjMUmYKKu3Mrf9rO14KipyKZtk/
↪1nkuH0J5qYtDPhvCNaZQIRKZZIOLuSmFSlO1IrQDdkE7SeG5MjNY48Mg1MYUPBr4gOOTymEyU6vJIDEi6OspvlGJfTM+pVbaSbS
↪l+wi4cT2hX2ye3OvrOT2Q+zfWBDY3Ee2x4G3RejE9nCnZnOZ9vkgd2AbcNvcwO2SUxQlcb+Vz5yCQ4Bf5h6m/
↪swdZGJW8guHgeX93H+Kjjc/9OkSpmivduub0pYU1tyxVNuflk24i5daw1e99oSZKvomZrvm29r/
↪xmdyLc2LmXoRzZqZt+QNfviOkiv9P3uKUW8Kj1oNnBnEjWZkT3JTgrHJuHdTCnTKuWrnAMybGwfLFSdqfJ/
↪yjv2PNI6tt9qZSdfnHxK0p/WTpR83CxwO7K15i+u4kg8Nv309EjtCXVg/
↪3q1BqGugGNzUzZo1Tscm467ijmv41foWjOdzCzgD3lTzuAfZ99aW+bAVIxsFXGBLlKEbLPVkFaW2juNUYqc4ms/
↪spF2V1A/P9k7oBCsrkI2D1wEzIwneh3c0i+9ko6tBBaquwnYUNBpRqIungz7sx4r2ehtuPqO8d/
↪WYCybKbV12ohHdr/
↪wVTFgP41sbeLw4h627F7psNFW4di9d4DbZkuLfRHyDV+xzwZuC8GRbVaKimeEn5fZOj8H2g9KA/
↪n2prXusVUoY8YALPsvr6QlQvbnoFZofCx/
↪CcoOjk29GGHZD1606xZtULuHxrBtPHzHIZt7sS1kI9+Q3d5h3veYzsaGH4Hrd+zCc0chW5ryczOxk0Fy9uljbcNK5gfbL3jCRlN
↪/YPrLZuJcde/wExfS8Mtfb+OjlWPyYyaYipbpSy6CulHlA/cZ6cD0f7EgT09T/
↪0KhqcSAfREVRhLWJcVZjz1m55lKGcWff2bNStWyTktlvZmREwIT2iEIAAAAASUVORK5CYII=">
        </div>
    </div>
    <!-- Mode normal -->
    <div ng-if="!compare_mode" class="container" style="top: 5cm; left: 1.5cm;">
        <label class="legend_headline">Légende</label>
        <div id="map_legend"></div>
    </div>
    <!-- Mode comparaison -->
    <div ng-if="compare_mode" class="container" style="top: 5cm; left: 1.5cm;">
        <label class="legend_headline">Légende</label>
        <div id="map_legend"></div>
        <div id="map_legend_compare"></div>
```

(suite sur la page suivante)

(suite de la page précédente)

```
    </div>
    <!-- Mode normal -->
    <div ng-if="!compare_mode" class="container" style="top: 4cm; right: 1.1cm;">
        <img id="map_image" class="map_image" src="images/transparent.png">
    </div>
    <!-- Mode comparaison -->
    <div ng-if="compare_mode">
        <div class="container" style="top: 4cm; right: 1.1cm;">
            <img id="map_image" class="map_image_compare_mode" src="images/
↪transparent.png">
        </div>
        <div class="container" style="top: 15cm; right: 1.1cm;">
            <img id="map_image_compare" class="map_image_compare_mode" src="images/
↪transparent.png">
        </div>
    </div>

    <div class="container" style="top: 24.5cm; left: 1.7cm; z-index: 9;">
        <!-- Pour afficher l'overview utiliser id="map_overview"-->
        <img id="map_overview">
    </div>


    <div id="footer_container" class="container" style="top: 26.7cm; left: 1.5cm">
        <div class="container" style="top: 0cm; right: 0cm;">
            <div class="footer_text">
                <label class="color_blue">Source(s):</label>

                <label class="color_blue">{{sources}}</label>
            </div>
        </div>
        <div class="container" style="top: 0cm; left: 5.5cm;">
            <!-- Pour afficher l'échelle actuelle utiliser {{map_scale}}-->
            <div class="footer_text">
                <label class="color_blue"></label>

                <label class="color_blue">Echelle: {{map_scale}}</label>
            </div>
        </div>
    </div>
</div>
```


vMap
Légende
Local Lampes
Lampe Eteinte
Lampe (Non pré
Lampe 100w et
Lampe 500w et
Lampe 2000w et
Local Route
Route
Cadastre
Piscine
Etang, lac
Tunnel
Limite ne forma
Parapet de pont
Cimetière
Cimetière israéli
Cimetière musul
Voie ferrée
Commune
Section
Lieu-dit
Bâtiment dur
Bâtiment léger
Subdivision fisca
Contour de parc
Parcelle
Cap d'Ona
Toto
25 plot
passe
Lampe n°4
888
Lampe n°666
Lampe n°6
Lampe à Laurence
le mour n°1
Lampe n°10
Lampe n°3
Lampe n°2
Lampe n°9
test1
Lampe fonctionnelle (5000w)
Quai Lacaze Duthiers
Allée Aristide Maillol
Avenue du Fontaule
Banyuls-sur-Mer
Rue Henry Perra
Rue Richelieu
Rue Saint-Sébastien
Rue Jean Bart
D 914
Anse du Fontaulé
Banyuls-sur-Mer
Carrefour express
Avenue du Fontaule
Quai Georges Petit
Lampadaire 328
25 plot passe
Lampe n°9
public token
Echelle: 1:6,658
Source(s):

styles

impression

Variables disponibles

Tous les paramètres qui vous affecterez depuis le mode Paramètres seront interprétés comme des variables et pourront être incluses dans le résultat en les mettant entre crochets dans le définition.

Au même titre que les paramètres définis, certains paramètres dynamiques sont automatiquement affectés :

- **map_scale** : échelle de la carte imprimée
- **date** : date du jour
- **layer_sources** : sources des couches
- **user_name** : nom de l"utilisateur
- **user_login** : login de l"utilisateur
- **user_company** : société de l"utilisateur
- **user_department** : service de l"utilisateur
- **user_email** : adresse mail de l"utilisateur
- **user_user_id** : identifiant de l"utilisateur

Il est également possible de créer des variable à partir de code javascript.

Exemple : ajout de la date du jour de génération de l"impression

```
<script>
n =  new Date();
y = n.getFullYear();
m = n.getMonth() + 1;
d = n.getDate();
document.getElementById("date").innerHTML = d + "/" + m + "/" + y;
</script>
```

Insertion du code html permettant de faire remonter la variable

```
<p style="text-align: left;">Le <a id="date"></a></p>
```

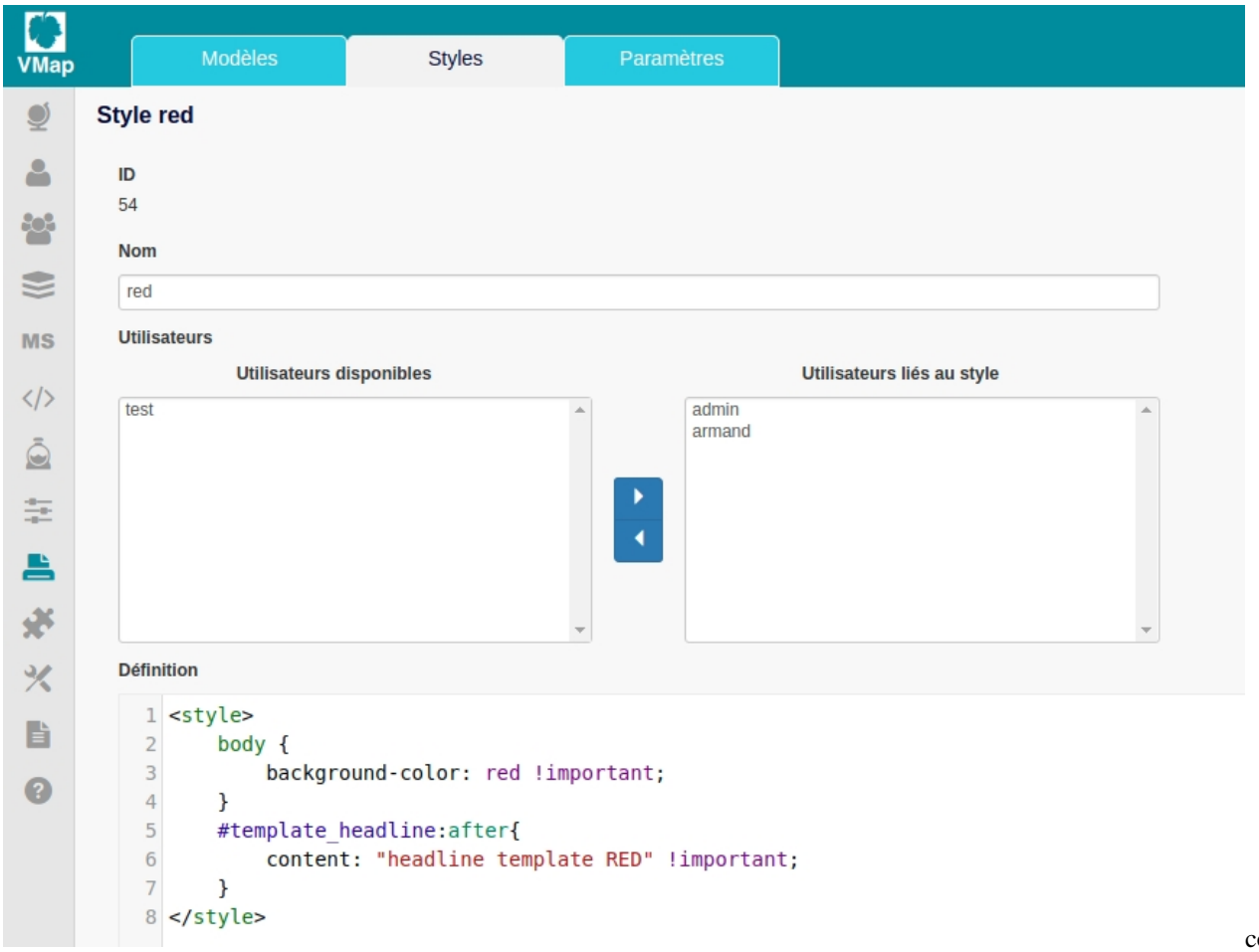
4.5.2 Objet Styles

En renseignant des styles d"impressions, alors l"utilisateur pourra choisir le style qu"il souhaite utiliser lors de la phase de préparation.

styles impression

Pour être utilisable chaque style devra être lié à un ou plusieurs utilisateurs, comme pour les modèles il faudra dans la partie définition écrire ne HTML un style CSS.

Ce style va venir surcharger celui du modèle et comme pour ce dernier, pour que les couleurs s"affichent correctement il faudra utiliser la syntaxe « !important ».



configuration styles impression

4.5.3 Objet Paramètres

Ce dernier onglet va permettre à l'"administrateur de définir les paramètres à saisir lors de la phase de préparation à l'"impression, chaque paramêtre est lié à un modèle d'"impression et est facultatif pour l'"utilisateur.

Si un paramètre est non modifiable alors il sera caché dans le formulaire mais le résultat sera inclus dans l'"impression.

configuration styles impression

4.6 Mode développement

Le mode développement permet de créer des objets métiers pour rendre les calques interrogeables

4.6.1 Onglet objets métiers

1. Définition

Un objet métier est une entité qui associe à un calque, les attributs d'une table ou vue de base de données. De la sorte, les attributs associés au calque sont affichables et éditables, dans le requêteur et dans le formulaire de création d'objet, accessibles dans le mode Carte.

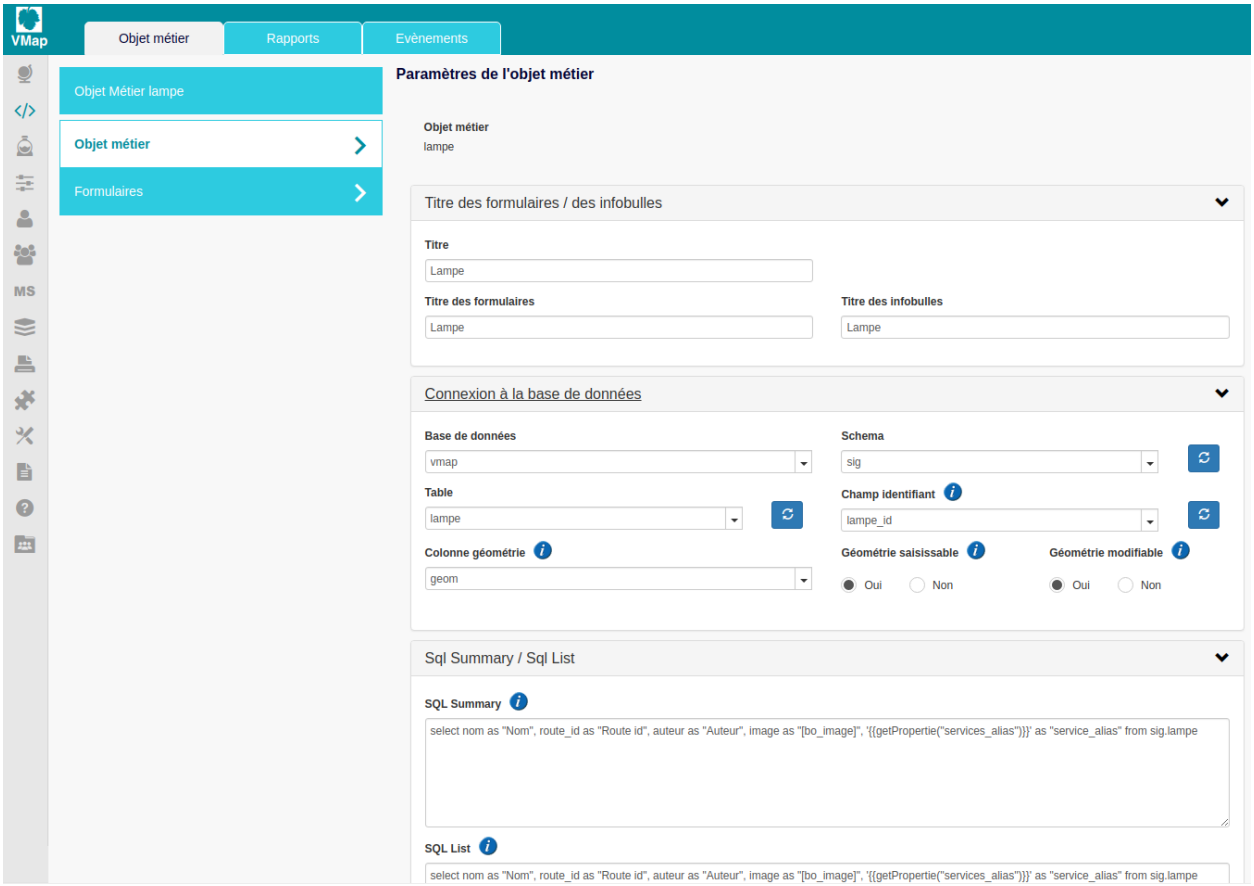
Un objet métier permet donc de gérer des données vectorielles stockées en base.

Le mode Développement permet l'ajout, l'édition et la suppression d'objets métier.

La création d'un objet métier s'opère en deux temps :

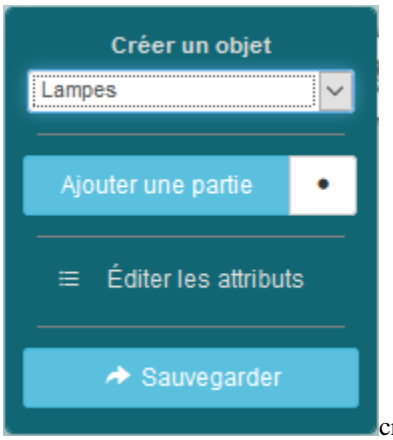
— La déclaration de l'objet et des paramètres d'affichage du requêteur.
— La construction des formulaires d'affichage, de création, d'édition et de recherche de l'objet métier via le studio.

2. Création d'un objet métier



Renseigner les champs suivants :

— **Titre :** nom de l'objet métier tel qu'il apparaîtra dans le requêteur et dans le formulaire de création d'objet

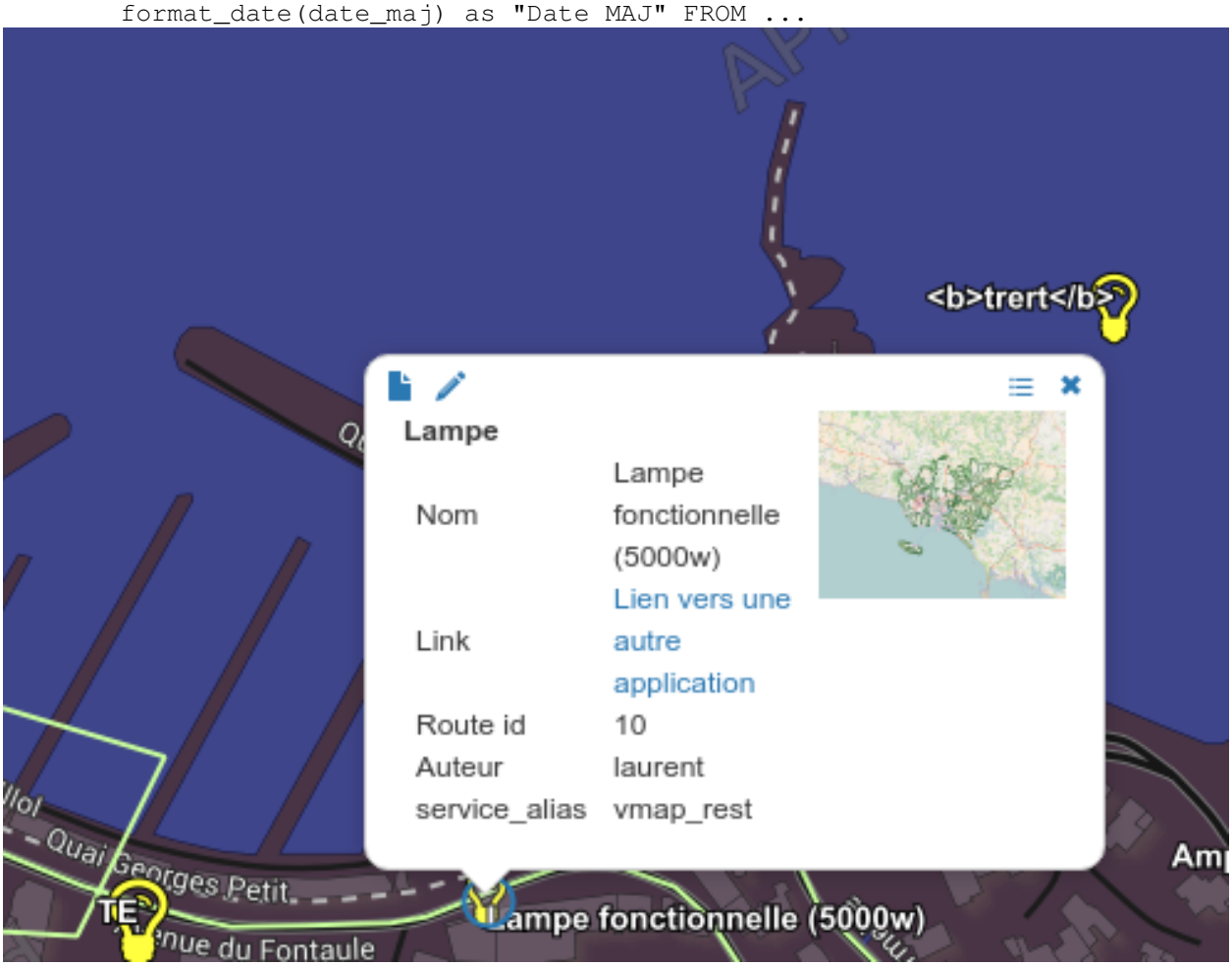


création d'objet

— **Titre des formulaires :** le titre qui va apparaître dans les formulaires, vous pouvez y intégrer le résultat d"un des attributs du formulaire en le mettant entre accolades (ex : Commune n°{{id_com}})

— **Titre des infobulles :** le titre qui va apparaître dans les infobulles, vous pouvez y intégrer le résultat d"un des attributs de la requête SQL Summary en le mettant entre accolades (ex : Commune n°{{ID commune}})

- **Champs id :** champ identifiant de la table.
- **Base de données :** nom de la base de données à laquelle se connecter
- **Schéma :** schéma de la base de données
- **Table :** table de la base de données
- **SQL Summary :** requête SQL pour définir les champs à afficher dans l'infobulle d'un objet.
  - **Images et liens** : Il est possible depuis la version 2018.01.00 d"intégrer des liens et des images en utilisant les balises HTML `<a>` et `<img>`
  - **Dates** : Il est possible depuis la version 2018.03.00 de formater les dates en fonction de la configuration de l"utilisateur en utilisant la fonction SQL `s_vitis.format_date()` exemple : `SELECT s_vitis.format_date(date_maj) as "Date MAJ" FROM ...`



- **SQL List :** requête SQL pour définir les champs à afficher dans la liste des objets sélectionnés du requêteur.
  - **Images et liens** : Il est possible depuis la version 2018.01.00 d"intégrer des liens et des images en utilisant les balises HTML `<a>` et `<img>`
  - **Dates** : Il est possible depuis la version 2018.03.00 de formater les dates en fonction de la configuration de l"utilisateur en utilisant la fonction SQL `s_vitis.format_date()` exemple : `SELECT s_vitis.format_date(date_maj) as "Date MAJ" FROM ...`

		Lampe id	Nom	Route id	Puissance	Portée	Allumée	Auteur	Date création	Date MAJ
☐		106	l1	96	100	10	true	margot	2017-05-26 00:00:00	2017-05-26 12:18:30.400335
☐		101	ma lampe	61	200	10	true	margot	2017-05-16 00:00:00	2017-05-16 10:53:10.308931

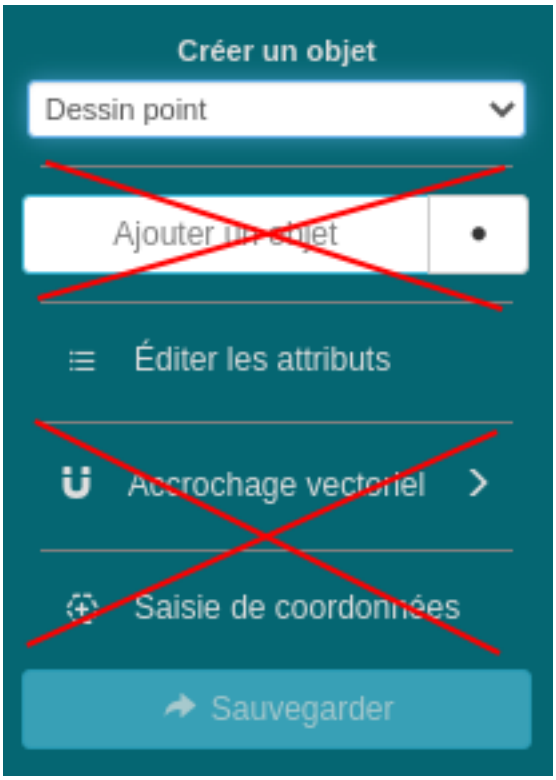
2.1 Description de certains champs

Géométrie saisissable, modifiable

— Géométrie saisissable : Option disponible si le nom de la cononne géométrique est définie.
  — Oui : Saisie d “une nouvelle géométrie autorisée.
  — Non : Impossibilité de saisir une nouvelle géométrie.

Ce paramétrage peut être utile si la table comporte un champ géométrique mais que la géométrie de l”objet est calculée dans un second temps (dans une vue par exemple).

Au moment de d”ajouter un objet  , si l”option vaut non les éléments barrés n”apparaissent pas.

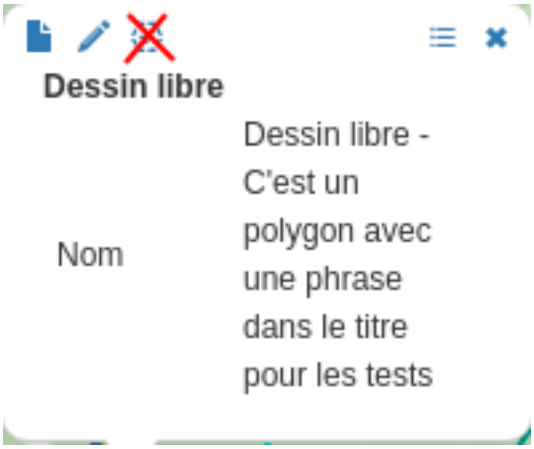


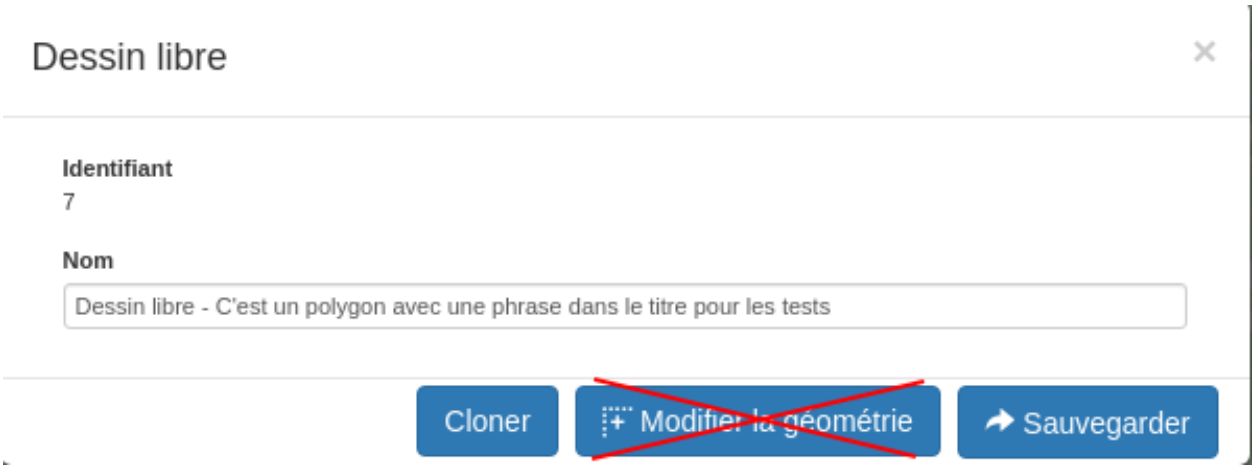
— Géométrie modifiable : Option disponible si le nom de la cononne géométrique est définie.
  — Oui : Modification de la géométrie autorisée.

— Non : Impossibilité de modifier une géométrie.

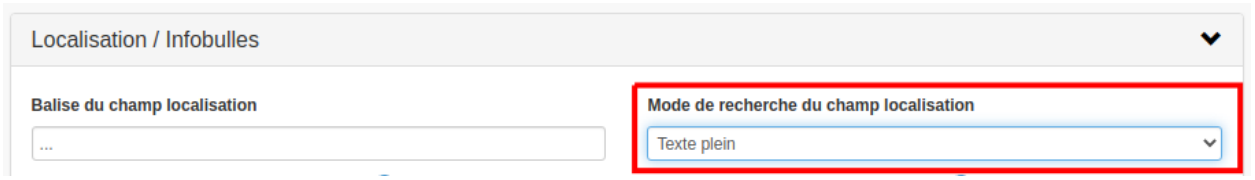
Ce paramétrage peut être utile si la table comporte un champ géométrique mais que la géométrie de l'objet est modifiée dans un second temps (dans une vue par exemple).

Si l'option vaut non :





Mode de recherche du champ localisation



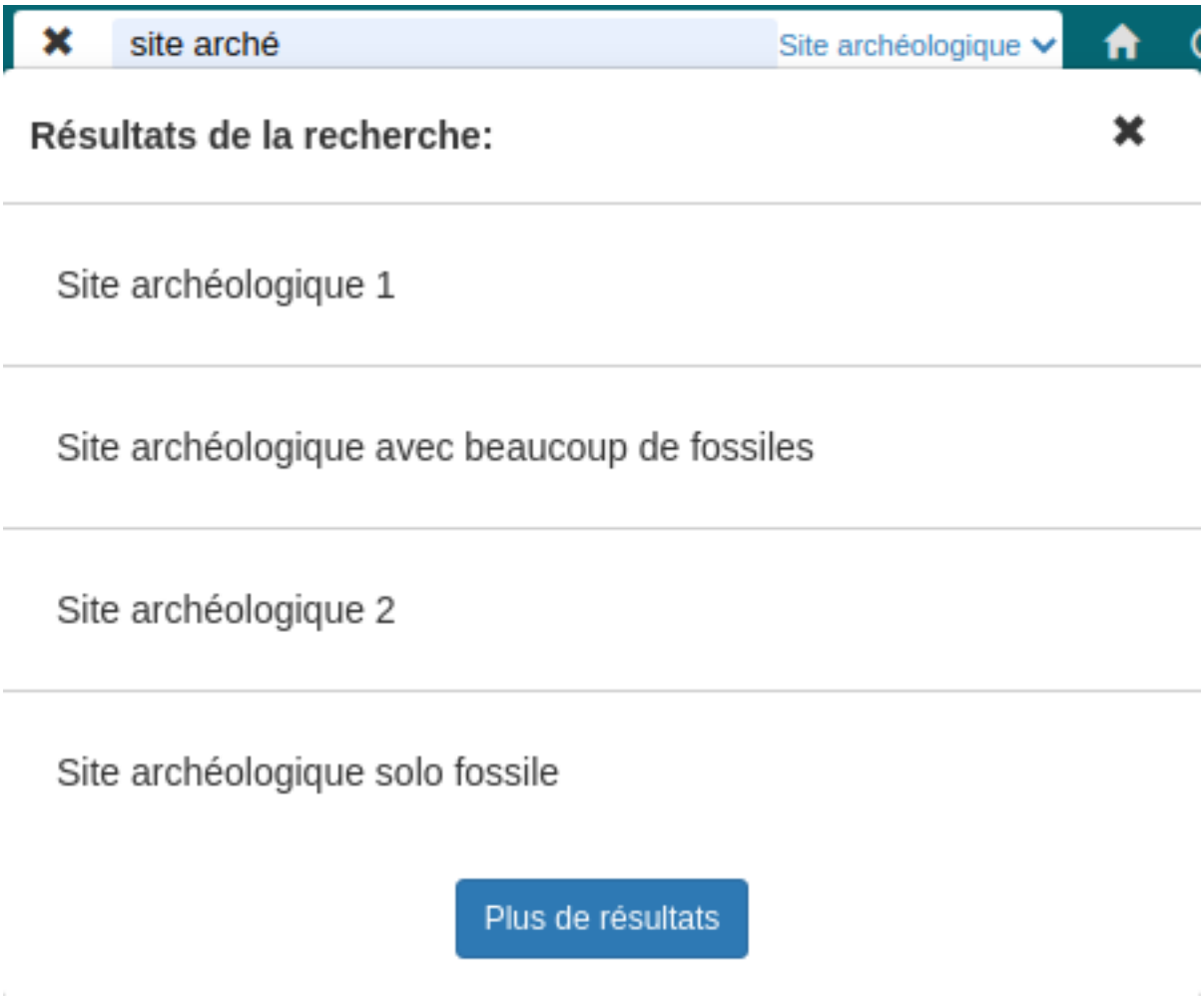
Les modes de recherche pour le champs localisation sont :

- Stricte
  - Cherche exactement le texte
- Début de chaine
  - Cherche le texte saisie (dans un seul bloc) en début de chaîne
  - En sql, cela correspond à WHERE [le_champ] LIKE « texte_recherché% »
- Fin de chaine
  - Cherche le texte saisie (dans un seul bloc) comme une fin de chaîne
  - En sql, cela correspond à WHERE [le_champ] LIKE « %texte_recherché »
- Milieu de chaine

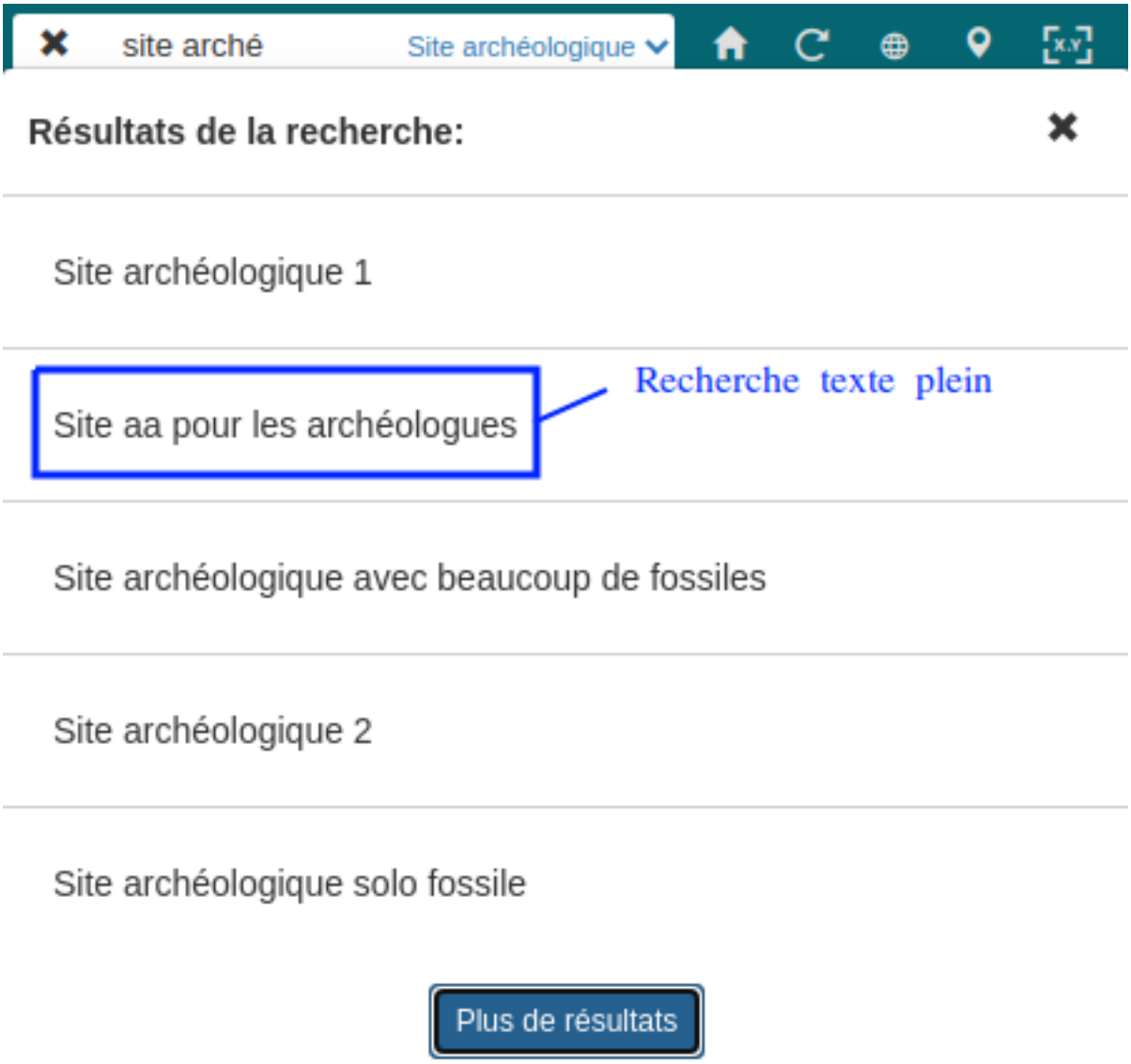
    - Cherche le texte saisie (dans un seul bloc) dans tout la chaîne
    - En sql, cela correspond à LIKE « %texte_recherché% »
- Texte plein
    - Recherche un ou plusieurs mots de manière indépendante dans le texte.
    - Corresponds à la recherche « plain text » de postgresql
    - Le résultat d'une recherche « texte plein » sera prétraité, trié et retourné à l'utilisateur en fonction d'un « score ». Les meilleurs résultats seront ceux dont le score sera le plus important.
      Les avantages de la recherche texte plein sont :
        - Une recherche plus large car chaque mot est cherché individuellement.
        - Un tri plus pertinant grâce à un score attribué à chaque mot

Exemple :

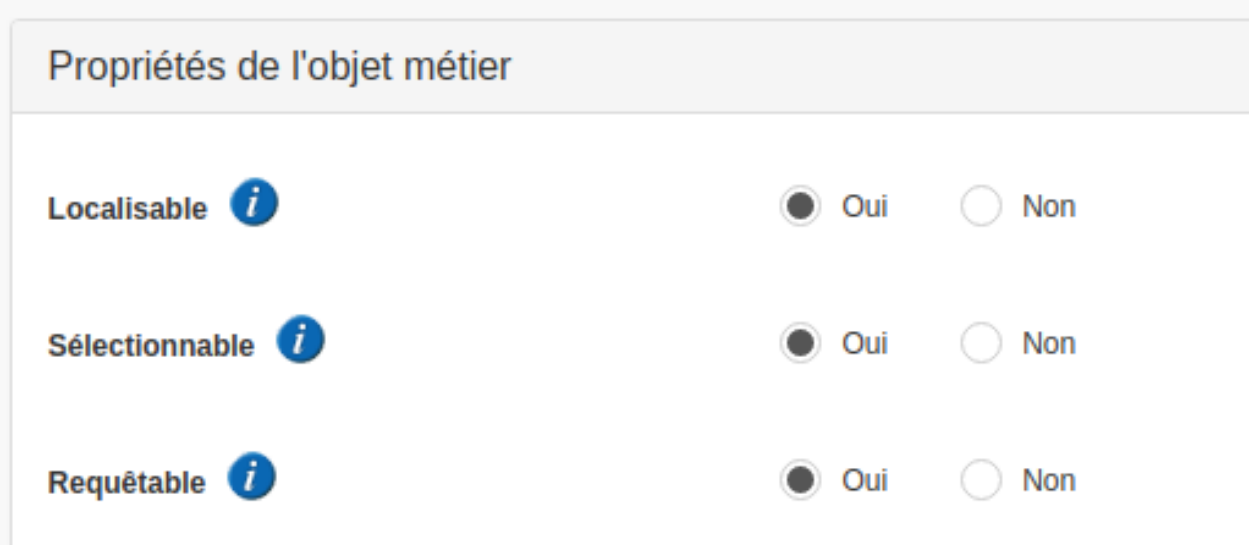
Recherche en Debut de chaîne :

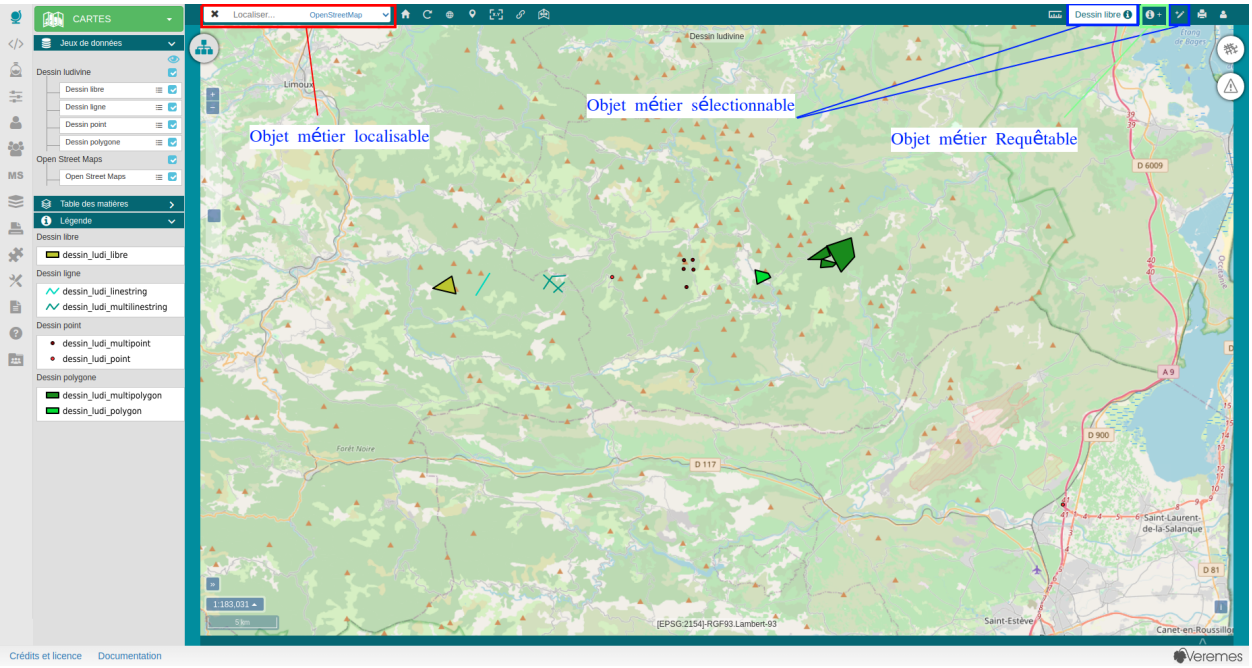


Recherche en Texte plein : La recheche texte plein cherche les mots individuellement.

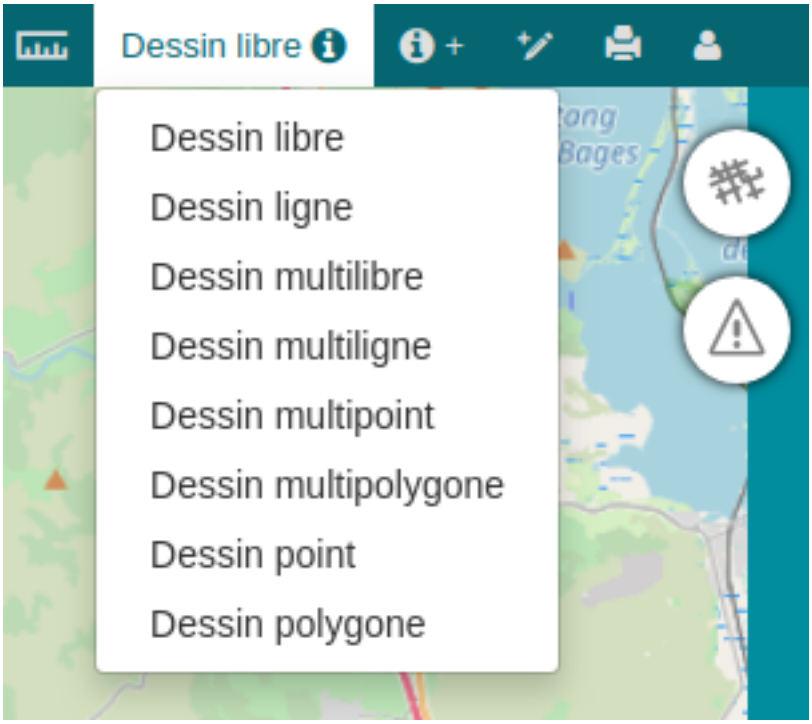

site arché
Site archéologique
Résultats de la recherche:
Site archéologique 1
Site aa pour les archéologues
Recherche texte plein
Site archéologique avec beaucoup de fossiles
Site archéologique 2
Site archéologique solo fossile
Plus de résultats

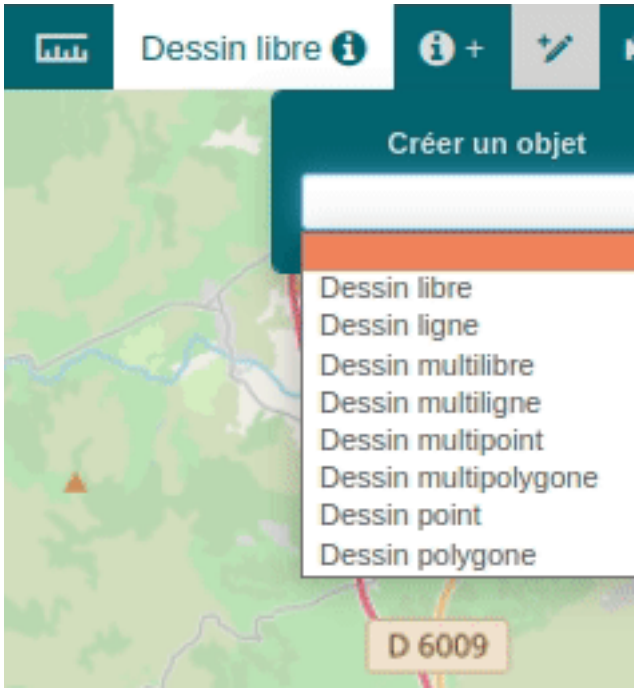
Objet localisable, sélectionnable et requêtable



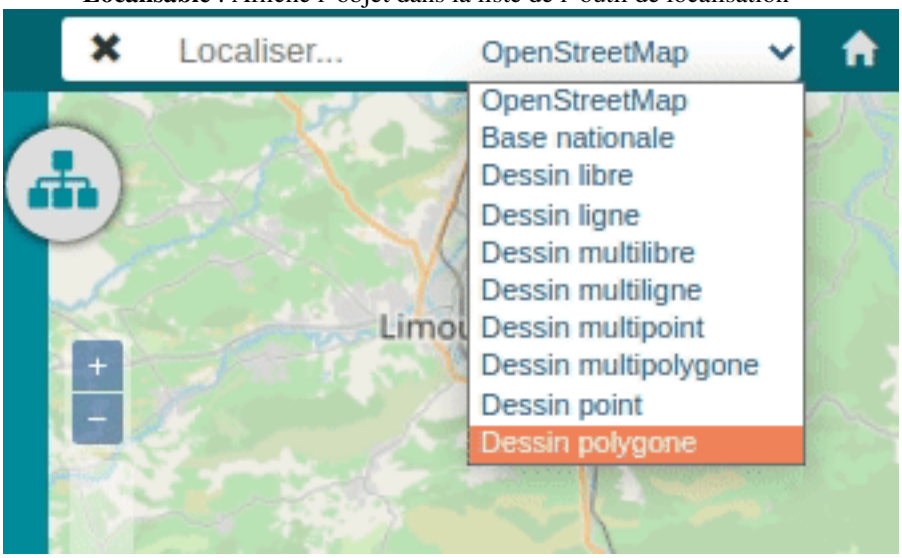


— **Sélectionnable** : Affiche l'"objet dans les listes des outils i et insertion

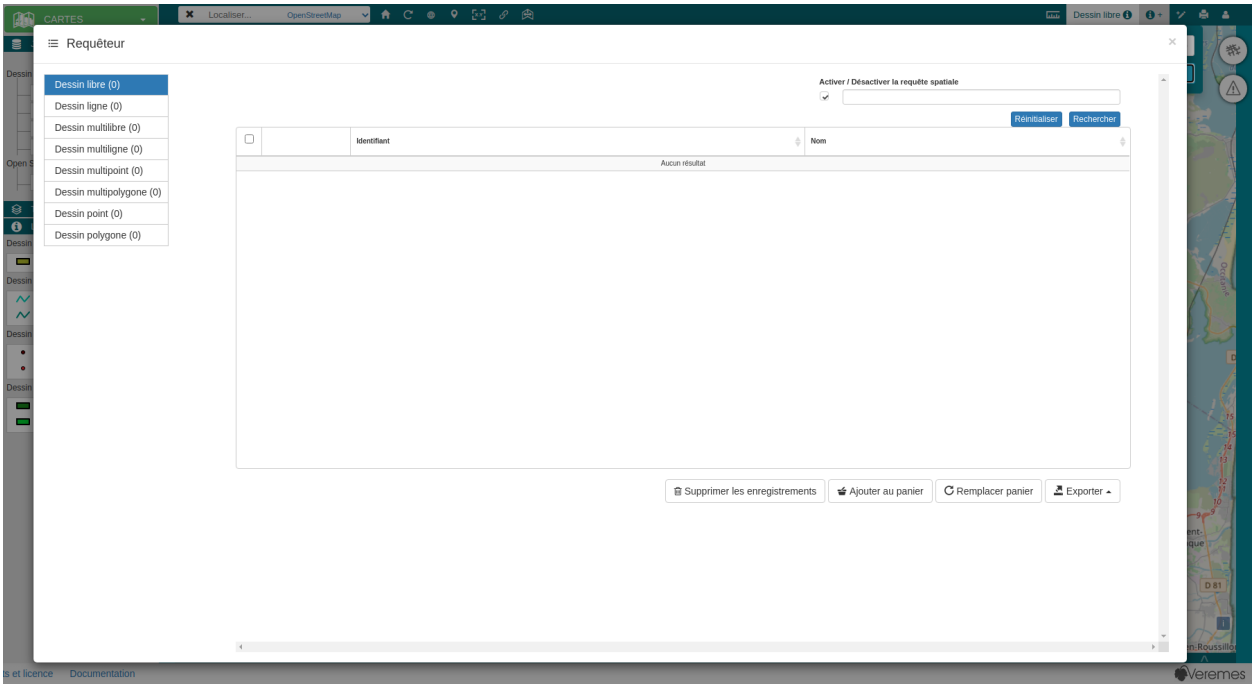




— **Localisable** : Affiche l'objet dans la liste de l'outil de localisation

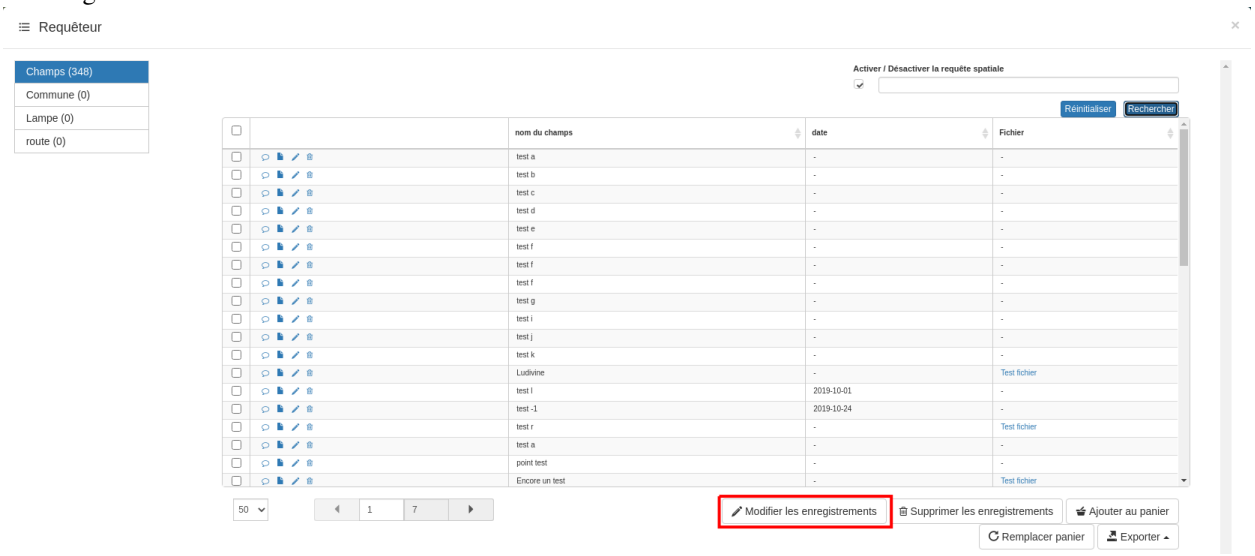


— **Requêtable** : Affiche l'objet dans les listes de l'outil i+



Objet disponible en édition multiple

— **Disponible en édition multiple** : Affiche dans le requêteur un bouton permettant de modifier plusieurs enregistrements



Le bouton « modifier les enregistrements » ouvre le formulaire d'édition multiple.

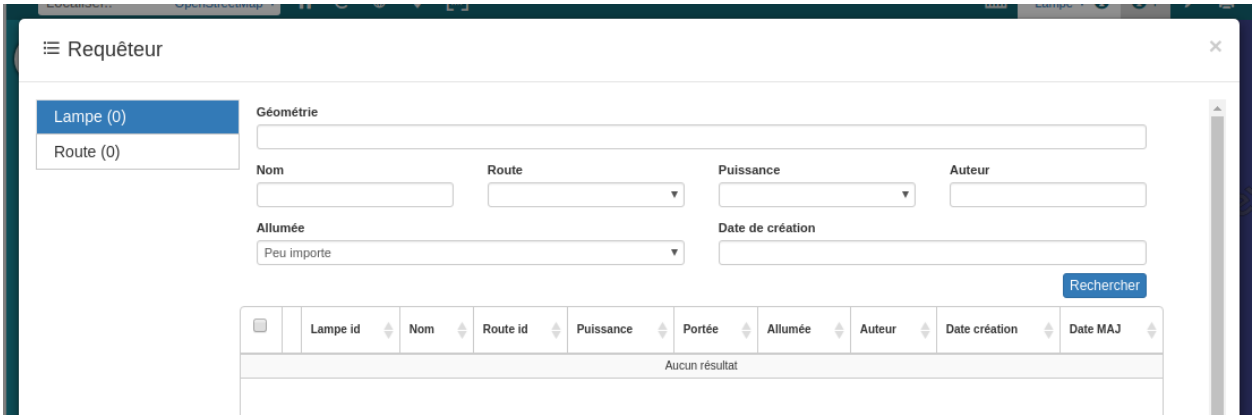
3. Formulaires

3.1. Définitions

Pour chaque objet métier, plusieurs formulaires sont utilisables et paramétrables :

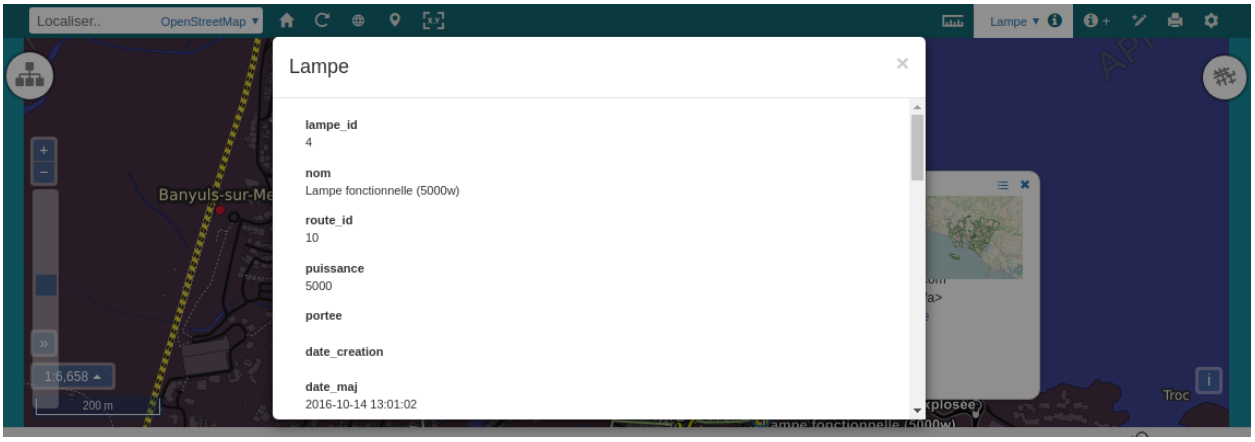
3.1.1. Formulaire de recherche de l'objet métier (search)

Utilisable dans le requêteur et disponible pour les utilisateurs ayant des **droits en consultation** sur la table liée, il permet de faire des recherches filtrées sur les enregistrements de l''objet métier.



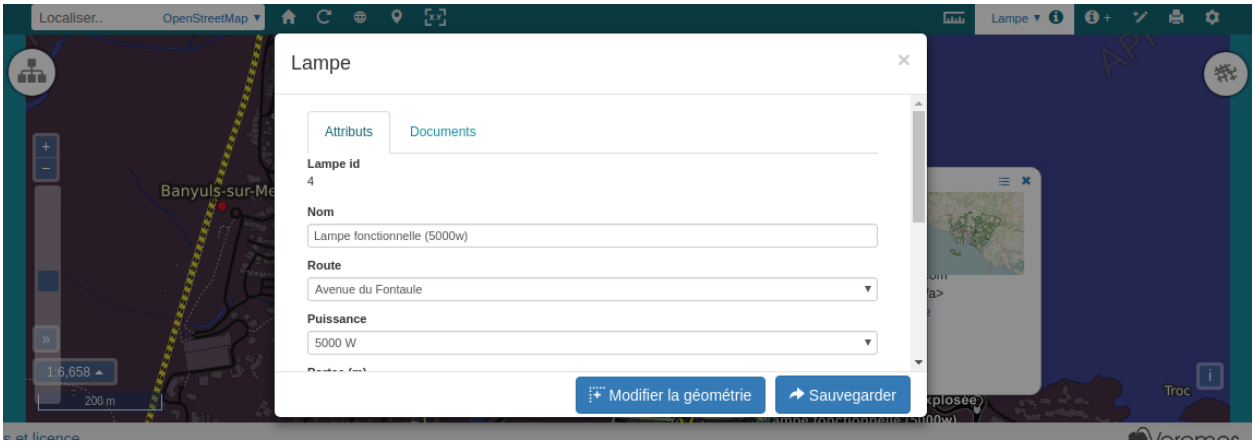
3.1.2. Formulaire d'affichage de l'objet métier (display)

Utilisable par les personnes ayant des **droits en consultation** sur la table liée, il permet d''afficher des informations en consultation pour l''enregistrement sélectionné.



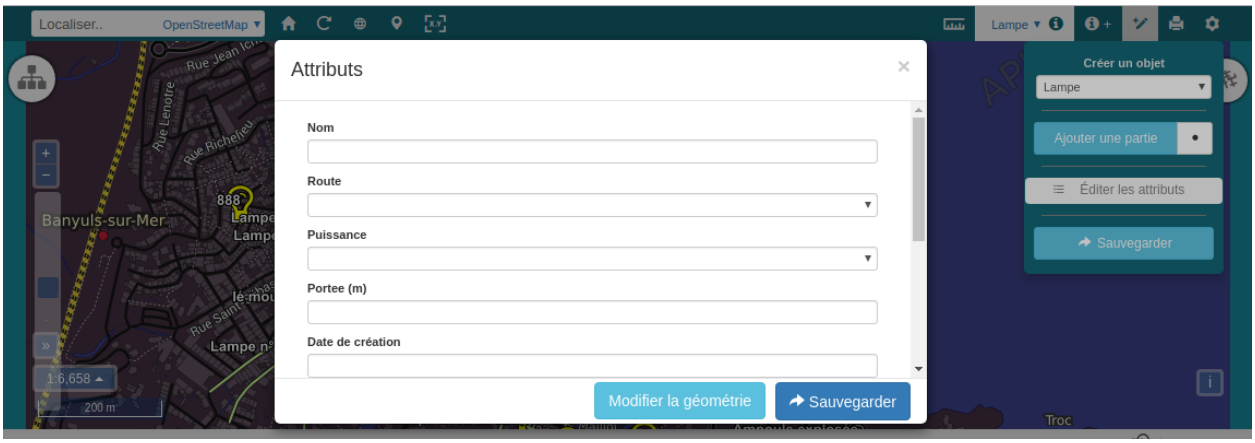
3.1.3. Formulaire de mise à jour de l'objet métier (update)

Utilisable par les personnes ayant des **droits de mise à jour** sur la table liée, il permet de mettre à jour les attributs de l''enregistrement en édition.



3.1.4. Formulaire de création de l'objet métier (insert)

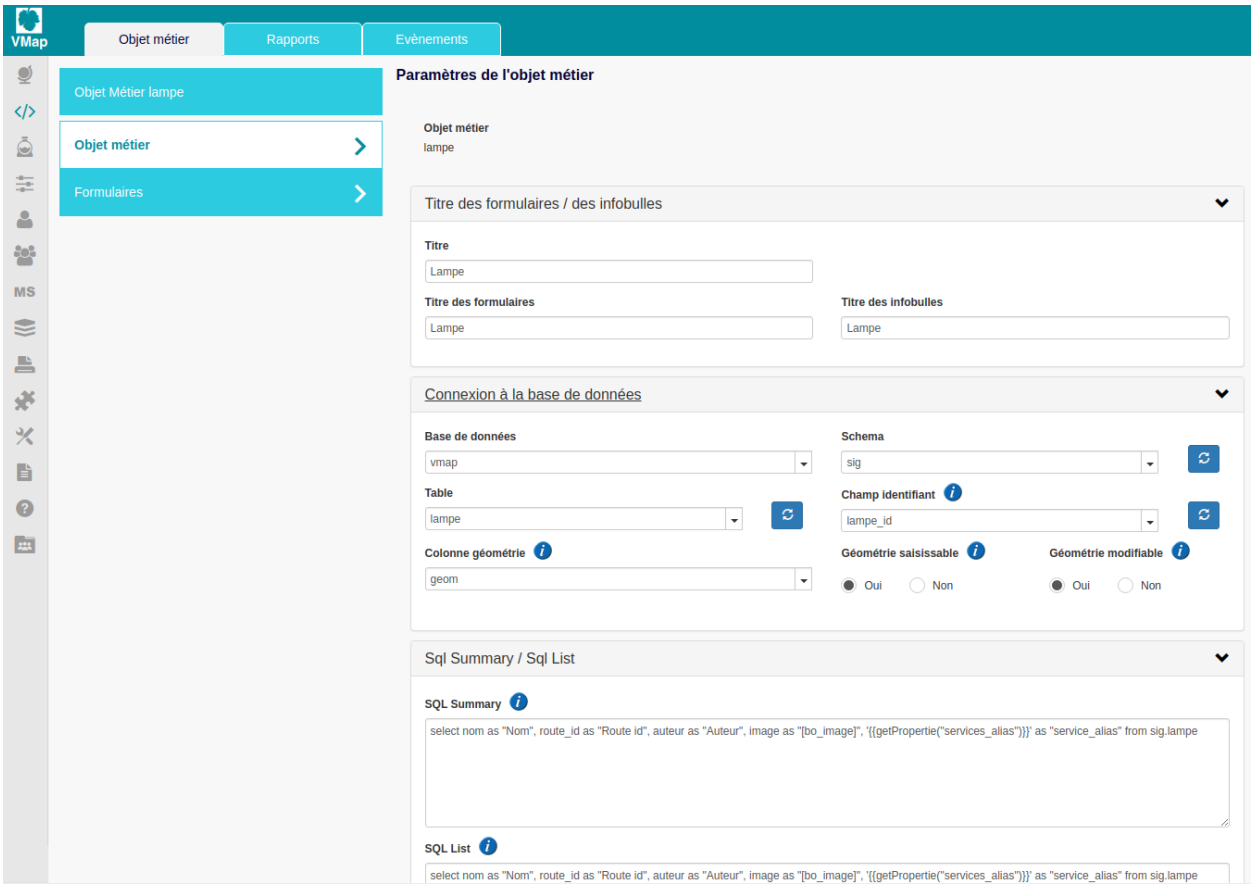
Utilisable par les personnes ayant des **droit en insertion** sur la table liée et accessible par le bouton « **Éditer les attributs** », il permet à l'"utilisateur de créer un objet et de renseigner ses attributs.



3.2 Studio

Un studio a été developpé pour gérer graphiquement les différents formulaires des objets métier.

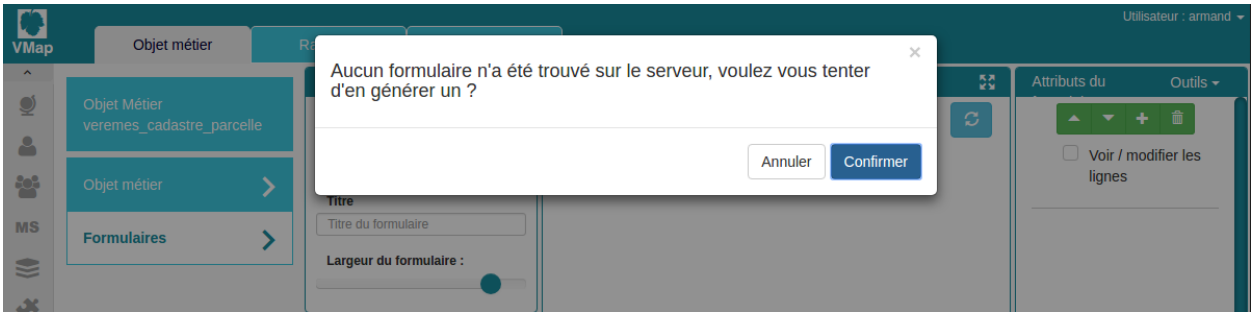
Le studio est accessible via la section Formulaire après avoir édité un objet métier.



3.2.1. Génération automatique des formulaires

La première chose à faire lorsqu"on veut créer un ensemble de formulaires est de demander à l"application de les générer en fonction des colonnes présentes sur la table liée. Si le typage en base de données est bien fait et que cela est possible, le type de champ affiché dans le formulaire sera également implémenté (texte, nombre, date etc. . .).

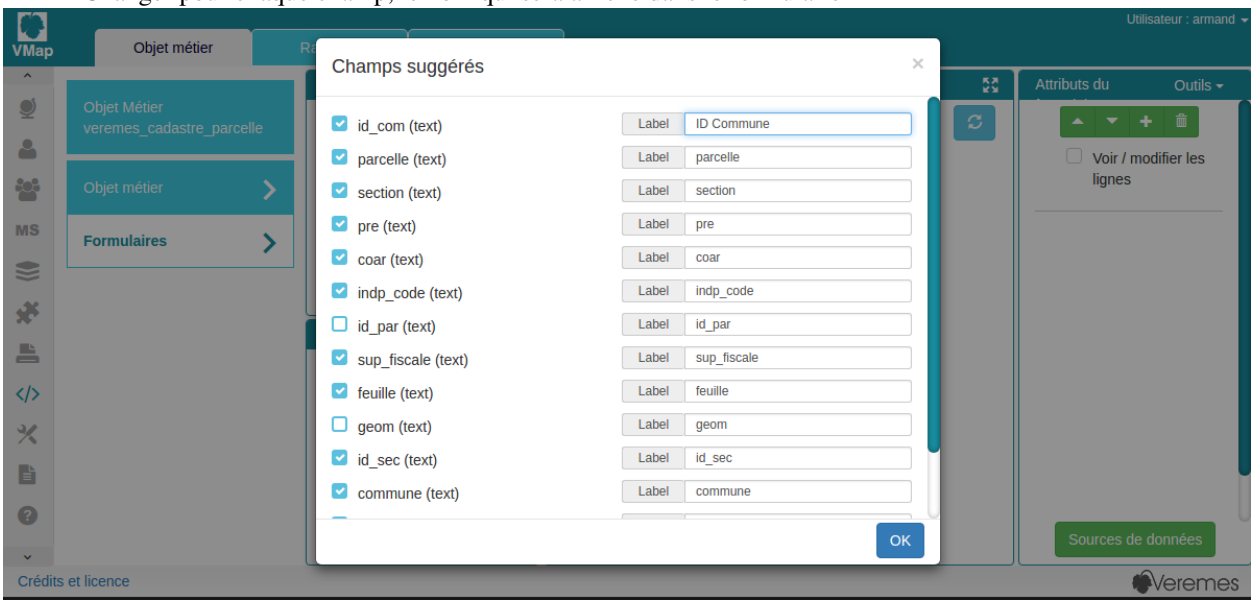
Pour cela, il suffit de cliquer sur **confirmer** lors de l"affichage du message suivant :



On peut également cliquer dans le **formulaire par défaut** sur **Régénérer le formulaire par défaut**.

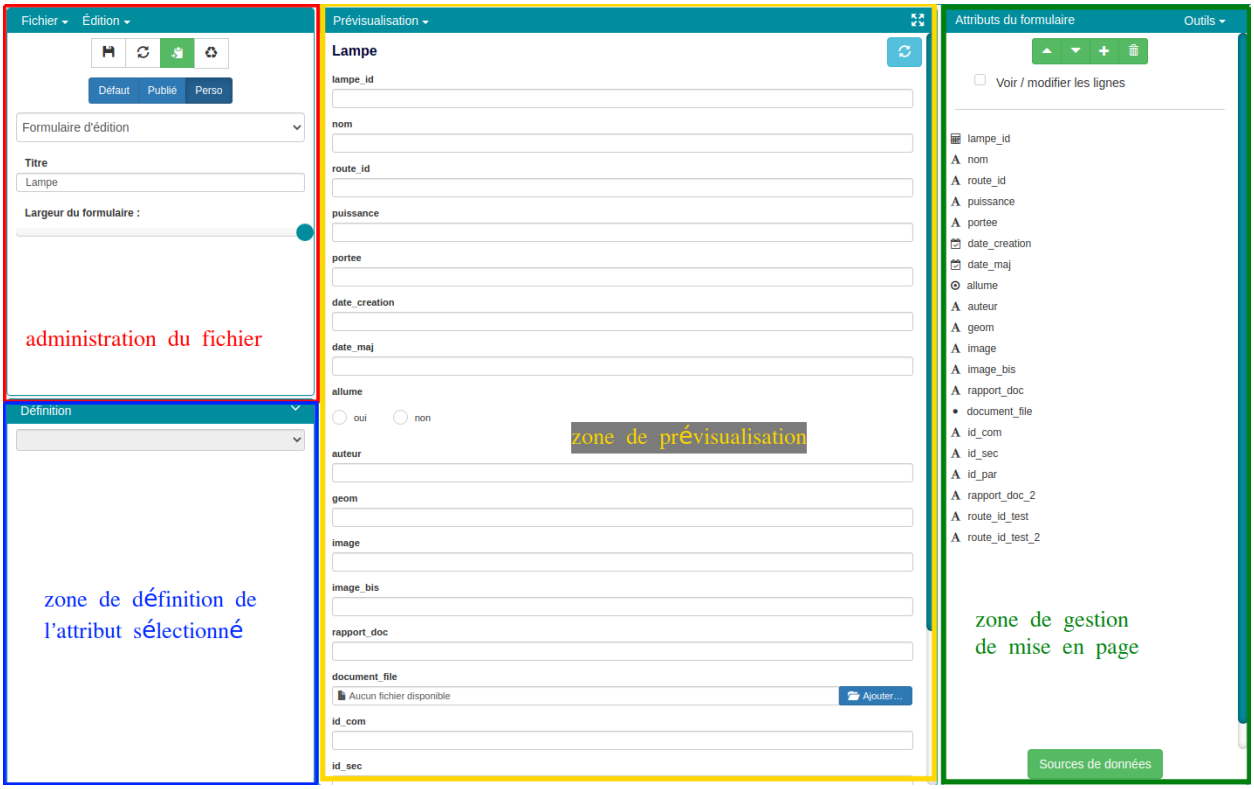
La fenêtre suivante apparaît et l'"utilisateur peut :

— Sélectionner les arguments à afficher
— Changer pour chaque champ, le nom qui sera affiché dans le formulaire



3.2.2. Utilisation du studio

Le studio est divisé en quatre principales zones de gestion des formulaires :

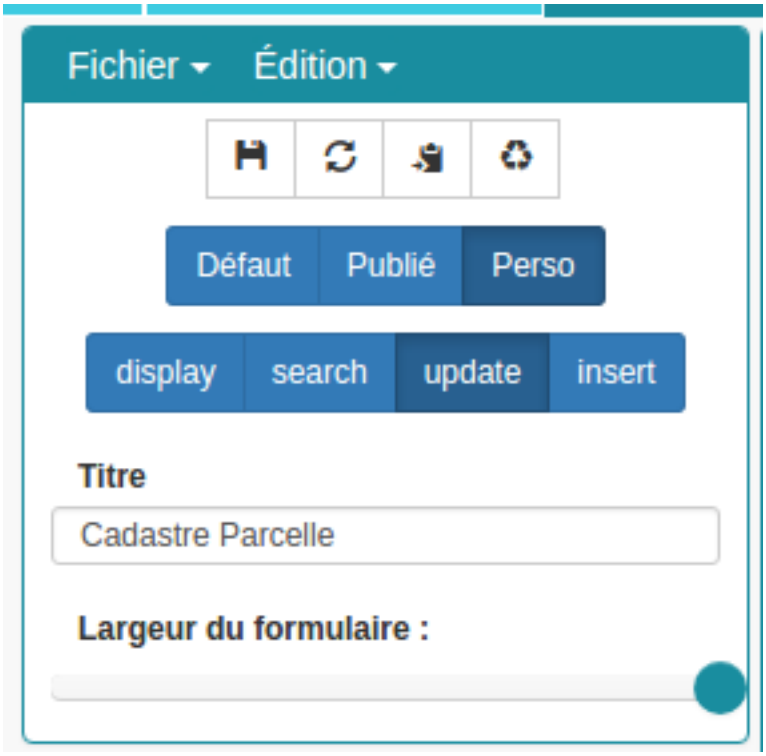


3.2.2.1. La zone d'"administration du fichier

Il s'"agit d'"une des zones fondamentales car elle permet la sauvegarde et l'"affichage des fichiers.

Il existe trois types de formulaires :

- le **formulaire par défaut**, formulaire généré automatiquement. Le développeur peut choisir de conserver en l'"état ce formulaire ou de le personnaliser.
- le **formulaire publié**, formulaire en cours d'"utilisation dans l'"application
- le **formulaire personnalisé**, formulaire en cours d'"édition.



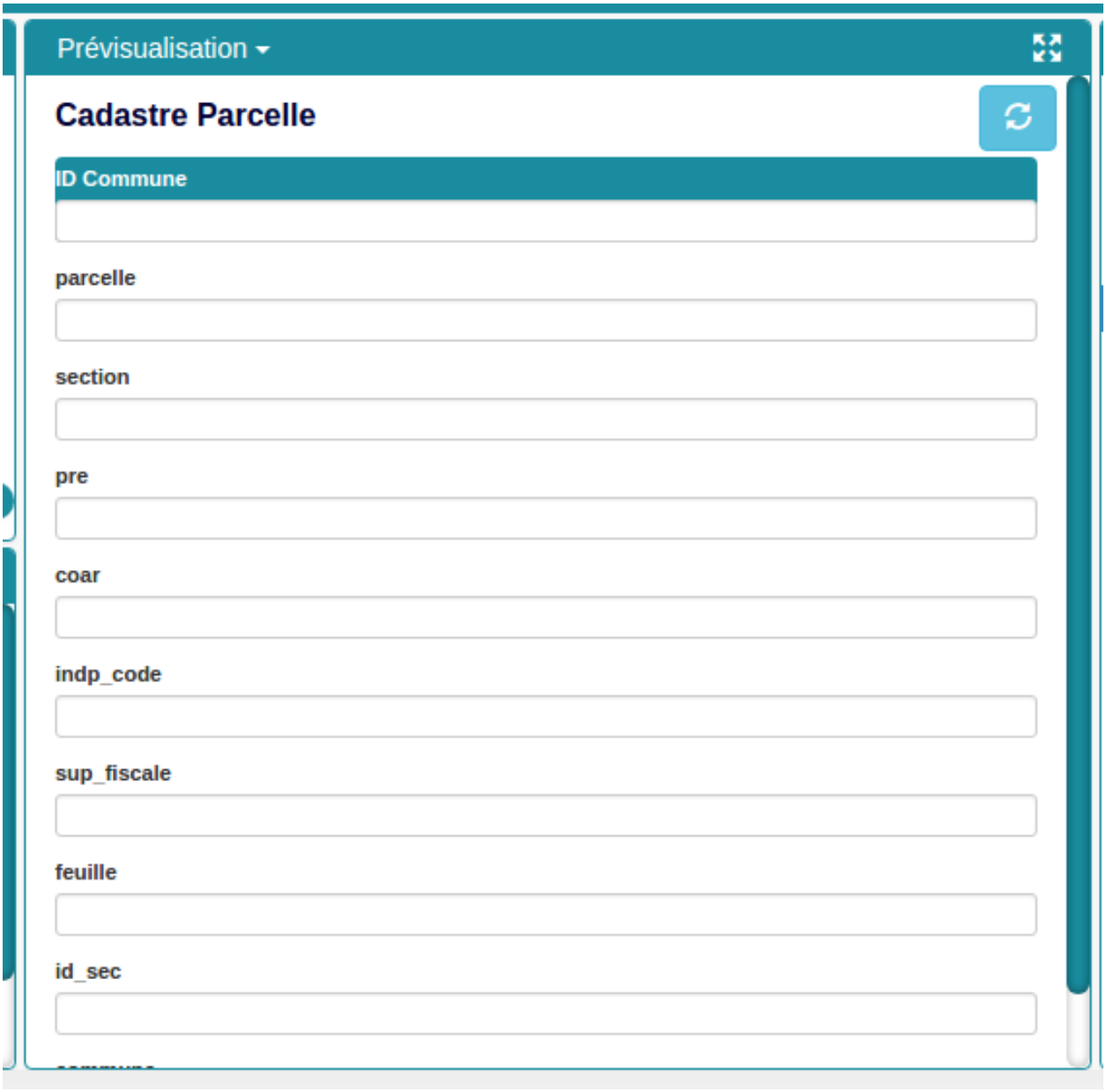
Pour modifier un formulaire, l'"administrateur clique sur le bouton **Perso** puis sélectionne le type de formulaire sur lequel il veut travailler (display, search, update, insert). Il édite ce dernier et **publie le formulaire personnalisé** car sans cela les modifications ne seraient pas visibles par les utilisateurs finaux.

Le menu déroulant **Fichier** permet de gérer les versions des formulaires (publier le formulaire personnalisé, régénérer le formulaire par défaut etc..)

Le menu déroulant **Édition> Gestion des onglets** permet d'"agencer le formulaire en répartissant les attributs sur plusieurs onglets.

3.2.2.2. La zone de prévisualisation

La zone de prévisualisation permet à l'"administrateur de visualiser en direct le formulaire en cours.

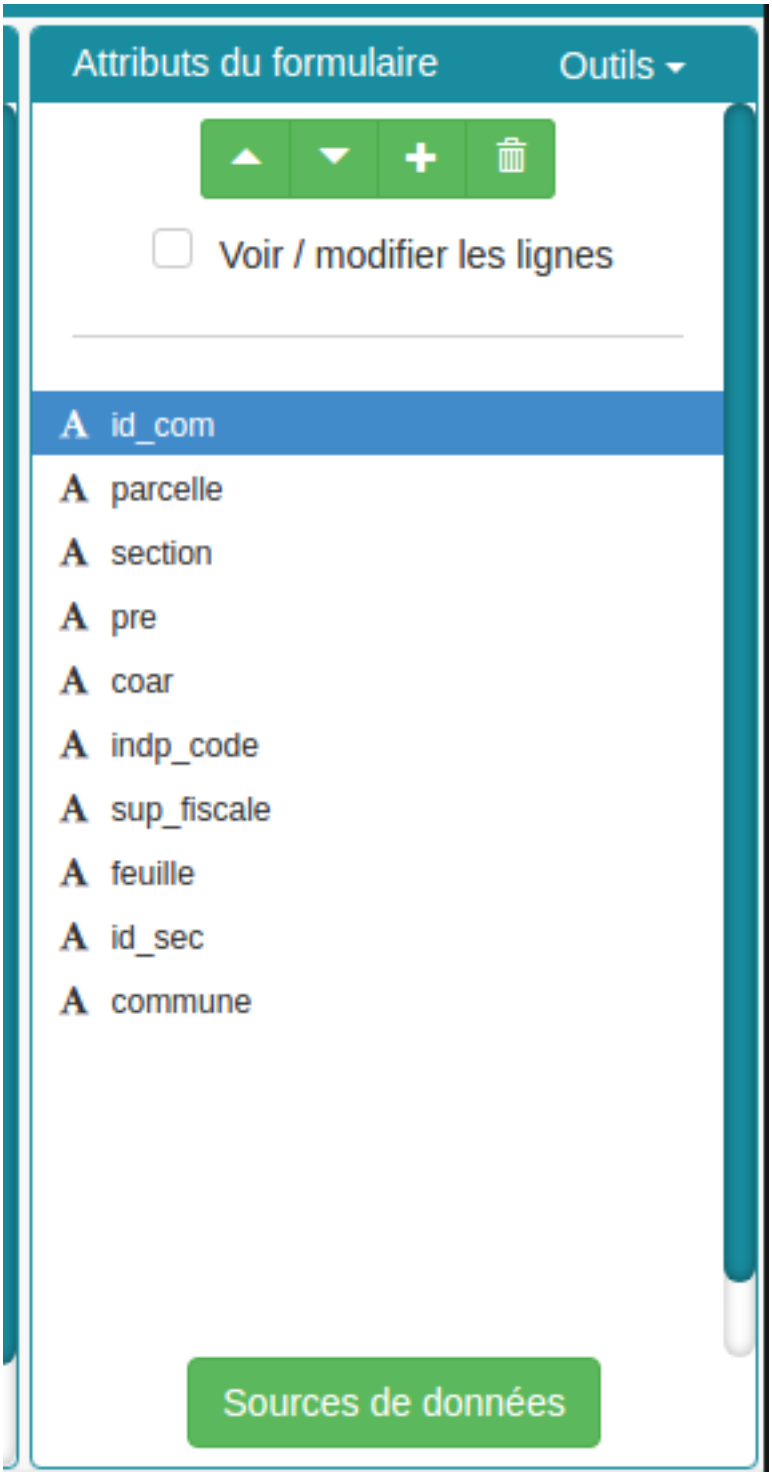


Le menu déroulant **Prévisualisation** permet l'affichage et la modification de la définition du formulaire au format JSON ainsi que l'ajout de JavaScript au formulaire.

> **note**
> Les onglets ne sont volontairement pas affichés dans cette zone. Ils sont affichés lors de l'utilisation réelle du formulaire.

3.2.2.3. La zone de gestion de mise en page

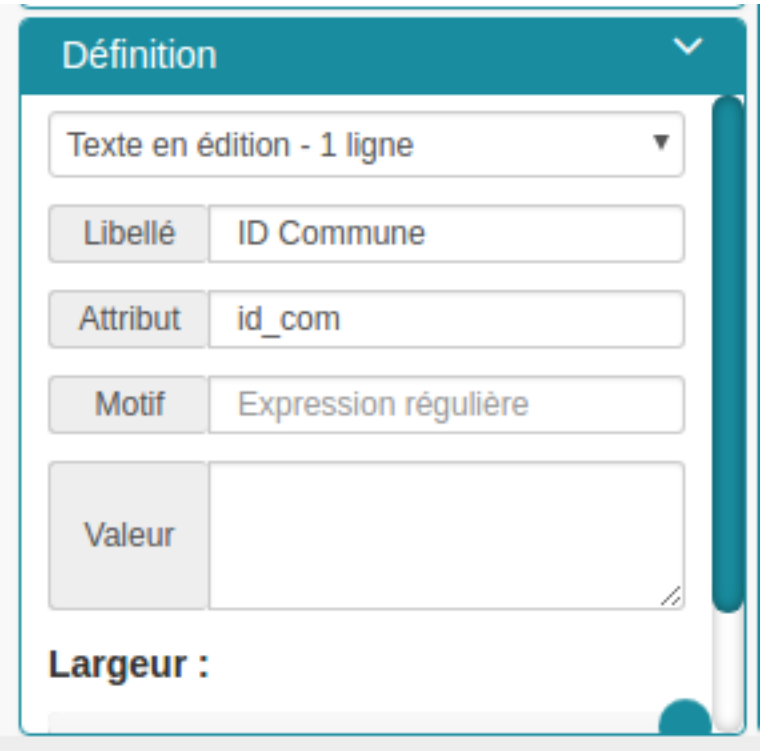
Dans cette zone, l'administrateur peut modifier l'ordre d'affichage des attributs, et via la case « Voir / modifier les lignes », il peut regrouper plusieurs éléments sur une même ligne.



Le bouton **Sources de données** en bas de zone, permet la configuration des attributs de type liste.

3.2.2.4. La zone de définition de l'"attribut sélectionné

Dans cette zone, l'"administrateur pourra gérer le type de saisie qui sera faite, le libellé à afficher sur le formulaire, le nom de la colonne auquel il est lié et bien d'"autres paramètres en fonction du type d'"attribut.

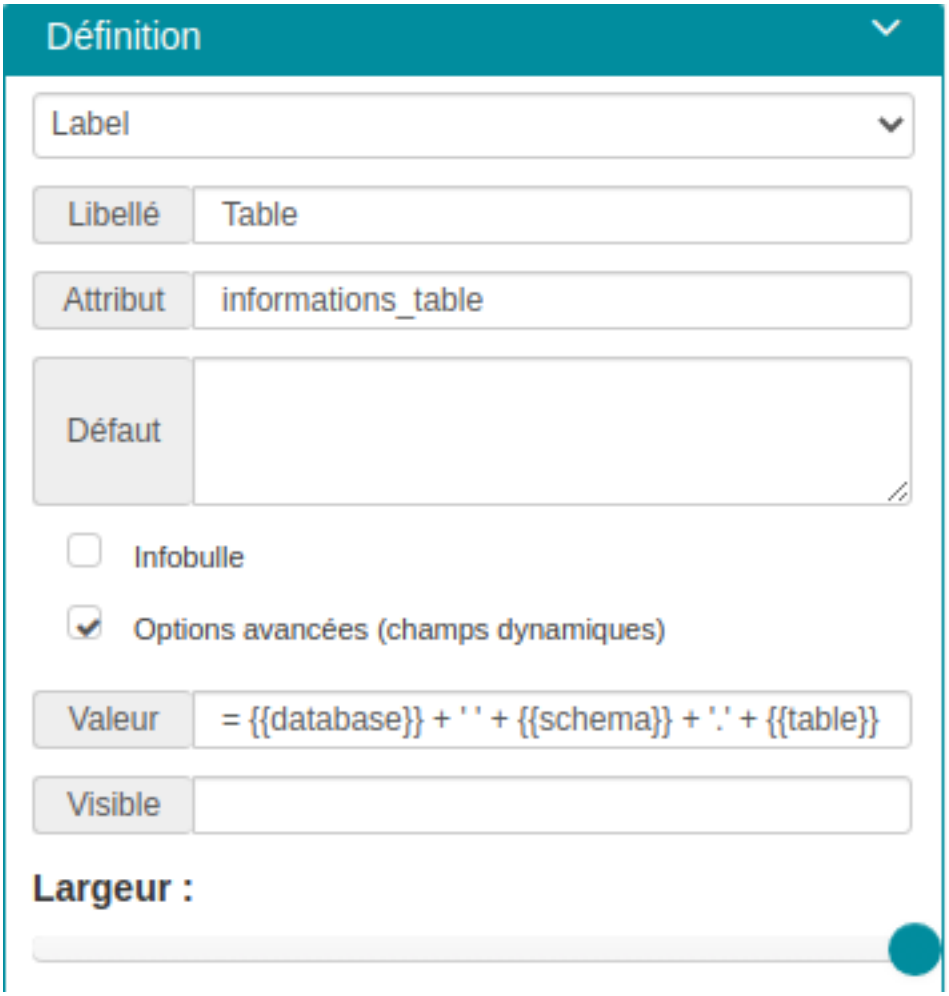


Depuis la version 2020.02 de vMap, il est possible d'"accéder aux informations de connexion de l'"objet métier directement dans les formulaires (consultation, saisi, modification).

3 nouvelles variables sont ainsi disponibles :

- database
- schema
- table

Exemple :

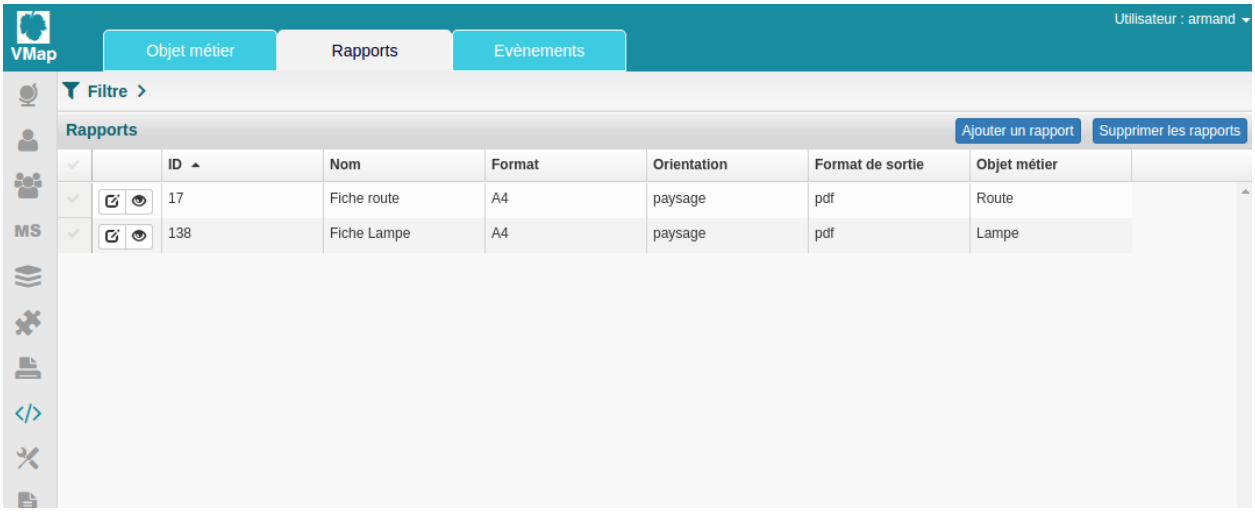


3.2.3. Utilisation du studio

Pour comprendre comment utiliser le studio consulter les documents suivants

- Utilisation du studio
- Exemples d'objets métier dans vMap

4.6.2 Onglet rapports



1. Définition

Un rapport sur un objet métier permet de générer des fichiers au format .pdf ou .doc sur les informations relative à un un objet sélectionné dans le panier.

Deux types de rapports sont à distinguer :

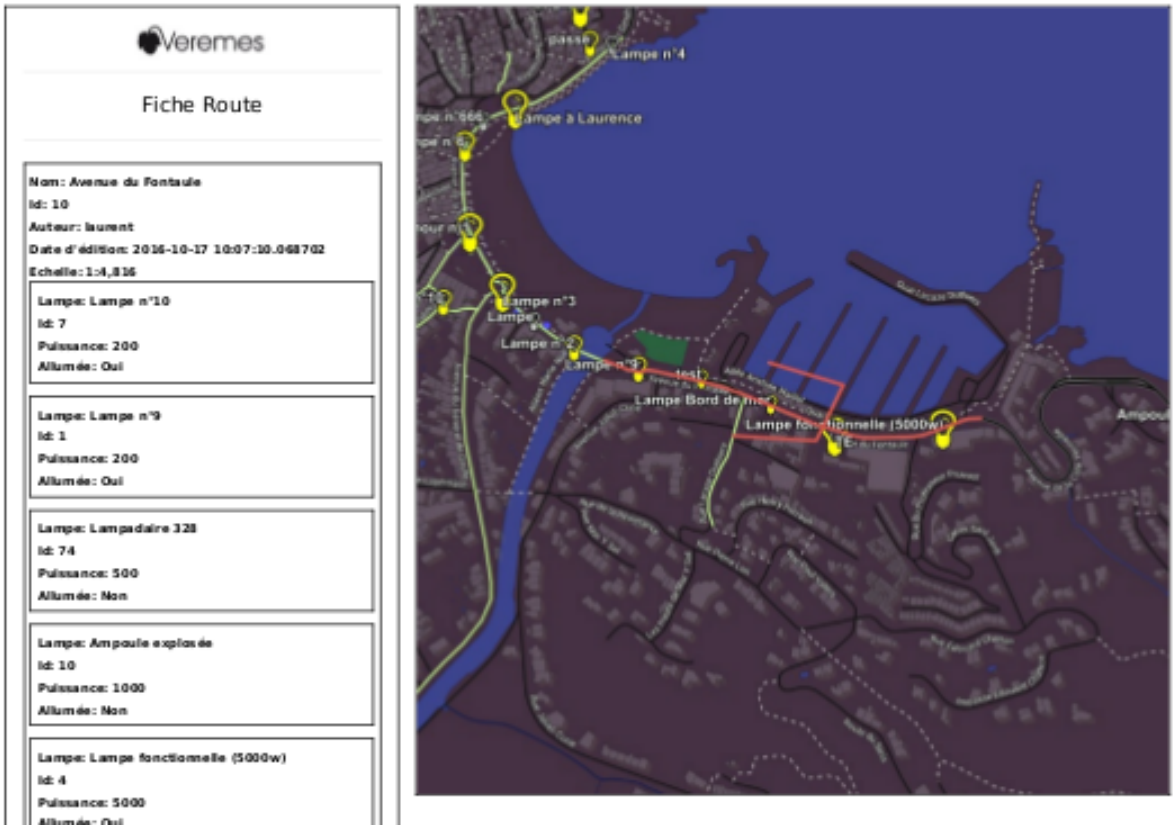
— Les rapports sur un élément 

logo rapport simple

— Les rapports sur plusieurs éléments 

logo rapport multi

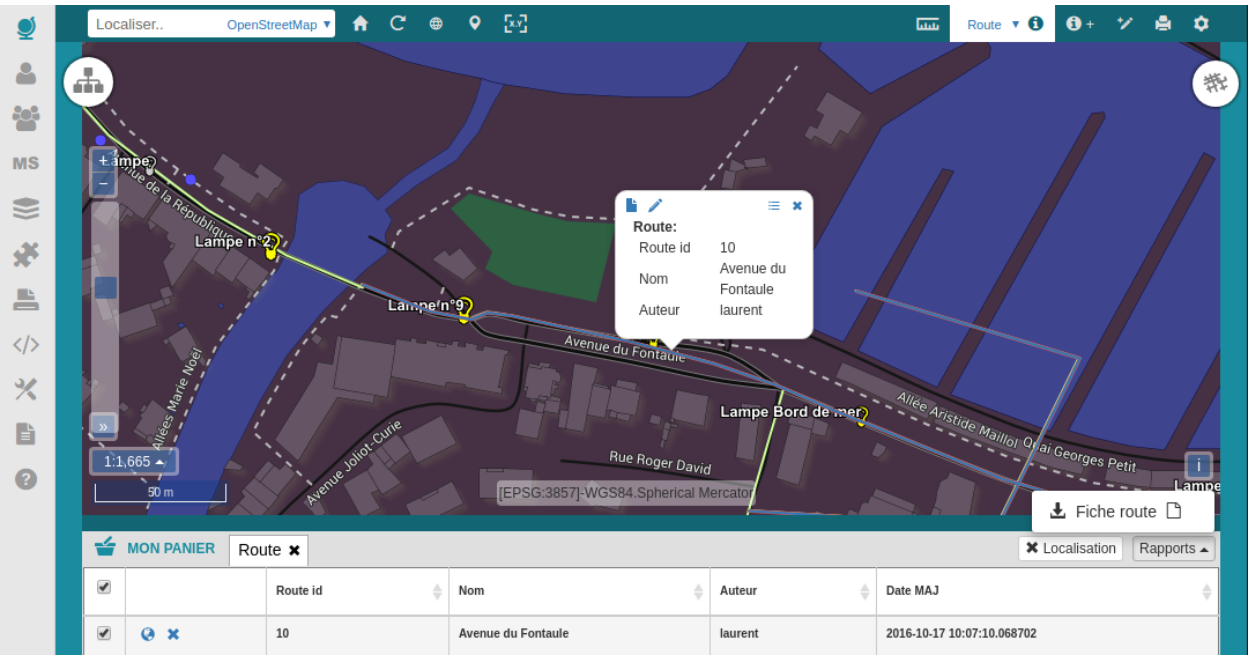
Si un utilisateur sélectionne plusieurs entités et lance un rapport sur un élément, alors plusieurs fichiers sont générés. Inversément, si il lance un rapport sur plusieurs éléments, un seul fichier contenant les informations de chacun des éléments est généré.



2. Utilisation

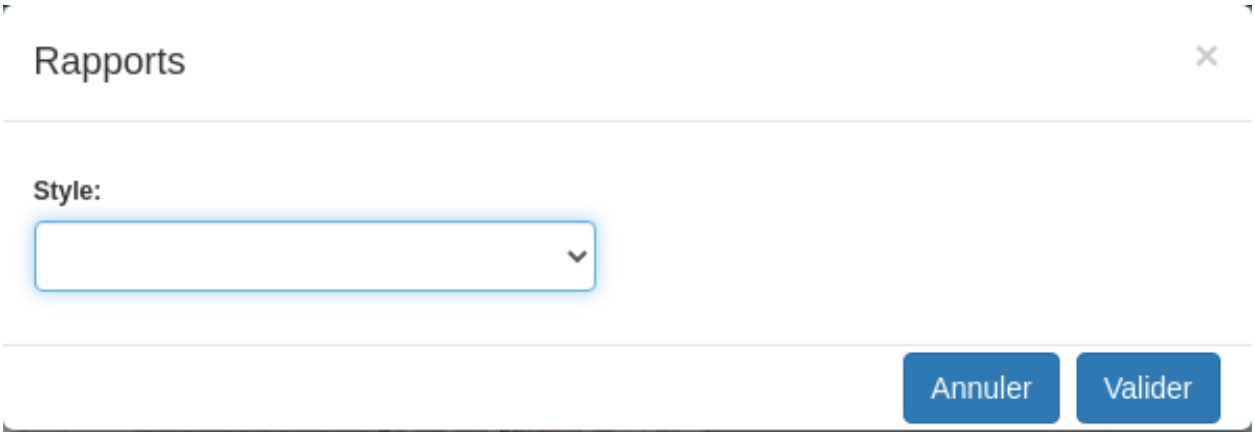
1 - Depuis le panier

Pour générer un rapport sur objet métier depuis le panier, sélectionner un objet sur la carte en cliquant dessus et l"ajouter au panier Une fois dans le panier, sélectionner les objets et générer le rapport voulu à l"aide du bouton « Rapports » (en haut à droite du panier).

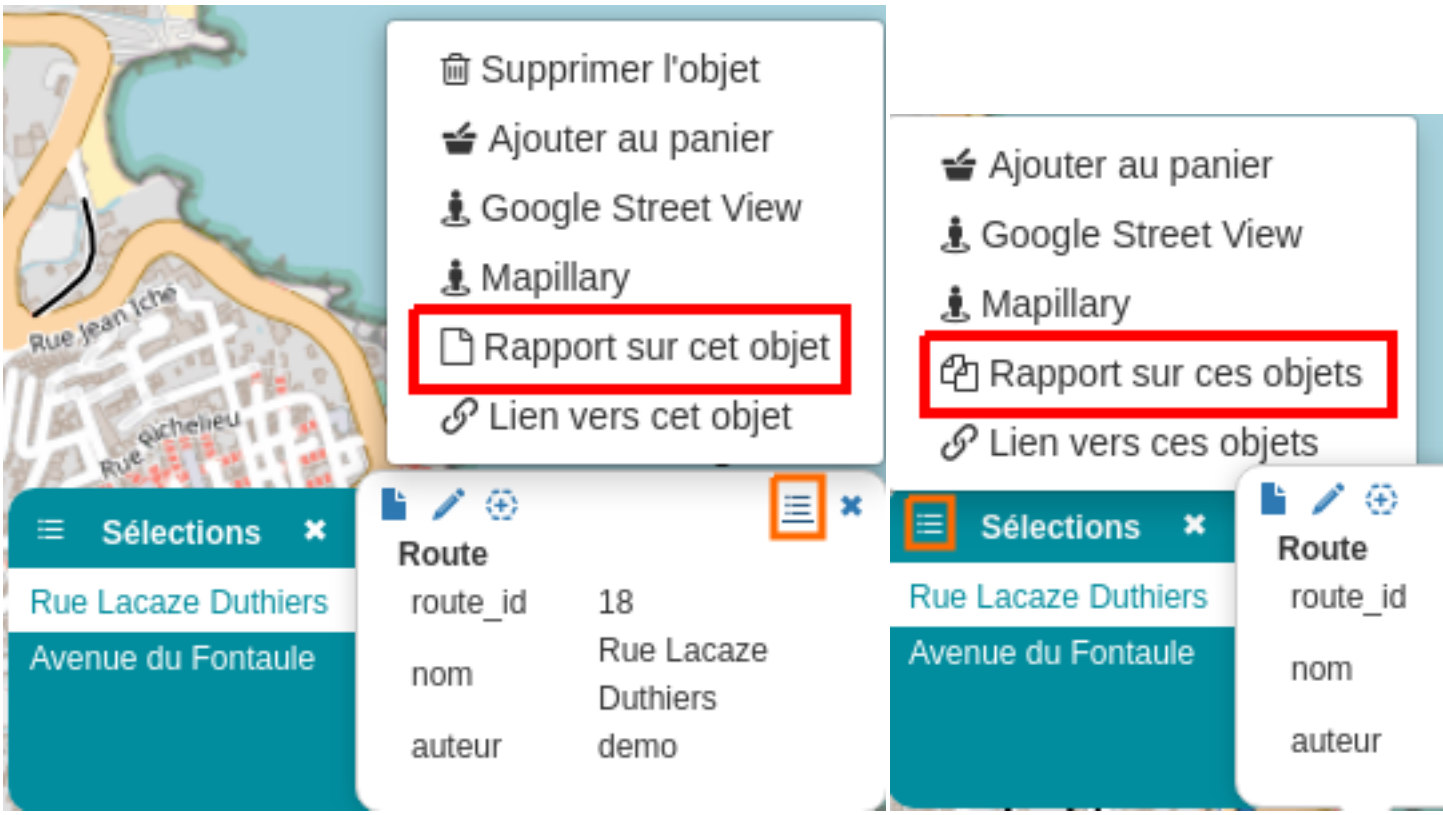


Depuis vMap 2020.02, la création d'"un rapport depuis le panier donne la possibilité de sélectionner un style. Il faut néemmoins que ce dernier soit associé au rapport ainsi qu'"à l'"utilisateur qui est connecté.

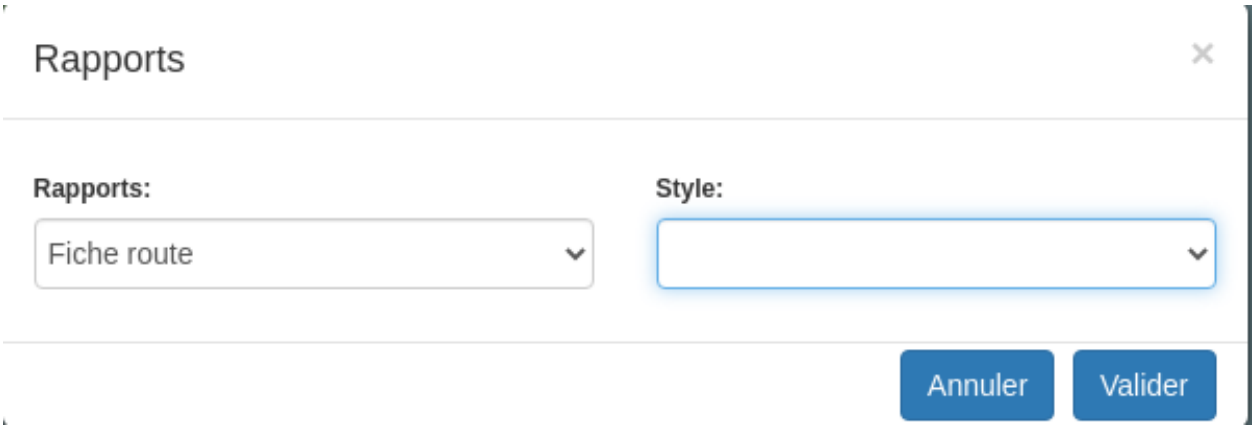
Si aucun style n'"est associé au rapport, il se génère directement.



2 - Depuis l'"infobulle



Pour générer un rapport sur objet métier depuis les infobulles de vMap, sélectionner un objet sur la carte en cliquant dessus puis cliquer pis ouvrir le menu déroulant de l'"infobulle permettant de générer un « Rapport sur ces objets ». Si un style est associé à l'"utilisateur connecté ainsi qu'"au rapport, l'"application permettra de choisir le style dans une fenêtre.



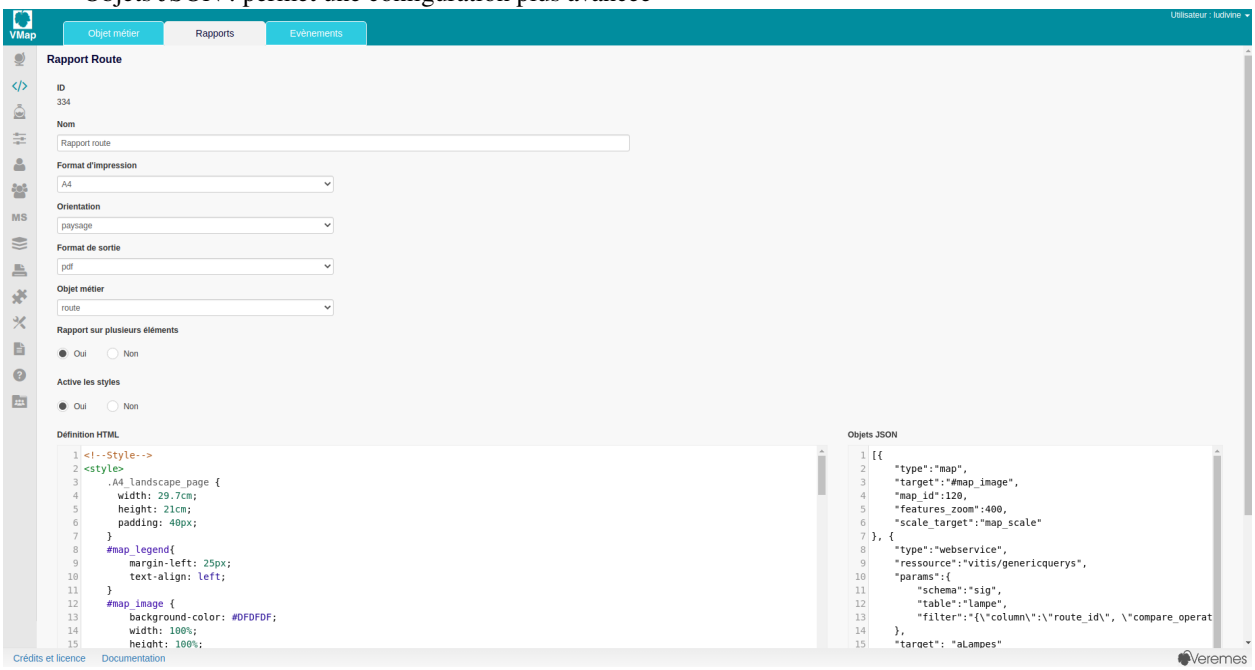
3. Administration

L'"onglet Rapports du menu Développement permet la création, l'"édition et la suppression de rapports.

Dans l'"interface d'"administration renseigner les éléments suivants :

— Nom : nom affiché dans l'"interface
— Format d'"impression : A4/A3

— Orientation : portrait/paysage
— Format de sortie : pdf/doc
— Objet métier : objet métier sur lequel le rapport doit être associé
— Rapport sur plusieurs éléments : pour générer un ou plusieurs documents lors de sélections multiples
— Active les styles : permet d'ajouter un style à un rapport
— Définition HTML : permet de configurer la mise en page
— Objets JSON : permet une configuration plus avancée



3.1. Configuration de la définition HTML

Le bloc de définition HTML permet de configurer la mise en page du rapport. Il est recommandé de procéder en trois parties :

— le style : balise style qui contiend la définition CSS à utiliser.
— le corps : balises HTML de mise en page.
— le script : balise script qui lance du JavaScript lors de la génération (gestion des sauts de page, par exemple).

3.1.1. Utilisation des variables

Dans le corps, la librairie AngularJS est accessible, c'est à dire que l'on peut utiliser la syntaxe suivante pour afficher le contenu d'une variable :

```
<label class="fichelabel">Nom: {{BO.nom}}</label>
```

Dans l'exemple ci-dessus, la variable BO est présente par défaut et contient les attributs de l'objet résultant (notez que pour un rapport à plusieurs éléments, elle se compose d'un tableau contenant les divers objets retournés).

Avec la librairie AngularJS, on peut facilement effectuer des boucles, des conditions, des changements de style etc..

Ci-après, un exemple permettant de faire une boucle et lister les lampes d'une route :

```
<!--Description des lampes de la route-->
<div ng-repeat="oLampe in aLampes" ng-if="oLampe.lampe_id!=undefined" class=
↪"description_box border_container">
```

(suite sur la page suivante)

(suite de la page précédente)

```
    <label class="fiche_label">Lampe: {{oLampe.nom}}</label>
    <label class="fiche_label">Id: {{oLampe.lampe_id}}</label>
    <label class="fiche_label">Puissance: {{oLampe.puissance}}</label>
    <label class="fiche_label">Allumée: {{oLampe.allume ? 'Oui' : 'Non'}}</label>
</div>
```

3.1.2. Affichage de la carte dans un rapport

Si on veut afficher une ou plusieurs cartes dans un rapport, créer dans une première partie, une balise image avec l" »id » de son choix (il est conseillé d"utiliser un fond transparent au cas où les tuiles ne se chargent pas lors de l"impression) :

```
<img id="map_image" src="images/transparent.png">
```

La seconde partie de la manipulation consiste à paramétrer un objet JSON pour indiquer à vMap la carte à utiliser et la façon dont l"utiliser. Se référer à la partie *3.2.1. Configuration des cartes à utiliser dans le template HTML*

3.2. Configuration des objets JSON

Pour bien configurer un rapport, il est utile de configurer la partie Objets JSON. Le but est de pouvoir ajouter des cartes au rapport, interroger des webservices ou afficher des images. Pour cela, créer en JSON, un tableau contenant les différentes configurations. Chacune d"elle est typée avec l"argument « type ».

Exemple :

```
[{
    "type":"map",
    "target":"#map_image",
    "map_id":120,
    "features_zoom":400,
    "scale_target":"map_scale"
}, {
    "type":"webservice",
    "ressource":"vitis/genericquerys",
    "params":{
        "schema":"sig",
        "table":"lampe",
        "filter":"{\"column\":\"route_id\", \"compare_operator\":\"=\", \"value\": \"{
↪{BO.route_id}}\"}"
    },
    "target": "aLampes"
}, {
    "type":"object",
    "content":{
        "company":"Veremes"
    },
    "target": "scope"
}]
```

3.2.1 Configuration des cartes à utiliser dans le template HTML

On peut inclure des cartes dans les formulaires en utilisant des objets de type « map » avec les paramètres suivants :

— target : cible sur laquelle doit se poser la carte (»# » + l"identifiant de votre balise image)
— map_id : identifiant de la carte à utiliser
— features_zoom : coefficient de zoom par rapport à la feature où la valeur 100 correspondrait à un zoom maximum.
— scale_target : nom de la variable qui contient l"échelle de la carte dans le template HTML

Exemple :

```
{
    "type":"map",
    "target":"#map_image",
    "map_id":120,
    "features_zoom":400,
    "scale_target":"map_scale"
}
```

Ici on vient afficher le(s) objets métier sur la carte 120 dans la balise image « #map_image » tout en mettant son échelle dans la variable « map_scale ».

3.2.2. Configuration des webservices

On peut demander à effectuer des requêtes vers des webservices vMap (PHP) pour afficher le résultat dans la vue HTML au travers de variables nommées. Il faut, pour cela, utiliser le type « webservice » et utiliser les paramètres suivants :

— ressource : ressource à interroger
— params : paramètres à utiliser lors de l"interrogation
— target : nom de la variable créée qui contiend les informations retournées

Important : tout comme dans la Définition HTML, on peut utiliser des doubles accolades pour utiliser une variable BO.

Exemple :

```
{
    "type":"webservice",
    "ressource":"vitis/genericquerys",
    "params":{
        "schema":"sig",
        "table":"lampe",
        "filter":"{\"column\":\"route_id\", \"compare_operator\":\"=\", \"value\": \"{
↪{BO.route_id}}\"}"
    },
    "target": "aLampes"
}
```

Dans cet exemple, une requête au webservice vitis/genericquerys permet d"interroger de façon générique des tables. Avec cet appel et en utilisant les doubles accolades {{BO.route_id}}, l"ensemble des lampes contenues dans la route sont affichées.

3.2.2. Configuration des images

On peut afficher des images pré-définies en utilisant le type image et les paramètres suivants :

— imageUrl : URL de l"image (peut être une définition base-64)
— target : cible sur laquelle doit se poser l"image (»# » + l"identifiant de votre balise image)

Exemple :

```
{
    "type":"image",
    "imageUrl":"...",
    "target":"#img1"
}
```

4. Exemple complet

Ci-dessous un exemple complet actuellement visible sur https://demo.veremes.net/vmap/?map_id=29. Dans cet exemple, un projet d'éclairage public contient deux entités :

— les routes
— les lampes

Chaque lampe est associée à une route

4.1 Définition HTML

```
<!--Style-->
<style>
    .A4_landscape_page {
      width: 29.7cm;
      height: 21cm;
      padding: 40px;
    }
    #map_legend{
        margin-left: 25px;
        text-align: left;
    }
    #map_image {
        background-color: #DFDFDF;
        width: 100%;
        height: 100%;
    }
    #map_image2 {
        background-color: #DFDFDF;
        width: 100%;
        height: 100%;
    }
    #map_overview {
        background-color: #DFDFDF;
        height: 4cm;
        width: 4cm;
    }
    .border_container{
        border: 1px solid black;
    }
    .description_box{
        text-align: left;
        padding: 5px;
        margin-bottom: 10px;
    }
    .fiche_urb_label {
        font-size: 10px;
        width: 100%;
        margin-bottom: 0px;
```

(suite sur la page suivante)

(suite de la page précédente)

```
    }
    #img1{
        height: 1cm;
        margin-top: 10px;
        margin-bottom: -10px;
    }
    .main_infos_column{
        height:100%;
        width:100%;
        position: relative;
        min-height: 1px;
        padding-right: 15px;
        padding-left: 15px;
    }
    .infos_column {
      height: 100%;
      border: 1px solid black;
    }
</style>

<!-- A4 print Template -->
<div id="A4_landscape_template" class="A4_landscape_page" style="text-align: center">

    <div class="row" style="padding-left: 10px;">
        <div class="col-xs-4">
            <div class="border_container main_infos_column infos_column">
                <img id="img1" src="images/transparent.png">
                <hr>
                <h4>Fiche Route</h4>
                <hr>

                <!--Description de la route-->
                <div class="description_box border_container">
                    <label class="fiche_urb_label">Nom: {{BO.nom}}</label>
                    <label class="fiche_urb_label">Id: {{BO.route_id}}</label>
                    <label class="fiche_urb_label">Auteur: {{BO.auteur}}</label>
                    <label class="fiche_urb_label">Date d'édition: {{BO.date_maj}}</
↪label>
                    <label class="fiche_urb_label">Echelle: {{map_scale}}</label>
                </div>

                <br>

                <!--Description des lampes de la route-->
                <div ng-repeat="oLampe in aLampes" ng-if="oLampe.lampe_id!=undefined"␣
↪class="description_box child_description_box border_container">
                    <label class="fiche_urb_label">Lampe: {{oLampe.nom}}</label>
                    <label class="fiche_urb_label">Id: {{oLampe.lampe_id}}</label>
                    <label class="fiche_urb_label">Puissance: {{oLampe.puissance}}</
↪label>
                    <label class="fiche_urb_label">Allumée: {{oLampe.allume ? 'Oui' :
↪'Non'}}</label>
                </div>
            </div>
        </div>
        <div class="col-xs-8" style="height: 710px">
            <div style="height: 100%; border: 1px solid black;">
```

(suite sur la page suivante)

(suite de la page précédente)

```
                <img id="map_image" src="images/transparent.png">
            </div>
        </div>
    </div>
</div>

<script>
setTimeout(function () {

  var aElems = $('.child_description_box');
  var aPages = [$('#A4_landscape_template')];
  var currentPage = 0;
  var aBottom = [];
  var iTotalHeight = 0;

  var createPage = function() {
    // Page
    var newPage = document.createElement("div");
    $(newPage).addClass('A4_landscape_page');
    // Zone d'informations
    var newInfosColumn = document.createElement("div");
    $(newInfosColumn).addClass('infos_column');
    $(newInfosColumn).css({
      "padding": "15px"
    });
    // Ajout des éléments
    $(newPage).append(newInfosColumn);
    $('#A4_landscape_template').parent().append(newPage);
    // Sauvegarde de la page
    aPages.push($(newPage));
    currentPage++;
    // Mise à jour de iTotalHeight
    iTotalHeight = getPagesHeight();
    return newPage;
  }

  var getBottomPositions = function(aElems) {
    var aBottoms = [];
    for (var i = 0; i < aElems.length; i++) {
      var iTop = $(aElems[i]).position().top;
      var iHeight = $(aElems[i]).height();
      var iBottom = iTop + iHeight;
      aBottoms.push(iBottom);
    }
    return aBottoms;
  }

  var getPagesHeight = function() {
    var aPagesBotomPositions = getBottomPositions(aPages);
    return aPagesBotomPositions[aPagesBotomPositions.length -1];
  }

  var moveElements = function(aElemsToMove, iPage) {
    for (var i = 0; i < aElemsToMove.length; i++) {
      $(aElemsToMove[i]).appendTo($(aPages[iPage]).find('.infos_column'));
    }
  }
```

(suite sur la page suivante)

(suite de la page précédente)

```
  var pagineChilds = function(){

    aBottom = getBottomPositions(aElems);
    iTotalHeight = getPagesHeight();

    for (var i = 0; i < aElems.length; i++) {

      // Quand un élément est plus bas que la dernière page
      if (aBottom[i] > iTotalHeight - 20) {

        // Crée une nouvelle page
        var newPage = createPage();

        // Déplace les éléments qui suivent sur la nouvelle page
        var aElemsToMove = [];
        for (var ii = i; ii < aElems.length; ii++) {
          aElemsToMove.push(aElems[ii]);
        }
        moveElements(aElemsToMove, aPages.length - 1);

        // Relance la fonction
        pagineChilds();
        return 0;
      }
    }
  }

  pagineChilds();
});
</script>
```

4.2. Objets JSON

```
[{
    "type":"map",
    "target":"#map_image",
    "map_id":120,
    "features_zoom":400,
    "scale_target":"map_scale"
}, {
    "type":"webservice",
    "ressource":"vitis/genericquerys",
    "params":{
        "schema":"sig",
        "table":"lampe",
        "filter":"{\"column\":\"route_id\", \"compare_operator\":\"=\", \"value\": \"{
↪{BO.route_id}}\"}"
    },
    "target": "aLampes"
}, {
    "type":"image",
    "imageUrl":"...",
    "target":"#img1"
```

(suite sur la page suivante)

(suite de la page précédente)

```
}, {
    "type":"object",
    "content":{
        "company":"Veremes"
    },
    "target": "scope"
}]
```

4.6.3 Onglet Evénements

Documentation en cours de rédaction..

4.7 Mode saisie ANC

Documentation en cours de rédaction..

4.8 Interrogation GetFeatureInfo

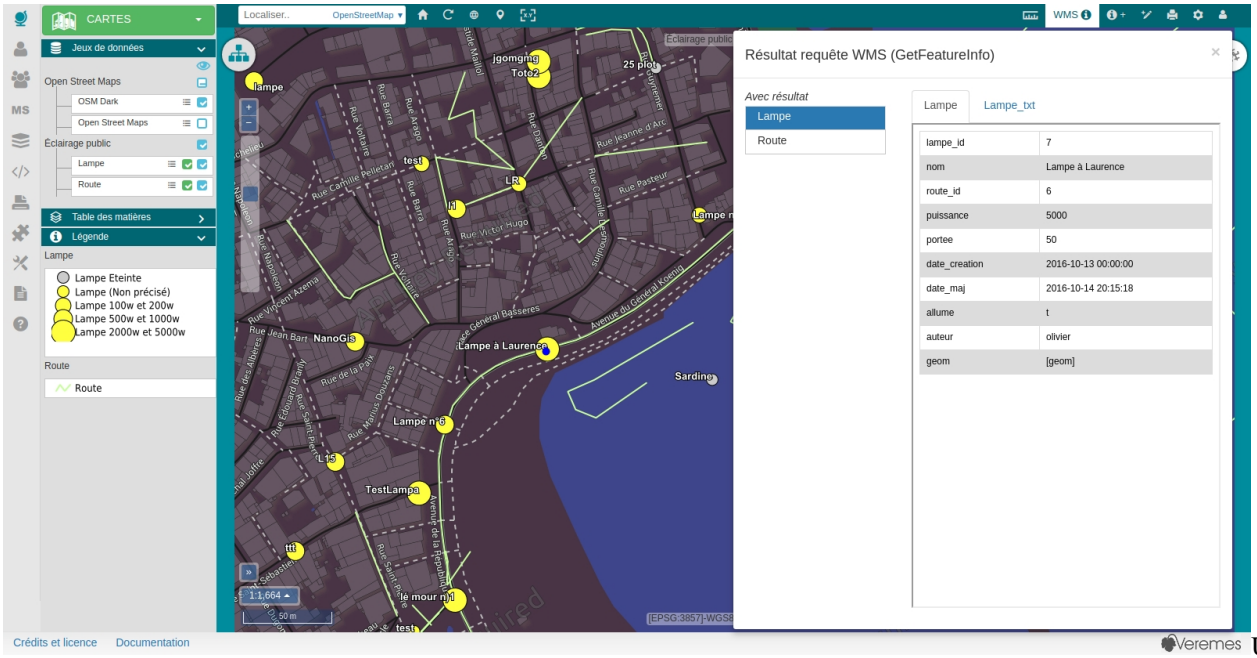
Disponible uniquement à partir de vMap 2018.03.00

Dans vMap il est possible d'interroger les calques au travers d'objets métiers qui permettront en plus de la consultation des attributs, de mettre à jour la base de données, effectuer des jointures, visualiser des formulaires personnalisés interactifs etc. . .

Les objets métiers sont adaptés à l'utilisation dans vMap : ils ne permettront pas d'interroger une couche externe, ni d'interroger de façon attributaire une couche vMap depuis un logiciel tiers (FME, QGIS . . .). Pour faire cela il faudra utiliser les fonctionnalités d'interrogation WMS par requête GetFeatureInfo.

Le GetFeatureInfo est une fonction du protocole WMS permettant d'intérroger une couche dans le but de recevoir les informations attributaires au format JSON, HTML, image ou texte. Dans la majeur partie des cas et dans vMap c'est le format HTML qui sera utilisé.

Depuis le mode cartographie, il suffira de cliquer sur la carte pour interroger toutes les couches actives au GetFeatureInfo. L'administrateur pourra définir à l'avance les couches interrogeables depuis le mode **calques et cartes** et l'utilisateur pourra à tout moment cocher/dé-cocher les couches interrogeables depuis le menu **jeux de données** situé sur la gauche.

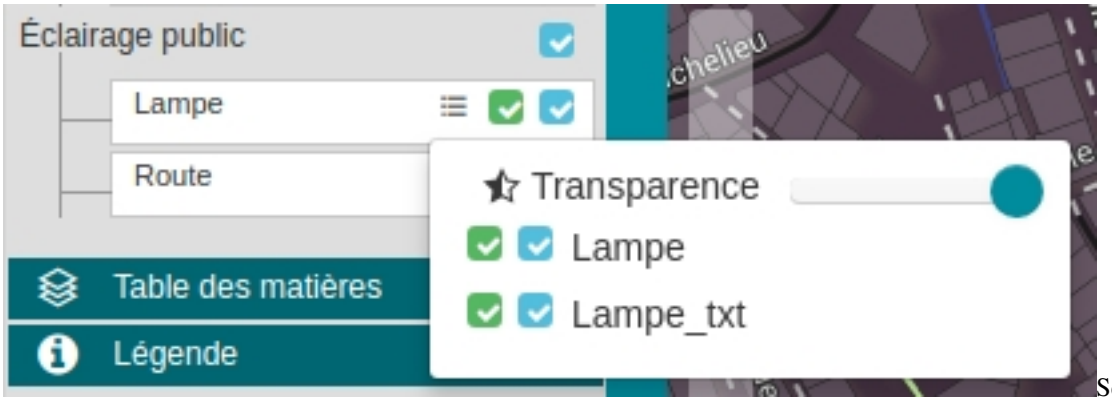


Utilisation GetFeatureInfo dans vMap

4.8.1 Utilisation dans le mode cartographie

Sélection des calques et couches à interroger

L'administrateur aura déterminé à l'avance quels sont les calques interrogeables via GetFeatureInfo (voir partie administration), lors du dépliage du volet de gauche pour les calques définis comme interrogeables GetFeatureInfo une nouvelle boite à cocher apparaîtra et permettra à l'utilisateur de définir si le calque ainsi que ses couches seront interrogés. Les boîtes à cocher peuvent avoir 3 états pour gérer les configurations intermédiaires (calques partiellement interrogeables).

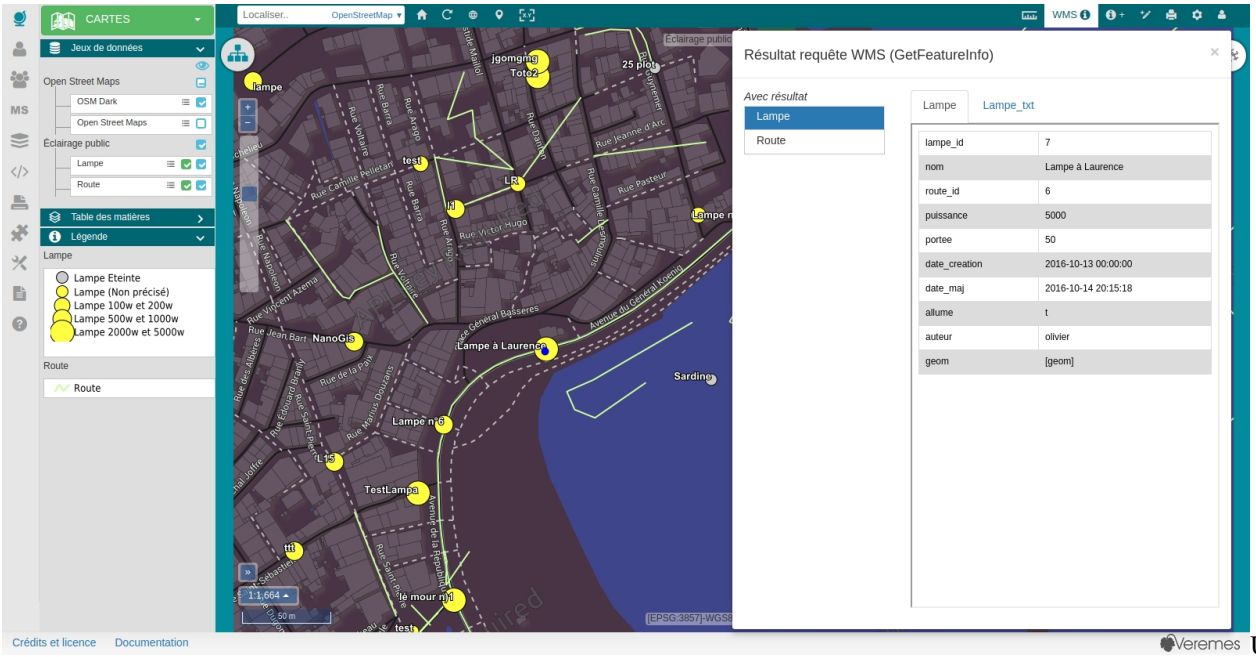


Selection couches GetFeatureInfo

Les calques interrogeables sont ceux pour lesquels le service GetFeatureInfo a été défini. Cette définition n"est pas obligatoire, elle reste de la responsabilité du fournisseur de service.

Clic sur la carte

Après avoir sélectionné la méthode d'interrogation, quand on clique sur un objet cartographique de la carte une fenêtre apparaît avec les différentes couches ainsi que le résultat HTML de la requête. Comme la hauteur et la largeur du résultat peuvent être grandes, des barres de défilement apparaissent automatiquement.



Utilisation GetFeatureInfo dans vMap

On voit apparaître sur la carte en bleu le point qui a été cliqué, sur la droite de la carte une fenêtre résultante apparaît : on y retrouve la liste des couches potentiellement interrogeables de la carte, en cliquant sur un des éléments de cette dernière le résultat de la requête GetFeatureInfo correspondant s'inscrit sur la partie de droite de la fenêtre.

Sur la liste des couches interrogeables, on distinguera les couches avec et sans résultat.

Si pendant que la fenêtre résultante est affichée l'utilisateur effectue un autre clic sur la carte, alors les informations se rechargent, si une couche était sélectionnée dans la fenêtre elle restera affichée même si aucun résultat n'est ressorti.

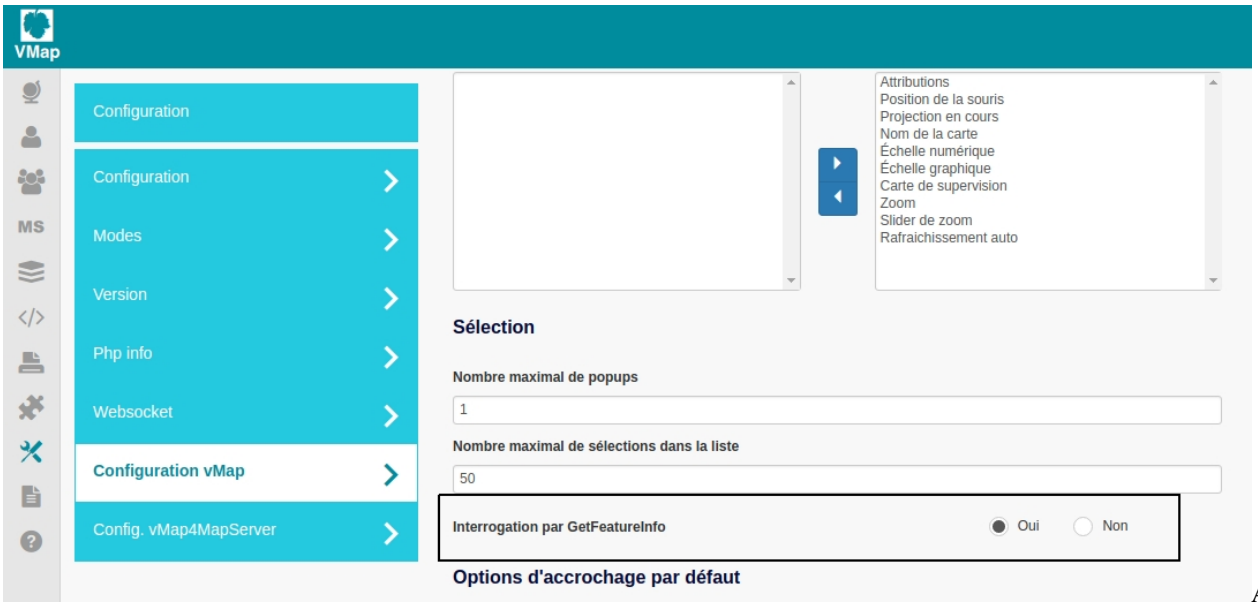
La fermeture de la fenêtre contenant les informations entraîne la suppression du point bleu (localisation du clic). Le résultat affiché est celui de la requête GetFeatureInfo, il peut donc être potentiellement incohérent si le serveur renvoie un message d"erreur ou des documents non html (xml, json. . .).

4.8.2 Activation depuis l"interface d"administration

Pour activer les fonctionnalités d"interrogation WMS par requête GetFeatureInfo il faudra agir sur la **configuration de l"application**, sur les couches depuis le mode **MapServer** et sur les calques depuis le mode **calques et cartes**.

Configuration

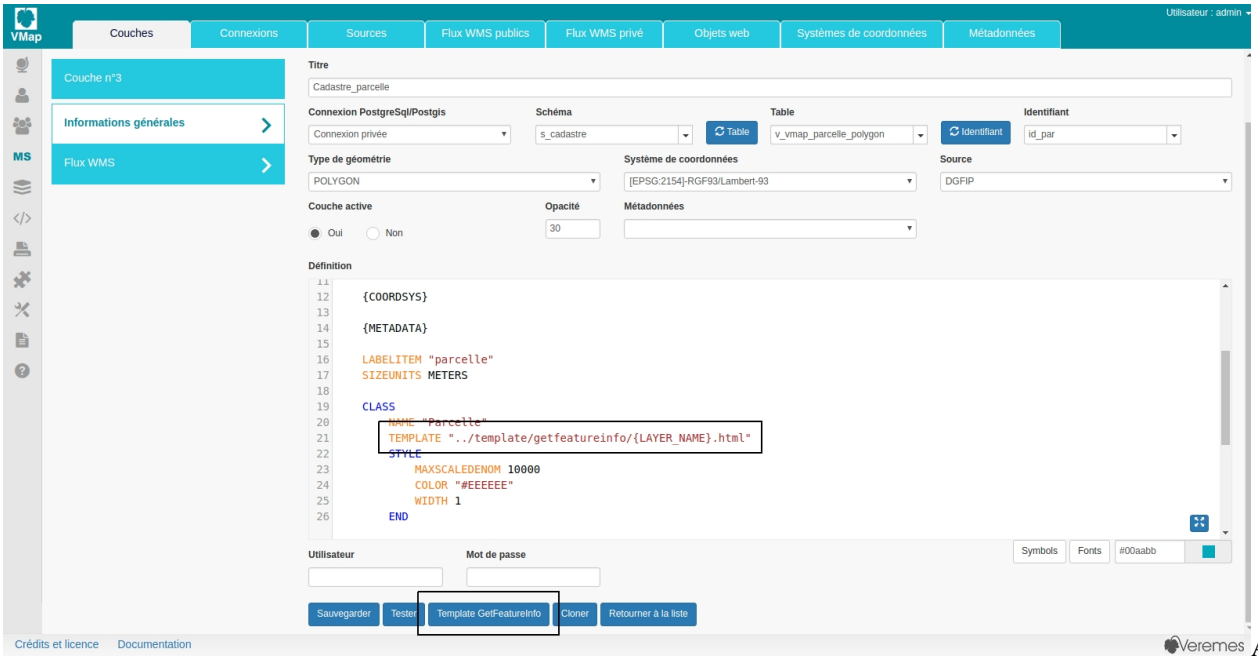
Dans le mode **configuration** puis dans la section **Configuration vMap** il est possible d"activer la fonctionnalité.



Activation GetFeatureInfo

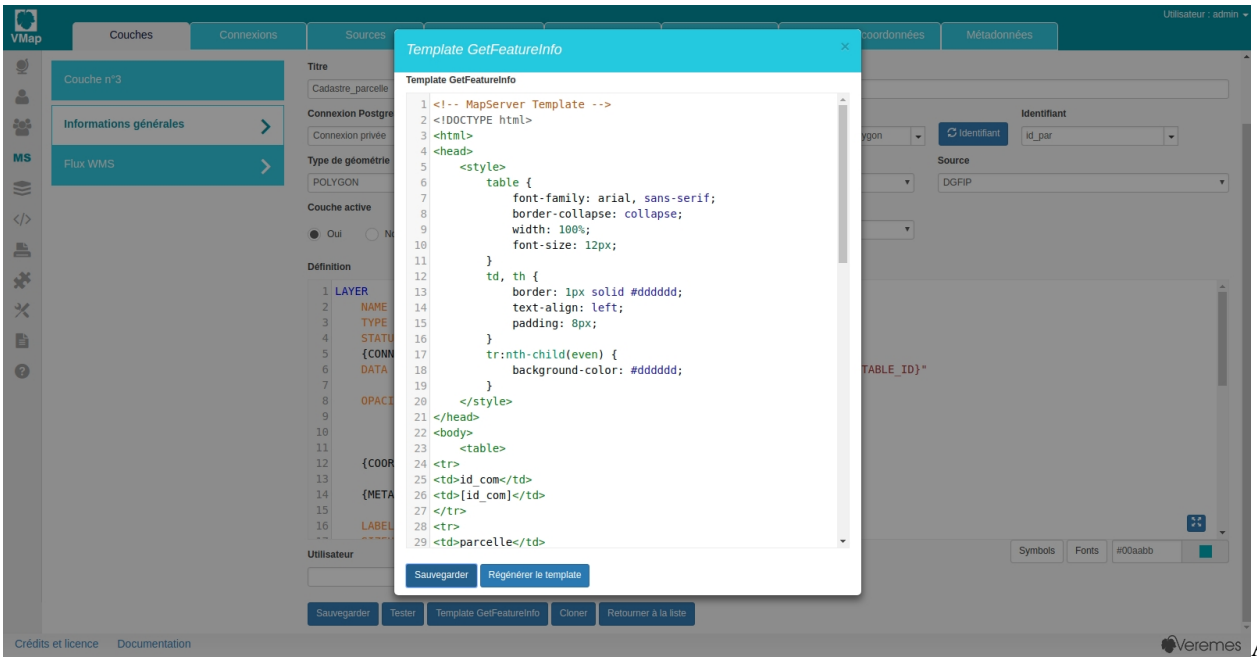
Mode MapServer

Pour que les couches puissent générer du GetFeatureInfo il faudra leur associer un **template**, pour cela il faudra le générer puis renseigner son emplacement dans la définition.



Administration couche GetFeatureInfo

Pour administrer le template il faudra de cliquer sur le bouton **Template GetFeatureInfo** qui va permettre d''éditer, générer et enregistrer le fichier au format HTML sur le serveur.



Administration couche GetFeatureInfo

Une fois le fichier enregistré, il faudra dans la définition de la couche placer dans chaque balise **CLASS** la ligne suivante :

```
TEMPLATE "../template/getfeatureinfo/{LAYER_NAME}.html"
```

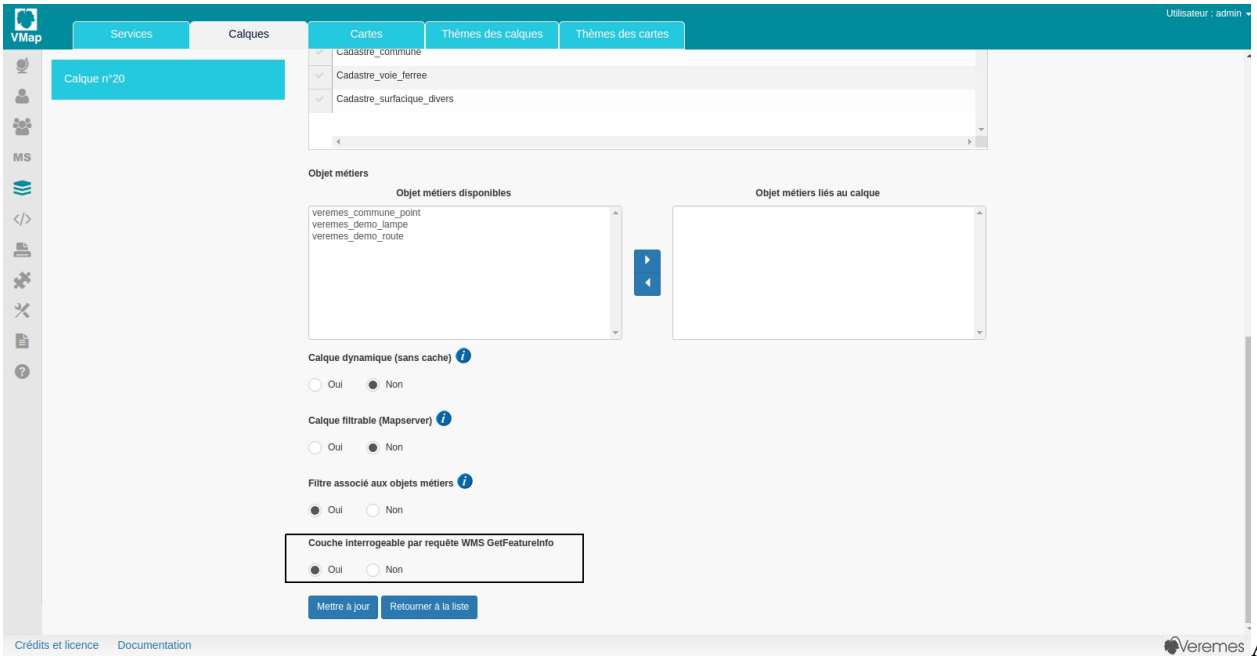
Une dernière étape : renseigner le mime type au flux, pour cela il faudra ajouter dans le mode MapServer à l'objet web `"wms_feature_info_mime_type" "text/html"`

Exemple :

```
WEB
        METADATA
                "wms_title" "{WMSSERVICE_ID}"
                "wms_onlineresource" "{WMSSERVICE_URL}"
                "wms_feature_info_mime_type" "text/html"
                "wms_srs" "EPSG:2154"
                "wms_enable_request" "*"
        END
  # Les attributs TEMPLATE, IMAGEPATH et IMAGEURL sont ajoutés dynamiquement lors des␣
↪tests. Ils sont inutiles en production
END
```

Mode calques et cartes

Enfin il faudra au travers du formulaire du calque définir que ce dernier est interrogeable au travers de la requête GetFeatureInfo.



Administration couche GetFeatureInfo

Retour du GetFeatureInfo au format GML

La configuration du GetFeatureInfo réalisée ci-dessus permet d'obtenir des résultats au format « text/html », le format exploité par vMap. Afin de retourner un résultat au format GML, plusieurs étapes sont à réaliser. Malgré tout, ce résultat ne pourra être exploité dans l'application. Seul un logiciel tier aura la possibilité de le faire. Dans la documentation qui va suivre, allons donc détailler l'ensemble des étapes à réaliser pour retourner en GML le résultat d'une requête GetFeareInfo. Toutes les étapes ci-dessous n'impacteront pas le fonctionnement normal de vMap. Ainsi :

— Etape 1 : Créer un nouvel objet web avec la définition ci-dessous en adaptant les paramètres (wms_onlineresource, wms_srs...) si nécessaire :

```
WEB
        METADATA
                "wms_title" "{WMSSERVICE_ID}"
                "wms_onlineresource" "{MS_CGI_URL}public/{WMSSERVICE_ID}"
                "wms_getfeatureinfo_formatlist" "application/vnd.ogc.gml"
                "wms_feature_info_mime_type" "gml"
                "wms_srs" "EPSG:4326 EPSG:2154 EPSG:3857"
                "wms_enable_request" "*"
        END
  # Les attributs TEMPLATE, IMAGEPATH et IMAGEURL sont ajoutés dynamiquement lors des␣
↪tests. Ils sont inutiles en production
END
```

Par rapport à un objet web « standard » de vMap, le paramètre `"wms_getfeatureinfo_formatlist" "application/vnd.ogc.gml"` a été ajouté, la valeur `"gml"` a remplacé la valeur `"text/html"` dans le paramètre `"wms_feature_info_mime_type"`.

— Etape 2 : Créer un nouveau Flux wms public et lui affecter l'objet web créé à l'étape 1.

— Etape 3 : Créer un nouvelle métadonnée contenant le paramètre `"gml_include_items" "all"`

Exemple :

```
METADATA
        "wms_title"                                    "{LAYER_TITLE}"
        "wms_srs"                                     "EPSG:3857 EPSG:2154 EPSG:4326"
        "wms_name"                                     "{LAYER_NAME}"
        "wms_server_version"          "1.3.0"
        "wms_format"                             "image/png"
        "wms_enable_request"           "*"
        "wms_extent"                            "-357823.2365 6037008.6939 1313632.3628␣
↪7230727.3772"
        "gml_include_items" "all"
END
```

Attention, veillez à modifier certains paramètres (wms_srs, wms_server_version, wms_extent. . .) si nécessaire.

— Etape 4 : Cloner la couche que vous souhaitez interroger en GetFeatureInfo et dont le résultat sera retourné en GML.

Il serait possible de ne pas réaliser cette étape si et seulement si l"interrogation en GetFeatureInfo n"a pas pour but d"être réalisée sous vMap. Si c"est le cas, il est indispensable de cloner la couche auquel cas le GetFeatureInfo ne fonctionnera pas dans vMap pour cette dernière. Idée : Vous pouvez nommer la couche du même nom que la couche cloner en la suffixant par « _GML ». Exemple : Commune_point_GML.

— Etape 5 : Mettre à jour la couche créée à l"étape 4 et lui affecter la métadonnée créé à l"étape 3

Dans le même temps, s"assurer que la couche est bien interrogeable en GetFeatureInfo (cf. Documentation ci-dessus)

— Etape 6 : Test

Pour tester cette nouvelle configuration, exécuter une requête GetFeatureInfo dans un navigateur sur la couche que vous avez paramétré. Ce dernier va alors télécharger un nouveau fichier dont le contenu se retrouve au format GML.

Exemple de requête GetFeatureInfo (modifier les paramètres nécessaires) :

```
https://[serveur]/wms/public/[nom_flux_public]?SERVICE=WMS&VERSION=1.3.0&
↪REQUEST=GetFeatureInfo&FORMAT=image/png&TRANSPARENT=true&QUERY_LAYERS=Commune__
↪point_GML&LAYERS=Commune__point_GML&STYLES=&INFO_FORMAT=GML&I=50&J=50&CRS=EPSG
↪%3A2154&WIDTH=101&HEIGHT=101&BBOX=642170.3063538198,6159025.597572117,661784.
↪8780762318,6178640.16929453&vitis_version=3009
```

Attention : Le paramètre `INFO_FORMAT` doit impérativement avoir `GML` comme valeur.

Exemple de résultat :

```
<?xml version="1.0" encoding="UTF-8"?>

<msGMLOutput
         xmlns:gml="http://www.opengis.net/gml"
         xmlns:xlink="http://www.w3.org/1999/xlink"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <Commune__point_GML_layer>
        <gml:name>Commune__point_GML</gml:name>
                <Commune__point_GML_feature>
                        <gml:boundedBy>
                                <gml:Box srsName="EPSG:2154">
                                        <gml:coordinates>679240.481423,6154435.540671␣
↪679240.481423,6154435.540671</gml:coordinates>
                                </gml:Box>
                        </gml:boundedBy>
                        <code>66049</code>
                        <nom>Ceret</nom>
                        <pop90>7289</pop90>
                        <nbr_service>1</nbr_service>
                </Commune__point_GML_feature>
```

(suite sur la page suivante)

(suite de la page précédente)

```
        </Commune__point_GML_layer>
</msGMLOutput>
```

4.9 Guide du développeur

4.9.1 Utilisation du studio

Attributs de formulaire

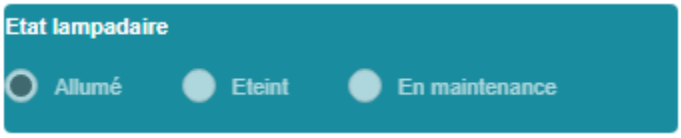
1. Définition

Les attributs d'un formulaire peuvent être créés, édités et supprimés dans le studio. Certains attributs exploitent des valeurs issues de base de données ou de services web, d'autres sont des composants de mise en page destinés à personnaliser l'interface du formulaire, et d'autres sont des attributs classiques directement configurables dans vMap.

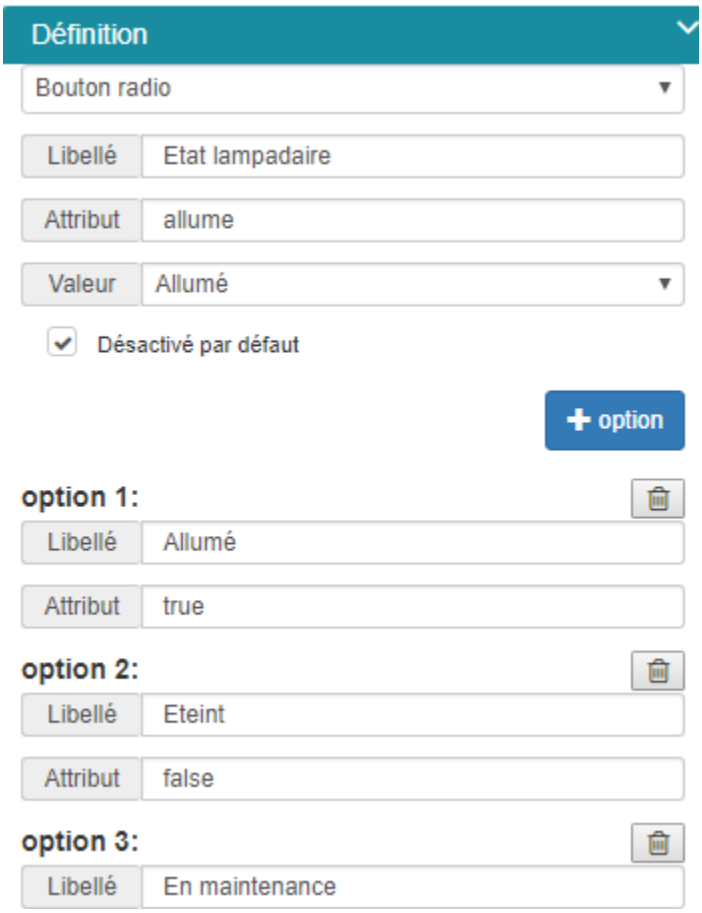
Il existe plus d'un vingtaine de types d'attributs paramétrables dans la fenêtre de définition en bas à gauche du studio.

2. Édition des attributs

2.1. Bouton radio

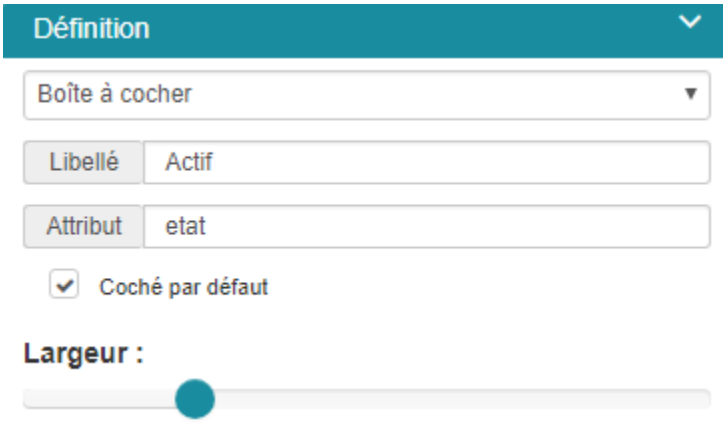


Nommer l'attribut et son libellé tel qu'il sera affiché dans le formulaire. Définir la valeur par défaut et déterminer si le bouton radio doit être désactivé ou pas. Définir ensuite les options possibles en entrant le libellé du bouton et la valeur envoyée en base. Le bouton « Option » permet l'ajout d'une option supplémentaire.



2.2. Boîte à cocher

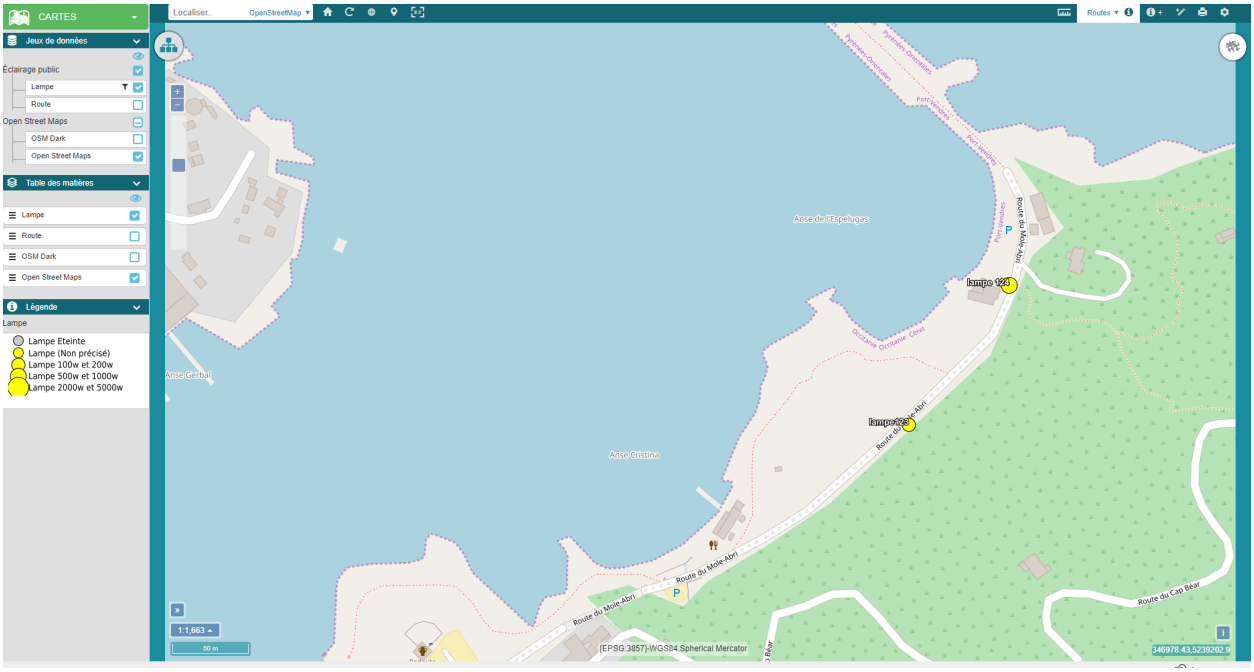
Nommer l'"attribut et son libellé tel qu"il sera affiché dans le formulaire. Définir si la boîte doit être cochée par défaut ou pas.



2.3. Paramètres de type Carte Bing, OSM, vMap

vMap permet d'exploiter les services web OSM, Bing Maps ou Vitis vMap pour personnaliser un formulaire en exploitant leurs ressources cartographiques.

2.3.1. Carte OSM



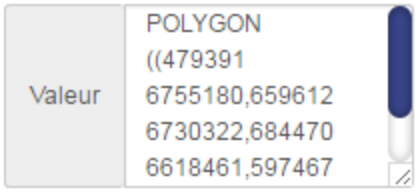
Nommer le paramètre et définir le libellé qui sera affiché dans le formulaire de demande. Définir la hauteur et la largeur de la carte et indiquer si ce paramètre est obligatoire ou pas en cochant la case Requis.

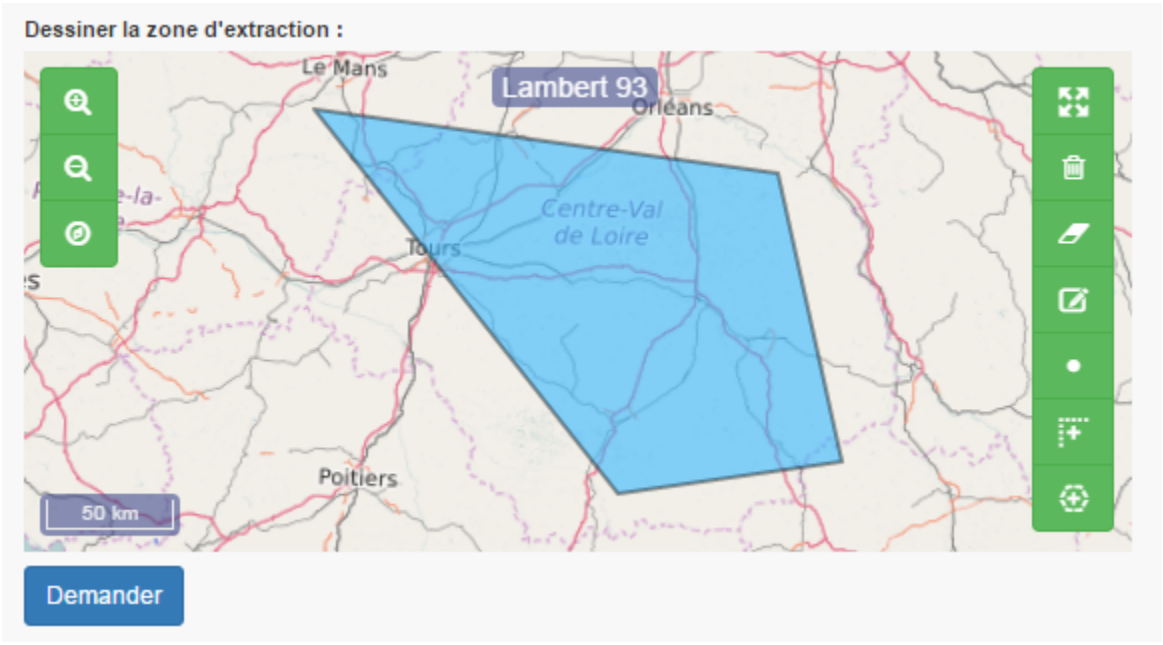
Paramétrer ensuite les options spécifiques aux éléments de type carte :

- La projection de la carte : WGS84 ou Lambert 93. En Lambert 93, l'étendue par défaut correspond à l'ensemble de la France métropolitaine.
- Méthode de centrage de la carte : l'auteur choisit si le centre de la carte est défini par un point défini via des coordonnées X/Y et une échelle d'affichage, ou si le centre de la carte est paramétrée par son étendue définie par les coordonnées X et Y Min et Max.

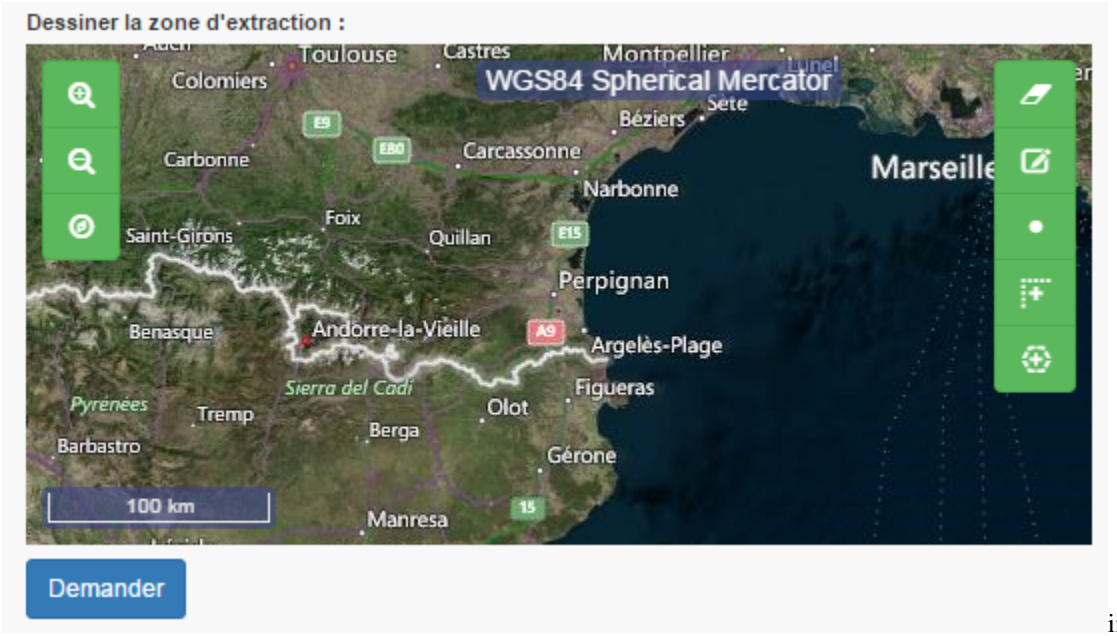
Choisir ensuite les éléments de dessin et de navigation qui seront affichés sur la carte du formulaire de demande :

- Position de la souris : affichage dynamique des coordonnées de la souris selon la projection définie.
- Boutons de zoom : affichage des boutons de navigation classique zoom avant, zoom arrière et retour à l'étendue par défaut.
- Echelle : affichage de l'échelle.
- Projection de la carte : affichage de la projection Lambert 93 ou WGS 84.
- Multiples géométries : possibilité ou pas de saisir des géométries de type différent (point, ligne et polygone).
- Plein écran : permet d'afficher la carte en mode plein écran.
- Suppression générale : Suppression de toutes les géométries saisies sur la carte.
- Edition : modification de la géométrie sélectionnée.
- Dessiner un point.
- Dessiner une ligne.
- Dessiner un polygone.
- Le champ Valeur permet à l'auteur de définir une géométrie qui sera affichée par défaut dans le formulaire. Cette géométrie est décrite via une chaîne WKT :





2.3.2. Carte Bing



image

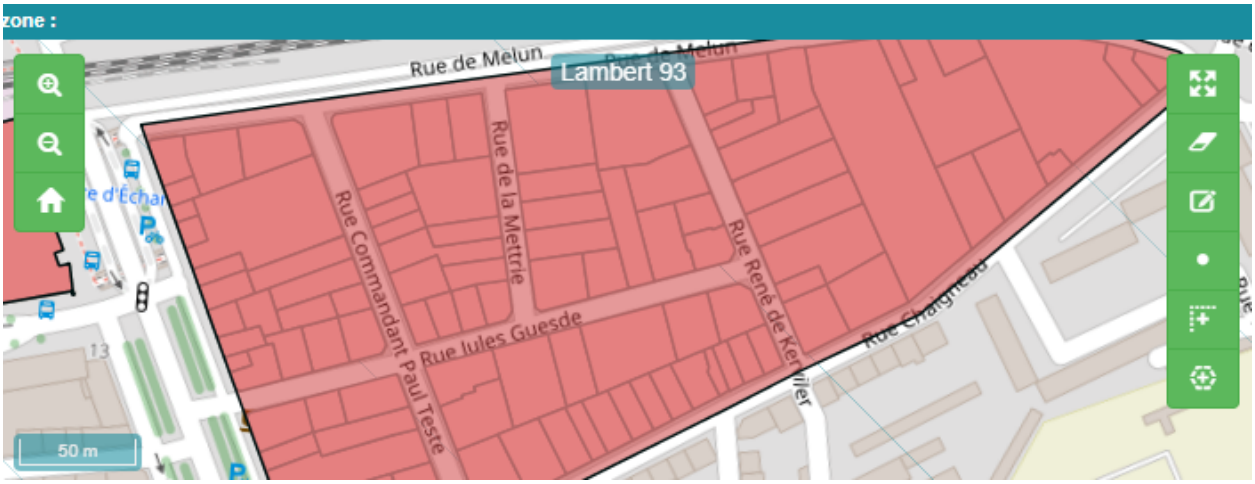
Tous les paramètres de personnalisation d'une carte Bing Maps sont identiques à ceux des cartes OSM. Il faut fournir en plus, une clé d'accès pour pouvoir exploiter ce service web cartographique.

Générer une clé Bing Maps sur le site .. : https://www.bingmapsportal.com/

Une fois obtenue, entrer la clé dans le champs Clé et sélectionner la carte à afficher dans le formulaire de demande :

- Aerial
- Aerial WithLabels
- Road

2.3.3. carte vMap



Pour pouvoir exploiter une carte vMap, il faut au préalable, dans vMap, exporter la définition de la carte.

L'export d'une carte vMap génère un fichier map.json que l'auteur du formulaire doit télécharger (champ Fichier local) pour pouvoir l'intégrer dans un formulaire.

Procéder ensuite de la même façon qu'avec les autres ressources de type carte, en nommant le paramètre et son libellé, puis en paramétrant l'affichage des outils propres aux cartes.

2.6. Champ caché

Un attribut de type Champ caché permet de masquer un attribut. Il est exploité dans le formulaire mais n'est pas apparent. Nommer le paramètre et définir la valeur à exploiter.

2.7. Couleur

Un attribut de tye Choix de la couleur insère un sélecteur de couleurs. Nommer le paramètre et le libellé à afficher dans le formulaire, et définir la couleur par défaut.

2.8. Curseur

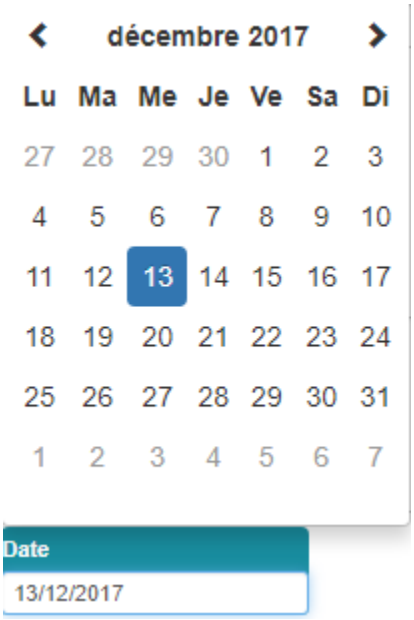
Un attribut de tye curseur insère un curseur dans le formulaire. Nommer le paramètre et le libellé à afficher et définir les valeurs minimales et maximales de la plage de données ainsi que la valeur par défaut.

2.9. Paramètres de type Date

2.9.1. Date

Un attribut de type Date insère une date sous la forme jj/mm/aaaa. Un calendrier s"affiche dans le formulaire pour faciliter la date à entrer.

Nommer le paramètre et le libellé à afficher et définir la valeur par défaut.



2.9.2. Date/heure

Un attribut de type Date et heure insère une date sous la forme jj/mm/aaaa hh :mm. Un calendrier et une montre facilite la saisie la date et de l"heure dans le formulaire.

Nommer le paramètre et le libellé à afficher et définir la valeur par défaut.

2.10. Dessin

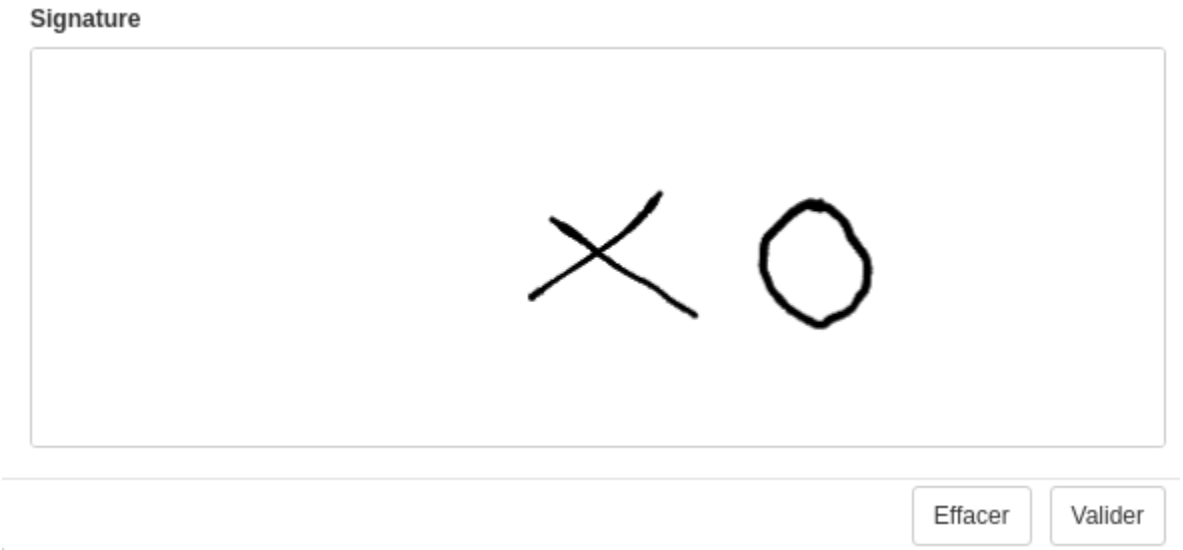
Un attribut de type Dessin permet de réaliser à main levée une « forme géométrique ». Le dessin réalisé est enregistré sous forme d'image. Cet attribut peut être utilisé sur smartphone, tablette ou ordinateur de bureau.

— Prérequis :

Pour faire fonctionner un champ de type dessin, rajouter un champ de type « Character Varying (255) » en base de données.

— Exemple d'utilisation :

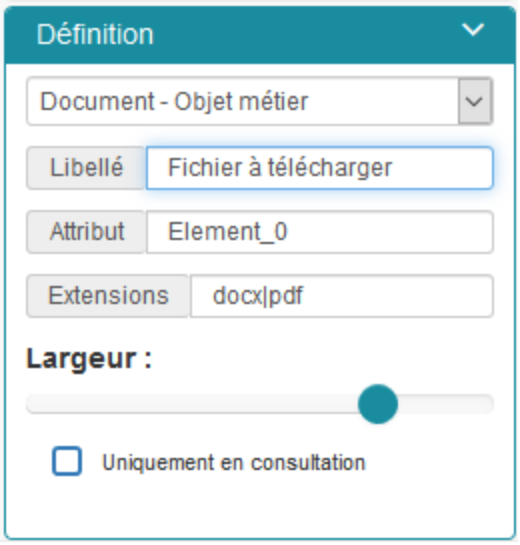
Un attribut de type dessin peut notamment être utilisé pour faire signer un locataire lors d'un état des lieux de sortie.



2.11. Document - objet métier

Un attribut de type Document - objet métier est un champ de chargement de documents.

Nommer le paramètre et le libellé à afficher.



Définir le format des documents téléchargeables en indiquant leurs extensions possibles séparées par un |.

La boîte à cocher « Uniquement en consultation » indique si le document est uniquement consultable ou si il doit être téléchargé.

Un unique fichier peut être associé à un attribut. Il faut donc compresser les documents en un unique fichier zip pour pouvoir les associer à un même attribut.

Obtenir un exemple d"insertion d"attribut de type Document

2.12. Image Objet métier

Un attribut de type Image - objet métier est un champ de chargement d"images.

Nommer le paramètre et le libellé à afficher.

La boîte à cocher « Uniquement en consultation » indique si l"image est uniquement consultable ou si elle doit être téléchargée.

Un unique fichier peut être associé à un attribut. Il faut donc compresser les images en un unique fichier zip pour pouvoir les associer à un même attribut.

2.13. Décimal

Nommer l"attribut et le libellé qui seront affichés dans le formulaire et définir la valeur par défaut. Définir si ce paramètre est obligatoire ou pas en cochant la case Requis.

2.14. Entier

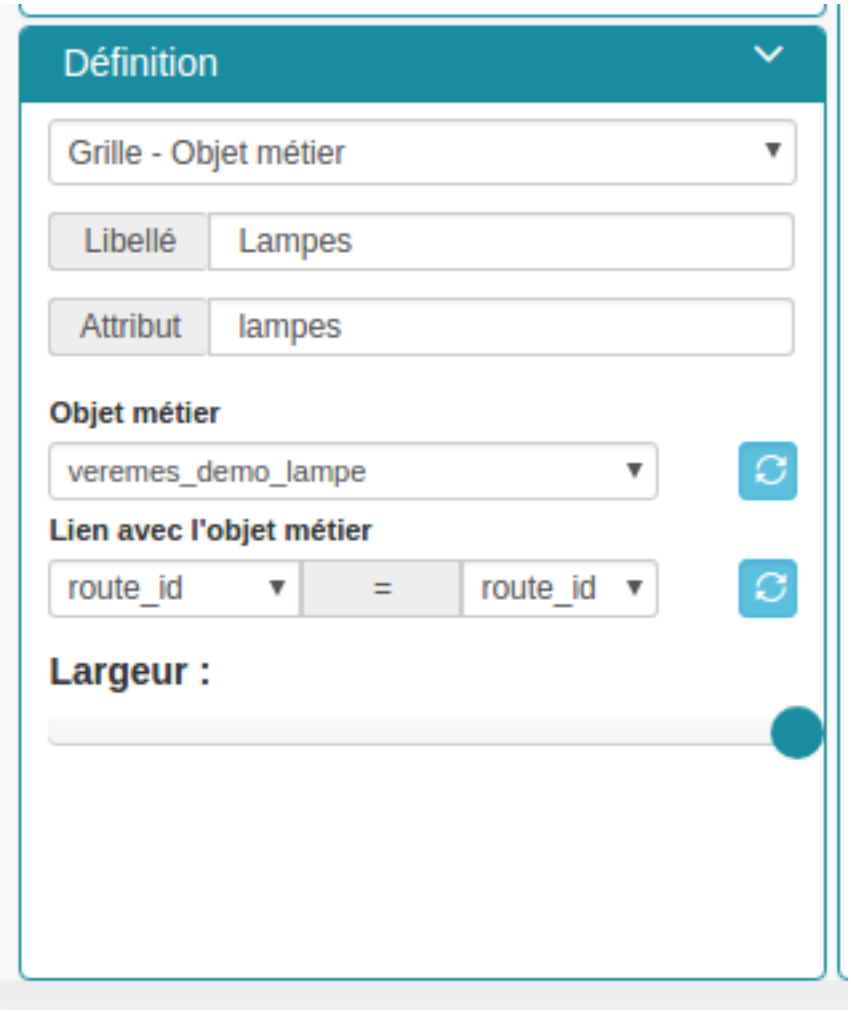
Nommer l"attribut et le libellé qui seront affichés dans le formulaire et définir la valeur par défaut. Définir si ce paramètre est obligatoire ou pas en cochant la case Requis.

2.15. Editeur de code CodeMirror

2.16. Grille objet métier

Un attribut de type Grille objet métier permet d"associer à un élément un sous élément. La grille objet métier permet l"insertion d"objet enfant associé à un objet parent.

Nommer le paramètre et le libellé à afficher, puis sélectionner l"objet métier enfant et définir l" attributs parent et enfant surlesquels reposent l"ascendance.



2.17. Grille section vitis

2.18. Image URL

Un attribut de type image URL permet d"afficher dans un formulaire l"url d"un image. Indiquer le libellé, l"attribut et l"url dans le champs valeur.

2.19. Interface bouton

2.20. Interface ligne de séparation

2.21. Label

2.22. Lien

Un attribut de type Lien permet d'insérer des liens vers d'autres plateformes. Nommer le paramètre et le libellé à afficher.

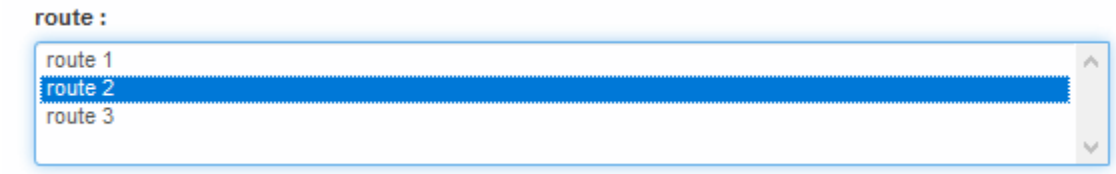
Définir ensuite les paramètres du lien :
- Texte : texte à afficher
- Cible : si laissé vide, la page s'ouvre dans un nouvel onglet
- Valeur : adresse du lien

2.23. Liste

3 types de listes sont paramétrables dans vMap. Le gestionnaire de source de données permet une configuration fine des listes, de leurs sources et de leurs modalités d'affichage.

2.23.1. Liste simple

On entend par liste simple, une liste de choix dans laquelle l'utilisateur sélectionne un élément directement en cliquant parmi une des ocurrences de la liste :



Après avoir indiqué le libellé de l'objet et l'attribut à partir duquel il est défini, sélectionner la source de données dans la liste ou cliquer sur le bouton Sources de données pour en ajouter.

2.23.2. Liste double

Une liste double permet d'afficher dans un formulaire les éléments d'une liste répartis en deux blocs : les éléments disponibles en sélection et les éléments sélectionnés.

2.23.3. Liste déroulante

Un attribut de type liste déroulante permet d'afficher la liste de choix sous forme de liste déroulante.

2.24. Texte

Nommer l'attribut et son libellé tel qu'il sera affiché dans le formulaire. Une expression régulière peut être définie pour containdre la saisie de ce champ, et une valeur pas défaut peut également être définie.

Le Gestionnaire de source de données

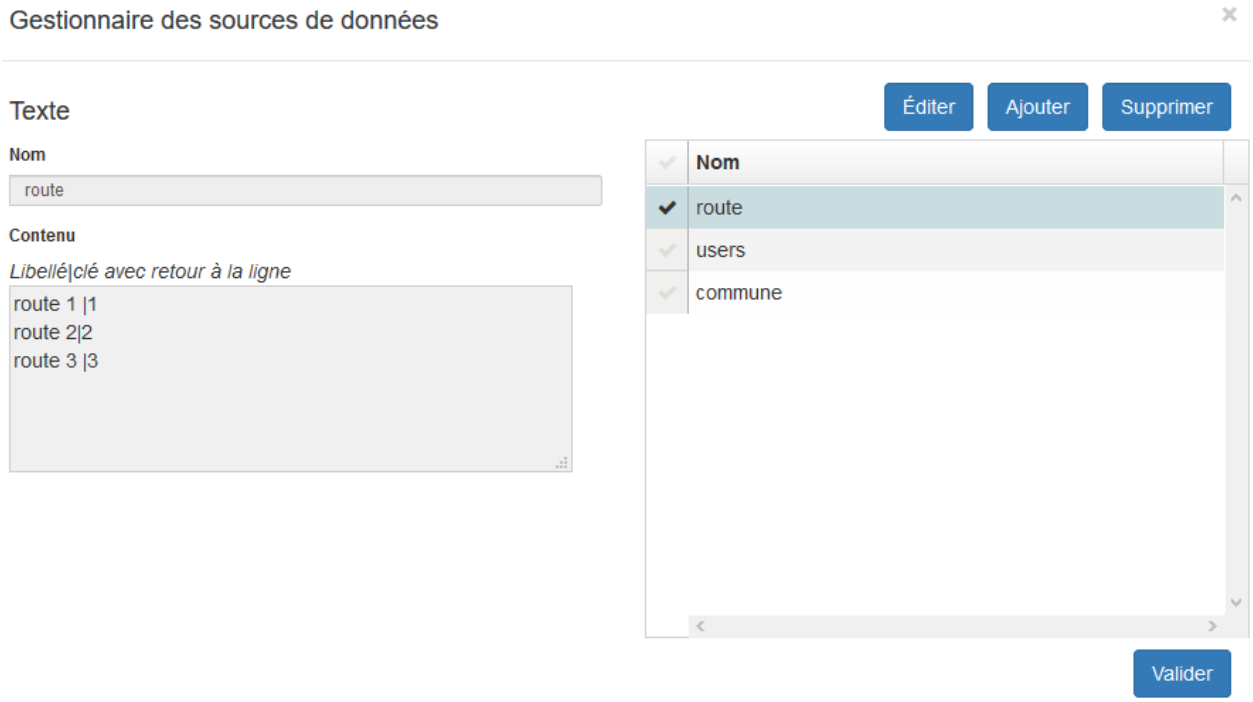
Le gestionnaire de sources de données permet la création, l'édition et la suppression de sources de données à associer à des attributs de type :

- liste
- liste déroulante
- liste double

Le gestionnaire de sources de données permet d'exploiter des données :

- Texte : valeurs saisies directement dans le gestionnaire
- Valeur de table locale : valeurs issues d'une table de base de données installée sur le serveur
- Base de données externe : valeurs importées d'une table d'une base de données externe
- Service web Vitis : permet d'exploiter un service web pour en récupérer les ressources
- Objet métier : permet d'exploiter un objet métier déjà configuré

Le bouton **Sources de données**, en bas à droite du studio permet d'ouvrir le gestionnaire de source de données.



Une fois une source de données définie dans le gestionnaire, on peut créer dans le studio, un attribut de type « *Liste* » et choisir la source de données créée précédemment.

1. Source de données de type texte

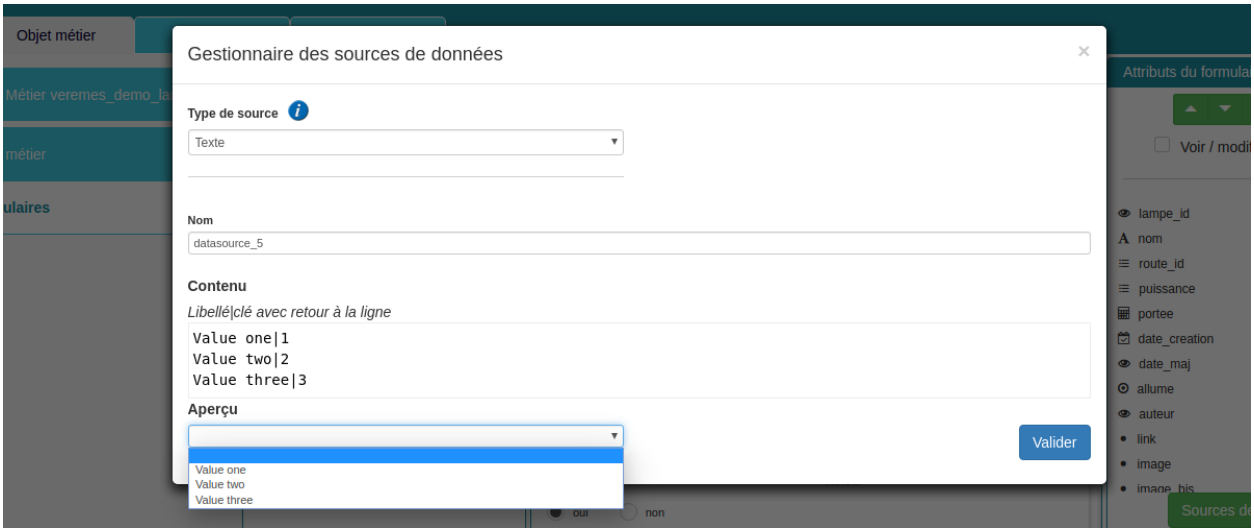
Le type texte permet de renseigner soi-même le contenu de la source de données.

```
libellé 1|clé 1
libellé 2|clé 2
libellé 3|clé 3
```

Chaque entité est composée :

- d'une **clé** qui est la valeur retenue
- d'un **libellé** qui est le contenu affiché dans le formulaire.

Les deux sont séparés (sans espace) par le caractère « | ». Répéter l'opération autant de fois que d'occurrences, en retournant à la ligne pour chaque élément.



2. Source de données de type valeurs d'une table locale

Ce type de source permet de récupérer directement en base de données le contenu d'une table. Définir :

- le nom de la base de données
- le schéma
- la table

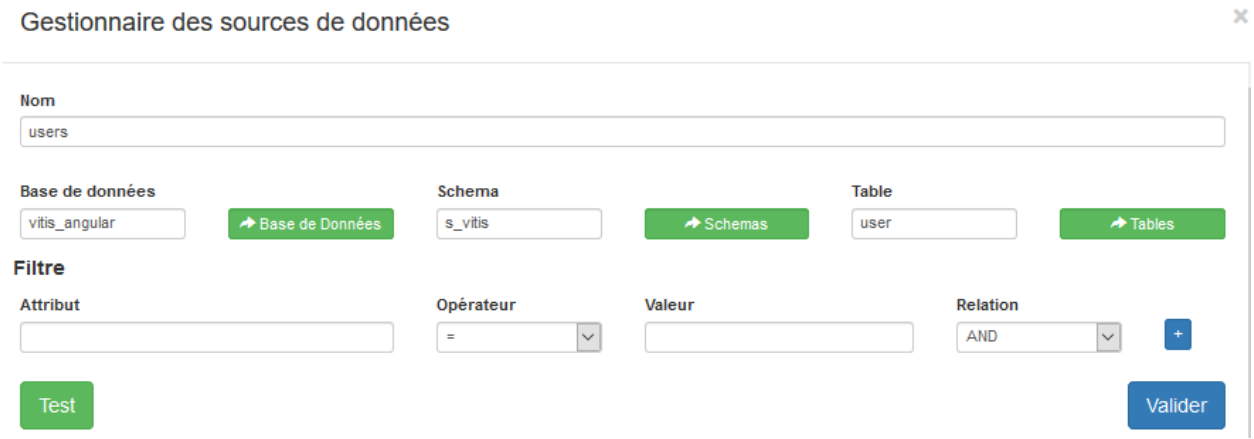
2.1 Affichage de la liste

On peut filtrer les enregistrements à afficher dans la liste. Pour cela renseigner :

- l'attribut qui est le nom de la colonne sur laquelle porte le filtre
- l'opérateur
- la Valeur à utiliser pour définir la condition.

Le bouton « + » permet l'ajout de conditions et on peut déterminer si les multiples conditions sont de type « *AND* » ou « *OR* ».

Important : l'insertion de ce type de source de données (tables locales) utilise le token de connexion de l'utilisateur. Il faut donc faire attention à ce que **tous les utilisateurs susceptibles d'utiliser le formulaire aient des droits en consultation sur la table.**

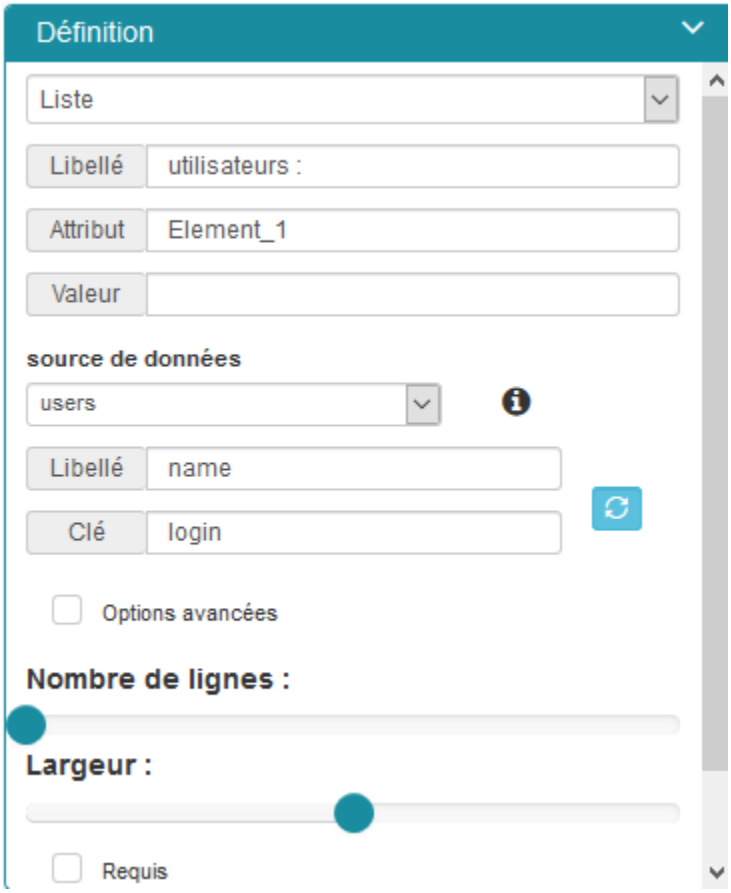


2.2 Clé et libellé

Dans le studio, définir l'attribut et son libellé à afficher dans le formulaire.

Puis sélectionner la source de données précédemment créée dans le Gestionnaire de source de données.

Définir ensuite le libellé et la clé des occurences de la liste :



Dans l'exemple ci-dessus, on souhaite pouvoir sélectionner dans le formulaire les utilisateurs dont les noms (colonne name) seront affichés dans une liste « utilisateurs ». La clé (colonne login) de chaque utilisateur est la valeur réellement utilisée.

2.3 Options avancées = affiner l'affichage des listes

Il est possible d'affiner l'affichage des éléments de la liste via les commandes :

- commandes de tri (order_by et sort_order) sur les attributs dont on spécifie les noms dans le champs Attributs. Ils doivent êtres séparés par le caractère |.
- clause distinct pour distinguer les valeurs identiques.
- filtre pour filtrer les valeurs à afficher via une clause Where dont on spécifie les arguments dans le champ " filter".

2.4 Listes en cascade

On peut choisir d'afficher une liste dont le contenu varie en fonction des valeurs sélectionnées dans une autre liste. L'option Cascade permet le paramétrage de telles listes en spécifiant l'attribut parent et l'attribut enfant sur lequel

effectuer le filtre d'affichage.

Attributs de filtrage et Signe de comparaison

Après avoir défini quel est l'élément parent, il faut définir le champ enfant sur lequel repose l'ascendance ainsi que le signe de comparaison sur lequel doit reposer la comparaison entre les champs liant la table parent à la table enfant.

Les signes de comparaison sont :

- = Egalité parfaite entre le champ parent et le champ enfant
- > Ne seront affichés dans la liste enfant que les enregistrements dont la valeur est supérieure à l'attribut de filtre parent.
- < Ne seront affichés dans la liste enfant que les enregistrements dont la valeur est inférieure à l'attribut de filtre parent.

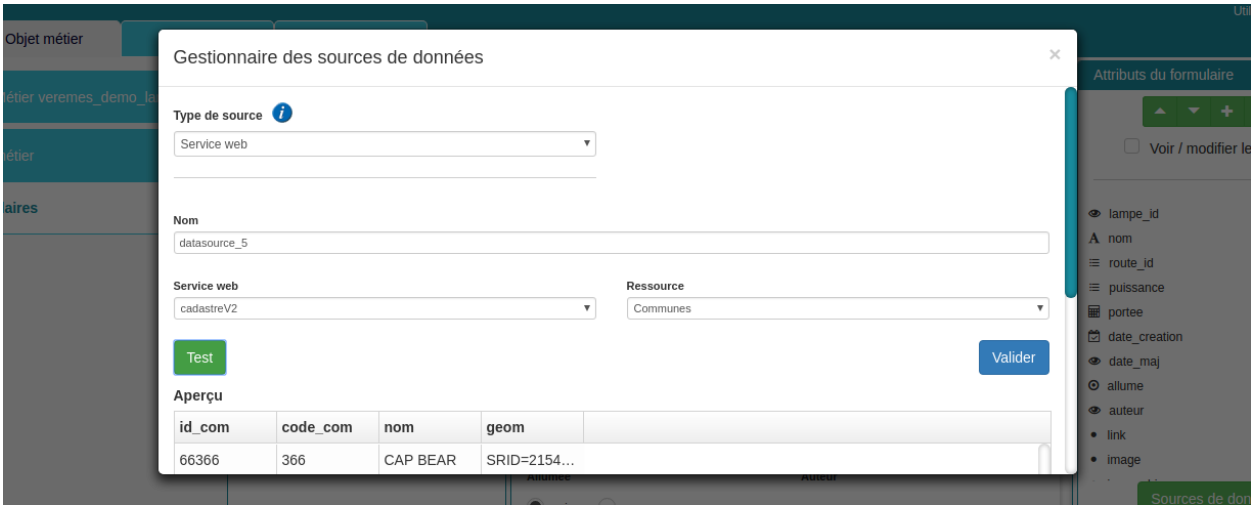
**Attendre le parent **

Pour forcer l'affichage de la liste des éléments enfants uniquement lorsqu'un élément parent est sélectionné, il faut cocher la case "Attendre le parent". Ainsi, si aucun élément parent n'est sélectionné, aucun élément enfant n'apparaîtra dans la liste.

3. Source de données de type service web

On peut exploiter une ressource d'un service web précédemment créé, afin d'effectuer des requêtes complexes. On peut également se servir d'un services de l'application.

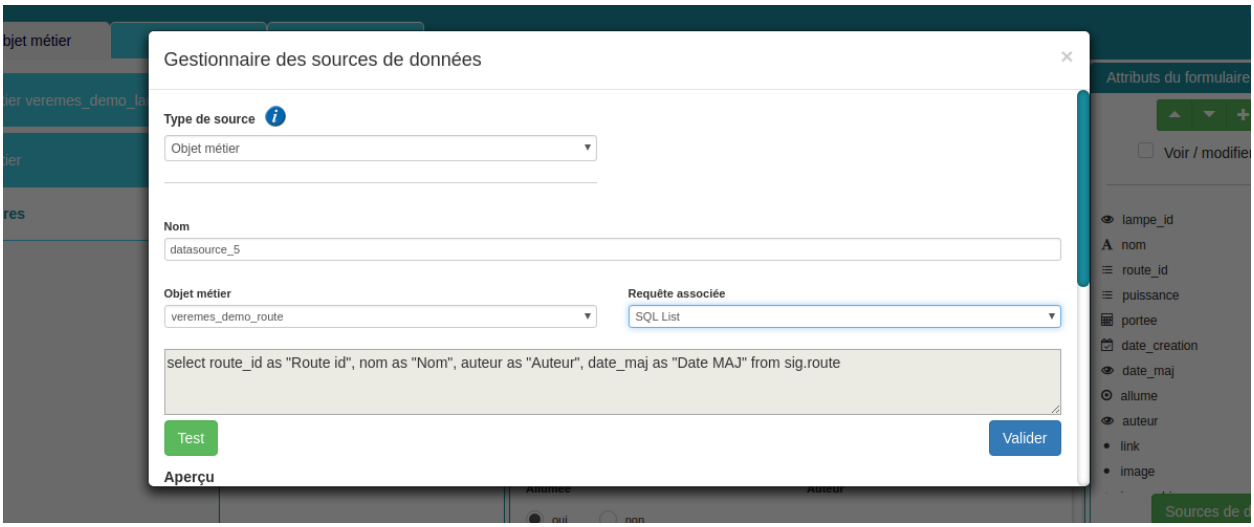
Le type de source « *Service web* » effectue une requête de type « *GET* » à la ressource sélectionnée.



4. Source de données de type objet métier

Il est également possible d'interroger directement un objet métier selon une des trois solutions suivantes :

- **Form :** renvoie l'ensemble des colonnes de la table associée à l'objet métier
- **SQL Summary :** renvoie de résultat de la requête définie par SQL Summary
- **SQL List :** renvoie de résultat de la requête définie par SQL List



5. Source de données de type base de données externe

Plus complexe mais plus puissant, ce type de source permet d'interroger des bases de données d'un serveur externe selon un login et un mot de passe fourni.

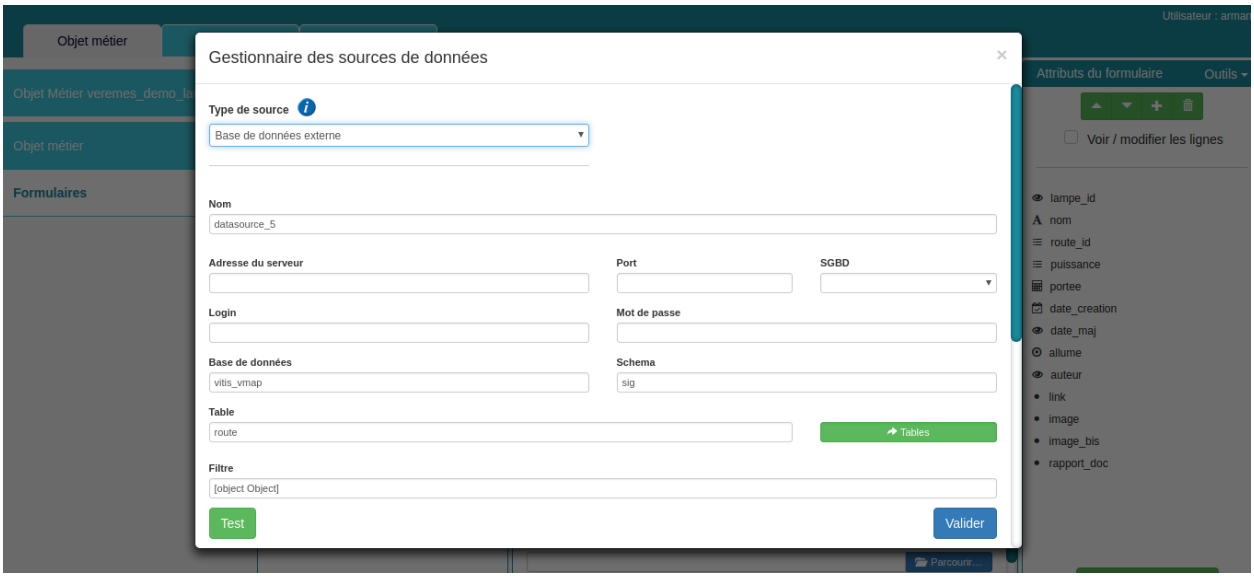
Après avoir saisi le nom de la nouvelle source de données, saisir les paramètres de connexion à la base de données :

- serveur
- port
- sgbd
- login
- mot de passe
- tables. Le bouton "Tables" permet d'afficher la liste des tables de la base et de la sorte de s'assurer de la réussite de la connexion.

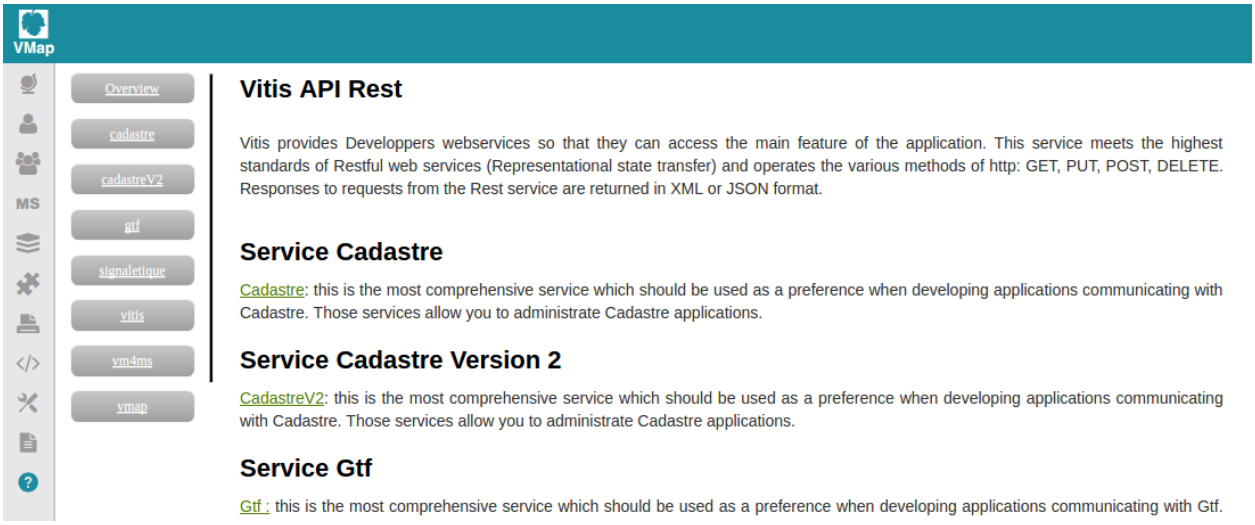
Il est également possible de filtrer les données à importer via une clause de type Where, saisie dans le champs Filtre.

Le bouton "Valider" permet de fermer la fenêtre en cours et revenir à liste des sources de données. Cliquer, à nouveau, sur Valider pour fermer le gestionnaire de source de données et revenir au studio.

Important : les login et mot de passe renseignés doivent être publics car les utilisateurs finaux pourraient avoir accès à cette information.



4.9.2 Services web



1. Définition

Les web services sont la partie back-end de l'application, ils se composent de plusieurs ressources qui permettent au client d'interroger la base de données, de lire/modifier des fichiers et d'effectuer des opérations sur la machine physique du serveur.

Dans vMap et autres produits Veremes, ils sont mis en place par une API-REST, ce qui signifie que l'on accède aux données selon des règles bien spécifiques.

Exemple de requête permettant de lister les cartes vMap :

```
https://corbieres/vmap_rest/vmap/maps
```

Exemple de requête permettant de voir les informations de la carte ayant pour identifiant "15" :

```
https://corbieres/vmap_rest/vmap/maps/{15}
```

L””API-REST retourne au client, un résultat JSON ou XML contenant les informations demandées :


```
{
  "maps": [
    {
      "theme_name": "Thème Géobretagne",
      "theme_description": "Cartes Géobretagne avec fond OSM",
      "crs_name": "[EPSG:2154]-RGF93.Lambert-93",
      "map_id": 15,
      "crs_id": "EPSG:2154",
      "name": "Carte OSM Géobretagne",
      "description": "Carte Geobretagne avec un fond osm",
      "extent": "140807|6725441|303007|6799494",
      "catalog_index": 3,
      "thumbnail": "https://corbieres/vmap_ws_data/vitis/vmap_admin_map_vmap_admin_
↪map/documents/15/thumbnail/geobret.png?d=1499068782",
      "maptheme_id": 1,
      "groups": "",
      "groups_label": ""
    }
  ],
  "status": 1
}
```


2. Utilisation

2.1. En-têtes

Il y a diverses en-têtes essentielles à l”utilisation des ressources.

2.1.1. Accept

Valeurs possibles :

- application/json
- application/xml
- application/x-vm-json

Définition

L”en-tête détermine le format de réponse demandé par le client. Les formats application/json et application/xml retournent un objet possédant un tableau qui porte le nom de la ressource (dans l”exemple ci-dessus, il s”agit de « maps »). Le format application/x-vm-json diffère en donnant comme nom du tableau « data », permettant de faire des requêtes génériques par le client.

2.1.2. Token

Le token de connexion identifie l”utilisateur de l”application, c”est grâce à lui que la ressource identifie si le demandeur possède les droits suffisants pour avoir un résultat, et c”est par son intermédiaire que se font les connexions à la base

de données.

Pour des raisons de sécurité il est strictement interdit de passer le token en tant que paramètre dans l'URL. Il faut donc le passer en-tête : si une personne malveillante a accès au réseau (man in the middle), elle pourrait alors voir ce token et donc usurper l'identité d'un autre utilisateur.

2.1.3. X-HTTP-Method-Override

Lorsque l'on utilise régulièrement l'API-REST, il est possible que l'on soit confronté à des problèmes de longueur d'URL : à partir d'un certain nombre de caractères, les navigateurs refusent d'exécuter la requête et affichent l'erreur suivante :

```
414 URI Too Long
```

Pour palier à cette contrainte, une en-ête X-HTTP-Method-Override a été mise en place pour envoyer une requête de type POST avec des paramètres figurant dans le body (sans limite de taille) et interprétables comme des requêtes GET :

```
General
    Request Method:POST

Request Headers
    X-HTTP-Method-Override: GET
```

2.2. Paramètres génériques

2.2.1. order_by

Permet de définir l'ordre d'affichage lorsqu'il y a plusieurs données. Par défaut il vaut l'identifiant de la ressource.

2.2.2. sort_order

Couplé au paramètre « order_by », il permet de définir l'ordre avec les valeurs suivantes :
- asc : ordre ascendant
- desc : ordre descendant

2.2.3. limit

Si le paramètre limit est fourni, alors le tableau retourné se limite à « n » éléments.

2.2.4. offset

Souvent couplé avec les paramètres « limit » et « order_by », il peut permettre, par exemple, d'effectuer une pagination sur une liste.

2.2.5. attributs

Définit les attributs qui seront retournés par le client. Pour les renseigner, il faut écrire ces attributs en les séparant par le caractère « | ».

2.2.6. distinct

True/false permet de distinguer les valeurs résultantes.

2.2.7. filter

Donne la possibilité à l'utilisateur de filtrer les données. Il faut écrire un objet JSON composé de **relations** et d'"**opérateurs**.

2.2.7.1. Relations

Les relations définissent le type de condition à utiliser selon la structure JSON suivante :

```
{
    "relation": "AND",
    "operators":[{
        "..."
    }, {
        "..."
    }]
}
```

Dans cet exemple, on demande d'"ajouter les filtres définis par les opérateurs selon la relation « AND ». On peut également utiliser une relation « OR ».

Il est aussi possible de faire dans une même requête du AND et du OR en incorporant une relation comme ci c'"était un opérateur :

```
{
    "relation": "AND",
    "operators":[{
        "..."
    }, {
        "relation": "OR",
        "operators": [{
            "..."
        }, {
            "..."
        }]
    }]
}
```

Ainsi, on obtient une requête constituée de AND et de OR (voir l'"exemple ci-après).

2.2.7.2. Opérateurs

Plus simples à comprendre, les opérateurs se composent de trois ou quatre arguments :

- **column** : nom de la colonne sur laquelle appliquer le filtre
- **value** : valeur du filtre
- **compare_operator** : type de comparaison (»= », « != », « <> », « >= », « <= », « > », « < », « IN », « NOT IN », « IS NULL », « IS NOT NULL », « LIKE », « INTERSECT »)
- **compare_operator_options (optionnel)** : ajoute des options suivant le type de compare_operator.

La structure est la suivante :

```
{
    "column": "...",
    "compare_operator": "...",
    "value": "...",
    "compare_operator_options": {
        "..." : "..."
    }
}
```

2.2.7.3. Exemples

Pour être plus parlant, voici quelques exemples avec leur équivalent sous forme SQL.

En utilisant une relation AND on peut filtrer sur plusieurs opérateurs :

```
{
    "relation": "AND",
    "operators":[{
        "column": "auteur",
        "compare_operator": "=",
        "value": "Laurent"
    }, {
        "column": "allume",
        "compare_operator": "=",
        "value": "true"
    }, {
        "column": "route_id",
        "compare_operator": "=",
        "value": 10
    }]
}
```

Équivalent SQL

```
auteur='laurent' AND allume='true' AND route_id=10
```

Si un seul opérateur est utilisé, alors il n"est pas nécessaire de renseigner de relation :

```
{
    "column":"auteur",
    "compare_operator":"=",
    "value":"laurent"
}
```

Équivalent SQL

```
auteur='laurent'
```

En utilisant des relations imbriquées, on peut effectuer des filtres complexes :

```
{
    "relation": "AND",
    "operators":[{
        "column":"auteur",
        "compare_operator":"=",
        "value":"laurent"
    }, {
        "relation": "OR",
        "operators": [{
            "column":"allume",
            "compare_operator":"=",
            "value":"true"
        }, {
            "column":"route_id",
            "compare_operator":"=",
            "value":10
        }]
    }]
}
```

Équivalent SQL

```
auteur='laurent' AND (allume='true' OR route_id=10)
```

On peut utiliser « compare_operator » = « IN » en utilisant des valeurs situées dans un tableau :

```
{
    "relation": "AND",
    "operators":[{
        "column":"auteur",
        "compare_operator":"=",
        "value":"laurent"
    }, {
        "relation": "OR",
        "operators": [{
            "column":"allume",
            "compare_operator":"=",
            "value":"true"
        }, {
            "column":"route_id",
            "compare_operator":"IN",
            "value":[5,10]
        }]
    }]
}
```

Équivalent SQL

```
auteur='laurent' AND (allume='true' OR route_id IN (5, 10))
```

Il est possible d"utiliser « compare_operator » = « LIKE » avec des valeurs suivies ou précédées du caractère « % » :

```
{
    "column":"auteur",
```

(suite sur la page suivante)

(suite de la page précédente)

```
    "compare_operator":"LIKE",
    "value":"laur%"
}
```

Équivalent SQL

```
auteur LIKE 'laur'%
```

En utilisant « compare_operator_options.case_insensitive » sur un type « LIKE », on peut rendre le filtre insensible à la casse :

```
{
    "column":"auteur",
    "compare_operator":"LIKE",
    "compare_operator_options":{
        "case_insensitive": true
    },
    "value":"%laur%"
}
```

Équivalent SQL

```
LOWER(auteur) LIKE LOWER('%lAur%')
```

Utilisation de « IS NOT NULL »

```
{
    "column": "nom",
    "compare_operator": "NOT NULL"
}
```

Équivalent SQL

```
nom IS NOT NULL
```

On peut effectuer des intersections géométriques utilisant PostGIS :

```
{
    "column":"geom",
    "compare_operator":"intersect",
    "value":"SRID=3857;POINT(349627.744690664 5237367.243157785)"
}
```

L'option « source_proj » utilisée ici n'est pas obligatoire mais conseillée si on connaît le système de projection de la table :

```
{
    "column":"geom",
    "compare_operator":"intersect",
    "compare_operator_options":{
        "source_proj": 2154
    },
    "value":"SRID=3857;POINT(349627.744690664 5237367.243157785)"
}
```

On peut utiliser un buffer lors de l'"intersection, et même spécifier sur quel type de géométrie s"applique le buffer :

```
{
    "column":"geom",
    "compare_operator":"intersect",
    "compare_operator_options":{
        "source_proj":"2154",
        "intersect_buffer":8.31909066896183,
        "intersect_buffer_geom_type":"point|line"
    },
    "value":"SRID=3857;POINT(349643.2709620344 5237383.963757724)"
}
```

3. Exemple de création d"un web service et de ses ressources

Dans une installation classique, les web services se trouvent sous forme de dossiers dans le répertoire vmap/vas/rest/ws. Dans ces dossiers se trouvent les fichiers indispensables ainsi que les ressources des web services.

Dans cet exemple, nous allons créer un web service « customWS » dans lequel créer une ressource « villes ».

3.1. Création du dossier et des fichiers indispensables

Parmi les fichiers indispensables, se trouvent les fichiers suivants :

- **overview.phtml** : permet d"afficher la ressource dans la page d"aide au développement
- **CustomWS.class.inc** : classe mère du projet
- **CustomWS.class.sql.inc** : fichier contenant les requêtes SQL du projet. Il doit contenir au moins les requêtes « Définition des requêtes de l"api Vitis ».

3.2. Création de la première ressource

Dans cet exemple, nous cherchons à créer la ressource « villes » qui permet de lister les villes contenues dans la table « f_villes_l93 » installée par défaut avec vMap.

Chaque ressource est définie par deux fichiers PHP :

- l"un pour la définition unitaire d"un objet (ici Ville.class.inc)
- l"autre pour agir sur une liste complète d"objets (ici Villes.class.inc). Le « s » (obligatoire) permet de faire la différencie entre la liste et l"unitaire.

3.2.1 La ressource unitaire (Ville.class.inc)

Il s"agit d"une classe PHP qui doitau moins contenir les éléments suivants :

3.2.1.1 Inclusions des fichiers

```
require_once 'CustomWS.class.inc';
require_once __DIR__ . '/../../class/vitis_lib/Connection.class.inc';
```

Inclusion de la classe mère du web service ainsi que la classe permettant d'effectuer des connexions à la base de données.

3.2.1.2 Classe

```
class Ville extends CustomWS {
    ...
}
```

Définition de la classe Ville.

3.2.1.3 Constructeur

```
/**
 * construct
 * @param type $aPath url of the request
 * @param type $aValues parameters of the request
 * @param type $properties properties
 * @param type $oConnection connection object
 */
function __construct($aPath, $aValues, $properties, $oConnection) {
    $this->aValues = $aValues;
    $this->aPath = $aPath;
    $this->aProperties = $properties;
    $this->oConnection = $oConnection;
    $this->aSelectedFields = Array(...);
}
```

Constructeur de la classe. La variable **$this->aSelectedFields** définit attributs à afficher lors des requêtes.

3.2.1.4 Fontion GET

```
/**
 * @SWG\Get(path="/villes/{code}",
 *   tags={"villes"},
 *   summary="Get Ville",
 *   description="Request to get Ville by id",
 *   operationId="GET",
 *   produces={"application/xml", "application/json", "application/x-vm-json"},
 *   @SWG\Parameter(
 *     name="token",
 *     in="query",
 *     description="user token",
 *     required=true,
 *     type="string"
 *   ),
```

(suite sur la page suivante)

(suite de la page précédente)

```
 *   @SWG\Parameter(
 *     name="code",
 *     in="path",
 *     description="",
 *     required=true,
 *     type="integer"
 *   ),
 *   @SWG\Response(
 *         response=200,
 *         description="Poprerties Response",
 *         @SWG\Schema(ref="#/definitions/villes")
 *     )
 *  )
 */

/**
 * get informations about villes
 */
function GET() {
    require $this->sRessourcesFile;
    $this->aFields = $this->getFields('sig', 'f_villes_l93', 'code');
}
```

Deux commentaires se trouvent au dessus de cette fonction :

- le premier est utilisé par swagger pour générer la documentation en ligne interactive
- le second est le commentaire de la fonction utilisée pour décrire aux développeurs ce que fait la fonction.

Les paramètres décrits dans les commentaires swagger passés dans le chemin l'"URL par la relation in= »path »(comme ici « *code* ») sont disponibles via la variable **$this->aPath**.

Les paramètres décrits dans les commentaires swagger passés dans l'"URL par la relation in= »query » (comme ici « *token* ») sont disponibles via la variable **$this->aValues**.

La ligne **require $this->sRessourcesFile** permet de récupérer le contenu du fichier *CustomWS.class.sql.inc*.

La fonction **$this->getFields** permet de récupérer en base de données, les informations la ville en question en utilisant le paramètre « *code* » passé dans l'"URL.

Le résultat stocké dans **$this->aFields** est retourné lors de la requête http.

3.2.2 La ressource multiple (Villes.class.inc)

3.2.2.1 Inclusions des fichiers

```
require_once 'CustomWS.class.inc';
require_once 'Ville.class.inc';
require_once __DIR__ . '/../../class/vitis_lib/Connection.class.inc';
require_once __DIR__ . '/../../class/vmlib/BdDataAccess.inc';
```

Require de la classe mère du web service ainsi que la classe unitaire et les fichiers permettant l'"utilisation de la base de données.

3.2.2.2 Classe

```
class Villes extends CustomWS {
    ...
}
```

Définition de la classe Villes

3.2.2.3 Constructeur

```
/**
 * construct
 * @param type $aPath url of the request
 * @param type $aValues parameters of the request
 * @param type $properties properties
 */
function __construct($aPath, $aValues, $properties) {
    $this->aValues = $aValues;
    $this->aPath = $aPath;
    $this->aProperties = $properties;
    $this->oConnection = new Connection($this->aValues, $this->aProperties);
    $this->aSelectedFields = Array(...);
}
```

Contrairement à la ressource unitaire, la connexion est instanciée.

3.2.2.4 Fontion GET

```
/**
 * @SWG\Get(path="/villes",
 *   tags={"Villes"},
 *   summary="Get Villes",
 *   description="Request to get Villes",
 *   operationId="GET",
 *   produces={"application/xml", "application/json", "application/x-vm-json"},
 *   @SWG\Parameter(
 *     name="token",
 *     in="query",
 *     description="user token",
 *     required=true,
 *     type="string"
 *   ),
 * @SWG\Parameter(
 *     name="order_by",
 *     in="query",
 *     description="list of ordering fields",
 *     required=false,
 *     type="string"
 *   ),
 * @SWG\Parameter(
 *     name="sort_order",
 *     in="query",
 *     description="sort order",
```

(suite sur la page suivante)

(suite de la page précédente)

```
 *     required=false,
 *     type="string"
 *   ),
 * @SWG\Parameter(
 *     name="limit",
 *     in="query",
 *     description="number of element",
 *     required=false,
 *     type="integer",
 *     default="4"
 *   ),
 * @SWG\Parameter(
 *     name="offset",
 *     in="query",
 *     description="index of first element",
 *     required=false,
 *     type="string"
 *   ),
 * @SWG\Parameter(
 *     name="attributs",
 *     in="query",
 *     description="list of attributs",
 *     required=false,
 *     type="string"
 *   ),
 * @SWG\Parameter(
 *     name="filter",
 *     in="query",
 *     description="filter results",
 *     required=false,
 *     type="string"
 *   ),
 * @SWG\Parameter(
 *     name="distinct",
 *     in="query",
 *     description="delete duplicates",
 *     required=false,
 *     type="boolean"
 *   ),
 *   @SWG\Response(
 *         response=200,
 *         description="Poprerties Response",
 *         @SWG\Schema(ref="#/definitions/villes")
 *     )
 *  )
 */

/**
 * get Villes
 * @return the array of objects
 */
function GET() {
    $aReturn = $this->genericGet('sig', 'f_villes_l93', 'code');
    return $aReturn['sMessage'];
}
```

Tous les paramètres génériques sont listés dans les commentaires swagger, et sont disponibles sur les variables **

$this->aPath ** et ** $this->aValues **.

Ici c"est la fonction **genericGet()** qui est utilisée et la fonction retourne du texte.

3.3. Ressource complexe avec executeWithParams()

Nous avons vu ci-dessus comment créer une ressource standard qui permet d"aller chercher en base de données les informations d"une table et de les renvoyer.

Imaginons que l"on veuille dans la classe Ville, faire une deuxième requête en base de données (cette fois définie dans *CustomWS.class.sql.inc*) pour aller chercher les monuments associés à la ville.

CustomWS.class.sql.inc :

```
$aSql['getVilleMonuments'] = "SELECT * FROM sig.f_monuments WHERE \"code\"=[sCode]";
```

Ville.class.inc :

```
function GET() {
    require $this->sRessourcesFile;
    $this->aFields = $this->getFields('sig', 'f_villes_l93', 'code');

    $aSQLParams = array(
        'sCode' => array('value' => $this->aFields['code'], 'type' => 'string')
    );
    $oResult = $this->oConnection->oBd->executeWithParams($aSql['getVilleMonuments'],
↪$aSQLParams);
    if (gettype($oResult) == 'object') {
        $this->aFields['monuments'] = Array();
        while ($aLigne = $this->oConnection->oBd->ligneSuivante($oResult)) {
            array_push($this->aFields['monuments'], $aLigne);
        }
    }
}
```

Ci-dessus, la fonction **executeWithParams()** permet d'exécuter une requête SQL. Le résultat est alors rajouté dans $this->aFields["monuments"].

4. Fonction executeWithParams()

4.1 Définition

Pour effectuer des requêtes SQL en PHP, il est impératif d"utiliser la fonction executeWithParams() qui va exécute une requête avec un tableau de paramètres passé en option.

Il ne faut surtout pas concaténer des variables à une requête SQL au risque d"exposer l"application à une faille de type SQLi

Il faut écrire dans la requête une balise contenant le nom de la variable, et fournir un tableau de variables à execute-WithParams().

Les différents formats sont :

- **string**, **number**, **integer** : pour les valeurs de variables à passer entre simple quotes.
- **group** : pour les valeurs à passer entre simple quotes et séparées par des virgules.
- **geometry** : pour les géométries à passer entre simple quotes.
- **quoted_string** : comme string mais pour intégrer des caractères spéciaux ex : "ma lampe%".

— **column_name**, **schema_name**, **table_name** : pour les noms de colonnes, tables, schémas. Attention car pour ces types de paramètre executeWithParams() ne s"occupera pas des quotes, il faut donc les mettre à l"avance ex : SELECT « [column_name] » FROM [schema_name].[table_name] WHERE table="[table_name]".

4.2 Exemples

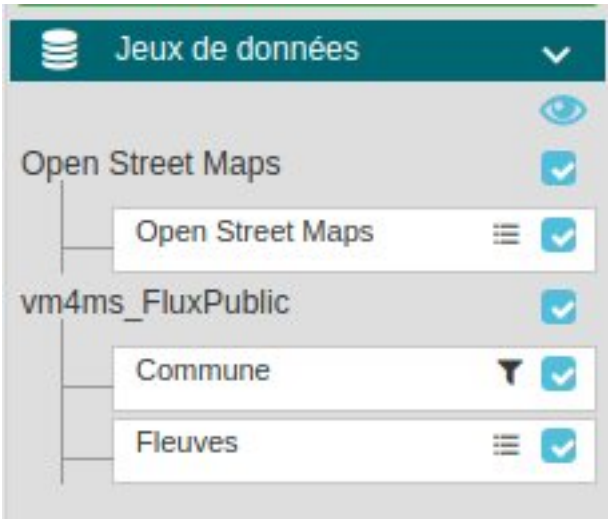
```
$aSQLParams = array(
    'sSchema' => array('value' => $this->aProperties['schema_vmap'], 'type' =>
↪'column_name'),
    'sGroups' => array('value' => $sGroups, 'type' => 'group')
);
$sSql = "SELECT map_id, group_id FROM [sSchema].map_group WHERE \"group_id\" in␣
↪([sGroups])";
$oResult = $this->oConnection->oBd->executeWithParams($sSql, $aSQLParams);
```

```
$aSQLParams = array(
    'sSchema' => array('value' => $this->aProperties['schema_vmap'], 'type' =>
↪'column_name'),
    'sMapId' => array('value' => $map_id, 'type' => 'number')
);
$sSql = "SELECT * FROM [sSchema].map_layer WHERE \"map_id\" = [sMapId]";
$oResult = $this->oConnection->oBd->executeWithParams($sSql, $aSQLParams);
```

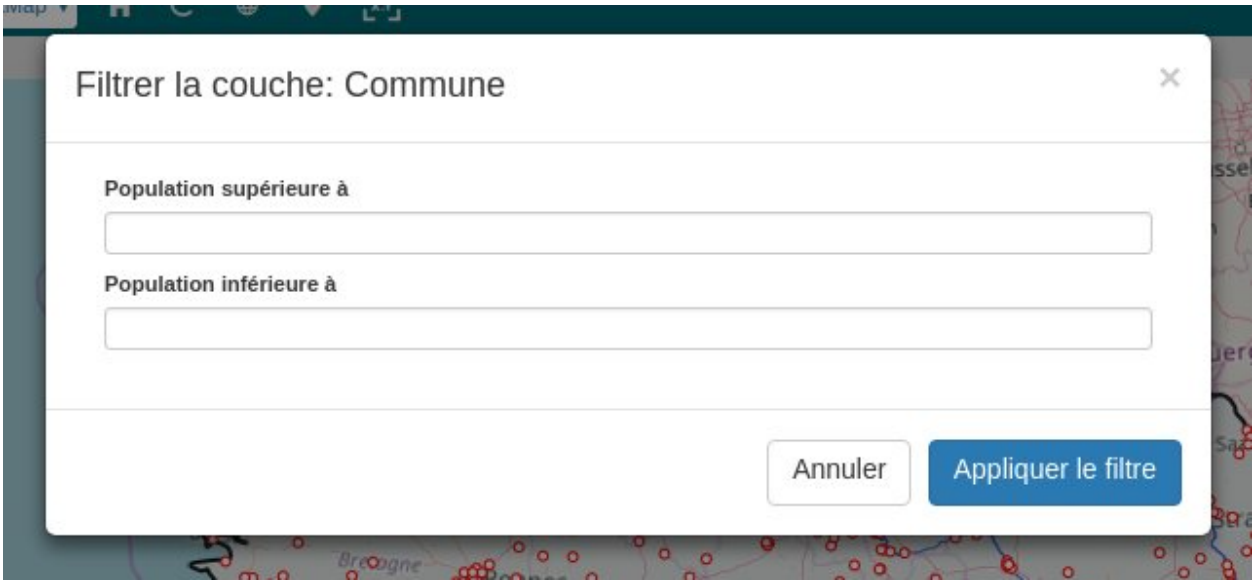
4.9.3 Filtres sur les couches

Il est possible d"ajouter des formulaires de filtre sur les couches dans le but de visualiser certaines données uniquement. Alors les utilisateurs du mode cartographique auront accès à un bouton faisant apparaître le formulaire de filtrage, sur la carte « Carte de France » créée par défaut lors de l"installation de l"application un filtre est mis en place à titre d"exemple.

Sur la couche « Commune » un bouton de filtre apparaît



En cliquant dessus le formulaire suivant est affiché dans une fenêtre modale



L'utilisateur peut alors filtrer la couche pour afficher uniquement les communes ayant une population comprise dans la fourchette saisie par l'utilisateur, ainsi si on saisit 50000 à « Population suppérieure à » nous obtenons la carte suivante :

Pour ce faire il faudra effectuer plusieurs étapes détaillées ci-dessous.

1 - Écrire le filtre sur la couche Mapserver

Après avoir saisit le filtre, vMap va ajouter aux URL permanentant de récupérer les tuiles les paramètres saisis, dans le module Mapserver il est possible de récupérer et d'utiliser ces valeurs. Bien évidemment cette opération est également fonctionnelle si vous utilisez un autre générateur de flux WMS.

Sur le module Mapserver il faudra se rendre dans la définition de la couche pour y renseigner un filtre, nous allons détailler l'exemple des communes, mais si vous voulez plus de détails vous trouverez toute l'information disponible sur la documentation Mapserver http ://mapserver.org/fr/cgi/runsub.html#filters

Dans notre exemple voici ce qui est écrit :

```
FILTER ([pop90] > '%pop90_sup%' and [pop90] < '%pop90_inf%')
VALIDATION
    'pop90_sup' '^[0-9]*$'
    'default_pop90_sup' '0'
    'pop90_inf' '^[0-9]*$'
    'default_pop90_inf' '100000000'
END
```

Dans la balise FILTER on écria la condition à respecter, les noms des colonnes devrons être écrites entre crochets ex : [pop90] et les noms des attributs récupérés à travers le l'URL (c'est à dire le formulaire de filtre) devront être écris entre pourcentages ex : %pop90_sup%

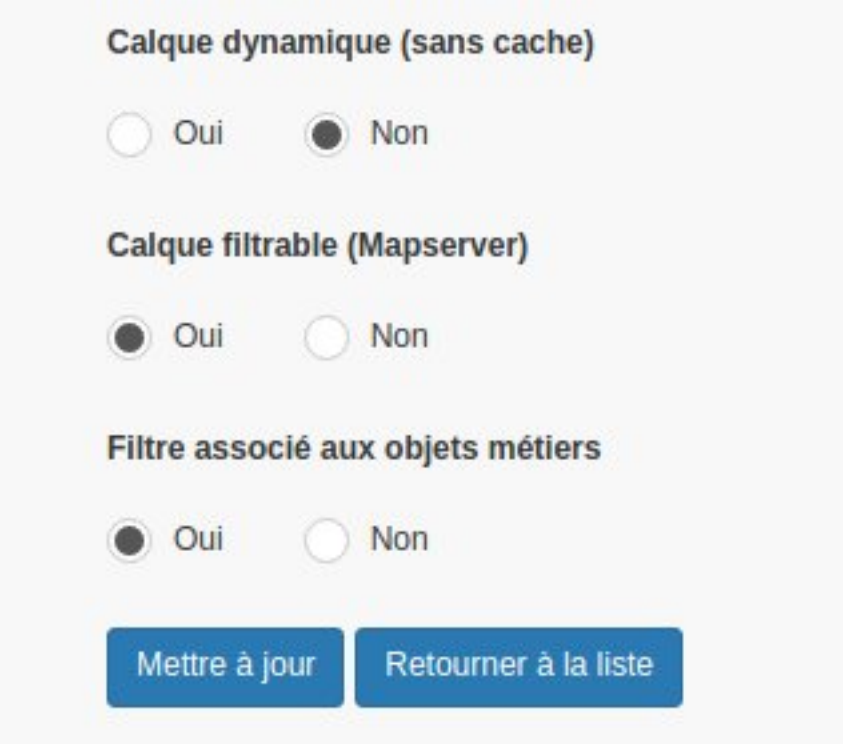
La balise VALIDATION est obligatoire et on doit y écrire pour chaque attribut récupéré à travers l'URL (c'est à dire le formulaire de filtre) une expression régulière empéchant les personnes mal intentionnées de faire des injections SQL, dans notre exemple nous avons écrit "^[0-9]*$" ce qui signifie autant de chiffres entre 0 et 9 que souhaité. **Pour que la carte affiche des valeurs lorsque les filtres sont vides** il faudra définir des valeurs par défaut en écrivant default_[nom de votre attribut], dans notre exemple par défaut la carte filtre les villes entre 0 et 100000000 d'habitants.

On peut également utiliser la balise **FILTERITEM** pour faire un filtre plus générique :

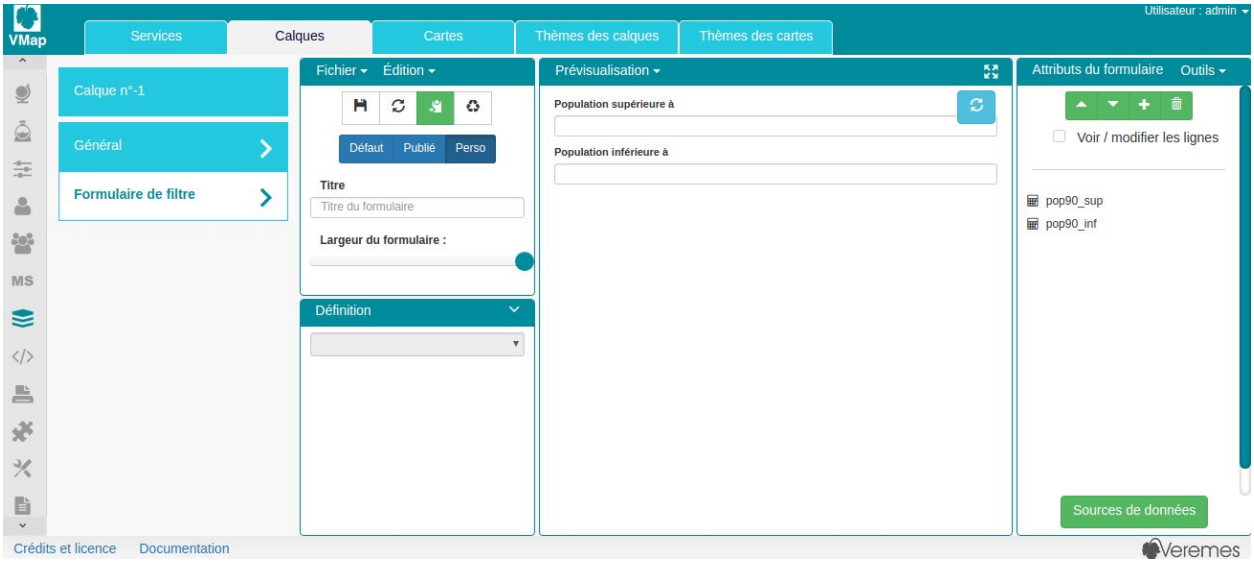
```
FILTERITEM "id_com"
FILTER /%id_com%/
VALIDATION
    "id_com" "^[a-zA-Z0-9_|]*$"
    "default_id_com" "^[a-zA-Z0-9_|]*$"
END
```

2 - Mettre en place un formulaire de filtre

Pour mettre en place le formulaire il faudra aller dans la partie **calques** et mettre Oui à « Calque filtrable (Mapserver) »



Après avoir mis à jour le calque, une nouvelle section « Formulaire de filtre » va apparaître dans lequel vous pourrez mettre en place votre filtre.



4.9.4 Autres exemples d"utilisation des filtres dans vMap

1 - Filtre textuelle non obligatoire et fonctionnement similaire au LIKE en SQL

Dans le studio il faudra définir un champ de type texte ou une liste déroulante avec une source de données adaptée.

Il faudra définir votre filtre pour qu"il utilise une valeur par défaut qui permettra de court-circuiter le filtre. (Dans le cas présent **"empty"**)

Si vous voulez un fonctionnement plus proche d"un LIKE SQL utilisez **~** à la place de **=**. (voir exemple ci-dessous)

```
        FILTER (([type] = '%type%' or '%type%' = 'empty') and ([liketext] ~ '%liketext
↪%' or '%liketext%' = 'empty')
    VALIDATION
        'type' '^empty|value_type_1|value_type_2|value_type_3$'
        'liketext' '^empty|.{1,}$'
        'default_type' 'empty'
        'default_liketext' 'empty'
    END
```

2 - Filtre sur une colonne de type date/timestamp

Pour filtrer sur un attribut unique de type date/timestamp référez vous à la documentation de MapServer http ://map-server.org/ogc/wms_time.html.

3 - Filtre sur plusieurs colonnes de type date/timestamp

Vue que la spécification d"un champ de type date/timestamp se fait dans la partie **METADATA**, il est impossible pour MapServer de traiter deux champs de ce type en théorie.

En réalité par un moyen détourné, il est possible de gérer autant de champs date/timestamp que vous voulez.

Pour l"exemple, je vais prendre trois champs en base creation_date (date de création d"un objet, type postgres timestamp with time zone), date_debut_travaux (une date de début de travaux pour un objet métier par exemple, type timestamp with time zone), duree_travaux_jour (la durée des travaux en jours à partie de la date de début, type integer)

a - Adapter votre vue

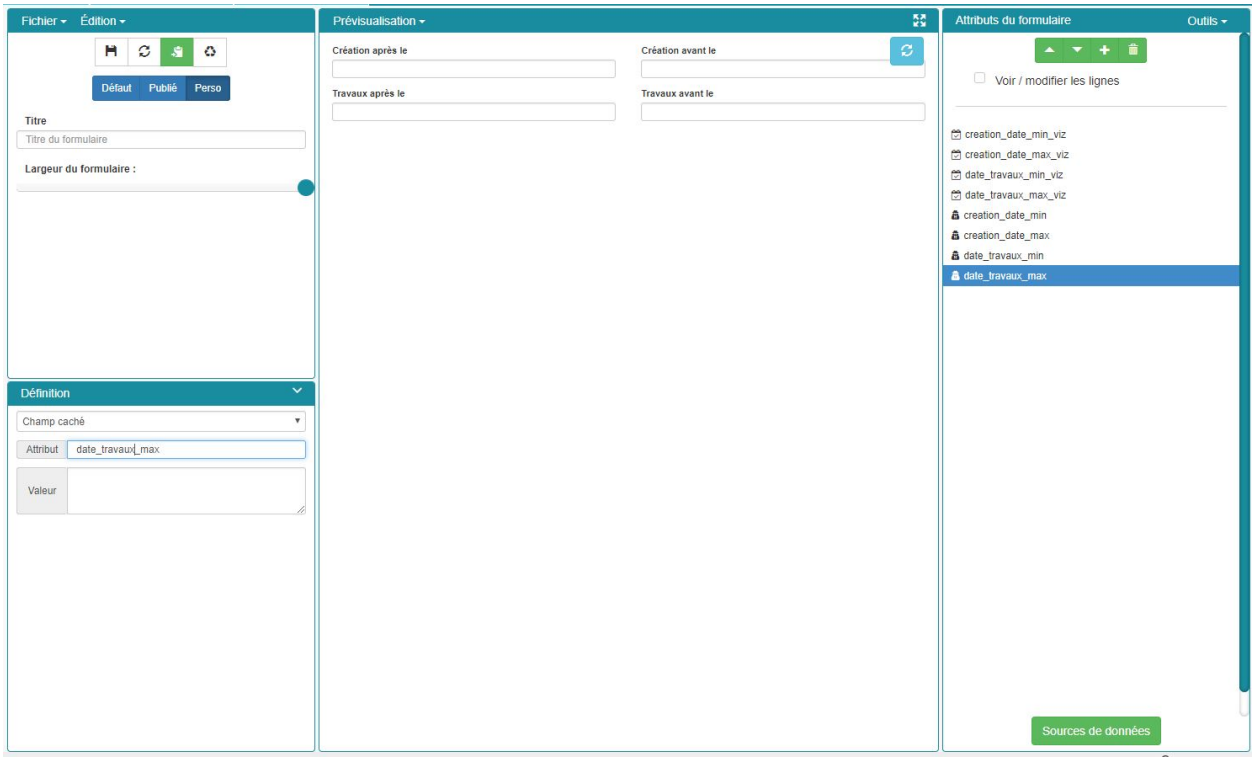
Pour permettre à MapServer de les traiter comme des entiers il va falloir que la vue retourne des entiers.

Exemple de code permettant de faire cela :

```
        date_part('epoch'::text, table.creation_date)::integer AS mstmstp_creation_
↪date,
    date_part('epoch'::text, table.date_debut_travaux)::integer AS mstmstp_date_debut_
↪travaux,
    date_part('epoch'::text, table.date_debut_travaux + table.duree_travaux_jour * '1␣
↪day'::interval)::integer AS mstmstp_date_fin_travaux
```

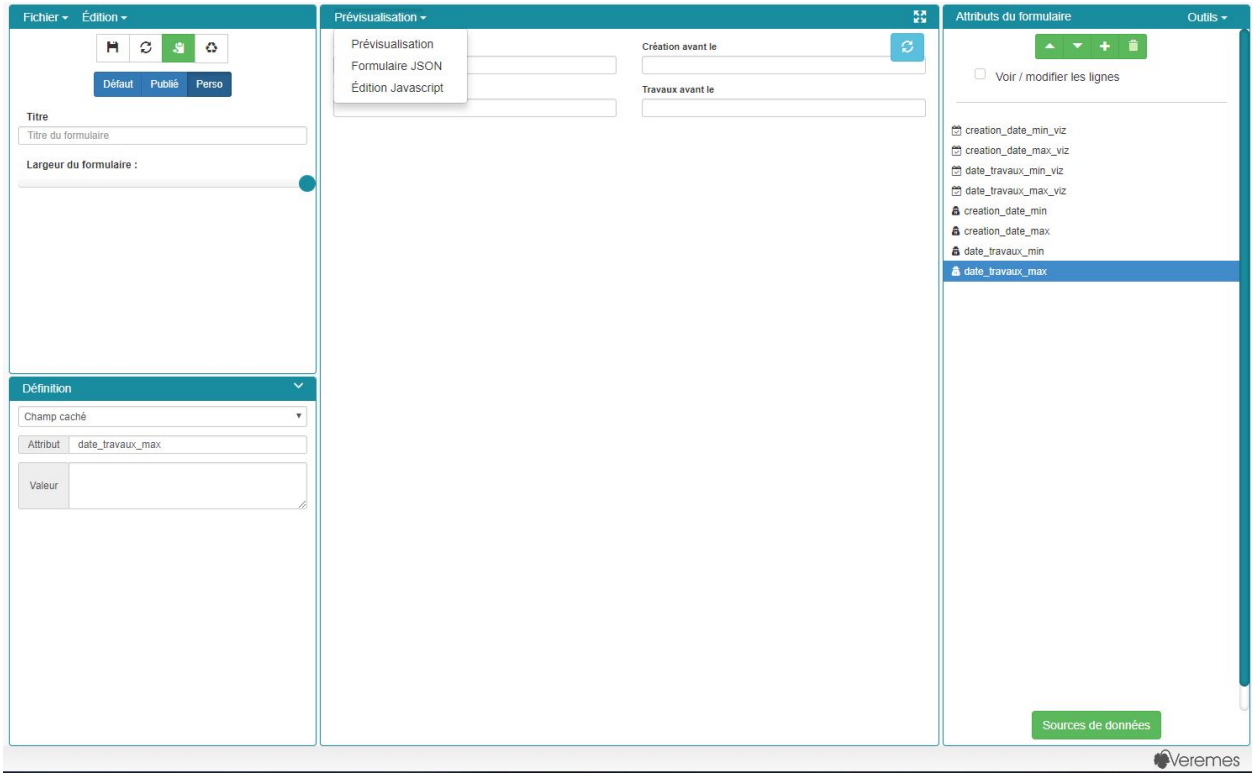
b - Adapter votre formulaire avec le studio

Ajouter les champs dates que vous voulez. Dans le cas présent on va laisser la possibilité de prendre une date au dessus, en dessous, ou d'encadrer la/les date(s). Pour chaque champ date il faut un champ caché. Le champ date va renvoyer une valeur formatée du type "DD/MM/YYYY" ou "YYYY-MM-DD", mais nous voulons un timestamp. Nous allons voir dans la prochaine partie comment utiliser un champ date pour remplir un champ caché avec un timestamp.



c - Permettre au formulaire de calculer les timestamps

Il va falloir ajouter du code javascript spécifique à ce formulaire. Pour ce faire il va falloir changer de mode d'utilisation du studio, pour pouvoir éditer du javascript.



Vous allez arriver devant un champ vide.

Voici le code permettant de remplir les champs cachés pour l'"exemple :

```
/* global angular, goog, moment, vitisApp, bootbox */

// goog fonctionne en mode décompilé mais pas en mode compilé

console.info("filtre_mapserver_couche_NOM loaded --> your functions are ready");
/*******************************************************************************
 filtre_mapserver_couche_NOM Javascript
 *******************************************************************************/

var oFormRequired = {
    "sUrl": "",
    "scope_": {},
    "toDestructor": []
};
/**
 * constructor_form
 * Fonction appelée à l'initialisation du formulaire
 * @param {type} scope
 * @param {type} s_url
 * @returns {undefined}
 */
var constructor_form = function (scope, s_url) {

    oFormRequired.sUrl = s_url;
    oFormRequired.scope_ = scope;

    var formaterStringFrom = "DD/MM/YYYY";
```

(suite sur la page suivante)

(suite de la page précédente)

```
    var formaterStringTo = "X"; // timestamp UNIX
    oFormRequired.toDestructor.push(oFormRequired.scope_.$watch("oFormValues." +␣
↪oFormRequired.scope_.sFormDefinitionName + ".creation_date_min_viz", function␣
↪(value) {
            if (typeof(value) !== "undefined" && value !== "") {
                oFormRequired.scope_["oFormValues"][oFormRequired.scope_[
↪"sFormDefinitionName"]]["creation_date_min"] = moment(value, formaterStringFrom).
↪format(formaterStringTo);
            } else {
                oFormRequired.scope_["oFormValues"][oFormRequired.scope_[
↪"sFormDefinitionName"]]["creation_date_min"] = null;
            }
        })
    );
    oFormRequired.toDestructor.push(oFormRequired.scope_.$watch("oFormValues." +␣
↪oFormRequired.scope_.sFormDefinitionName + ".creation_date_max_viz", function␣
↪(value) {
            if (typeof(value) !== "undefined" && value !== "") {
                oFormRequired.scope_["oFormValues"][oFormRequired.scope_[
↪"sFormDefinitionName"]]["creation_date_max"] = moment(value, formaterStringFrom).
↪format(formaterStringTo);
            } else {
                oFormRequired.scope_["oFormValues"][oFormRequired.scope_[
↪"sFormDefinitionName"]]["creation_date_max"] = null;
            }
        })
    );
    oFormRequired.toDestructor.push(oFormRequired.scope_.$watch("oFormValues." +␣
↪oFormRequired.scope_.sFormDefinitionName + ".tavaux_date_min_viz", function (value)
↪{
            if (typeof(value) !== "undefined" && value !== "") {
                oFormRequired.scope_["oFormValues"][oFormRequired.scope_[
↪"sFormDefinitionName"]]["travaux_date_min"] = moment(value, formaterStringFrom).
↪format(formaterStringTo);
            } else {
                oFormRequired.scope_["oFormValues"][oFormRequired.scope_[
↪"sFormDefinitionName"]]["travaux_date_min"] = null;
            }
        })
    );
    oFormRequired.toDestructor.push(oFormRequired.scope_.$watch("oFormValues." +␣
↪oFormRequired.scope_.sFormDefinitionName + ".tavaux_date_max_viz", function (value)
↪{
            if (typeof(value) !== "undefined" && value !== "") {
                oFormRequired.scope_["oFormValues"][oFormRequired.scope_[
↪"sFormDefinitionName"]]["travaux_date_max"] = moment(value, formaterStringFrom).
↪format(formaterStringTo);
            } else {
                oFormRequired.scope_["oFormValues"][oFormRequired.scope_[
↪"sFormDefinitionName"]]["travaux_date_max"] = null;
            }
        })
    );

};
/**
 * destructor_form
```

(suite sur la page suivante)

(suite de la page précédente)

```
 * Fonction appelée quand on quitte le formulaire (ne pas modifier)
 * @returns {undefined}
 */
var destructor_form = function () {
    console.log("Destructor");

    for (var i = 0; i < oFormRequired.toDestructor.length; i++) {
        oFormRequired.toDestructor[i] = undefined;
        delete oFormRequired.toDestructor[i];
    }

    //supprimer la balise script du js pour pouvoir le recharger si on revient sur le
↪formulaire plus tard
    angular.element('[src="' + oFormRequired.sUrl + '?version=' + oFormRequired.scope_
↪["oProperties"]["build"] + '"]').remove();

    oFormRequired = undefined;
    delete oFormRequired;

    constructor_form = undefined;
    delete constructor_form;
    destructor_form = undefined;
    delete destructor_form;
};
/*****************************************************************************/
```

Ce code créé une instance pour surveiller les changements des champs dates (**$watch**) et utilise une librairie intégrée à Vitis (**moment.js**) pour parser la date, la transformer en timestamp Unix et stocker le résultat dans le champ caché.

d - Intégration du filtre dans la couche MapServer

```
        FILTER (([mstmstp_date_debut_travaux] < '%travaux_date_max%' or [mstmstp_date_
↪fin_travaux] < '%travaux_date_max%') and ([mstmstp_date_debut_travaux] < '%travaux_
↪date_min%' or [mstmstp_date_fin_travaux] < '%travaux_date_min%') and ([mstmstp_
↪creation_date] < '%travaux_date_max%') and ([mstmstp_creation_date] > '%creation_
↪date_min%'))

    VALIDATION
        'creation_date_max' '^[0-9]{1,}$'
        'creation_date_min' '^[0-9]{1,}$'
        'travaux_date_max' '^[0-9]{1,}$'
        'travaux_date_min' '^[0-9]{1,}$'
        'default_creation_date_max' '10000000000000000000000000'
        'default_creation_date_min' '0'
        'default_travaux_date_max' '10000000000000000000000000'
        'default_travaux_date_min' '0'
    END
```

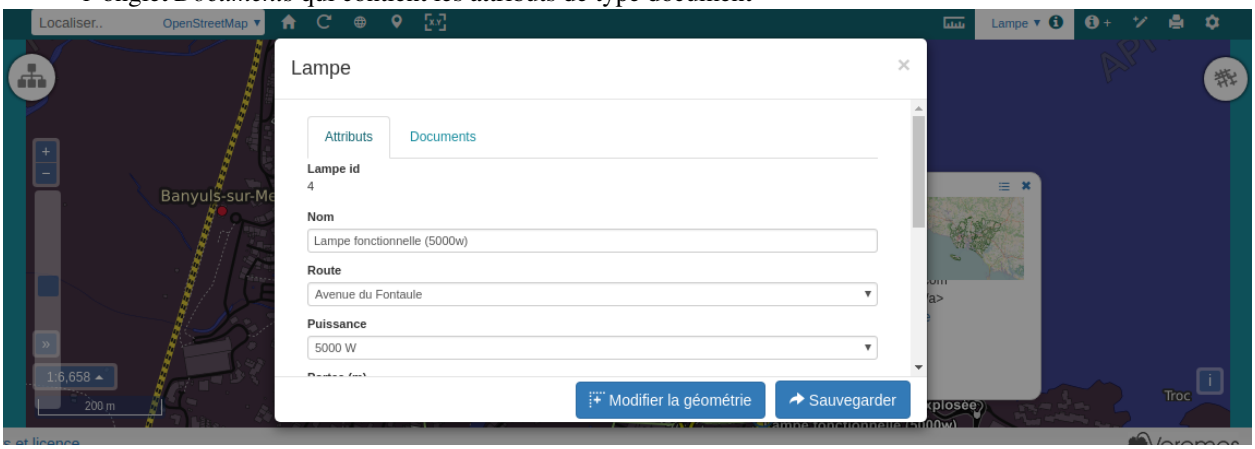
4.9.5 Exemples d”objets métier dans vMap

Ce document décrit des exemples d”utilisation du studio utiles pour comprendre son fonctionnement ainsi que l”intégralité de l”application.

1. Personnalisation d'un formulaire : répartition d'attributs sur plusieurs onglets

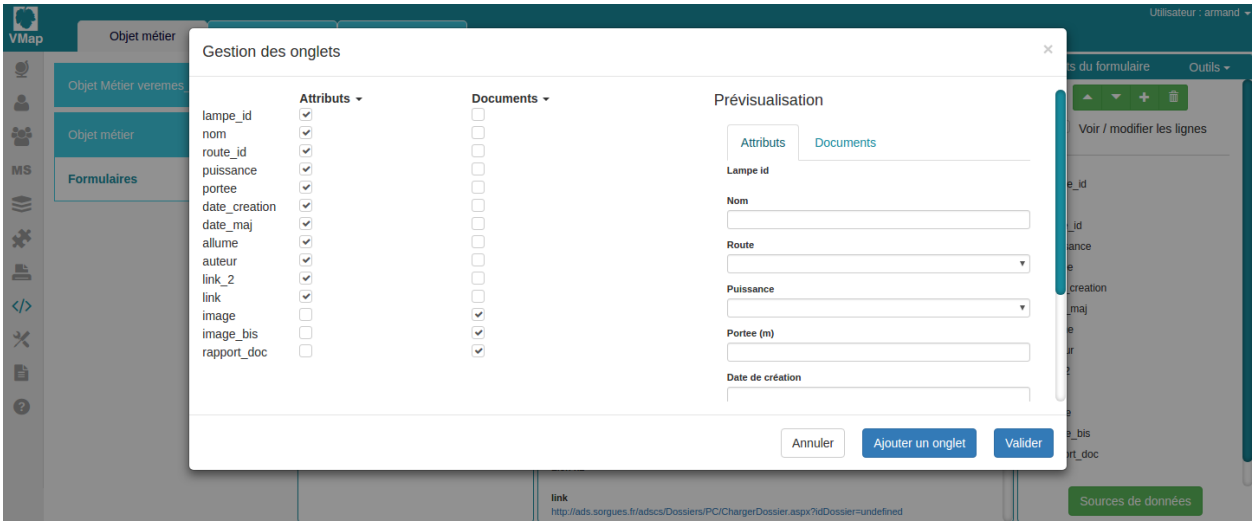
L'exemple ci-dessous illustre l'agencement d'un formulaire en deux onglets :

— l'onglet *Attributs* dans lequel les attributs de type textuel sont issus de base de données
— l'onglet *Documents* qui contient les attributs de type document



Le gestionnaire d'onglets est accessible via le bouton **Édition > Gestion des onglets** dans la zone d'administration des formulaires :

Une fois l'outil affiché, il est possible de cocher ou décocher les attributs à afficher sur les différents onglets tout en ayant un aperçu en zone de prévisualisation.



Le bouton *Ajouter un onglet* permet l'insertion de nouveaux onglets et on peut également effectuer des opérations comme renommer, supprimer ou déplacer des onglets en cliquant sur leur nom.

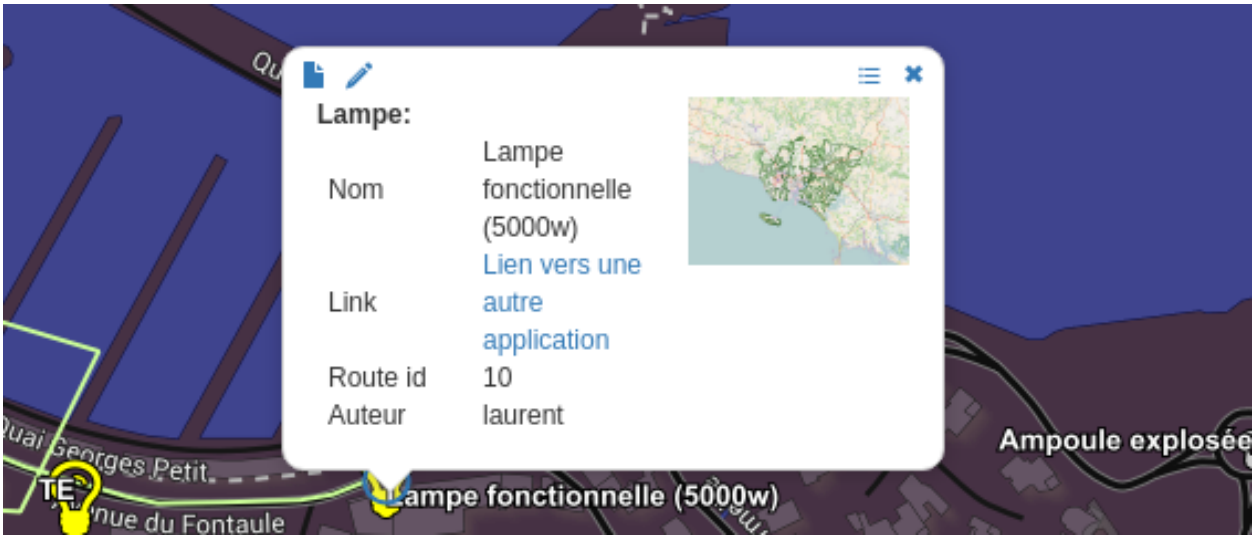
Remarque : un attribut peut se situer sur plusieurs onglets à la fois, ceci peut être utile pour afficher un label.

2. Personnalisation d'un formulaire : insertion d'un attribut de type lien

Il est souvent utile, lors de l'utilisation d'un objet métier, de mettre en place des liens vers d'autres plateformes.

Dans vMap ceci se fait à deux endroits distincts :

2.1. Dans l'info-bulle d'un objet



Lorsd de la déclaration de l'objet métier, modifier l'attribut **SQL Summary** et utiliser des balises *« bo_link »*.

Exemple :

```
select nom as "Nom", '[bo_link href="https://www.google.fr/?gws_rd=cr&
↪ei=h3hvWbHuJIORaPe3ofAG#q='||nom||'" target="_blank"]Lien vers une autre␣
↪application[/bo_link]' as "Link", route_id as "Route id", auteur as "Auteur", image␣
↪as "[bo_image]"  from sig.lampe
```

Il est possible de concaténer une des valeurs de l'enregistrement avec le lien : dans l'exemple ci-dessus la valeur « *nom* » est concaténée à la fin de l'URL pour effectuer une recherche Google du nom de l'enregistrement sélectionné.

Il est possible de concaténer une propertie avec le lien : en écrivant **{{getPropertie("[nom de la propertie]")}}**. Dans l'exemple ci-dessous, la valeur de la propertie « *services_alias* » est affichée dans l'info-bulle.

```
select nom as "Nom", route_id as "Route id", auteur as "Auteur", image as "[bo_image]
↪", '{{getPropertie(''services_alias'')}}' as "service_alias" from sig.lampe
```

On peut également définir l'attribut *target* qui permet de choisir un nouvel onglet à chaque fois si on donne pour valeur « *_blank* » ou alors de stipuler un nom pour utiliser toujours le même onglet.

2.2. Dans le formulaire

On peut effectuer la même opération en personnalisant le formulaire en insérant un attribut de type *« Lien »* et en utilisant les fonction *« concat »* et *« getFormValue »* dans le champ *« Valeur »*.

```
concat('https://www.google.fr/?gws_rd=cr&ei=h3hvWbHuJIORaPe3ofAG#q=', getFormValue(
↪'nom'))
```

En utilisant les champs *« Texte »* et *« Cible »* on peut également modifier le texte à afficher ainsi que l'onglet cible.

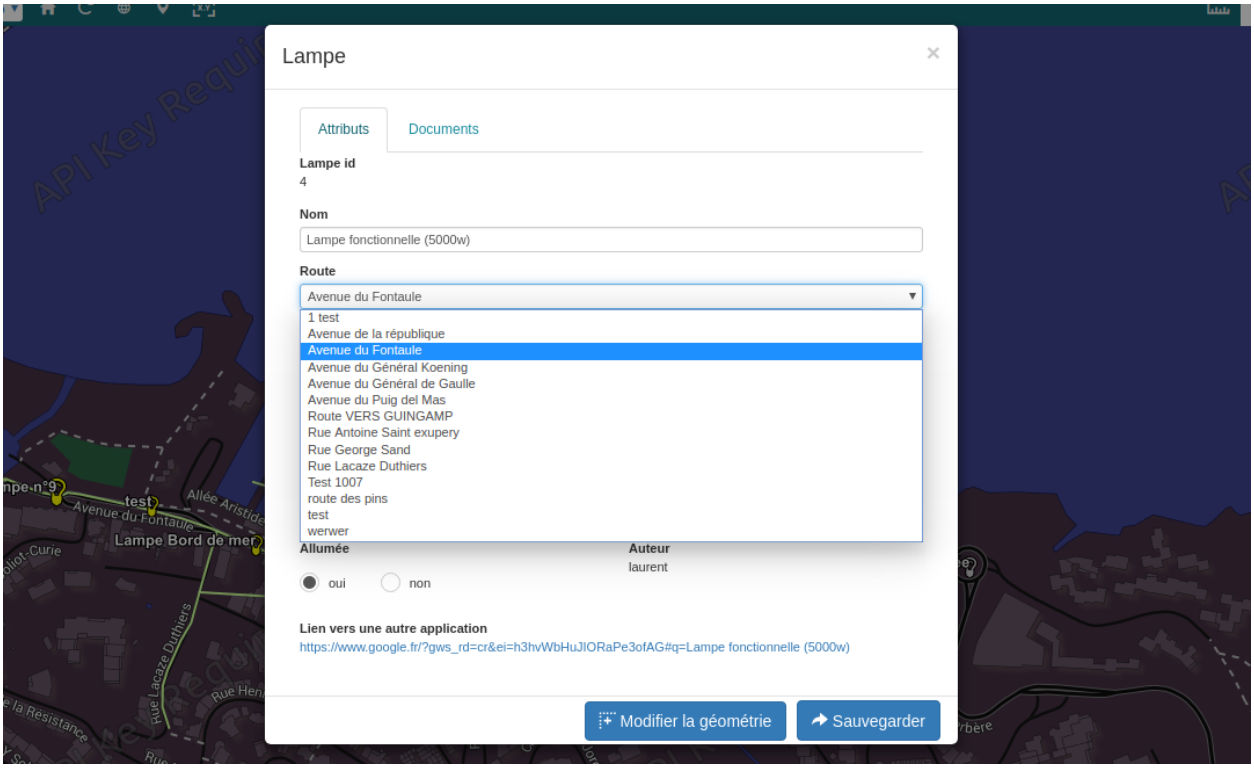
3. Utilisation du gestionnaire de source de données : insertion d'une liste déroulante

Le gestionnaire de sources de données permet la création, l'édition et la suppression de sources de données à associer à des attributs de type :

- liste
- liste déroulante

Le gestionnaire de sources de données permet d'exploiter des données :

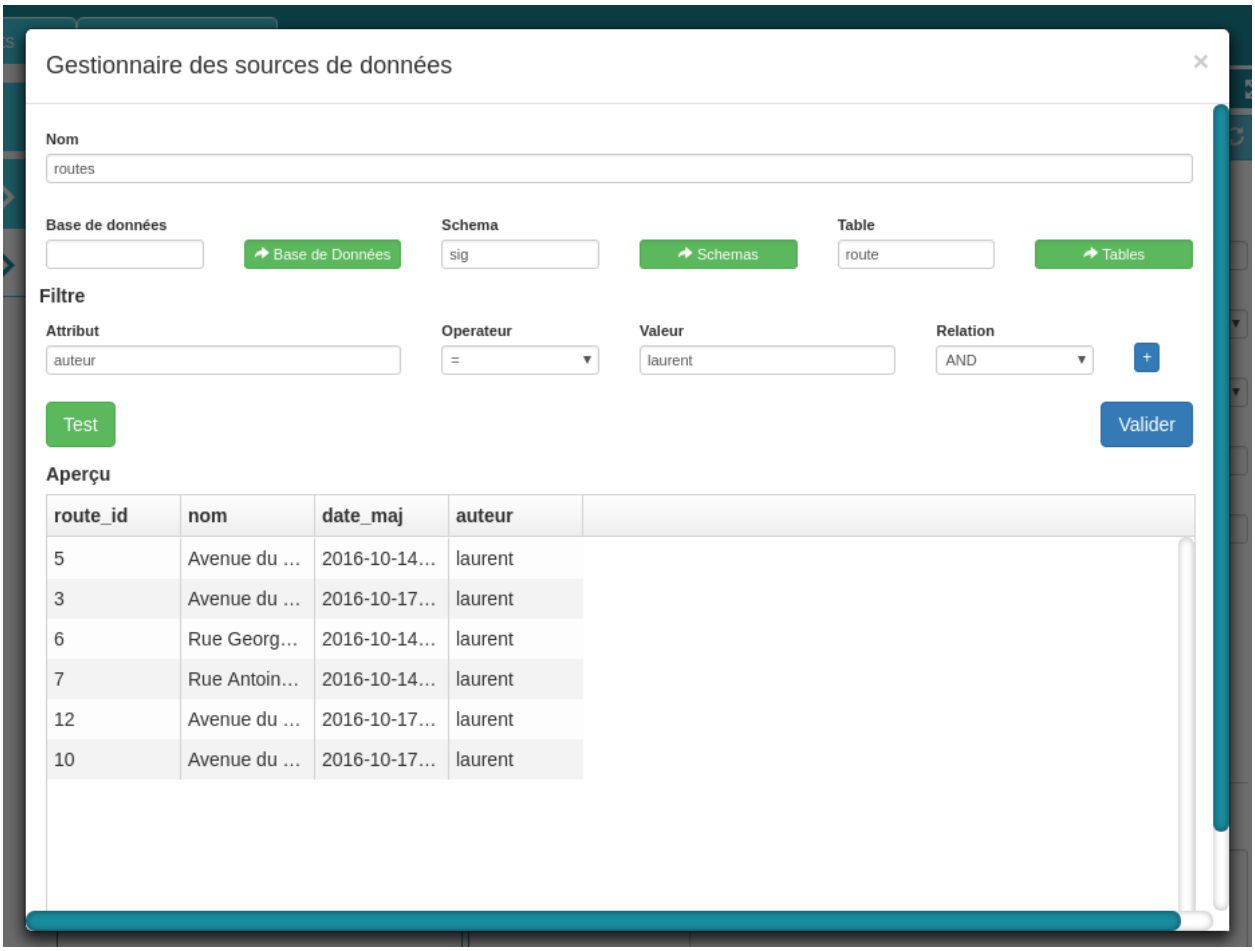
- Texte : valeurs saisies directement dans le gestionnaire
- Valeur de table locale : valeurs issues d'une table de base de données installée sur le même serveur que vMap
- Base de données externe : valeurs importées d'une table d'une base de données externe
- Service web Vitis : permet d'exploiter un service web pour en récupérer les ressources
- Objet métier : permet d'exploiter un objet métier déjà configuré



Le bouton **Sources de données**, en bas à droite du studio permet d'ouvrir le gestionnaire de source de données.

Dans l'exemple ci-dessous, il s'agit d'afficher l'ensemble des routes contenues dans la table *« route »* et dont l'auteur est *« laurent »*.

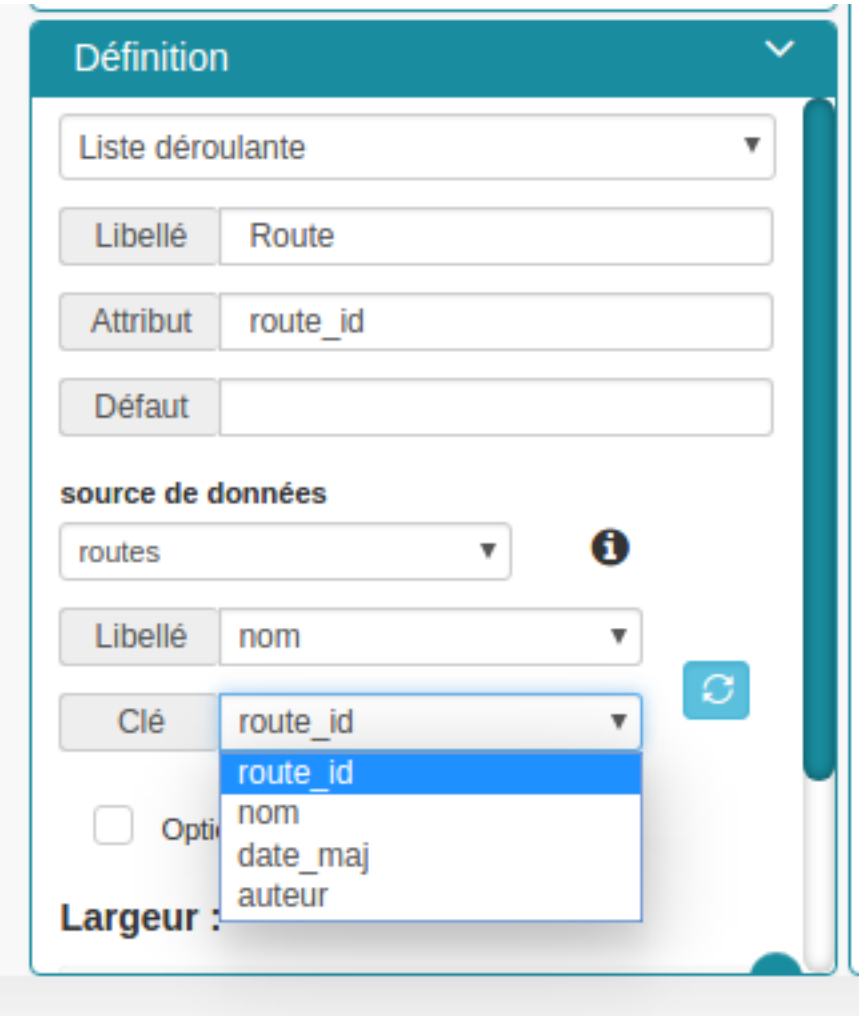
On peut utiliser le bouton *« + »* pour ajouter des nouveaux filtres et le bouton *« Test »* pour tester la source de données.



Une fois la source de données renseignée, on peut créer un attribut de type « *Liste déroulante* » (ou autre type de liste) et choisir la source de données mise en place précédemment.

Une liste est définie par une « *Clé* » qui est la valeur retournée lorsqu'on sélectionne un élément de la liste et d'un « *Libellé* » qui est ce que l'utilisateur voit dans la liste.

Dans cet exemple, on souhaite sélectionner une route à associer à la lampe en édition. Chaque route est définie par un identifiant numérique (route_id) et elle possède un nom textuel (nom) : on sélectionne donc « *nom* » en tant que libellé et « *route_id* » en tant que clé.

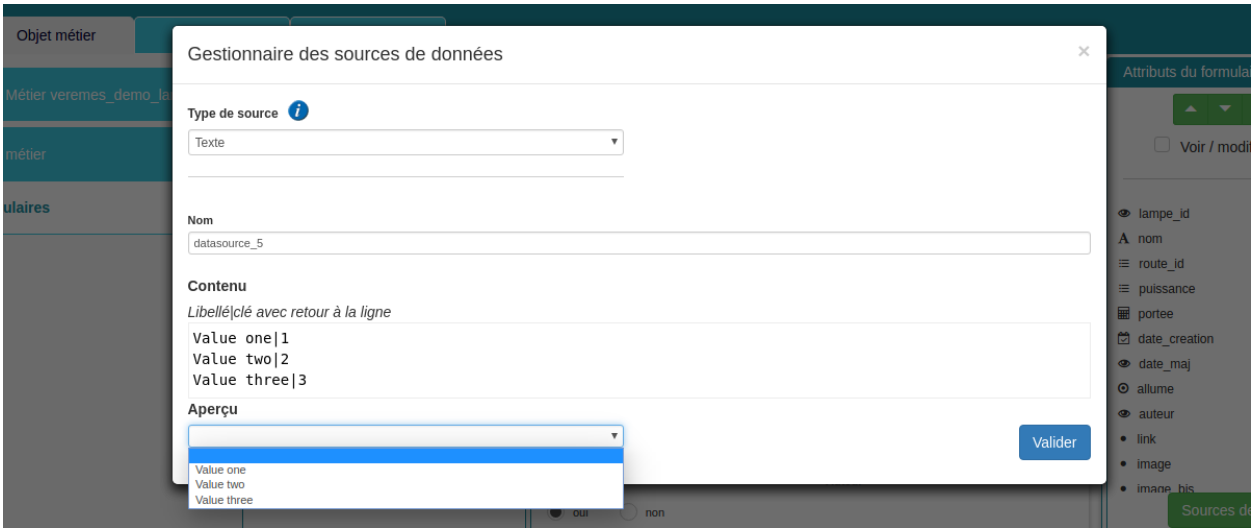


3.1. Source de données de type texte

Le type texte permet de renseigner soi-même le contenu de la source de données.

```
libellé 1|clé 1
libellé 2|clé 2
libellé 3|clé 3
```

Chaque entité est composée d'une **clé** qui est la valeur retenue et d'un **libellé** qui est le contenu affiché. Les deux sont séparés (sans espace) par le caractère « | ». On peut répéter l'opération autant de fois que l'on veut, en allant à la ligne pour chaque élément.

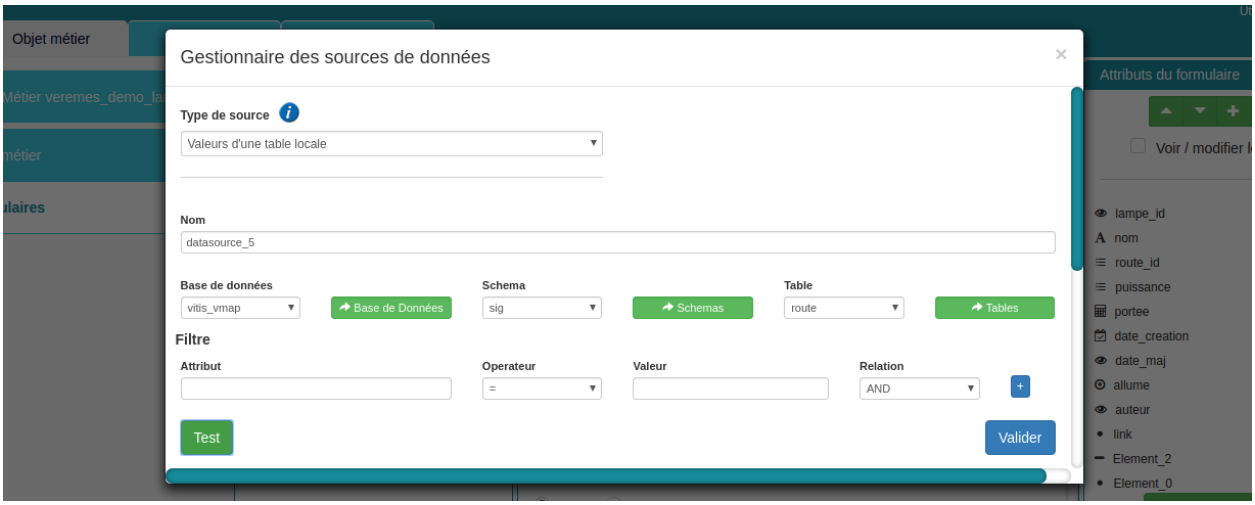


3.2. Source de données de type valeurs d"une table locale

Type utilisé lors de l"exemple précédent, il permet d"aller directement chercher en base de données (sur le serveur en cours) le contenu d"une table.

On peut également ajouter une ou plusieurs conditions à l"aide de filtre. Pour cela il suffit de renseigner une « *Valeur Clé* » qui est un nom de colonne de la table, un « *Opérateur* » dans le liste fournie et une « *Valeur* » qui correspond à la valeur à utiliser pour la condition. Le bouton « + » permettra d"ajouter des conditions et on peut également décider si les conditions sont de type « *AND* » ou « *OR* » grâce à une liste déroulante.

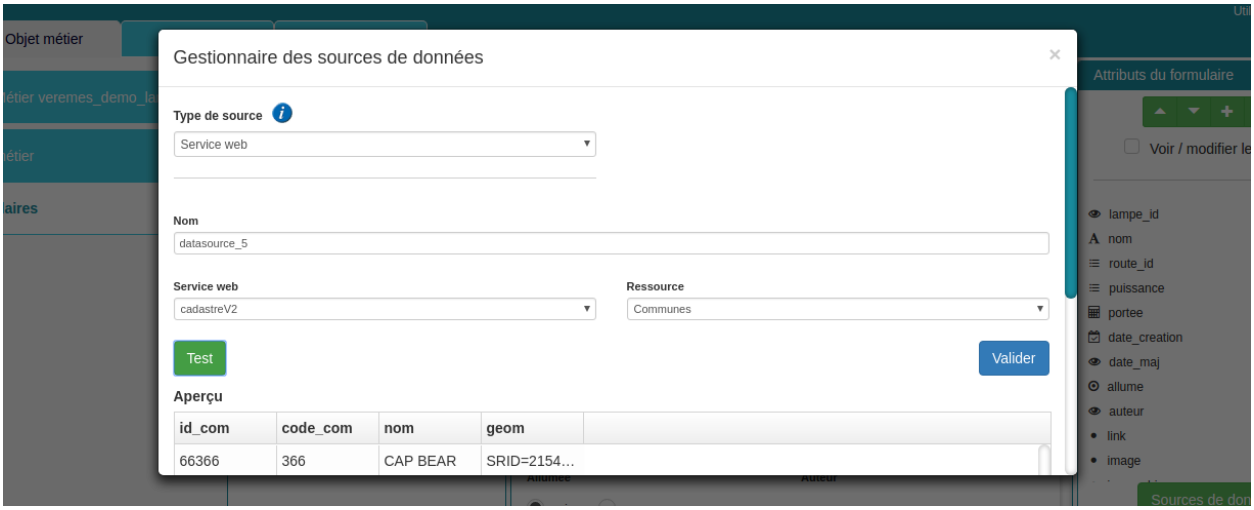
Important : lors de son utilisation, ce genre de source de données utilise le token de connexion de l"utilisateur. Il faut donc faire attention à ce que **tous les utilisateurs susceptibles d"utiliser le formulaire aient des droits en consultation sur la table.**



3.3. Source de données de Type service web

Parfois, le type « *Valeurs d"une table locale* » ne suffit pas car on veut utiliser une ressource d"un service web précédemment créé, afin d"effectuer des requêtes complexes. On peut aussi souhaiter se servir d"un services de l"application.

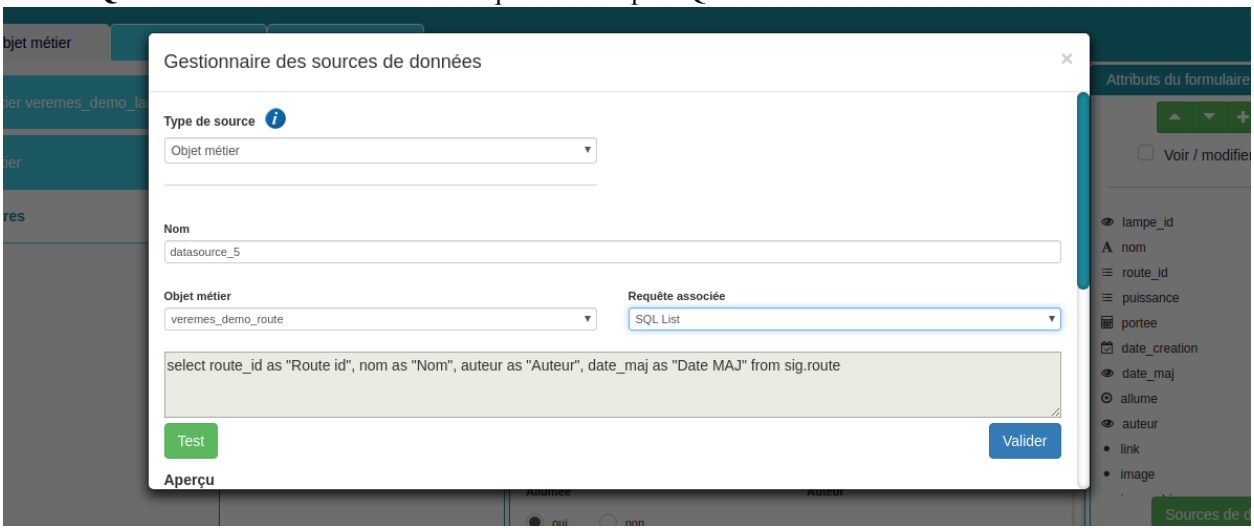
Pour cela, on utilise le type « *Service web* » qui effectue une requête de type « *GET* » à la ressource en question.



3.4. Source de données de type objet métier

Il est également possible d'interroger directement un objet métier selon une des trois solutions suivantes :

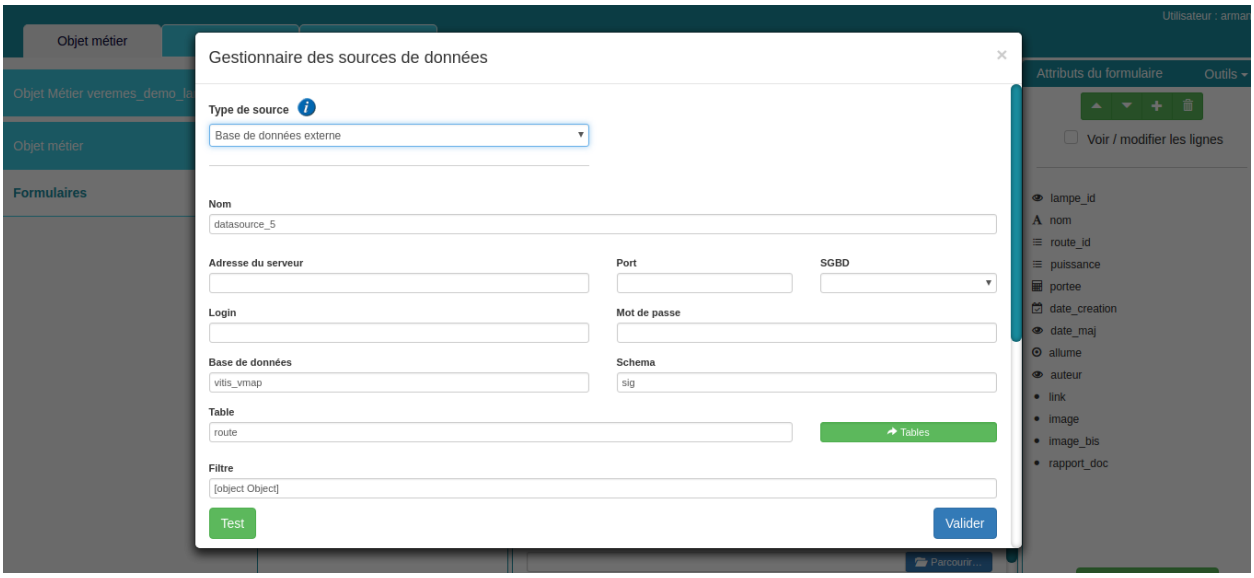
- **Form :** renvoie l'ensemble des colonnes de la table associée à l'objet métier
- **SQL Summary :** renvoie de résultat de la requête définie par SQL Summary
- **SQL List :** renvoie de résultat de la requête définie par SQL List



3.5. Source de données de type base de données externe

Plus complexe mais plus puissant, ce type de source permet d'interroger des bases de données d'un serveur externe selon un login et un mot de passe fourni.

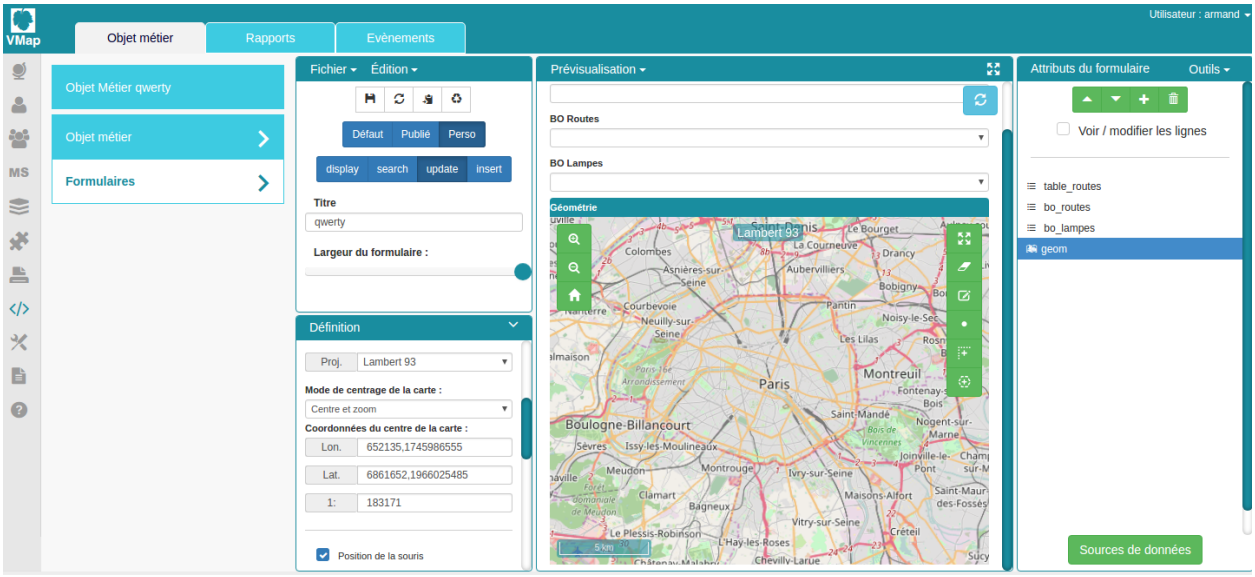
Important : les login et mot de passe renseignés doivent être publics car les utilisateurs finaux pourraient avoir accès à cette information.



4. Personnalisation d”un formulaire : insertion d”un attribut de type carte

Le studio permet d”exploiter les services web OSM, Bing Maps ou Vitis vMap pour personnaliser un formulaire en exploitant leurs ressources cartographiques.

L”utilisateur final peut, de la sorte, visualiser et saisir de la géométrie en exploitant la carte comme support de saisie.

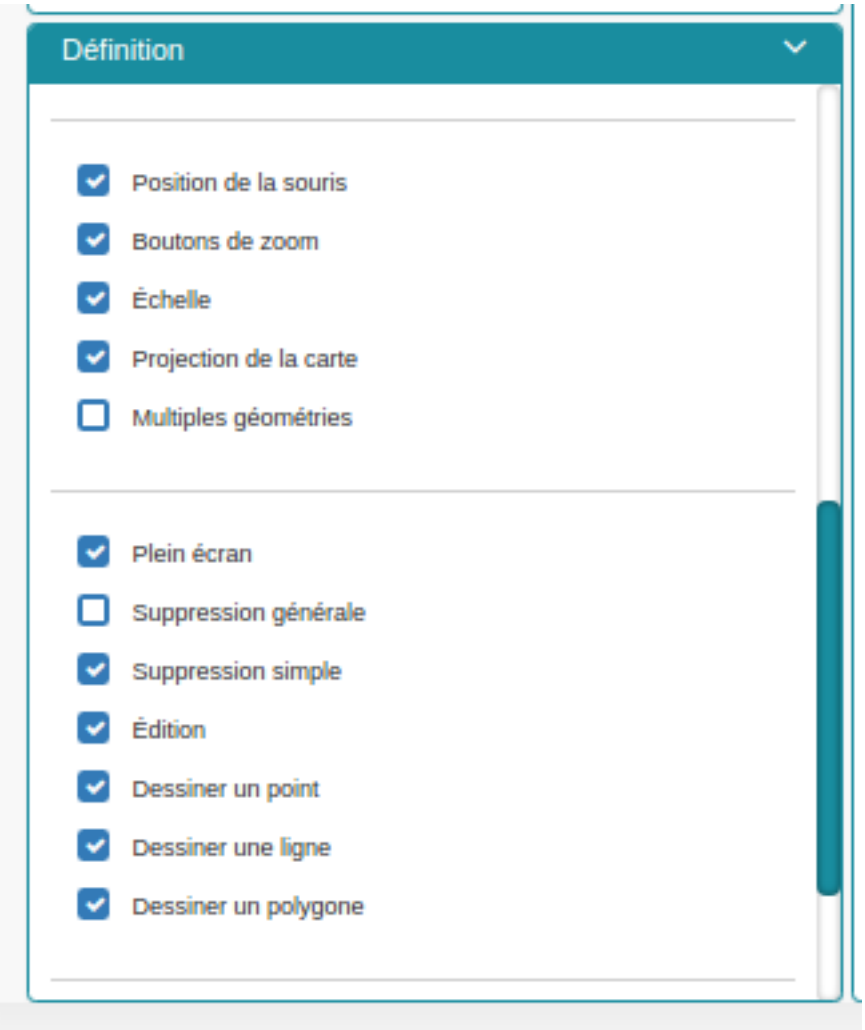


Trois types de cartes sont disponibles :

- **Carte OSM :** carte contenant une couche OSM
- **Carte Bing :** carte contenant une couche Bing (nécessite une clé)
- **Carte vMap :** carte complexe pouvant contenir plusieurs couches et définie par un fichier JSON téléchargeable depuis **Mode vMap > Cartes > Gestion des cartes > Ma carte > Télécharger**

Une fois la carte sélectionnée, l”administrateur peut définir l”emprise de la carte en naviguant simplement dessus ou en renseignant les champs « *Long* » pour la longitude, « *Lat* » pour la latitude et « *1 :* » pour l”échelle. Si le mode de centrage de la carte est défini sur « *Étendue* », saisir les valeurs « *XMin* », « *YMin* », « *XMax* », « *YMax*.

Les outils disponibles lors de l"utilisation sont configurables graphiquement via les boites à cocher de la zone « *Définition* ».



[doc]

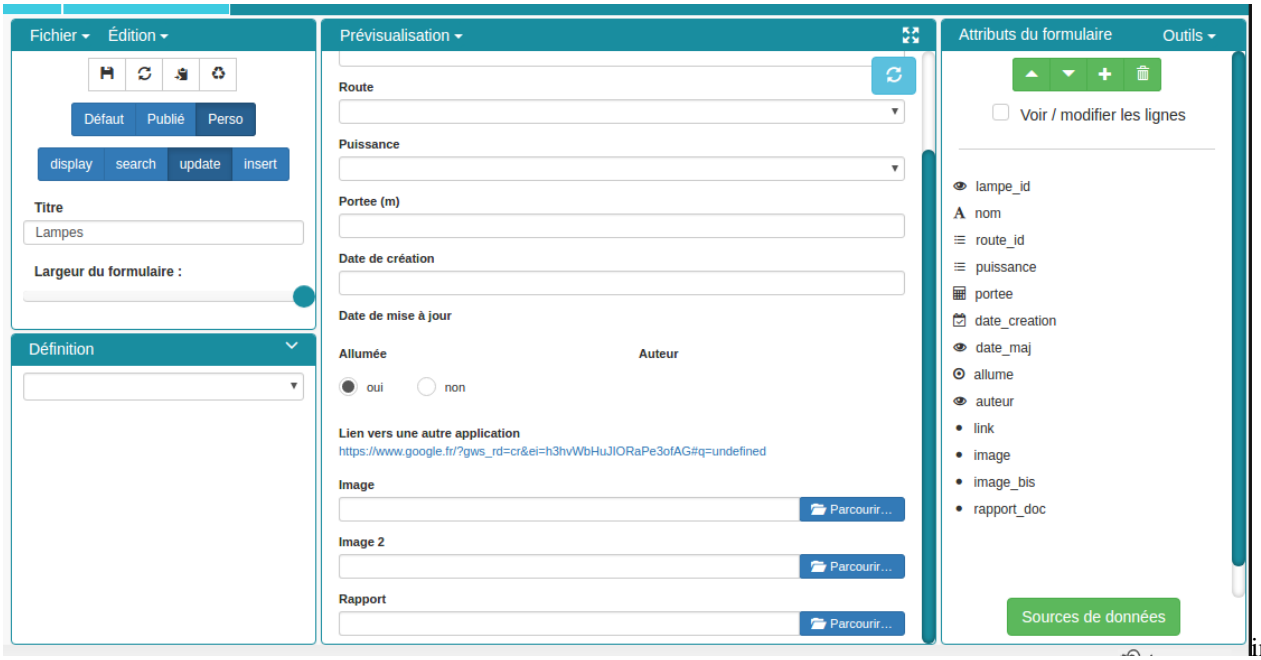
5. Personnalisation d"un formulaire : insertion d"un champ de chargement de Document/Image

Il est possible d"associer des documents ainsi que des images aux enregistrements liés à un objet métier en utilisant respectivement les types « *Document - Objet métier* » et « *Image - Objet métier* ».

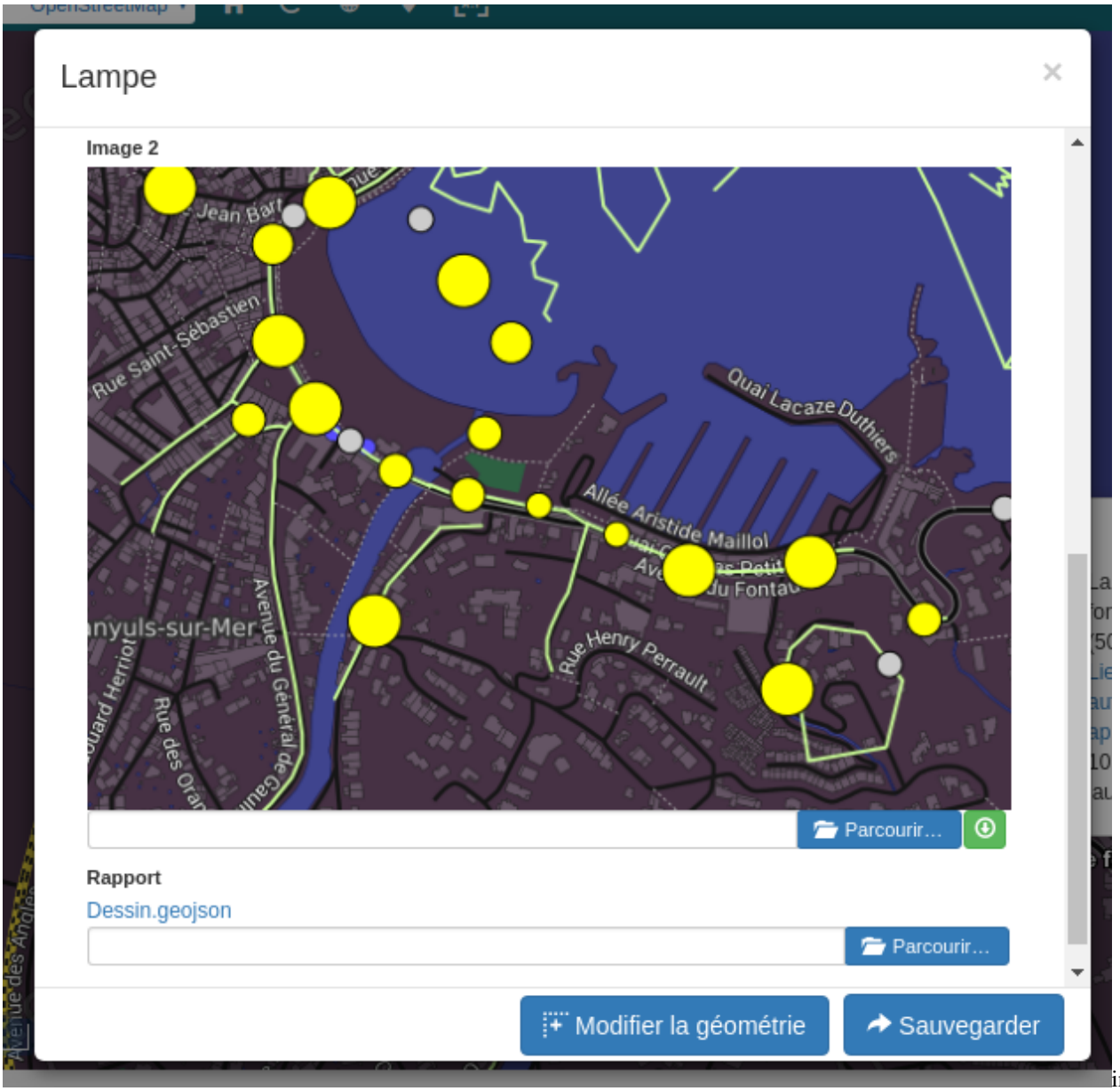
Une boite à cocher « *Uniquement en consultation* » permet de définir si l"utilisateur peut visualiser et éditer ce champs ou alors uniquement le visualiser.

Si elles existent, les images sont automatiquement affichées tandis que les documents sont disponibles en téléchargement.

| Studio | Résultat | | ——————————————————————- | ——————————————————————- | |


Fichier
Édition
Défaut
Publié
Perso
display
search
update
insert
Titre
Lampes
Largeur du formulaire :
Définition
Prévisualisation
Route
Puissance
Portee (m)
Date de création
Date de mise à jour
Allumée
Auteur
oui
non
Lien vers une autre application
https://www.google.fr/?gws_rd=cr&ei=h3hvWbHuJIORaPe3ofAG#q=undefined
Image
Parcourir...
Image 2
Parcourir...
Rapport
Parcourir...
Attributs du formulaire
Outils
Voir / modifier les lignes
lampe_id
nom
route_id
puissance
portee
date_creation
date_maj
allume
auteur
link
image
image_bis
rapport_doc
Sources de données

image



| image |

Les documents résultants sont stockés dans le répertoire suivant et seul leur nom est stocké en base :

```
{dossier vMap}/vas/ws_data/vitis/{nom de l'objet métier}/{identifiant de l
↪'enregistrement}/{nom de l'attribut}/{nom du fichier}
```

Remarque : seulement un fichier peut être associé à un attribut, si plusieurs fichiers doivent être téléversés, il faut créer plusieurs attributs ou sinon les compresser dans un fichier .zip

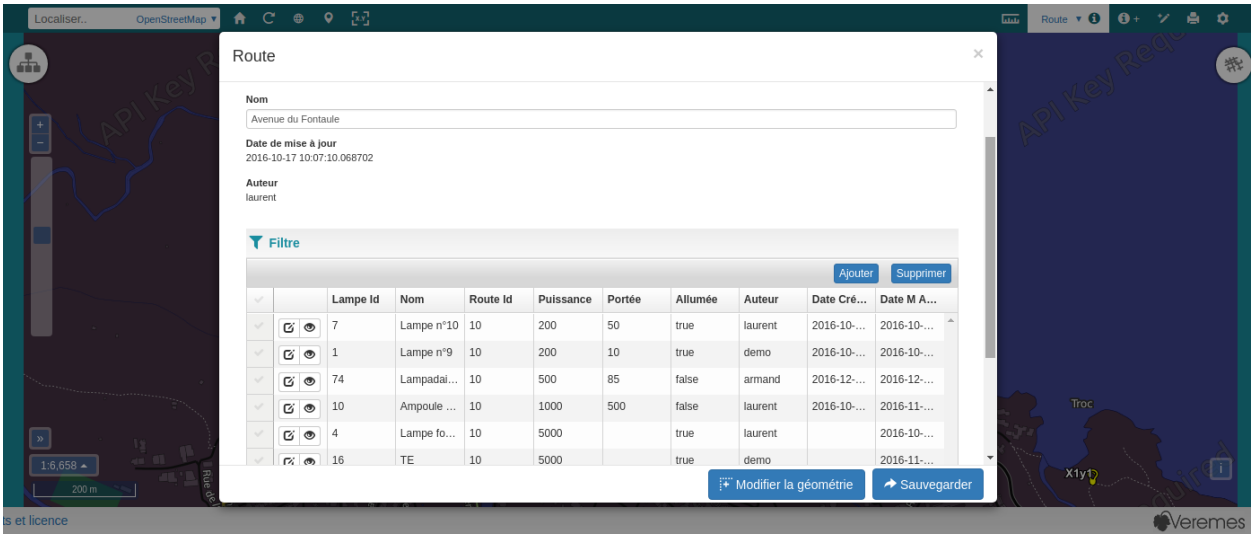
6. Personnalisation d"un formulaire : insertion d"une grille de sous-objets

Il est assez régulier d"avoir plusieurs objets métiers qui dépendent les uns des autres. Dans ce cas, il est très utile lors de l"édition d"un objet parent, de visualiser la liste des sous-objets liés à ce dernier.

Dans l"exemple ci-joint, c"est l"objet métier « *Route* » qui joue le rôle du parent. Un enregistrement peut etre constitué

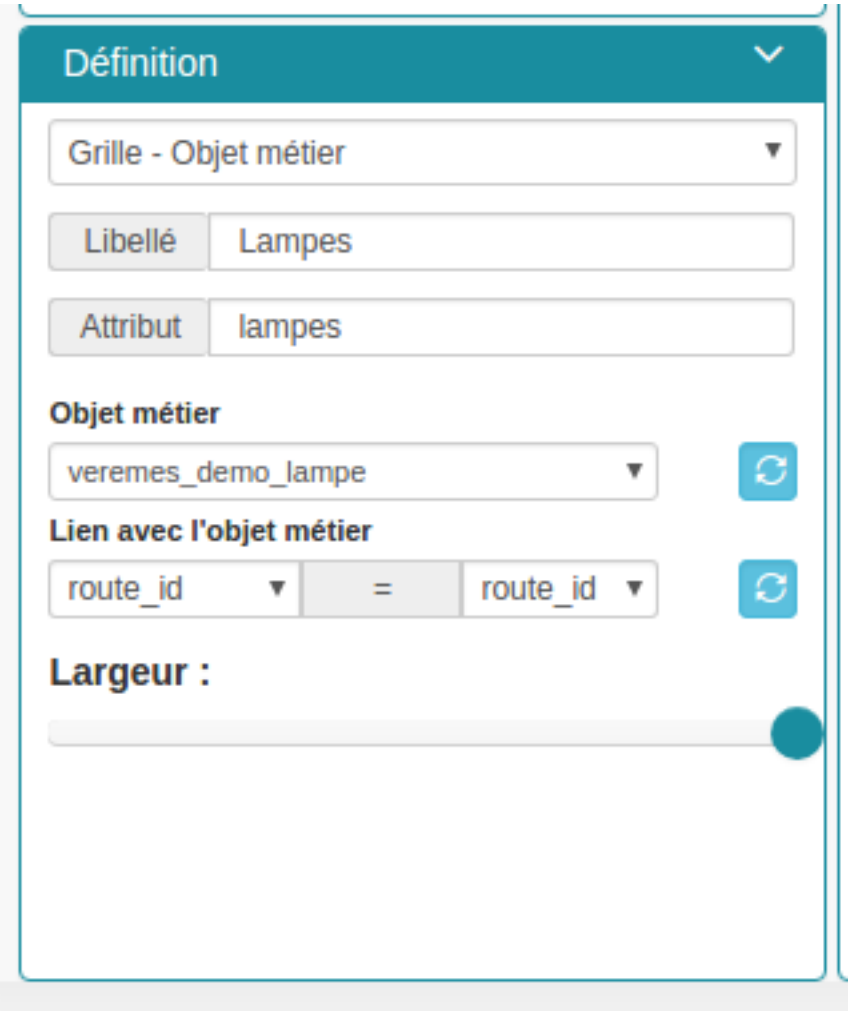
de plusieurs « *Lampes* ».

Dans vMap, il est possible d'"afficher les listes parents/enfants en donnant la possibilité d'"ajout, d'"édition et de suppression (en fonction des droits de l'"utilisateur) sur le sous-objet.



Dans le studio, il faut créer un élément de type « *Grille - Objet métier* », puis sélectionner l'"objet métier qui joue le rôle d'"enfant et renseigner le lien qui existe entre les deux objets.

Dans le champ « *Lien avec l'"objet métier* », le premier champ désigne la colonne de l'"enfant tandis que le deuxième celle de l'"enregistrement parent.



7. Edition de JavaScript associé à un formulaire : opérer une conversion rgb/rgba

vMap est un logiciel personnalisable. Il peut être utilise d"associer du code JavaScript aux différents formulaires.

Le code écrit dans ces formulaires est lancé lors de l"édition, l"insertion et la visualisation d"un objet métier. Il peut servir par exemple, à convertir des données avant et après saisie, faire des concaténations, des requêtes de type Ajax…

La section « *Édition JavaScript* » dans la partie « *Prévisualisation du studio* » permet d"ouvrir l"éditeur de code :

Le script doit être composé d'une fonction **constructor_form** appelée lors du chargement. Cette fonction est lancée avec le **scope** du formulaire en paramètre.

Testons le code suivant :

```
/**
 * constructor_form
 * Fonction appelé à l'initialisation du formulaire
 * @param {type} scope
 */
var constructor_form = function (scope) {
    console.log("constructor_form");

    alert('Hello world');

    console.log('scope:', scope);
};
```

Ceci va afficher une popup « Hello world » lors de l'affichage du formulaire, et va écrire le contenu de l'objet scope dans la console du navigateur (affichable dans les outils de développement).

Analysons le contenu de l'objet **scope** :

```
"": undefined$$
ChildScope: function b()
$$childHead: b
$$childTail: m
$$destroyed: false
$$isolateBindings: Object
$$listenerCount: Object
$$listeners: Object
$$nextSibling: m
$$phase: null
$$prevSibling: m
$$watchers: Array(13)
$id: 273
$parent: m
$root: mcloseModal: function (identifier)
compileTemplate: function ()
ctrl: formReader.formReaderController
custom-form: wd
executeButtonEvent: function ($event, buttonEvent)
getLinkFileName: function (url)
getValidationCssClass: function (sFieldName)
getWabField: function (oField)
iDisplayedTab: 0
initSubformGrid
Event_Element_0: function ()
initSubformGridEvent_counter: 9
isButtonPresent: function (oButton, oField, oTab)
isFieldPresent: function (oField, oTab, bCheckButtons)
isFormTextElement: function (sFormElementType)
isStringNotEmpty: function (element)
loadSubForm: function (opt_options)
oFormDefinition: Object
oFormEventsContainer: m
oFormValues: Object
oProperties: Object
oSubformValues: null
```

(suite sur la page suivante)

(suite de la page précédente)

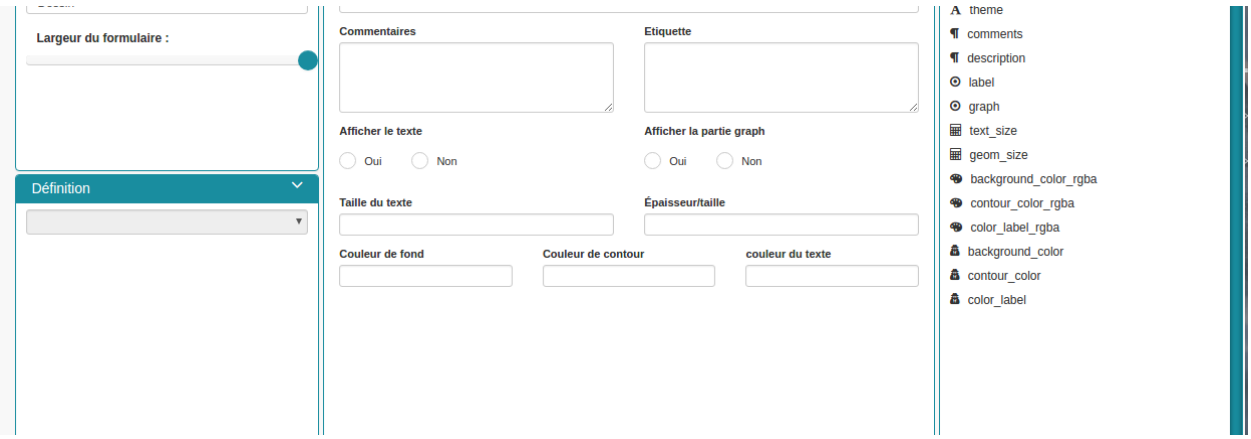
```
reloadSelectField: function (oParentSelect, sFormDefinitionName)
resetFileInputs: function ()
sFormDefinitionName: "update"
sFormUniqueName: 1500541427008
sendForm: function ()
setFormValues: function (oValues)
showTabs: true
submitButton: false
switchSelectedOptions: function (sFormDefinitionName, oFieldDefinition,␣
↪sFromSelectName, sToSelectName)
testElementsValidityTab: function (callback)
useWab: function ()
wabGroup: null
wabState: null
__proto__: Object
```

Dans cet objet, trois variables sont essentielles :

- **sFormDefinitionName :** nom du formulaire utilisé (update, display, insert etc..)
- **oFormDefinition :** définition JSON du formulaire
- **oFormValues :** valeurs courantes du formulaire

Dans notre cas nous voulons convertir les couleurs de « *rgba* » vers « *rgb* » et vise versa pour avoir un formulaire en « *rgba* » et une base de données en « *rgb* ».

Ces couleurs sont contenues en base dans les attributs « *background_color* », « *contour_color* » et « *color_label* ». Dans le formulaire, ces variables sont dans des champs cachés. Les attributs « *background_color_rgba* », « *contour_color_rgba* » et « *color_label_rgba* » sont également créés pour être exploités lors de l'"utilisation.



Dans le mode Edition du JavaScript, les fonctions de conversion suivantes sont crées créées :

```
var parseColorFromRGBA = function (rgba) {
    if (isRGBA(rgba)) {
        var matchColors = /rgba\((\d{1,3}),(\d{1,3}),(\d{1,3}),(\d{1,3})\)/;
        var match = matchColors.exec(rgba);
        var color = match[1] + ' ' + match[2] + ' ' + match[3];
    } else {
        color = rgba;
    }
    return color;
};

var parseColorToRGBA = function (color) {
```

(suite sur la page suivante)

(suite de la page précédente)

```
    if (isRGBA(color))
        var rgba = color;
    else
        var rgba = 'rgba(' + color.replace(/ /g, ',') + ',1)';
    return rgba;
};

var isRGBA = function (color) {
    if (color.substring(0, 4) === 'rgba')
        return true;
    else
        return false;
};
```

Le code suivant est généré pour convertir de « *rgb* » vers « *rgba* » lors du chargement du formulaire :

```
scope['oFormValues']['update']['background_color_rgba'] = parseColorToRGBA(scope[
↪'oFormValues']['update']['background_color']);
scope['oFormValues']['update']['contour_color_rgba'] = parseColorToRGBA(scope[
↪'oFormValues']['update']['contour_color']);
scope['oFormValues']['update']['color_label_rgba'] = parseColorToRGBA(scope[
↪'oFormValues']['update']['color_label']);
```

Et pour convertir le « *rgba* » vers « *rgb* », le code suivant est implémenté :

```
scope['oFormValues']['update']['background_color'] = parseColorFromRGBA(scope[
↪'oFormValues']['update']['background_color_rgba']);
scope['oFormValues']['update']['contour_color'] = parseColorFromRGBA(scope[
↪'oFormValues']['update']['contour_color_rgba']);
scope['oFormValues']['update']['color_label'] = parseColorFromRGBA(scope['oFormValues
↪']['update']['color_label_rgba']);
```

Le problème avec ce deuxième code c"est qu"il doit être lancé juste avant que le formulaire ne soit soumis par l"utilisateur car sinon les changements effectués par ce dernier ne seront pas appliqués.

Comment effectuer des opérations juste avant l"envoi du formulaire ?

Dans l"objet « *oFormDefinition* », il est possible de renseigner des événements :

- **beforeEvent :** événement appelé avant envoi au serveur
- **afterEvent :** événement appelé après l"envoi au serveur

De cette façon, écrire le code complet :

```
/**
 * constructor_form
 * Fonction appelé à l'initialisation du formulaire
 * @param {type} scope
 */
 var constructor_form = function (scope) {
    console.log("constructor_form");

    var parseColorFromRGBA = function (rgba) {
        if (isRGBA(rgba)) {
            var matchColors = /rgba\((\d{1,3}),(\d{1,3}),(\d{1,3}),(\d{1,3})\)/;
            var match = matchColors.exec(rgba);
            var color = match[1] + ' ' + match[2] + ' ' + match[3];
        } else {
            color = rgba;
```

(suite sur la page suivante)

(suite de la page précédente)

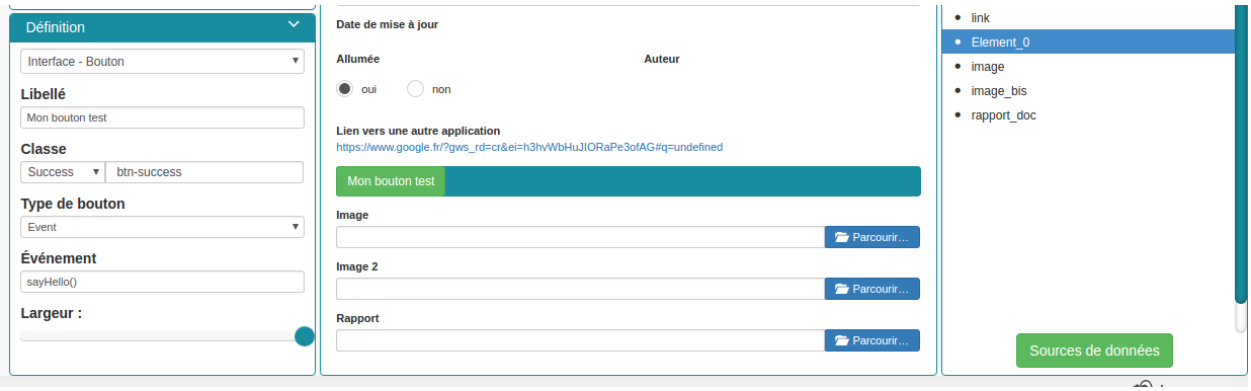
```
        }
        return color;
    };

    var parseColorToRGBA = function (color) {
        if (isRGBA(color))
            var rgba = color;
        else
            var rgba = 'rgba(' + color.replace(/ /g, ',') + ',1)';
        return rgba;
    };

    var isRGBA = function (color) {
        if (color.substring(0, 4) === 'rgba')
            return true;
        else
            return false;
    };

    // Lance la conversion de rgb vers rgba au chargement si on est en mode update
    if (angular.isDefined(scope['oFormValues']['update'])) {
        scope['oFormValues']['update']['background_color_rgba'] =␣
↪parseColorToRGBA(scope['oFormValues']['update']['background_color']);
        scope['oFormValues']['update']['contour_color_rgba'] = parseColorToRGBA(scope[
↪'oFormValues']['update']['contour_color']);
        scope['oFormValues']['update']['color_label_rgba'] = parseColorToRGBA(scope[
↪'oFormValues']['update']['color_label']);
    }

    // Lance la convertion de rgba vers rgb au beforeEvent
    var beforeEvent = function (sMode) {
        scope['oFormValues'][sMode]['background_color'] = parseColorFromRGBA(scope[
↪'oFormValues'][sMode]['background_color_rgba']);
        scope['oFormValues'][sMode]['contour_color'] = parseColorFromRGBA(scope[
↪'oFormValues'][sMode]['contour_color_rgba']);
        scope['oFormValues'][sMode]['color_label'] = parseColorFromRGBA(scope[
↪'oFormValues'][sMode]['color_label_rgba']);
    };

    // Ajoute BeforeEvent
    scope['oFormDefinition']['update']['beforeEvent'] = function () {
        beforeEvent('update');
    };
    scope['oFormDefinition']['insert']['beforeEvent'] = function () {
        beforeEvent('insert');
    };
};
```

8. Personnalisation d”un formulaire : insertion d”une fonction appelée depuis un Bouton - événement JavaScript

L”exemple précédent illustre la façon dont intégrer du code dans un formulaire objet métier via « *constructor_form* ». Dans ce nouvel exemple, une fonction appelée depuis un bouton dans l”interface est créée.

8.1. Bouton Hello world

Dans une première partie, une popup « Hello world » est affichée lors du clic sur le bouton. Il faut pour cela ajouter un attribut de type « *Interface - Bouton* » auquel on donne en événement, la fonction **sayHello**().



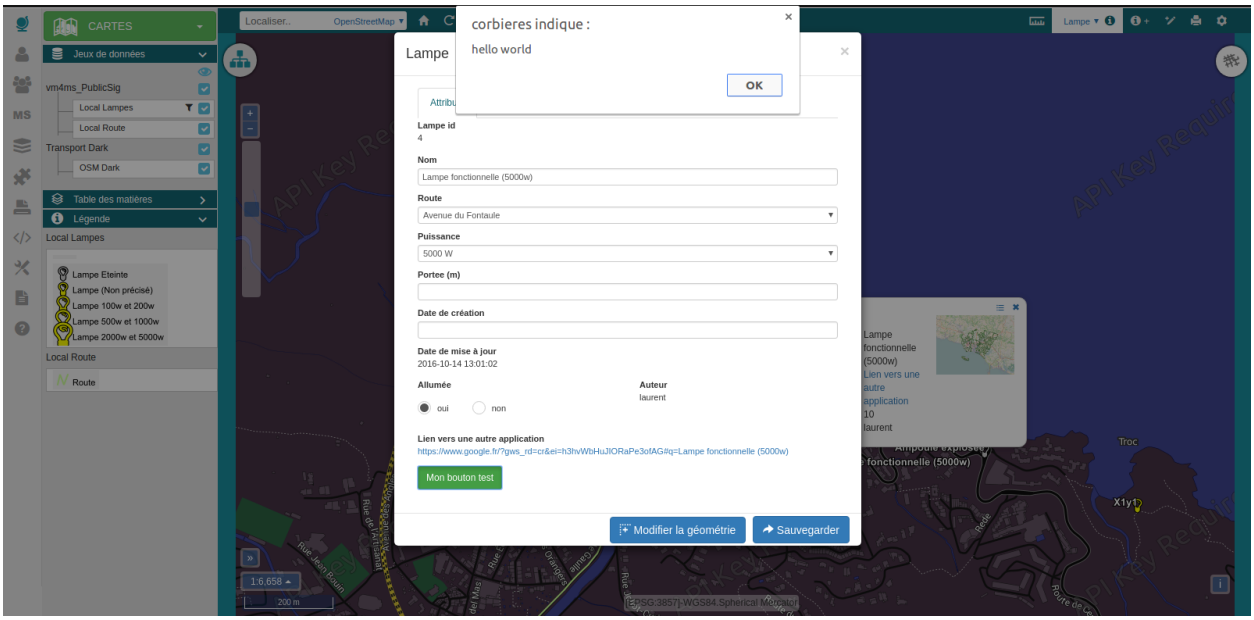
Côté JavaScript, il est important de placer la fonction sur le bon objet : il faut la placer sur **le scope de la Main Directive de Vitis**.

Pour y parvenir, il suffit d'appeler **angular.element(vitisApp.appMainDrtv).scope**() :

```
/**
 * constructor_form
 * Fonction appelé à l'initialisation du formulaire
 * @param {type} scope
 */
var constructor_form = function (scope) {
    console.log("constructor_form");

};

/**
 * Fonction à appeler par le bouton
 */
angular.element(vitisApp.appMainDrtv).scope()["sayHello"] = function(){
    alert('Hello world');
}
```

Remarque : il est important de vérifier via la console du navigateur que la fonction n'existe déjà pas car on pourrait remplacer par erreur une fonction déjà existante.

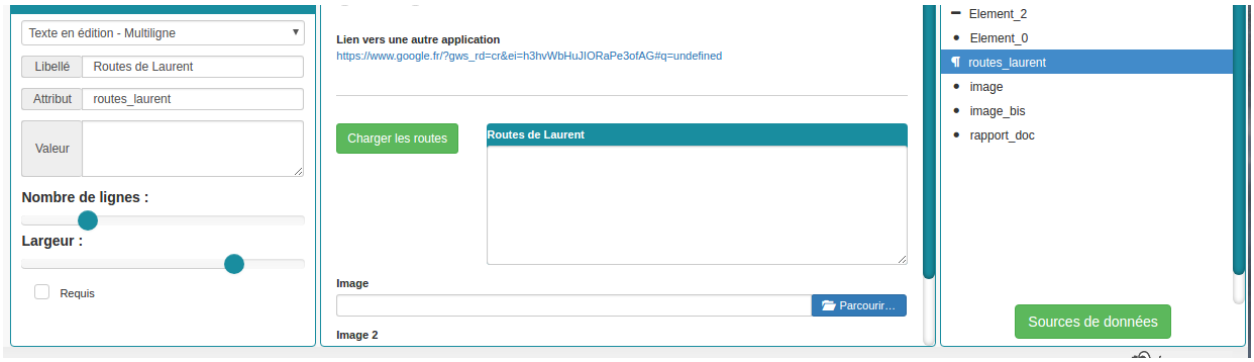
Voici le résultat côté client :



8.2. Bouton Ajax

Dans une deuxième partie, une requête Ajax est effectuée lors du clic sur le bouton. Elle permettra de récupérer les routes dont l'"auteur est « laurent » puis l'"on va les écrire dans un champ de type texte.

Pour cela, un bouton « *Charger les routes* » est crée. On y associe la fonction **loadLaurentRoutes**, et l'"on crée un champ de type « *Texte en édition - Multiligne* » nommé **routes_laurent**.



Pour effectuer la requête Ajax, il faut utiliser la fonction **ajaxRequest**() de vMap. Au moment de la réponse de la requête, on concatène chacun des résultats dans **oFormValues.update.routes_laurent** afin de voir apparaître le résultat dans l'"interface.

Pour avoir accès au scope depuis la fonction **loadLaurentRoutes**, on crée une variable globale **oFormRequired** dans laquelle on place le scope depuis **constructor_form**.

Voici le code final :

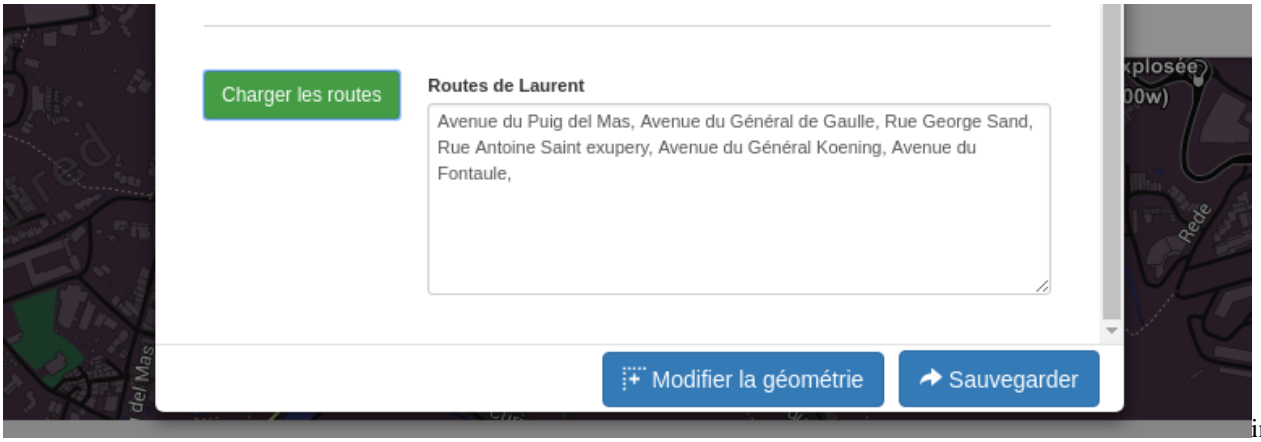
```
var oFormRequired = {
    scope_: {}
};

/**
 * constructor_form
```

(suite sur la page suivante)

(suite de la page précédente)

```
 * Fonction appelé à l'initialisation du formulaire
 * @param {type} scope
 */
 constructor_form = function (scope) {
    console.log("constructor_form");

    oFormRequired.scope_ = scope;
};

/**
 * Fonction à appeler par le bouton
 */
 angular.element(vitisApp.appMainDrtv).scope()["loadLaurentRoutes"] = function(){
    console.log('loadLaurentRoutes');

    showAjaxLoader();
    ajaxRequest({
        'method': 'GET',
        'url': oVmap['properties']['api_url'] + '/vitis/genericquerys',
        'headers': {
            'Accept': 'application/x-vm-json'
        },
        'params': {
            'schema':'sig',
            'table':'route',
            'filter':{"relation":"AND","operators":[{"column":"auteur","compare_
↪operator":"=","value":"laurent"}]}
        },
        'scope': oFormRequired.scope_,
        'success': function (response) {
            hideAjaxRequest();
            console.log('response', response);

            oFormRequired.scope_['oFormValues']['update']['routes_laurent'] = '';

            if (angular.isDefined(response['data'])){
                if (angular.isDefined(response['data']['data'])){
                    for (var i = 0; i < response['data']['data'].length; i++) {
                        oFormRequired.scope_['oFormValues']['update']['routes_laurent
↪'] += response['data']['data'][i]['nom'] + ', ';
                    }
                }
            }
        },
        'error': function (error){
            hideAjaxRequest();
            console.log('error', error);
        }
    });
};
```

Désormais, un clic sur le bouton « *Charger les routes* » remplit le champ « *Routes de laurent* »



image

4.10 Module anomalies

Aucune interface d'administration n'a été développée pour le module anomalie. L'ensemble des paramètres à éditer sont disponibles dans le fichier : /var/www/vmap/vas/rest/conf/anomalies/properties.inc

Ce fichier permet notamment de configurer le corps des emails qui sont envoyés.

4.10.1 Paramètres variables et paramétrables dans les emails

Pour mettre des valeurs dynamiques dans un email il faut utiliser la syntaxe suivante : `{{ anomaly.CLE }}`

Voici l'ensemble des variables qui peuvent être utilisées dans le corps des emails :

- **anomalies_id** -> Identifiant de l'anomalie
- **status_id** -> Identifiant du statut de l'anomalie
- **status_name** -> Nom du statut de l'anomalie
- **status_color** -> Couleur du statut de l'anomalie
- **theme_id** -> Identifiant du thème de l'anomalie
- **theme_name** -> Nom du thème de l'anomalie
- **theme_short_name** -> Abréviation du nom du thème de l'anomalie
- **title** -> Titre de l'anomalie
- **description** -> Description de l'anomalie
- **files** -> Nom des fichiers ajoutés avec le formulaire de l'anomalie
- **contact** -> Champ contact du formulaire de l'anomalie
- **user_id** -> Identifiant de l'utilisateur qui a créé l'anomalie
- **login** -> Login de l'utilisateur qui a créé l'anomalie
- **user_name** -> Nom de l'utilisateur qui a créé l'anomalie
- **user_email** -> Email de l'utilisateur qui a créé l'anomalie
- **anomalies_date** -> Date de la création de l'anomalie
- **last_update_date** -> Date de la dernière mise à jour de l'anomalie
- **closing_date** -> Date de clotûre de l'anomalie
- **map_id** -> Identifiant de la carte sur laquelle utilisateur était connecté lors de l'ajout de l'anomalie
- **admin_comment** -> Commentaire de l'administrateur

4.10.2 Propriété importante

Pour que le module fonctionne correctement il est important de définir une valeur à cette propriété.

$properties[»anomalies_email_admin »]

Doit contenir l'adresse email de l'administrateur qui recevra les emails. Aucune valeur par défaut n'a été définie.

4.10.3 Propriété générales des emails

$properties[»email_object_user_create_anomaly »]

Objet de l'email envoyé à l'utilisateur lors de la création d'une anomalie

Valeur par défaut :

```
"L'anomalie VMAP n°{{anomaly.anomalies_id}} a été créé"
```

$properties[»email_user_create_anomaly »]

Contenu de l'email envoyé à l'utilisateur lors de la création d'une anomalie

Valeur par défaut :

```
"Bonjour,
<br/><br/>
Vous venez de créer une nouvelle anomalie le {{anomaly.anomalies_date}} :
<br/>
Identifiant : {{anomaly.anomalies_id}}
<br/>
Thème : {{anomaly.theme_name}}
<br/>
Titre : {{anomaly.title}}
<br/>
Description : {{anomaly.description}}
<br/>
Coordonnées pour vous recontacter : {{anomaly.contact}}
<br/><br/>
Nous vous remercions pour votre retour et y donnerons suite dès que possible.
<br/>
Vous pouvez suivre la prise en charge de votre demande depuis l'onglet « Anomalies »␣
↪du MODULE ANOMALIES dans vmap.
<br/><br/>
Bien cordialement,
<br/>
L'équipe SIG"
```

$properties[»email_object_user_update_anomaly »]

Objet de l'email envoyé à l'utilisateur lors de la mise à jour d'une anomalie

Valeur par défaut :

```
"L'anomalie VMAP n°{{anomaly.anomalies_id}} a été mise à jour"
```

$properties[»email_user_update_anomaly »]

Contenu de l'email envoyé à l'utilisateur lors de la mise à jour d'une anomalie

Valeur par défaut :

```
"Bonjour,
<br/><br/>
La modification de votre anomalie n°{{anomaly.anomalies_id}} a bien été enregistrée␣
↪le {{anomaly.last_update_date}}.
<br/>
Identifiant : {{anomaly.anomalies_id}}
<br/>
Thème : {{anomaly.theme_name}}
<br/>
Titre : {{anomaly.title}}
<br/>
Description : {{anomaly.description}}
<br/>
Coordonnées pour vous recontacter : {{anomaly.contact}}
<br/><br/>
Bien cordialement,
<br/>
L'équipe SIG"
```

$properties[»email_object_user_status_in_progress_anomaly »]

Objet de l'email envoyé à l'utilisateur lorsque le statut de l'anomalie change pour « En cours de traitement »

Valeur par défaut :

```
"L'anomalie VMAP n°{{anomaly.anomalies_id}} est en cours de traitement"
```

$properties[»email_user_status_in_progress_anomaly »]

Contenu de l'email envoyé à l'utilisateur lorsque le statut de l'anomalie change pour « En cours de traitement »

Valeur par défaut :

```
"Bonjour,
<br/><br/>
Votre anomalie n°{{anomaly.anomalies_id}} « {{anomaly.title}} » du {{anomaly.
↪anomalies_date}} est en cours de traitement.
<br/><br/>
Bien cordialement,
<br/>
L'équipe SIG"
```

$properties[»email_object_user_status_waiting_anomaly »]

Objet de l'email envoyé à l'utilisateur lorsque le statut de l'anomalie change pour « En attente d'information »

Valeur par défaut :

```
"L'anomalie VMAP n°{{anomaly.anomalies_id}} est en attente d'information"
```

$properties[»email_user_status_waiting_anomaly »]

Contenu de l'"email envoyé à l'"utilisateur lorsque le statut de l'"anomalie change pour « En attente d'"information »

Valeur par défaut :

```
"Bonjour,
<br/><br/>
Nous sommes en attente d'informations pour le traitement de votre anomalie n°{
↪{anomaly.anomalies_id}} « {{anomaly.title}} » du {{anomaly.anomalies_date}}.
<br/>
N'hésitez pas à nous recontacter pour plus de précisions.
<br/><br/>
Bien cordialement,
<br/>
L'équipe SIG"
```

$properties[»email_object_user_status_cancel_anomaly »]

Objet de l'"email envoyé à l'"utilisateur lorsque le statut de l'"anomalie change pour « Annuler »

Valeur par défaut :

```
"L'anomalie VMAP n°{{anomaly.anomalies_id}} a été annulée"
```

$properties[»email_user_status_cancel_anomaly »]

Contenu de l'"email envoyé à l'"utilisateur lorsque le statut de l'"anomalie change pour « Annuler »

Valeur par défaut :

```
"Bonjour,
<br/><br/>
Votre anomalie n°{{anomaly.anomalies_id}} « {{anomaly.title}} » du {{anomaly.
↪anomalies_date}} a été annulée, à votre demande ou par votre administrateur SIG.
<br/>
N'hésitez pas à nous recontacter pour plus de précisions.
<br/><br/>
Bien cordialement,
<br/>
L'équipe SIG"
```

$properties[»email_object_user_status_finished_anomaly »]

Objet de l'"email envoyé à l'"utilisateur lorsque le statut de l'"anomalie change pour « Terminer »

Valeur par défaut :

```
"L'anomalie VMAP n°{{anomaly.anomalies_id}} a été traitée"
```

$properties[»email_user_status_finished_anomaly »]

Contenu de l'"email envoyé à l'"utilisateur lorsque le statut de l'"anomalie change pour « Terminer »

Valeur par défaut :

```
"Bonjour,
<br/><br/>
Votre anomalie n°{{anomaly.anomalies_id}} « {{anomaly.title}} » du {{anomaly.
↪anomalies_date}} a été traitée, la demande est clôturée le {{anomaly.closing_date}}.
<br/>
Remarques éventuelles : {{anomaly.admin_comment}}.
<br/><br/>
Merci encore pour vos retours,
<br/><br/>
Bien cordialement,
<br/>
L'équipe SIG"
```

4.10.4 Propriété des objets et contenus des emails pour l'"administrateur

$properties[»email_object_admin_create_anomaly »]

Objet de l'"email envoyé à l'"administrateur lors de la création d'"une anomalie.

Valeur par défaut :

```
"Anomalie VMAP n°{{anomaly.anomalies_id}} - {{anomaly.theme_name}} - {{anomaly.title}}
↪"
```

$properties[»email_admin_create_anomaly »]

Contenu de l'"email envoyé à l'"administrateur lors de la création d'"une anomalie.

Valeur par défaut :

```
"MODULE ANOMALIE VMAP
<br/><br/>
Une nouvelle anomalie a été signalée le {{anomaly.anomalies_date}} :
<br/>
Identifiant : {{anomaly.anomalies_id}}
<br/>
Thème : {{anomaly.theme_name}}
<br/>
Titre : {{anomaly.title}}
<br/>
Description : {{anomaly.description}}
<br/>
Coordonnées pour vous recontacter : {{anomaly.contact}}
<br/><br/>
Utilisateur : {{anomaly.login}}
<br/>
Coordonnées : {{anomaly.user_email}}"
```

$properties[»email_object_admin_update_anomaly »]

Objet de l'email envoyé à l'administrateur lors de la mise à jour d'une anomalie.

Valeur par défaut :

```
"Anomalie VMAP n°{{anomaly.anomalies_id}} - {{anomaly.theme_name}} - {{anomaly.title}}
↪"
```

$properties[»email_admin_update_anomaly »]

Contenu de l'email envoyé à l'administrateur lors de la mise à jour d'une anomalie.

Valeur par défaut :

```
"MODULE ANOMALIE VMAP
<br/><br/>
L' anomalie n°{{anomaly.anomalies_id}} a été modifiée le {{anomaly.last_update_date}}.
↪ :
<br/>
Identifiant : {{anomaly.anomalies_id}}
<br/>
Thème : {{anomaly.theme_name}}
<br/>
Titre : {{anomaly.title}}
<br/>
Description : {{anomaly.description}}
<br/>
Coordonnées pour vous recontacter : {{anomaly.contact}}
<br/><br/>
Utilisateur : {{anomaly.login}}
<br/>
Coordonnées : {{anomaly.user_email}}"
```

$properties[»email_object_admin_status_change_anomaly »]

Objet de l'email envoyé à l'administrateur lorsque le statut de l'anomalie change.

Valeur par défaut :

```
"Anomalie VMAP n°{{anomaly.anomalies_id}} - {{anomaly.theme_name}} - {{anomaly.title}}
↪"
```

$properties[»email_admin_status_change_anomaly »]

Contenu de l'email envoyé à l'administrateur lorsque le statut de l'anomalie change.

Valeur par défaut :

```
"MODULE ANOMALIE VMAP
<br/><br/>
L'anomalie n°{{anomaly.anomalies_id}} « {{anomaly.title}} » du {{anomaly.anomalies_
↪date}} a pris le statut {{anomaly.status_name}} le {{anomaly.last_update_date}}.
<br/><br/>
Utilisateur : {{anomaly.login}}
```

(suite sur la page suivante)

(suite de la page précédente)

```
<br/>
Coordonnées : {{anomaly.user_email}}"
```

$properties[»email_object_admin_finished_anomaly »]

Objet de l'email envoyé à l'administrateur lorsque le statut de l'anomalie change pour « Terminer ».

Valeur par défaut :

```
"Anomalie VMAP n°{{anomaly.anomalies_id}} - {{anomaly.theme_name}} - {{anomaly.title}}
↪"
```

$properties[»email_admin_finished_anomaly »]

Contenu de l'email envoyé à l'administrateur lorsque le statut de l'anomalie change pour « Terminer ».

Valeur par défaut :

```
"MODULE ANOMALIE VMAP
<br/><br/>
L'anomalie n°{{anomaly.anomalies_id}} « {{anomaly.title}} » du {{anomaly.anomalies_
↪date}} a été clôturée le {{anomaly.closing_date}}.
<br/><br/>
Remarques : {{anomaly.admin_comment}}.
<br/><br/>
Utilisateur : {{anomaly.login}}
<br/>
Coordonnées : {{anomaly.user_email}}"
```

4.10.5 Autre propriété

$properties[»anomalies_max_zoom »]

Nombre entre 1 et 24, défini le niveau de zoom maximum.

Valeur permettant de définir le niveau de zoom utilisé lorsqu'un utilisateur consulte (cartographiquement) une anomalie.

Cette valeur est une valeur maximale. Si l'anomalie est de type « multi-point » le centrage cartographique sera réalisé de telle manière à ce que l'ensemble des points soient visibles.

Cette valeur est comprise entre 1 et 24.

Valeur par défaut :

```
11
```

CHAPITRE 5

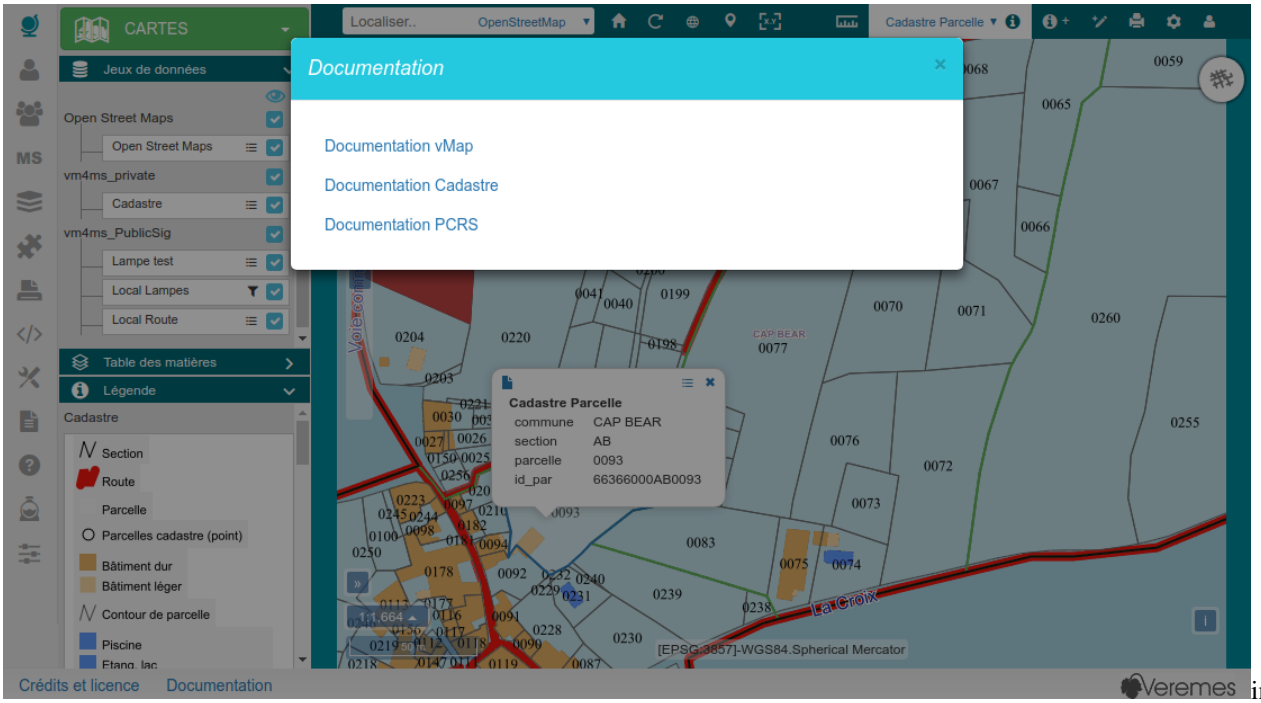
Configuration

5.1 Configuration générale

Documentation en cours de rédaction..

5.2 Gestion de la documentation

Depuis la version 2018 l"utilisateur peut visualiser des liens de documentation depuis l"interface de l"application vMap, GTF etc..



image

Par défaut se trouvera le lien vers la documentation de l"application utilisée mais l"administrateur pourra ajouter des liens vers de la documentation propre.

Pour ajouter des liens de documentation il faut éditer le fichier de configuration client situé dans **[répertoire de l"application]/client/conf/properties.json** dans ce fichier JSON doit se trouver un tableau d"objets appelé *« documentation »*.

Exemple pour l"application vMap

```
"documentation": [{
    "name": "Documentation vMap",
    "url": "https://vmap.readthedocs.io/fr/latest/"
}]
```

Pour ajouter de la documentation propre il suffira d"enrichir ce tableau comme dans l"exemple ci-dessous :

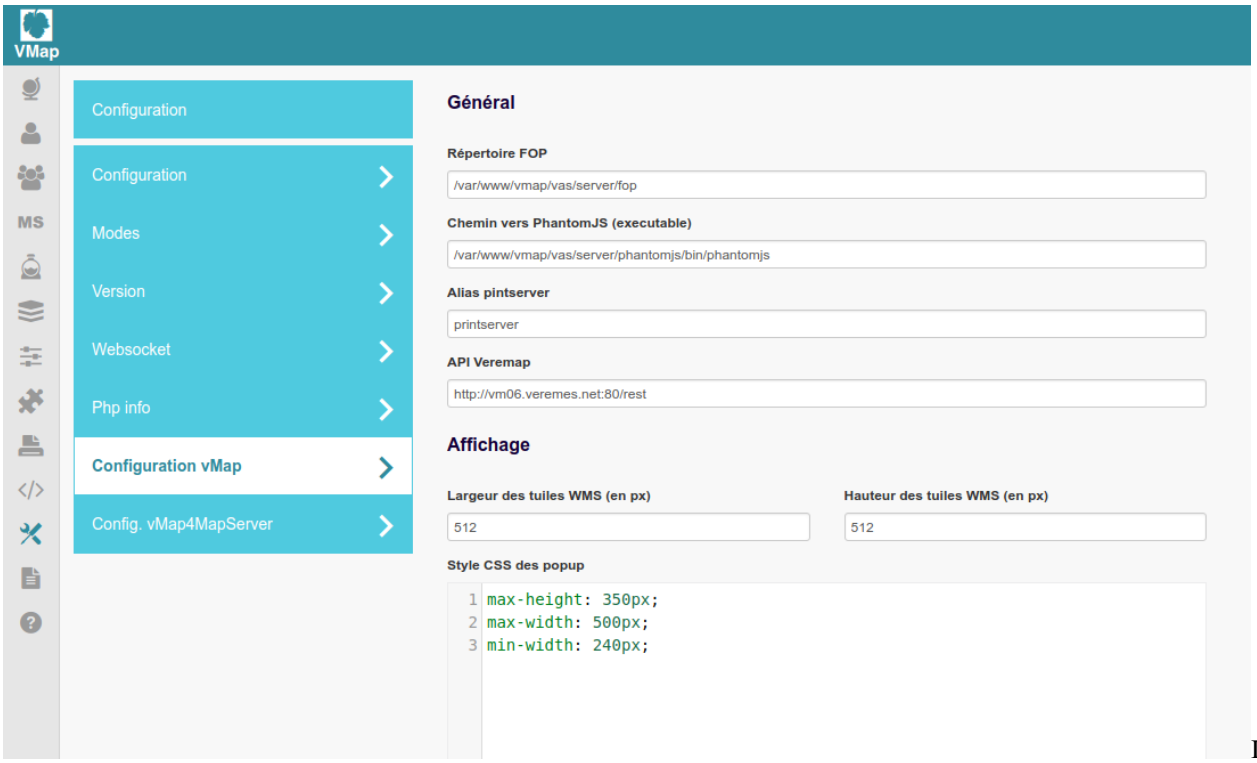
```
"documentation": [{
    "name": "Documentation vMap",
    "url": "https://vmap.readthedocs.io/fr/latest/"
}, {
    "name": "Documentation Cadastre",
    "url": "https://...."
}, {
    "name": "Documentation PCRS",
    "url": "https://...."
}]
```

Attention : lors d"une mise à jour ce fichier sera remplacé, il est donc important de le sauvegarder avant de lancer cette dernière

5.3 Configuration du mode cartographie

La configuration du monde cartographique peut se faire de manière totalement graphique depuis l'"interface mais pour des fonctions plus poussés elle se fait également en modifiant certains fichiers sur le serveur.

5.3.1 Depuis l'"interface de configuration



Interface de configuration de vMap

Général

Répertoire FOP

Chemin vers le répertoire où est installé FOP (nécessaire pour le module cadastre)

Défaut : */var/www/vmap/vas/server/fop*

Chemin vers PhantomJS (executable)

Chemin vers le répertoire où est installé PhantomJS (nécessaire pour les impressions)

Défaut : */var/www/vmap/vas/server/phantomjs/bin/phantomjs*

Alias pintserver

Alias Apache utilisé pour les impressions

Défaut : *printserver*

API Veremap

(Optionnelle) URL vers l'API de Veremap pour utiliser les flux privés de ce dernier

Affichage

Largeur des tuiles WMS (en px)

Largeur en pixels des tuiles pour les flux WMS

Défault : *512*

Hauteur des tuiles WMS (en px)

Hauteur en pixels des tuiles pour les flux WMS

Défault : *512*

Style CSS des popup

Style CSS à appliquer aux info-bulles, ceci permet entre autres de gérer la taille de ces dernières

Défault : *max-height : 350px ;max-width : 500px ;min-width : 240px ;*

Groupes de calques repliées par défaut dans « Jeux de données »

Définit si par défaut, dans la partie « Jeux de données » du bandeau de gauche, les groupes de calques sont repliés

Fenêtre « Jeux de données » repliée par défaut

Définit si par défaut, la partie « Jeux de données » du bandeau de gauche est repliée

Fenêtre « Table des matières » repliée par défaut

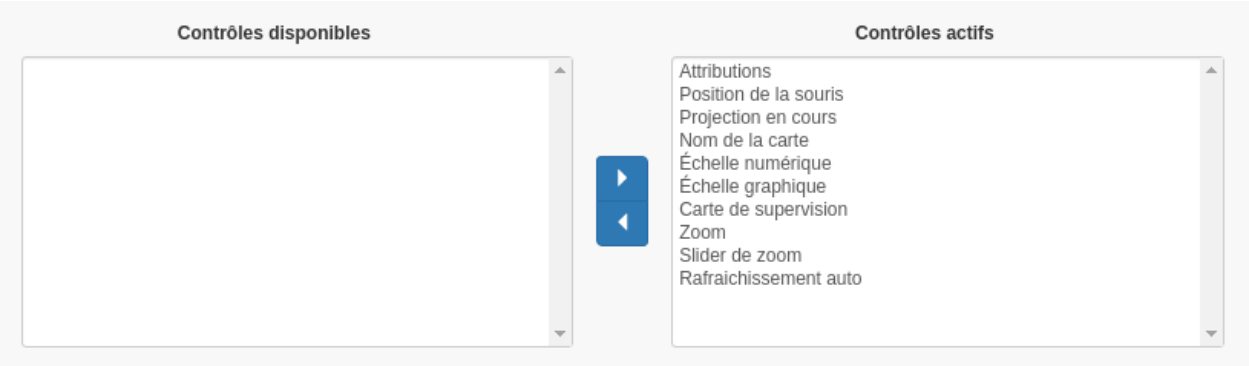
Définit si par défaut, la partie « Table des matières » du bandeau de gauche est repliée

Fenêtre « Légende » repliée par défaut

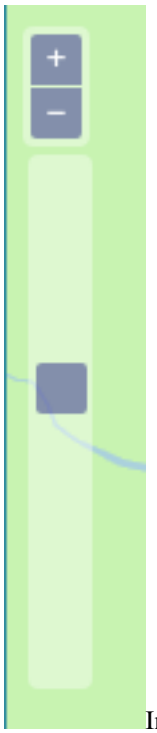
Définit si par défaut, la partie « Légende » du bandeau de gauche est repliée

Outils

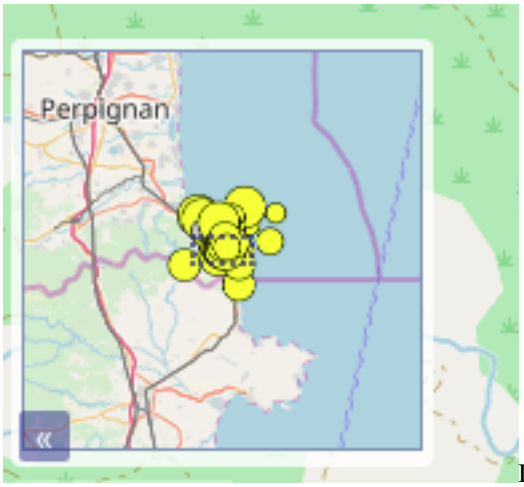
Contrôles



Active ou désactive les outils de cartographie suivants

— Slider de zoom 
Interface de configuration de vMap outils 2

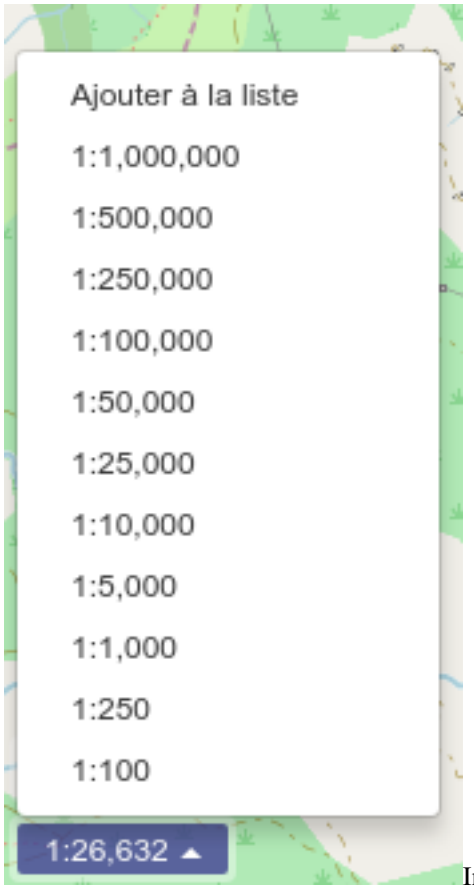
— Zoom 
Interface de configuration de vMap outils 3

— Carte de supervision 

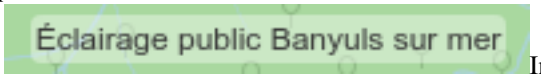
Interface de configuration de vMap outils 4

— Échelle graphique 

Interface de configuration de vMap outils 5

— Échelle numérique 

Interface de configuration de vMap outils 6

— Nom de la carte 

Interface de configuration de vMap outils 7

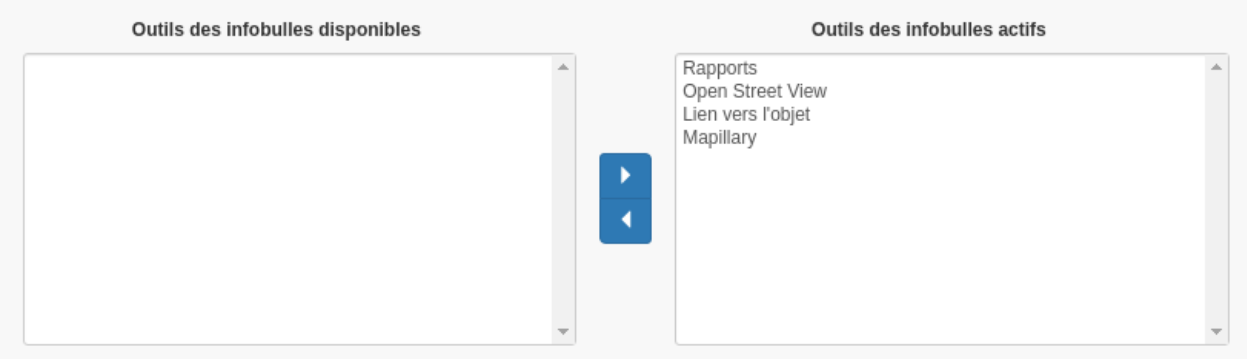
— Projection en cours 

Interface de configuration de vMap outils 8

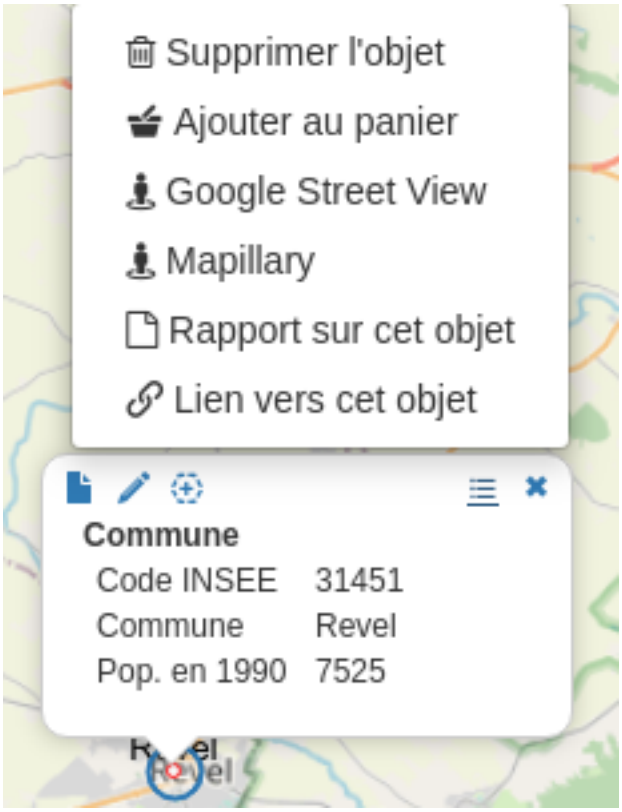
— Rafraichissement auto Rafaichit automatiquement grâce à la technologie websocket les couches quand deux

utilisateurs modifient la même donnée.

Outils des infobulles



Permet de gérer les outils présents dans les infobulles



— Rapports : Rapport sur cet objet

— Google street view : Google Street View

— Mapillary : Mapillary

— Lien vers l'"objet : Lien vers cet objet

Sélection

Nombre maximal de popups

Nombre maximal d'info-bulles affichables simultanément (valeur conseillée : 1)

Nombre maximal de sélections dans la liste

Nombre maximal d'enregistrements affichables simultanément dans le requêteur (valeur conseillée : 50)

5.3.2 Configuration serveur avec properties_server.inc

En modifiant le fichier *vmap/vas/rest/conf/properties_server.inc* il est possible de configurer des fonctions plus approfondies.

$properties["schema_vmap"]

Nom du schéma vMap utilisé par l'applicaiton

Défaut : *« s_vmap »*

$properties["use_proxy_for_tiles"]

True pour utiliser le proxy pour charger les tuiles, ceci va diminuer les performances mais améliorera la gestion du cross-origin.

Défaut : *false*

$properties["cadastre"]["api"]

Api du module cadastre à utiliser

Défaut : *« cadastreV2 »*

$properties["print"]["equality_timeout"]

Correspond au temps durant lequel le nombre de tuiles chargées est égal aux nombres de tuiles a charger a été mis en place.

En utilisant des services tuilés certaines tuiles peuvent mettre énormément de temps à se générer (certaines ne se généreront peut être jamais).Pour éviter que ceci bloque certaines impressions le paramètre equality_timeout l'impression se lancera même si certaines tuiles n'ont pas encore été chargées.

Augmenter equality_timeout améliorera vos chances d'avoir une meilleure impression mais augmentera le temps nécessaire à une impression

Défaut : *1000*

$properties[”print”][”tile_size”]

Taille des tuiles durant une impression, en augmentant ce chiffre on diminue les performances mais on diminue aussi le risque qu”un libellé placé entre deux tuiles soit tronqué.

Défaut : *1024*

$properties[”print”][”features_zoom”]

Pourcentage de zoom à appliquer quand on imprime une géométrie

Défaut : *100*

$properties[”print”][”quality”]

Pourcentage de qualité d”impression

Défaut : *100*

$properties[”use_veremap_api”]

Définit si il faut utiliser l”API Veremap pour générer des flux privés

Défaut : *false*

$properties[”use_vm4ms_api”]

Définit si il faut utiliser l”API du module Mapserver pour générer des flux privés

Défaut : *true*

$properties[”owner_login”]

Login à utiliser pour effectuer les actions de super utilisation

Défaut : *$properties[»vitis_owner_login »]*

$properties[”owner_pass”]

Mot de passe à utiliser pour effectuer les actions de super utilisation

Défaut : *$properties[»vitis_owner_pass »]*

$properties[”vmap_log_dir”]

Chemin où écrire les fichiers de log

Défaut : *« {$properties[”vas_home”]}/log/vmap »*

$properties["vmap_map_log_file"]

Chemin où écrire les fichiers de log pour l"utilisation des cartes

Défaut : *$properties["vmap_log_dir"] . "/map/" . date($properties[»log_period »]) . "/map.log"*

$properties["vmap_geocoders"]

Géocodeurs à utiliser par défaut

Défaut :

```
{"osm":{"title":"OpenStreetMap","url":"https://nominatim.openstreetmap.org/search?
↪format=json&addressdetails=1&limit=[limit]&extratags=1&namedetails=1&polygon_
↪geojson=1&countrycodes=fr&q=[search]","data_field":"data","title_field":"display_
↪name","geojson_field":"geojson","summary_fields":[{"key":"address.country","label":
↪"Pays"},{"key":"address.state","label":"Région"},{"key":"address.postcode","label":
↪"Code postal"},{"key":"extratags.population","label":"Population"}]},"national_
↪adresse":{"title":"Base nationale","url":"https://api-adresse.data.gouv.fr/search/?
↪q=[search]&limit=[limit]","data_field":"data.features","title_field":"properties.
↪label","geojson_field":"geometry","summary_fields":[{"key":"properties.context",
↪"label":"Département"},{"key":"properties.city","label":"Ville"},{"key":"properties.
↪postcode","label":"Code postal"}]}}'
```

Il est possible de modifier la liste des géocoders.

- Pour retirer les 2 geocoders par défaut il suffit mettre la valeur suivante `$properties['vmap_geocoders'] = ''`
- Pour ne conserver que OSM : `$properties['vmap_geocoders'] = '{"osm":{"title":"OpenStreetMap","url":"https://nominatim.openstreetmap.org/search?format=json&addressdetails=1&limit=[limit]&extratags=1&namedetails=1&polygo "data_field":"data","title_field":"display_name","geojson_field":"geojson", "summary_fields":[{"key":"address.country","label":"Pays"}, {"key":"address.state","label":"Région"},{"key":"address.postcode","label":"Code postal"},{"key":"extratags.population", "label":"Population"}]}'`
- Pour ne conserver que la Base nationale : `$properties['vmap_geocoders'] = '"national_adresse":{"title":"Base nationale","url":"https://api-adresse.data.gouv.fr/search/?q=[search]&limit=[limit]","data_field":"data.features","title_field":"properties.label","geojson_field":"geometry", "summary_fields":[{"key":"properties.context","label":"Département"}, {"key":"properties.city","label":"Ville"},{"key":"properties.postcode", "label":"Code postal"}]}}'`

$properties["vmap_default_geocoders"]

Géocodeur à utiliser par défaut

Défaut : *"osm"*

$properties["vmap_export"]["gtf_api_url"]

Permet l"extraction des données par GTF.

URL vers l"API de l"instance GTF à utiliser

$properties["vmap_export"]["gtf_workspace_id"]

Permet l"extraction des données par GTF.

Workspace id à utiliser lors de l"export

$properties["vmap_export"]["gtf_priority_id"]

Permet l"extraction des données par GTF.

Défaut : *"1"*

$properties["vmap_export"]["gtf_email_option_id"]

Permet l"extraction des données par GTF.

Défaut : *"1"*

$properties["vmap_export"]["gtf_public_token"]

Permet l"extraction des données par GTF.

Token public de l"instance GTF

Défaut : *"publictoken"*

$properties["vmap_export"]["gtf_export_formats"]

Permet l"extraction des données par GTF.

Formats supportés par GTF

Défaut :

```
$properties['vmap_export']['gtf_export_formats'] = '[{"label": "Shapefile", "value":
↪"shape"}]'
```

$properties["vmap_export"]["gtf_export_coordsys"]

Permet l"extraction des données par GTF.

Systèmes de coordonnées supportés par GTF

Défaut :

```
$properties['vmap_export']['gtf_export_coordsys'] = '[{"label":"2154 Lambert 93",
↪"value":"2154"},{"label":"3857 WGS84 Spherical Mercator","value":"3857"},{"label":
↪"4326 WGS84 Longitude latitude","value":"4326"},{"label":"27561 Lambert I Nord",
↪"value":"27561"},{"label":"27562 Lambert II Centre","value":"27562"},{"label":
↪"27563 Lambert III Sud","value":"27563"},{"label":"27564 Lambert IV Corse","value":
↪"27564"},{"label":"27571 Lambert I Carto","value":"27571"},{"label":"27572 Lambert␣
↪II Carto","value":"27572"},{"label":"27573 Lambert III Carto","value":"27573"},{
↪"label":"27572 Lambert IV Carto","value":"27572"}]'
```

$properties[”vmap_export”][”gtf_2020”]

Permet l”extraction des données par GTF en version 2020.

Si GTF est en version 2020 ou suppérieure il faudra écrire `$properties['vmap_export']['gtf_2020'] = true;`

5.3.3 Configuration client avec properties.json

En modifiant le fichier *vmap/client/conf/properties.json* il est possible de configurer la partie client de l”application

Mode mobile

En passant `mobile_interface` à `true` les utilisateurs utilisant des smartphones pourront utiliser le mode cartographie en version mobile.

Cet interface est étidié pour être utilisé depuis le navigateur chrome de tout smartphone dont l”écran est de au moins 4 pouces, il peut être également utilisé depuis un autre navigateur mais il se peut que cela engendre certains bugs. Il faudra activer le GPS du périphérique et autoriser la localisation depuis le navigateur pour utiliser les outils de localisation.

```
"mobile_interface": "true"
```

5.4 Configuration du Cadastre

L”administration du module cadastre se fait directement sur le serveur en modifiant le fichier de properties

```
vmap/vas/rest/conf/cadastreV2/properties.inc
```

Sur ce fichier on trouve un script PHP contenant un tableau de propriétés défini par la variable **$properties** grâce auquel il est possible de gérer les paramètres suivants.

5.4.1 Propriétés potentiellement modifiables

Fiche descriptive de la parcelle

Properties génériques pour obtenir des informations sur les intersections. Fonctionne avec des objets ponctuels (POINT), linéaires (LINE) et polygones (POLYGON). Fonctionne avec une ou plusieurs vues et/ou une ou plusieurs tables.

Exemple :

- $properties[»cadastre »][»descr_parcel »][»intersect »][»views »][»nom_vue »][»title »] = « Le titre »
- $properties[»cadastre »][»descr_parcel »][»intersect »][»views »][»nom_vue »][»type »]= « POLYGON »
- $properties[»cadastre »][»descr_parcel »][»intersect »][»views »][»nom_vue »][»intersect »] = « Libellé surface inter. »
- $properties[»cadastre »][»descr_parcel »][»intersect »][»views »][»nom_vue »][»champ1 »] = « Libellé champ »
- $properties[»cadastre »][»descr_parcel »][»intersect »][»views »][»nom_vue »][»champ2 »] = « Libellé champ »
- $properties[»cadastre »][»descr_parcel »][»intersect »][»views »][»nom_vue »][»… »] = « Libellé champ »
- $properties[»cadastre »][»descr_parcel »][»intersect »][»views »][»… »][»title »] = « Le titre »

— etc…

Si ces informations sont mal renseignées, la fiche descriptive d'une parcelle peut ne pas se générer correctement.

Exemple de personnalisation d'une intersection comprenant un lien href :

1- Construction du lien dans une vue de postgresql

L'affichage d'un lien dans les intersections se construit dans une vue de postgresql sous la forme "[link href= »URL » target= »_blank »]Libellé[/link]" as « link »

Exemple :

```
SELECT commune.nom, ('[link href="https://fr.wikipedia.org/wiki/'::text ||␣
↪lower(commune.nom::text)) || '" target="_blank"]Consulter[/link]'::text AS link␣
↪FROM sig.commune;
```

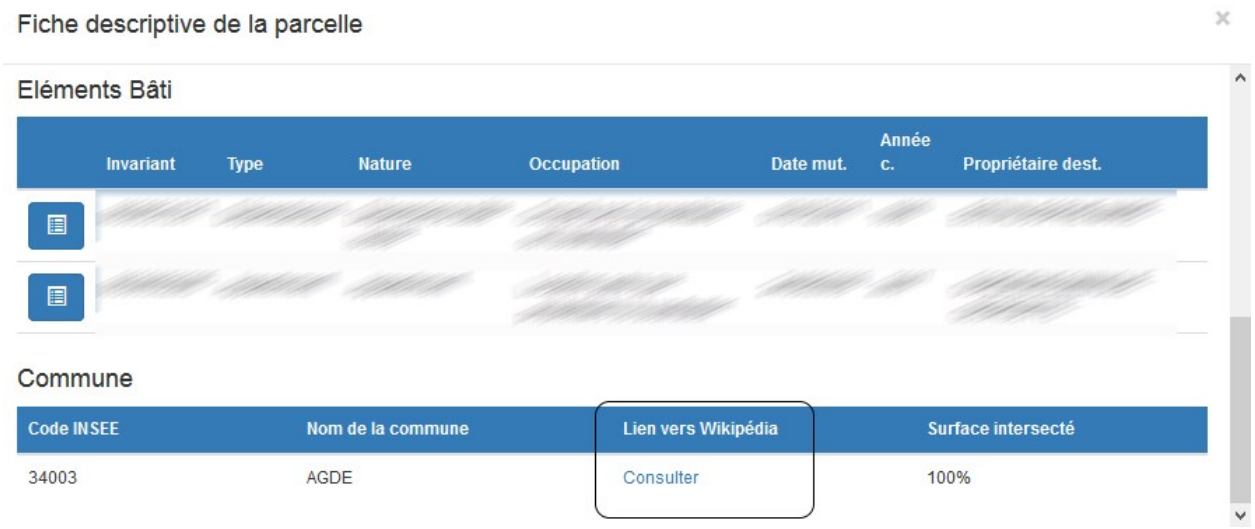
2- Affichage du lien dans les intersections

Une fois le lien construit dans la vue, ajouter une properties dans le fichier « . »\vas\rest\conf\cadastreV2\properties.inc »

Exemple :

```
$properties["cadastre"]["descr_parcel"]["intersect"]["views"]["nom_schema.nom_vue"][
↪"id_com"] = "Code INSEE"
$properties["cadastre"]["descr_parcel"]["intersect"]["views"]["nom_schema.nom_vue"][
↪"nom"] = "Nom de la commne"
$properties["cadastre"]["descr_parcel"]["intersect"]["views"]["nom_schema.nom_vue"][
↪"link"] = "Lien vers Wikipédia"
$properties["cadastre"]["descr_parcel"]["intersect"]["views"]["nom_schema.nom_vue"][
↪"intersect"] = "Surface intersecté";
```

Le rendu suivant est obtenu :



cadastre.descr_parcel.intersect.tolerance

Surface minimum à partir de laquelle une intersection est prise en compte (en %).

Fiche d'urbanisme

Properties génériques pour obtenir des informations sur les intersections. Fonctionne avec des objets ponctuels (POINT), linéaires (LINE) et polygones (POLYGON). Fonctionne avec une ou plusieurs vues et/ou une ou plusieurs tables.

Exemple :

- $properties[»cadastre »][»fiche_urb »][»intersect »][»views »][»nom_vue »][»title »] = « Le titre »
- $properties[»cadastre »][»fiche_urb »][»intersect »][»views »][»nom_vue »][»type »]= « POLYGON »
- $properties[»cadastre »][»fiche_urb »][»intersect »][»views »][»nom_vue »][»intersect »] = « Libellé surface inter. »
- $properties[»cadastre »][»fiche_urb »][»intersect »][»views »][»nom_vue »][»champ1 »] = « Libellé champ »
- $properties[»cadastre »][»fiche_urb »][»intersect »][»views »][»nom_vue »][»champ2 »] = « Libellé champ »
- $properties[»cadastre »][»fiche_urb »][»intersect »][»views »][»nom_vue »][»… »] = « Libellé champ »
- $properties[»cadastre »][»fiche_urb »][»intersect »][»views »][»… »][»title »] = « Le titre »
- etc…

Si ces informations sont mal renseignées, la fiche d'urbanisme d'une parcelle peut ne pas se générer correctement.

Il est aussi possible d'afficher ces informations sous forme de tableau de la manière suivante :

- $properties[»cadastre »][»fiche_urb »][»intersect »][»array_views »][0][»nom_vue »][»… »] = « Libellé champ 1 » ;
- $properties[»cadastre »][»fiche_urb »][»intersect »][»array_views »][0][»nom_vue »][»… »] = « Libellé champ 2 » ;
- $properties[»cadastre »][»fiche_urb »][»intersect »][»array_views »][0][»nom_vue »][»… »] = « Libellé champ 3 » ;
- $properties[»cadastre »][»fiche_urb »][»intersect »][»array_views »][1][»nom_vue 2 »][»… »] = « Libellé champ 1 » ;
- $properties[»cadastre »][»fiche_urb »][»intersect »][»array_views »][1][»nom_vue 2 »][»… »] = « Libellé champ 2 » ;
- $properties[»cadastre »][»fiche_urb »][»intersect »][»array_views »][1][»nom_vue 2 »][»… »] = « Libellé champ 3 » ;
- $properties[»cadastre »][»fiche_urb »][»intersect »][»array_views »][2][»nom_vue 3 »][»… »] = « Libellé champ 1 » ;
- $properties[»cadastre »][»fiche_urb »][»intersect »][»array_views »][2][»nom_vue 3 »][»… »] = « Libellé champ 2 » ;
- $properties[»cadastre »][»fiche_urb »][»intersect »][»array_views »][2][»nom_vue 3 »][»… »] = « Libellé champ 3 » ;

cadastre.fiche_urb.intersect.tolerance

Surface minimum à partir de laquelle une intersection est prise en compte (en %).

cadastre.fiche_urb.logo

Logo en base64 à afficher dans la fiche d'urbanisme

cadastre.fiche_urb.company

Nom du gestionnaire à afficher si il n'y a pas de logo

cadastre.fiche_urb.printtemplate_id

Identifiant du modèle d"impression à utiliser Par défaut vaut -1

cadastre.fiche_urb.map_id

Identifiant de la carte à utiliser Par défaut vaut -1

cadastre.fiche_urb.features_zoom

Pourcentage de zoom en fonction de la parcelle à utiliser Par défaut vaut 800

5.4.2 Propriétés à ne pas modifier

cadastre.selection_limit

Limite de sélection d"objets sur la carte. Par défaut vaut 100

cadastre.rp.rp_dir

Endroit où stocker les rapports générés Par défaut vaut $properties["vas_home"] . "/public/cadastreV2"

cadastre.rp.rp_url

Lien vers les rapports générés Par défaut vaut $properties["public_alias"] . "/cadastreV2"

cadastre.rp.login

Login de l"utilisateur utilisé pour générer les rapports Par défaut vaut u_vitis

cadastre.rp.pass

Mot de passe de l"utilisateur utilisé pour générer les rapports Par défaut est vide

cadastre.selection_buffer

Buffer de sélection cartographique Par défaut vaut 5

cadastre.database_projection

Projection de la base de données Par défaut vaut « EPSG :2154 »

cadastre.views.adresse

Vue utilisée pour l"objet adresse Par défaut vaut « v_vmap_nb_10_parcelle_light »

cadastre.views.commune

Vue utilisée pour l''objet commune Par défaut vaut « v_vmap_commune »

cadastre.views.description_parcelle

Vue utilisée pour l''objet description parcelle Par défaut vaut « v_vmap_nb_10_parcelle »

cadastre.views.invariant

Vue utilisée pour l''objet invariant Par défaut vaut « v_vmap_bati_0010_local »

cadastre.views.lieu_dit

Vue utilisée pour l''objet lieu dit Par défaut vaut « v_vmap_lieu_dit »

cadastre.views.parcelle

Vue utilisée pour l''objet parcelle Par défaut vaut « v_vmap_parcelle_all_geom »

cadastre.views.proprietaire

Vue utilisée pour l''objet propriétaire Par défaut vaut « v_vmap_maj_pc »

cadastre.views.section

Vue utilisée pour l''objet section Par défaut vaut « v_vmap_section_cadastrale »

cadastre.views.voie

Vue utilisée pour l''objet voie Par défaut vaut « v_vmap_maj_fv »

cadastre.view.bati

Vue utilisée pour l''objet bâti Par défaut vaut « s_cadastre.v_vmap_batiment »

CHAPITRE 6

Introduction

vMap est un puissant outil de webmapping français basé sur les meilleures solutions open-souce (Postgres/PostGIS, Mapserver, OpenLayers 3, AngularJS, Bootstrap. . .).

Utilisé comme lecteur de services cartographiques (WMS, WMTS, Bing, OSM, GPX, GeoJSON, KML, TopoJSON. . .) vMap permet aussi de publier ses propres services **WMS** par l'intermédiaire de **Mapserver**.

L'application utilise les droits **PostgreSQL** pour que chaque utilisateur dispose de ses propres droits pour accéder à la donnée de manière sécurisée.

6.1 Publier des couches interrogeables en utilisant Mapserver et WMS (GetFeatureInfo)

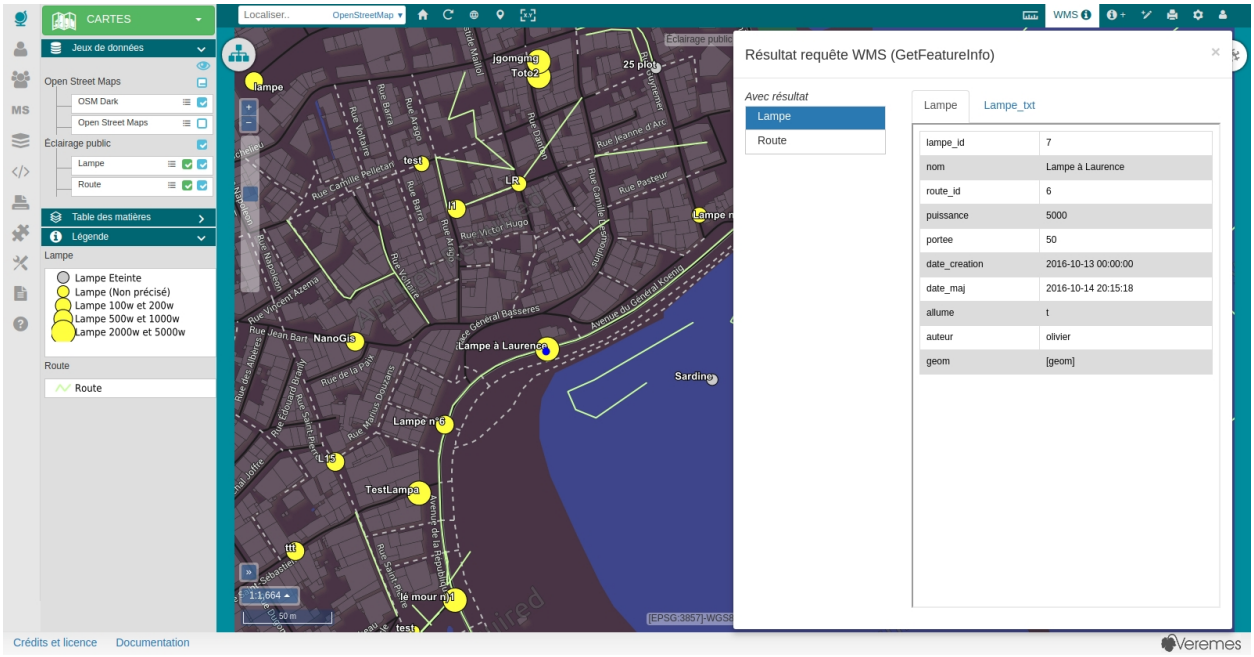
Le mode cartographique vous permet d'afficher les données attributaires retournées par la requête GetFeatureInfo directement en cliquant sur les objets de la carte

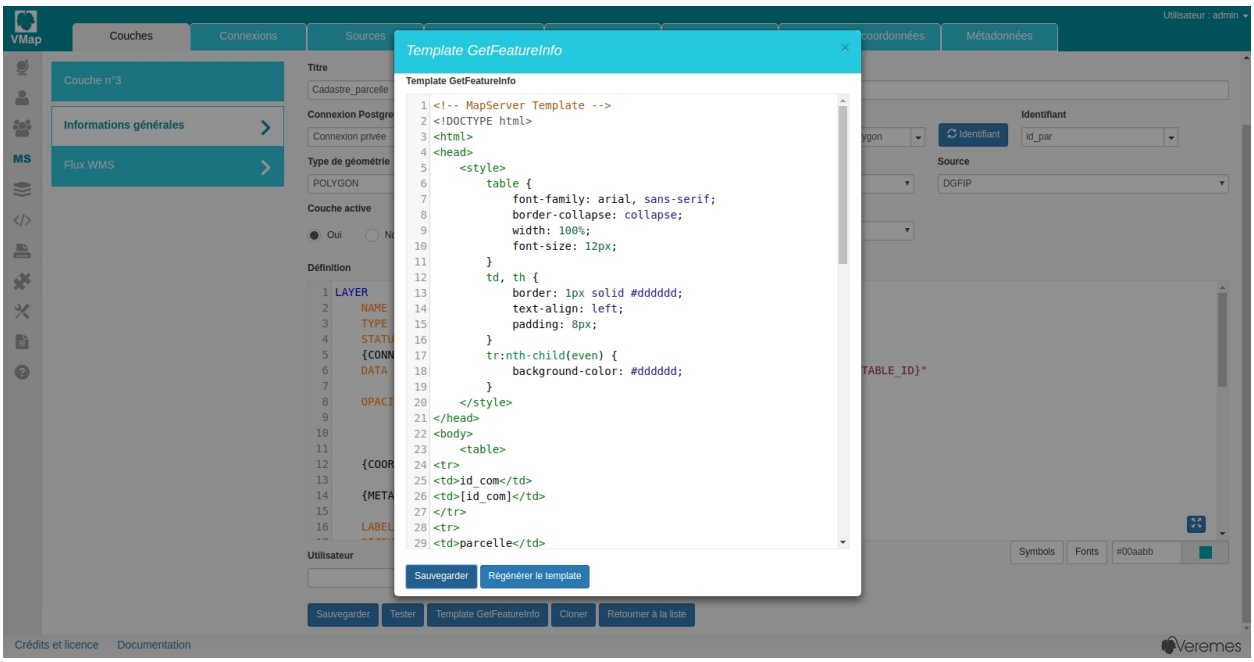
Utilisez le mode Mapserver pour créer des couches WMS interrogeables et génerer des templates HTML utilisés pour l'interrogation GetFeatureInfo

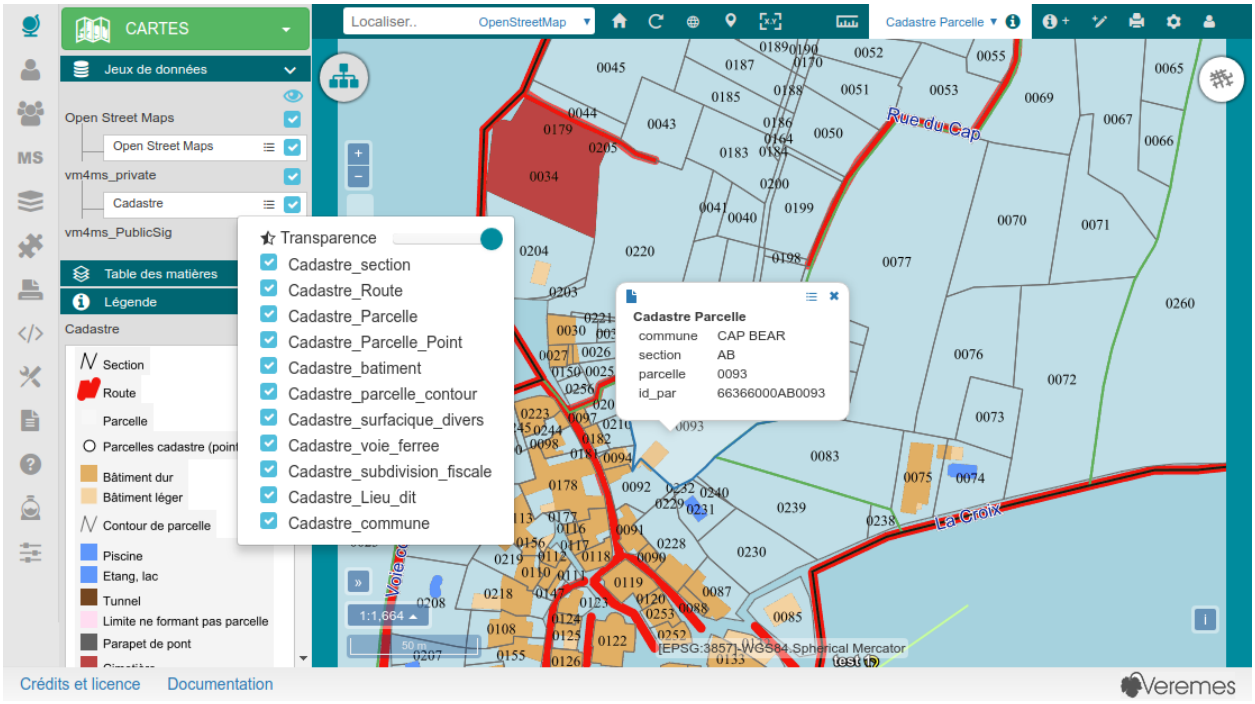
6.2 Publier des couches interrogeables en utilisant les objets métiers

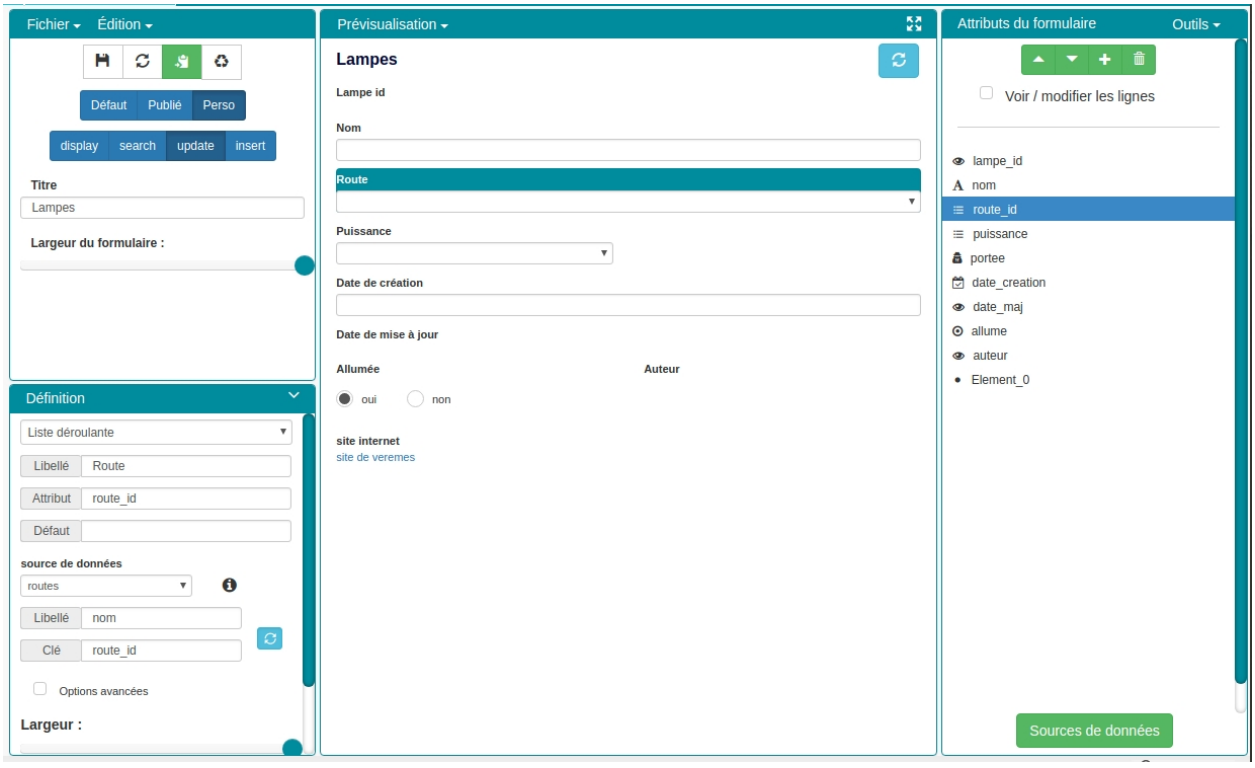
Les requêtes WMS GetFeatureInfo sont limitées, pour aller plus loin utilisez les **objets métiers vMap** qui vous permettront d'éditer la donnée en passant par des formulaires personnalisables.

Une puissante interface graphique vous permettra de créer des formulaires complexes avec plus de 30 types de champs, vous pourrez également charger de la donnée externe (Postgres, Oracle, API), utiliser des sous-objets et bien plus.


CARTES
Jeux de données
Open Street Maps
OSM Dark
Open Street Maps
Éclairage public
Lampe
Route
Table des matières
Légende
Lampe
Lampe Eteinte
Lampe (Non précisé)
Lampe 100w et 200w
Lampe 500w et 1000w
Lampe 2000w et 5000w
Route
Route
Localiser..
OpenStreetMap
WMS
Résultat requête WMS (GetFeatureInfo)
Avec résultat
Lampe
Route
Lampe
Lampe_txt
lampe_id 7
nom Lampe à Laurence
route_id 6
puissance 5000
portee 50
date_creation 2016-10-13 00:00:00
date_maj 2016-10-14 20:15:18
allume t
auteur olivier
geom [geom]
NanoGis
Lampe à Laurence
Sardine
Lampe n°8
TestLampa
Crédits et licence
Documentation
Veremes


VMap
Couches
Connexions
Sources
Métadonnées
Utilisateur : admin
Couche n°3
Informations générales
Flux WMS
Titre
Cadastre_parcelle
Connexion Postgre
Connexion privée
Type de géométrie
POLYGON
Couche active
Définition
Identifiant
Source
DGFIP
Utilisateur
Sauvegarder
Tester
Template GetFeatureInfo
Cloner
Retourner à la liste
Symbols
Fonts
Template GetFeatureInfo
Template GetFeatureInfo
<!-- MapServer Template -->
<!DOCTYPE html>
<html>
<head>
<style>
table {
font-family: arial, sans-serif;
border-collapse: collapse;
width: 100%;
font-size: 12px;
}
td, th {
border: 1px solid #dddddd;
text-align: left;
padding: 8px;
}
tr:nth-child(even) {
background-color: #dddddd;
}
</style>
</head>
<body>
<table>
<tr>
<td>id_com</td>
<td>[id_com]</td>
</tr>
<tr>
<td>parcelle</td>
Sauvegarder
Régénérer le template
Crédits et licence
Documentation
Veremes


CARTES
Jeux de données
Open Street Maps
Open Street Maps
vm4ms_private
Cadastre
vm4ms_PublicSig
Table des matières
Légende
Cadastre
Section
Route
Parcelle
Parcelles cadastre (point
Bâtiment dur
Bâtiment léger
Contour de parcelle
Piscine
Etang, lac
Tunnel
Limite ne formant pas parcelle
Parapet de pont
Localiser..
OpenStreetMap
Cadastre Parcelle
Transparence
Cadastre_section
Cadastre_Route
Cadastre_Parcelle
Cadastre_Parcelle_Point
Cadastre_batiment
Cadastre_parcelle_contour
Cadastre_surfacique_divers
Cadastre_voie_ferree
Cadastre_subdivision_fiscale
Cadastre_Lieu_dit
Cadastre_commune
Cadastre Parcelle
commune CAP BEAR
section AB
parcelle 0093
id_par 66366000AB0093
Rue du Cap
La Croix
1:1,664
[EPSG:3857]-WGS84 Spherical Mercator
Crédits et licence
Documentation
Veremes


Fichier
Édition
Défaut
Publié
Perso
display
search
update
insert
Titre
Lampes
Largeur du formulaire :
Définition
Liste déroulante
Libellé Route
Attribut route_id
Défaut
source de données
routes
Libellé nom
Clé route_id
Options avancées
Largeur :
Prévisualisation
Lampes
Lampe id
Nom
Route
Puissance
Date de création
Date de mise à jour
Allumée
Auteur
oui
non
site internet
site de veremes
Attributs du formulaire
Outils
Voir / modifier les lignes
lampe_id
nom
route_id
puissance
portee
date_creation
date_maj
allume
auteur
Element_0
Sources de données

6.3 Accrochage vectoriel en saisie

La **qualité des données** cartographiques est essentielle, pour éviter les trous et les superpositions, vous pouvez activer les fonctions d'accrochage en saisie.

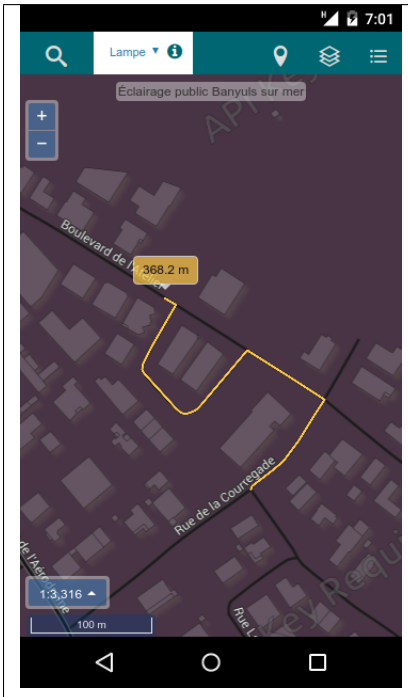
6.4 Comparaison de cartes

Vous pouvez utiliser le mode comparaison pour visualiser deux cartes différentes et effectuer des impression PDF.

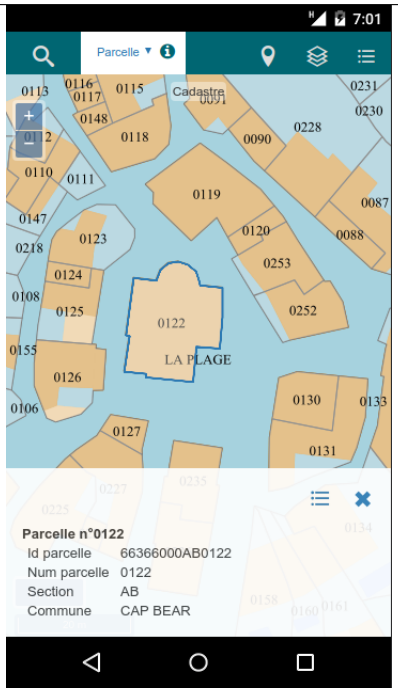
6.5 Version mobile

vMap est également disponible sur smartphones en utilisant un simple navigateur.

Vous serez alors capables d'afficher, insérer ou éditer la donnée de la même manière que sur votre ordinateur. Une connexion **GPS** est également disponible et permet de placer des points sur votre position réelle lors de la saisie.



Affichage



Interrogation

Édition

CHAPITRE 7

Liens utiles

— **Plus d'infos sur vMap :** https://www.veremes.com/produits/vmap
— **Documentation :** https://vmap.readthedocs.io/fr/latest/
— **Dépôt GitLab :** https://gitlab.veremes.net/open-source/vmap