Démarrer avec ODPI (dans Azure)

Solution Open Data Clé en main

Publication : Septembre 2014

Version: 2.1

Auteurs : Philippe Beraud

Contributeurs/Relecteurs : Rémi Olivier, Alexandre Fournier, Ludovic Panchaud

Pour les dernières informations sur la solution communautaire Open Data Clé en main, consultez

aka.ms/OpenDataFrance

Sommaire

[Licence d’ODPI 3](#_Toc398910263)

[Résumé 4](#_Toc398910264)

[Commentaires 4](#_Toc398910265)

[Prérequis 5](#_Toc398910266)

[Connaissances et attendu 5](#_Toc398910267)

[Présentation rapide de la plateforme ODPI 6](#_Toc398910268)

[Obtention des sources d’ODPI 8](#_Toc398910269)

[Organisation du code source d’ODPI 9](#_Toc398910270)

[Dossier ODPI (Open Data Platform Installer) 9](#_Toc398910271)

[Dossier *ODPI.Cloud* 10](#_Toc398910272)

[Dossier *blob files* 10](#_Toc398910273)

[Mise en place de la configuration de la souscription Azure 11](#_Toc398910274)

[Prérequis 11](#_Toc398910275)

[Etape 1 – Création du compte de stockage 11](#_Toc398910276)

[Etape 2 – Configuration du compte de stockage 15](#_Toc398910277)

[Publication d’ODPI dans Azure 18](#_Toc398910278)

[Etape 1 – Configuration des paramètres d’ODPI 18](#_Toc398910279)

[Etape 2 – Déploiement d’ODPI 18](#_Toc398910280)

[Mise à jour des solutions Open Data dans ODPI 21](#_Toc398910281)

[Etape 1 – Génération du package de service Azure d’une solution Open Data 21](#_Toc398910282)

[Pour OGDI DataLab… 22](#_Toc398910283)

[Pour ODAF Openturf… 22](#_Toc398910284)

[Pour le Portail Citoyen Open Data… 23](#_Toc398910285)

[Etape 2 – Mise à jour des fichiers dans ODPI 23](#_Toc398910286)

[Etape 3 – Mise à jour du paquet dans le compte de stockage 25](#_Toc398910287)

[Ajouter une solution Open Data dans ODPI 26](#_Toc398910288)

[Etape 1 – Ajout du package de service dans le compte de stockage blob 26](#_Toc398910289)

[Création du gabarit de configuration 26](#_Toc398910290)

[Envoi des fichiers vers le compte de stockage blobs 27](#_Toc398910291)

[Etape 2 – Création des nouveaux fichiers dans ODPI 27](#_Toc398910292)

[Etape 3 – Mise à jour des fichiers existants 31](#_Toc398910293)

[Etape 4 – Déploiement d’ODPI dans Azure 32](#_Toc398910294)

# Licence d’ODPI



**ODPI** (Open Data Platform Installer), l’un des accélérateurs de la [solution Open Data Clé en main](http://aka.ms/OpenDataCleEnMain)[[1]](#footnote-1), est publiée sous le contrat de licence de logiciel libre Microsoft Public License (Ms-PL) tel que décrit à l’adresse Internet <http://opensource.org/licenses/ms-pl.html>.

# Résumé

**ODPI** (Open Data Platform Installer), l’un des accélérateurs de la solution communautaire Open Data Clé en main, vise à permettre à toutes personnes, même les moins techniques, d’installer simplement et rapidement des solutions Open Data complexes conçues pour les fonctionnalités [Sites Web](http://azure.microsoft.com/fr-fr/services/websites/)[[2]](#footnote-2) (ASP.NET, PHP et Node.js), [Cloud Services](http://azure.microsoft.com/fr-fr/services/cloud-services/)[[3]](#footnote-3) et de l’environnement d’exécution et d’hébergement [Microsoft Azure](http://azure.microsoft.com/fr-fr/services/websites/)[[4]](#footnote-4) dans le cloud.

En ce sens, cette plateforme d’installation automatisée vise à offrir une installation simplifiée dans l’esprit de ce qui est proposé pour les sites Web Azure (ASP.NET, Java, PHP, Node.js et Python), non seulement pour ce type de soluton mais également vis-à-vis de solutions (workload) potentiellement plus complexes faisant appel, le cas échéant, à de multiples « Web role » et « worker role ».

Cette plateforme personnalisable permet ainsi de déployer avec un minimum d’effort des environnements comme les accélérateurs de la solution communautaire Open Data Clé en main.

Ce document détaille l’ensemble du processus de mise en œuvre et de publication sur Azure d’ODPI.

Un accompagnement pas-à-pas est ainsi proposé pour toutes les étapes du processus, que ce soit en termes de prérequis, de mise à jour des packages à installer par le biais de la plateforme, du déploiement en tant que telle de la plateforme sur Azure, etc.

Ce document aborde ainsi comment configurer et publier la plateforme ODPI dans votre souscription Azure. Ceci correspond à l’étape ultime avant d’avoir une solution entièrement fonctionnelle et opérationnelle.

De même, il s’intéresse à la mise à jour des accélérateurs de la solution Open Data Clé en main proposées par défaut en installation par ODPI et décrit comment ajouter une solution personnalisée au sein de la plateforme ODPI afin d’illustrer comme faire évoluer la plateforme en même temps que les solutions qu’elle propose à l’installation.

# Commentaires

Votre opinion nous intéresse. N’hésitez pas à nous transmettre vos commentaires via la boîte aux lettres Open Data France (<mailto:ogdifrance@live.fr>) afin que nous puissions améliorer la solution communautaire Open Data Clé en main.

Le [blog MSDN Open Data France](http://blogs.msdn.com/b/ogdifrance/)[[5]](#footnote-5) vous tient informé dans le même temps des dernières évolutions de la solution Open Data Clé en main, des différents accélérateurs proposés dans ce contexte, de l’actualité et des nouveautés relatives à ceux-ci.

# Prérequis

La plateforme d’installation automatisée ODPI à destination de l’environnement d’exécution Azure nécessite un ordinateur sous Windows Server (R2) 2008 ou ultérieur.

Le lecteur doit connaître les technologies des services Web RESTful, le langage C# et .NET Framework afin de pouvoir facilement comprendre les éléments mis ainsi à disposition et les éléments de configuration associés.

# Connaissances et attendu

La mise en œuvre d’ODPI requiert une expérience préalable :

* De l’environnement d’exécution et d’hébergement Microsoft Azure,
* De l’environnement de développement Microsoft Visual Studio 2013,
* De la programmation avec le Microsoft Framework .NET,
* Du développement et de l’utilisation d’interfaces Web de style REST (Representational State Transfer),
* Des dernières technologies Web comme ASP.NET MVC ou encore jQuery.

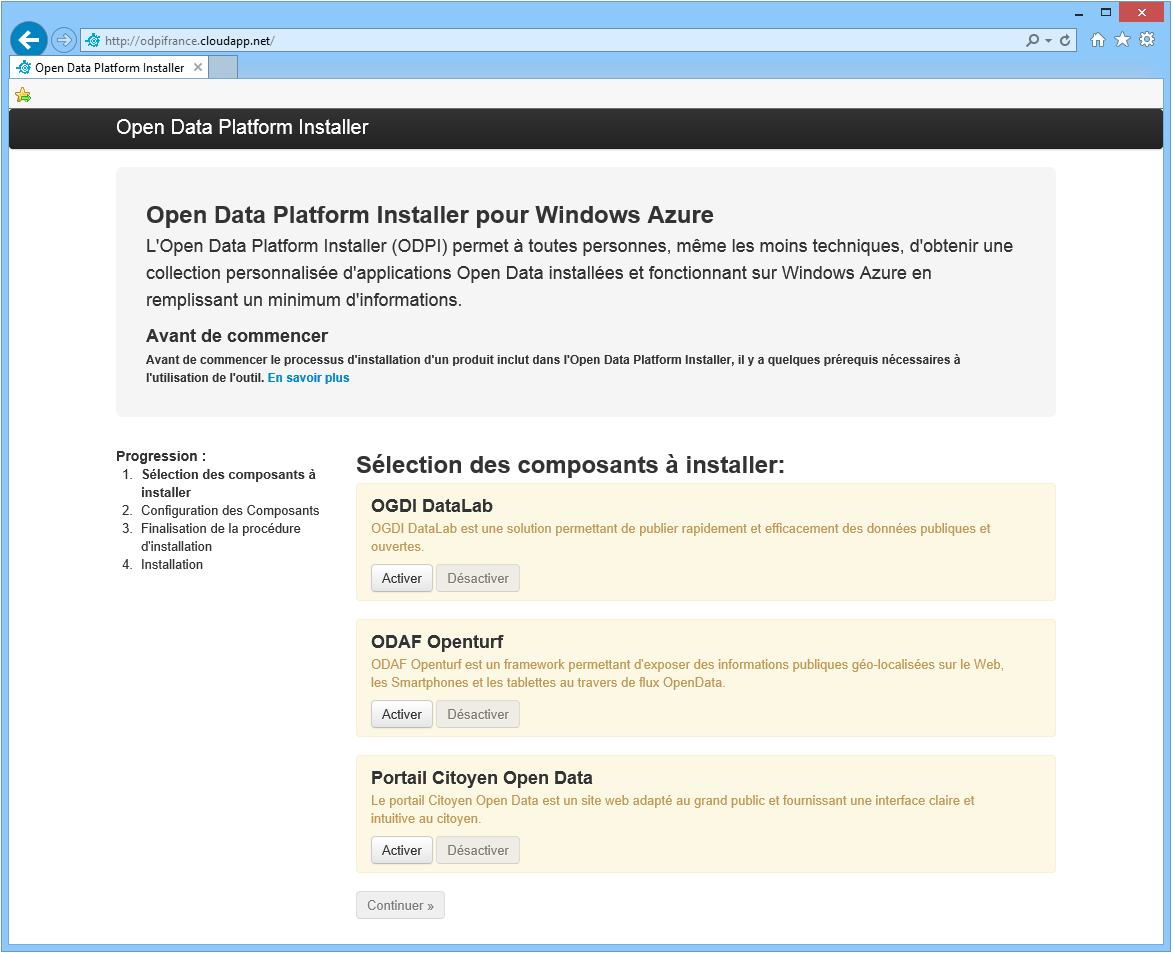
Remarque Microsoft met à disposition des formations gratuites téléchargeables sur ces environnements et technologies. Nous vous invitons à considérer le cas échéant le kit de formation [Azure Readiness Content](http://www.microsoft.com/en-us/download/details.aspx?id=8396)[[6]](#footnote-6) couvrant l’environnement Azure et l’environnement de développement Microsoft Visual Studio 2013. Ce dernier comprend des présentations, des vidéos de formation, des démonstrations accompagnées de leur script ainsi que des ateliers didactiques complets.

Remarque Le [blog du groupe Produit Microsoft Azure](http://blogs.msdn.com/b/windowsazure/)[[7]](#footnote-7) propose également de nombreuses ressources en termes de formation sur cet environnement d’exécution et d’hébergement. Celui-ci constitue également un excellent moyen de se tenir informé des évolutions proposées par l’environnement.

# Présentation rapide de la plateforme ODPI

La plateforme d’installation automatisée **ODPI** (Open Data Platform Installer) vise à permettre à toutes personnes, même les moins techniques, d’installer simplement et rapidement des solutions Open Data complexes conçues pour les fonctionnalités Sites Web (ASP.NET, PHP et Node.js), Cloud Services et de l’environnement d’exécution et d’hébergement Azure dans le cloud.

Vous pouvez tester une instance de cette plateforme ODPI à l’adresse <http://odpifrance.cloudapp.net>.



Dans ce contexte, la plateforme ODPI gère pour vous la configuration et le déploiement de ces solutions ainsi proposées à destination de Azure moyennant une saisie d’informations simplifiée. S’adressant en premier lieu à un public peu technique, la plateforme ODPI n’en demeure pas moins intéressante pour les développeurs et les intégrateurs désireux d’optimiser les déploiements et/ou de gagner du temps dans leur développement/personnalisation d’une plateforme/portail de données ouvertes.

La plateforme ODPI propose par défaut plusieurs solutions permettant d’entrer de plein pied dans le mouvement Open Data. Il s’agit en l’occurrence:

* De la plateforme de publication de données ouvertes **OGDI** (Open Government Data Initiative) **DataLab** avec sa gestion multi-catalogue et son service de donnée RESTful conforme au protocole ouvert de donnée OData,
* Du Framework applicatif **ODAF** (Open Data Application Framework) **Openturf** à destination des environnements mobiles,
* Et du **Portail** **Citoyen Open Data** permettant d’offrir une interface simple et intuitive aux citoyens.

[image](https://github.com/openlab/DataLab)

Ces solutions sont disponibles indépendamment sous licence libre [Microsoft Public License (MS-PL)](http://opensource.org/licenses/ms-pl.html)[[8]](#footnote-8) sur la [forge GitHub](https://github.com/openlab)[[9]](#footnote-9).

Pour plus de renseignements sur la plateforme ODPI, nous vous invitons à visiter le blog MSDN [OGDI France](http://blogs.msdn.com/b/ogdifrance/)[[10]](#footnote-10) et plus particulièrement le billet [ODPI, la facilité de mise en œuvre des solutions Open Data](http://blogs.msdn.com/b/ogdifrance/archive/2012/07/13/odpi-la-facilit-233-de-mise-en-uvre-des-solutions-open-data.aspx)[[11]](#footnote-11).

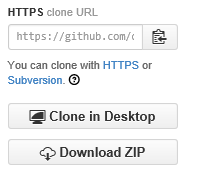
# Obtention des sources d’ODPI

L’accélérateur ODPI de la solution communautaire Open Data Clé en main est disponible sur la forge GitHub à l’adresse [https://github.com/openlab/](https://github.com/openlab/OGDI-DataLab)ODPI.

Les solutions communautaires évoluant avec l’investissement de la communauté, il est souhaitable de vos assurer que vous disposez de la dernière version, ne serait-ce que pour bénéficier, le cas échéant, de l’ensemble des correctifs.

[image](https://github.com/openlab/DataLab)

Pour télécharger les sources depuis la forge GitHub, le moyen le plus simple et rapide est de cliquer directement sur **Download ZIP**.



S’il constitue le moyen le plus simple, les autres options se révèlent plus adaptées pour mettre en place une gestion de versions.



A ce titre, nous vous invitons à consulter les billets [OGDI DataLab sur GitHub ! Impliquez-vous dans la communauté](http://blogs.msdn.com/b/ogdifrance/archive/2012/05/14/ogdi-datalab-sur-github-impliquez-vous-dans-la-communaut-233.aspx)[[12]](#footnote-12) et [GitHub for Windows](http://blogs.msdn.com/b/ogdifrance/archive/2012/06/14/github-for-windows.aspx)[[13]](#footnote-13), tous deux présents sur le blog MSDN [Open Data France](http://aka.ms/OpenDataFrance)[[14]](#footnote-14).

Par ailleurs, L’environnement Visual Studio 2013 propose une intégration directe avec GitHub.

Remarque La version 2012 suppose d’installer avec les [outils Visual Studio pour Git](http://visualstudiogallery.msdn.microsoft.com/abafc7d6-dcaa-40f4-8a5e-d6724bdb980c)[[15]](#footnote-15). L’article [Integrating and Using Github in Visual Studio 2012](http://www.codeproject.com/Articles/581907/IntegratingplusandplusUsingplusGithubplusinplusVis)[[16]](#footnote-16) vous propose un guide pas-à-pas dans le contexte de cette version.

Quelle que soit l’approche retenue, et une fois, les sources téléchargées, nous vous invitons à passer à la section suivante.

# Organisation du code source d’ODPI

L’intégralité du code source de la plateforme ODPI est située dans le sous-répertoire *Source* du répertoire d’extraction du fichier d’archive *ODPI.zip*.

Ce répertoire comprend 3 sous-dossiers principaux :

1. Le dossier *ODPI* qui correspond au site Web de la plateforme d’installation qui sera déployée.

Il regroupe différents sous-dossiers qui constituent, au final, l’application Web ASP.NET MVC destinée à l’environnement Azure et qui permet de visualiser les données provenant d’un service RESTful OData;

1. Le dossier *ODPI.Cloud* qui est un projet, comme son nom le laisse supposer, permettant de déployer le projet ODPI décrit précédemment dans Azure;
2. Le dossier *blob files* qui contient l’ensemble des packages de service des accélérateurs Open Data préconfigurées de la solution Open Data Clé en main et proposées à l’installation sur le site Web ODPI que vous déploierez.

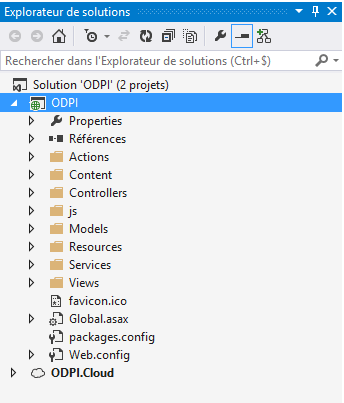
Par défaut, ces accélérateurs à destination des services de cloud computing (PaaS) de Azure sont au nombre de 3, en l’occurrence OGDI DataLab, ODAF Openturf et le Portail Citoyen Open Data.

La suite de cette section décrit les dossiers/éléments de configuration de ces différents sous-dossiers de façon à mieux cerner la solution proposée par ODPI au travers du fichier de solution *ODPI.sln* pour Visual Studio 2013 et ainsi vous l’approprier plus facilement en vue par exemple de sa personnalisation (légère ou avancée).

## Dossier ODPI (Open Data Platform Installer)

Cette première composante de la solution correspond à la plateforme d’installation automatisée sous forme d’une application Web ASP.NET MVC destinée aux services de cloud computing de l’environnement Azure.

Le dossier *ODPI* est composé des 8 dossiers suivants :



| **Dossier** | **Description** |
| --- | --- |
| Actions | Ce dossier contient les actions qui seront effectuées lors de la configuration et du déploiement des solutions Open Data déclarées et au nombre de 3 par défaut. Il contient notamment les classes de validation des connexions aux sources de données et les actions à effectuer une fois le déploiement réussi. |
| Content | Ce dossier est un dossier classique d’un site Web ASP.NET MVC, il contient les images et feuilles de styles. |
| Controllers | Ce dossier est un dossier classique d’un site Web ASP.NET MVC. |
| Js | Ce dossier est un des points clé de la plateforme ODPI. C’est à travers les différents fichiers JavaScript qu’il contient que l’interaction avec l’utilisateur final est possible. |
| Models | Au-delà de son utilité dans l’architecture ASP.NET MVC, ce dossier contient la définition et la configuration à adopter des différentes solutions Open Data qui seront proposées à l’installation sur le site Web ODPI. |
| Resources | Ce dossier contient l’ensemble des fichiers de localisation. Le site est actuellement entièrement traduit en français et en anglais en fonction de la langue du navigateur. |
| Services | Ce dossier contient la couche service qui permet, entre autres, la génération des certificats nécessaires à l’identification sur le portail Azure et qui permet également de configurer les packages en fonction d’un gabarit (template) précis puis de déployer dans Azure. |
| Views | Ce dossier est un dossier classique d’un site Web ASP.NET MVC. Il contient l’ensemble des pages accessibles en fonction de la route demandée. |

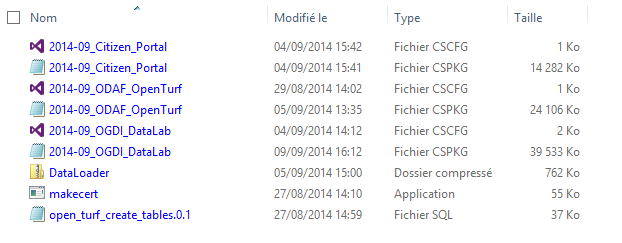
## Dossier *ODPI.Cloud*

Le dossier *ODPI.Cloud* correspond au projet permettant de déployer l’application ODPI dans Azure. Il référence donc le projet *ODPI* décrit précédemment et permet de le configurer.

L’ensemble des paramètres de configuration de la solution ODPI est regroupé au niveau des fichiers de configuration de services *ServiceConfiguration.Cloud.cscfg* et *ServiceConfiguration.Local.cscfg* du projet Azure *ODPI.Azure* relatif au rôle *ODPI*.

## Dossier *blob files*

Le dossier *blob files* contient l’ensemble des packages de service Azure relatifs aux solutions Open Data préconfigurées et proposées à l’installation par le site Web ODPI.



Il contient notamment les fichiers suivants :

* Pour OGDI DataLab : *2014-09\_OGDI\_DataLab.cscfg* et *2014-09\_OGDI\_DataLab.cspkg* et *DataLoader.zip*.
* Pour ODAF Openturf : *2014-09\_ODAF\_Openturf.cscfg*, *2014-09\_ODAF\_Openturf.cspkg* et *open\_turf\_create\_tables.0.1.sql*.
* Pour le Portail Citoyen Open Data : *2014-09\_Citizen\_Portal.cscfg* et *2014-09\_Citizen\_Portal.cspkg*.

Nous y revenons dans la suite de ce document.

# Mise en place de la configuration de la souscription Azure

## Prérequis

La mise en œuvre de la plateforme ODPI suppose au préalable simplement une souscription Azure avec un compte Microsoft valide.

Vous pouvez accéder à Azure avec un paiement à l’usage, sans engagement, ou avec des offres d’abonnements, ou encore dans le cadre d’un accord d’entreprise Microsoft comme décrit à l’adresse <http://azure.microsoft.com/fr-fr/pricing/purchase-options/>.

Si vous ne disposez pas d’ores et déjà d’un tel compte, naviguez simplement vers <http://aka.ms/free/trial> et cliquez sur **essayer-la gratuitement** (en vert) pour vous créer un compte de test. Un assistant vous guide alors dans les différentes étapes.



## Etape 1 – Création du compte de stockage

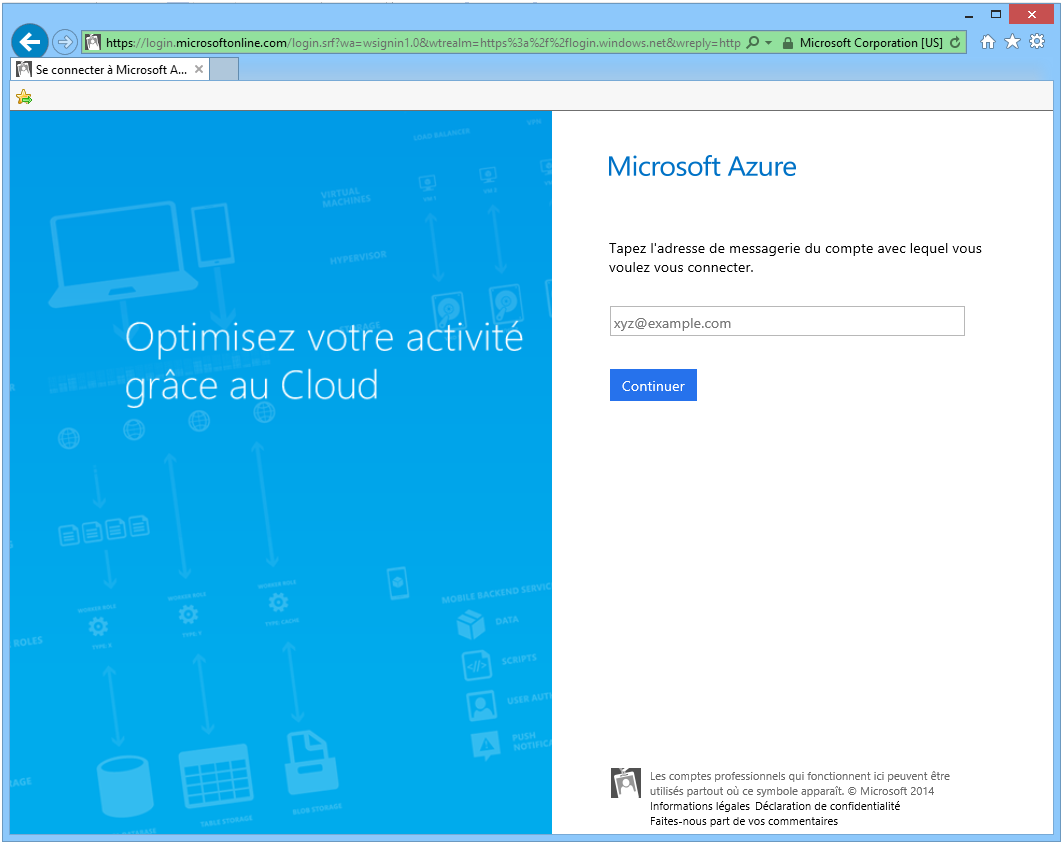
La plateforme ODPI nécessite 1 compte de stockage Azure pour son fonctionnement.

Ce compte de stockage à travers le [service de blobs Azure](http://msdn.microsoft.com/en-us/library/dd135733)[[17]](#footnote-17) permet de stocker les packages de service Azure contenant les solutions Open Data proposées à l’installation ainsi que leurs gabarits associés.

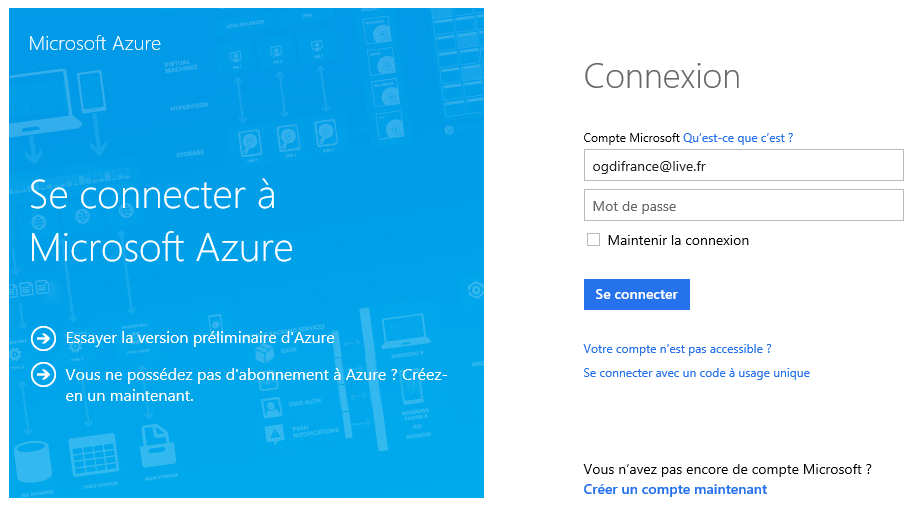
Lors de cette première étape, vous allez créer le compte de stockage utilisé par la plateforme ODPI depuis le portail de gestion Azure.

Pour créer le compte de stockage, procédez comme suit :

1. Ouvrez votre navigateur, naviguez vers <http://manage.windowsazure.com/>.

****

1. Précisez l’adresse mèl de votre compte Microsoft relatif à votre souscription Microsoft Azure et cliquez sur **Continuer**.



1. Précisez le mot de passe de votre compte Microsoft lorsque vous y êtes invité(e) et cliquez sur **Se connecter**.
2. Une fois dans le portail de gestion Azure, la première chose à faire est de créer un groupe d’affinité.

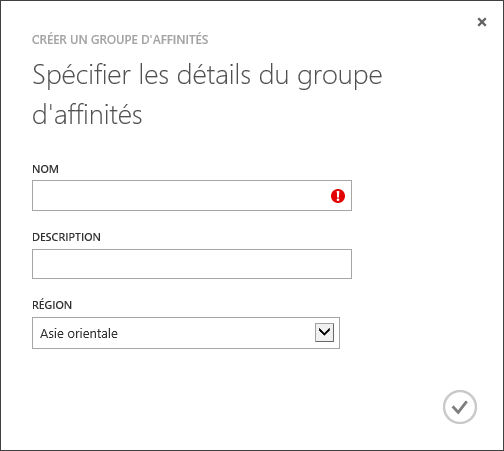
Remarque Vous créez un groupe d’affinités ici de façon à pouvoir ensuite déployer la plateforme ODPI dans la même localisation géographique, ceci afin d’optimiser la bande passante et de réduire la latence entre les applications et les données dont elles dépendent.

Cliquez sur **PARAMETRES** au bas du bandeau gauche. Le volet **paramètres** s’affiche. Cliquez sur **GROUPES d’AFFINITES**.

Si vous n’avez pas encore ajouté un groupe d’affinité, le volet doit être similaire à la capture suivante :



1. Cliquez sur le lien **AJOUTER UN GROUPE D’AFFINITES** (ou cliquez sur **AJOUTER** dans le bandeau en bas du portail). Le dialogue suivant s’ouvre.

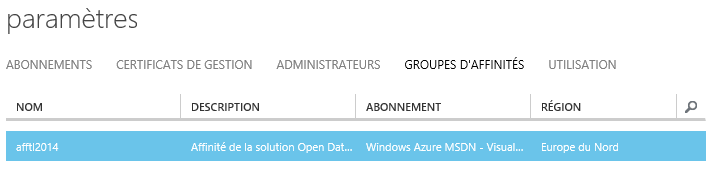


1. Renseignez les champs comme suit et cliquez sur la coche :

* **NOM** : choisissez un nom pour votre groupe, par exemple *afftl2014*
* **DESCRIPTION** : Affinité pour ODPI
* **REGION : Europe du Nord** étant conseillé pour un accès depuis la France métropolitaine

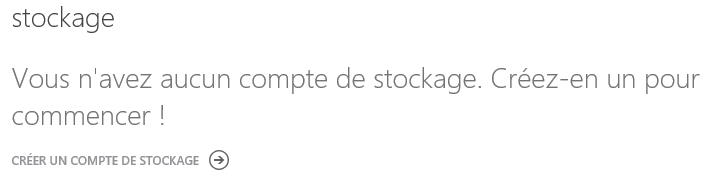
**Europe du Nord** correspond au centre de données de Dublin, **Europe de l’Ouest** à celui d’Amsterdam.

Après un temps de chargement pouvant durer quelques minutes, la création du groupe devient effective. A l’issue de la création du compte, le statut suivant s’affiche.



1. Maintenant, pour créer le compte de stockage, cliquez sur **STOCKAGE** au milieu du bandeau gauche.

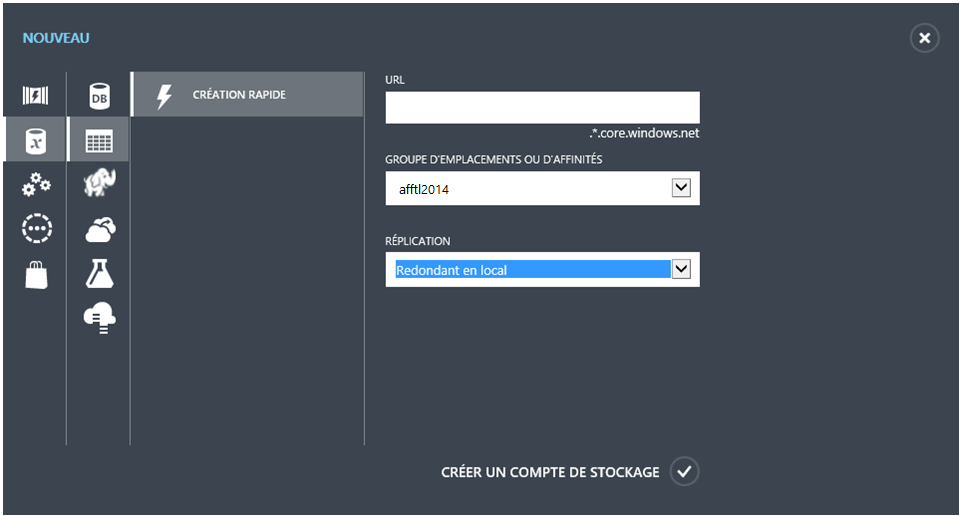
Si vous ne disposez d’aucun compte de stockage, le volet **stockage** doit être similaire à la capture suivante :



1. Cliquez sur le lien **CREER UN COMPTE DE STOCKAGE** (ou cliquez sur **NOUVEAU** dans le bandeau en bas du portail). Le panneau **NOUVEAU** s’ouvre dans le bas du portail d’administration Microsoft Azure.



1. Cliquez sur **CREATION RAPIDE**.



1. Renseignez les champs comme suit et cliquez sur **CREER UN COMPTE DE STOCKAGE** :

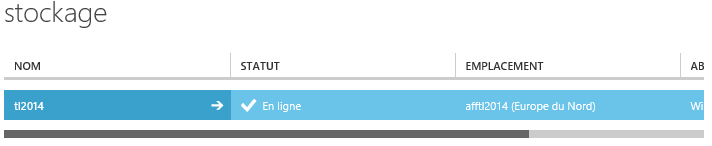
* **URL** : entrez le nom de votre compte de stockage, par exemple *tl2014*

Remarque Azure va utiliser cette valeur pour générer les URLs servant de points d’entrée aux services (tables, blobs et files d’attente) du compte de données. Attention, ce nom doit être unique, contenir entre 3 et 24 caractères et n’utiliser que des lettres en minuscule et des chiffres !

Remarque Le portail s'assure que le nom est valide en vérifiant i) que le nom est conforme aux règles de noms et ii) qu’il est actuellement disponible. Une erreur de validation sera affichée si vous entrez le nom qui ne satisfait pas les règles.

* **GROUPE DE REGIONS OU D’AFFINITES** : Sélectionnez le groupe d’affinité précédemment créée
* **REPLICATION**: **Redondant en local**

A l’issue de la création du compte, le statut suivant s’affiche.

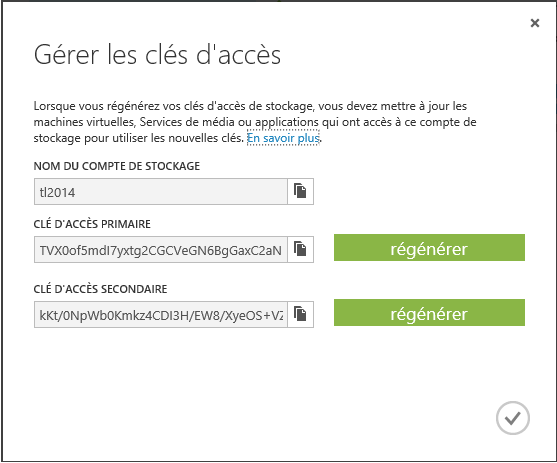


1. Une fois le compte de stockage créé, cliquez sur **GERER LES CLES D’ACCES** dans le bas du portail pour voir les clés associées à ce compte.



Remarque Le bouton GERER LES CLES D’ACCES donne accès à la clé primaire du compte nécessaire pour pouvoir accéder à ces services via les [API REST de gestion des services](http://msdn.microsoft.com/fr-fr/library/ee460799.aspx)[[18]](#footnote-18) Azure.

Le dialogue **Gérer les clés d’accès** s’ouvre.



Votre compte de stockage est à présent créé. Vous devez maintenant le préparer pour fonctionner avec la plateforme ODPI.

## Etape 2 – Configuration du compte de stockage

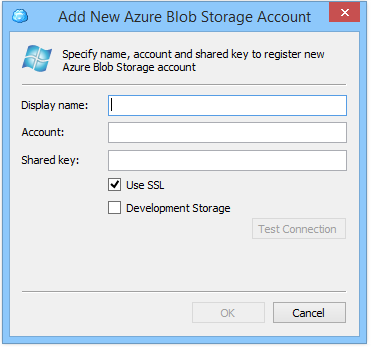
La plateforme ODPI nécessite 1 seul compte de service en plus d’un compte de stockage pour s’exécuter. Le compte de stockage sera utilisé pour ses fonctionnalités blobs destinées à héberger les packages de service des accélérateurs de la solution Open Data Clé en main ainsi que leurs gabarits.

Dans la suite de cette étape, vous aurez besoin d’un outil permettant d’explorer et de gérer les stockages blobs dans Azure.

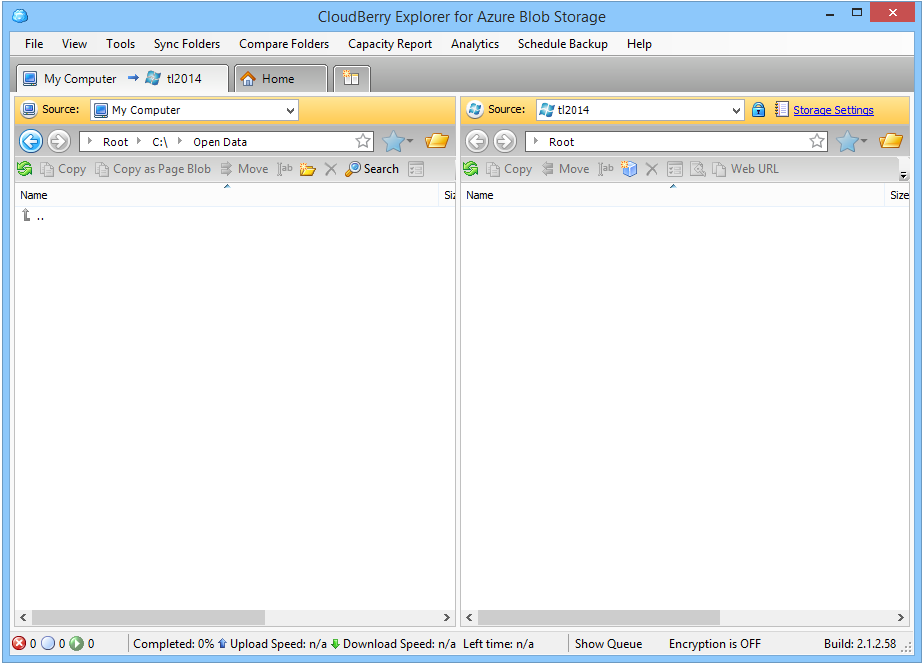
Nous utiliserons ici l’application [CloudBerry Explorer For Microsoft Azure](http://www.cloudberrylab.com/free-microsoft-azure-explorer.aspx)[[19]](#footnote-19) qui a l’avantage d’être gratuite (dans va version de base) en plus d’être parfaitement adaptée à notre besoin.

Procédez comme suit :

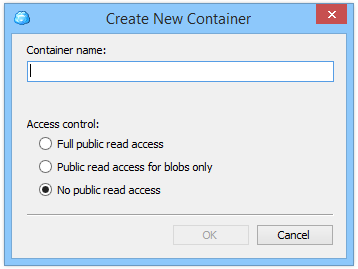
1. Exécutez **CloudBerry Explorer for Azure Blob Storage**.
2. Dans **CloudBerry Explorer**, dans la liste déroulante **Source**, cliquez sur **<New Azure Blob Storage Account>** pour ajouter votre compte de stockage précédemment créé. La fenêtre **Add New Azure Blob Storage Account** s’ouvre.



1. Remplissez les informations nécessaires. **Shared key** correspond à la clé d’accès primaire visualisé dans l’étape 11 de la section précédente (champs **CLE D’ACCES PRIMAIRE**).
2. Cliquez sur **OK** pour vous connecter et explorer votre compte de stockage qui pour l’instant est vide.



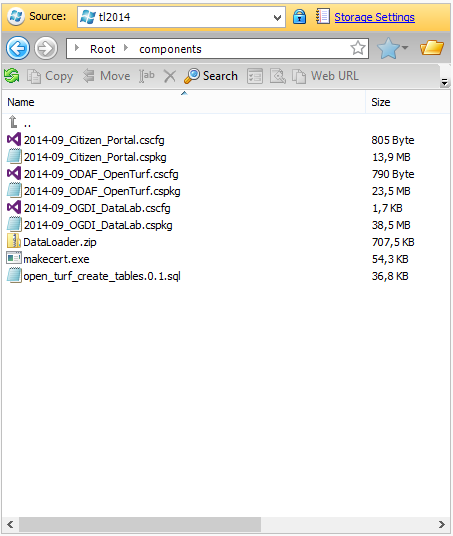
1. Vous allez à présent créer un conteneur pour héberger les packages de service des services cloud computing de Azure contenant les accélérateurs de la solution Open Data à proposer. Pour cela, créez un conteneur nommé **components** en cliquant sur le bouton en forme de cube. Un dialogue **Create New Container** s’ouvre.



1. Renseignez les champs comme suit et cliquez sur **CREER UN COMPTE DE STOCKAGE** :

* **Container name** : **components**
* **Access control** : **No public read access**

1. A présent, faîtes glisser l’ensemble des fichiers du dossier *blob files* fournis avec le code source de la solution ODPI dans le conteneur *components*.



Votre compte de stockage est à présent configuré pour permettre à la plateforme ODPI une fois déployée de déployer à son tour les accélérateurs de la solution Open Data Clé en main ainsi proposées au travers des packages de service copiés.

# Publication d’ODPI dans Azure

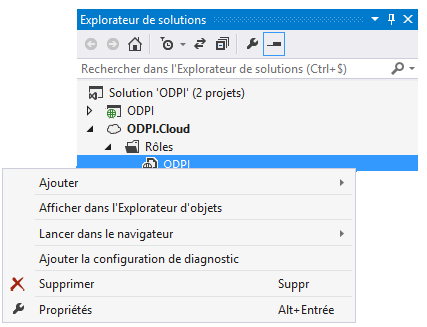
Vous allez à présent aborder les différentes étapes visant à déployer totalement votre solution ODPI dans votre souscription Azure.

## Etape 1 – Configuration des paramètres d’ODPI

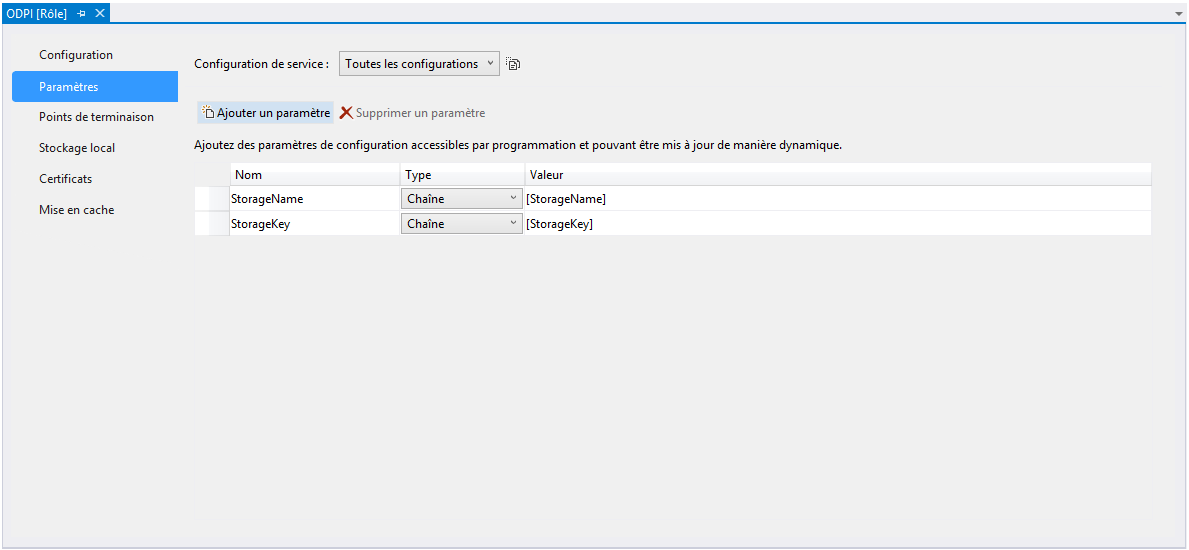
La configuration centralisée de la plateforme ODPI regroupe les éléments nécessaires au fonctionnement du site Web et notamment au bon déploiement des solutions.

Pour correctement configurer la plateforme d’installation ODPI à déployer, procédez comme suit :

1. Dans Visual Studio 2013, dans **Explorateur de solutions**, avec la solution *ODI.sln* ouverte, développez le dossier **Rôles** du projet *ODPI.Cloud*. Ce projet de type Azure vous sert à déployer le projet *ODPI*.
2. Faites un clic droit sur *ODPI* sous le dossier **Rôles**, puis sélectionnez **Propriétés.**



1. Un assistant de configuration s’ouvre. Sélectionnez l’onglet **Paramètres**.



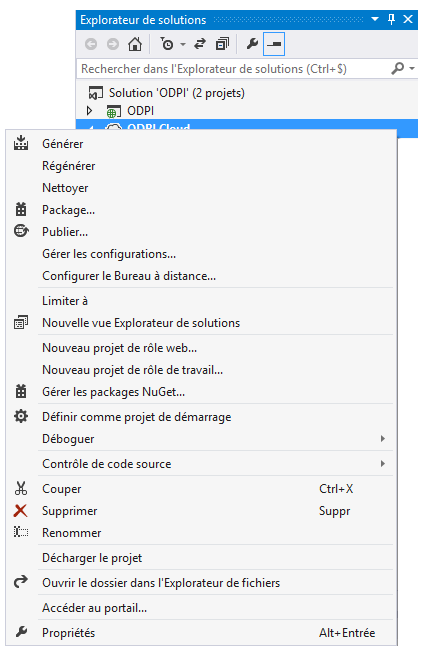
1. Saisissez les informations du compte de stockage créé à la section § Etape 1 – Création du compte de stockage en page 11 et contenant à présent vos packages de service pour les solutions Open Data à installer de façon automatisée via la plateforme.

## Etape 2 – Déploiement d’ODPI

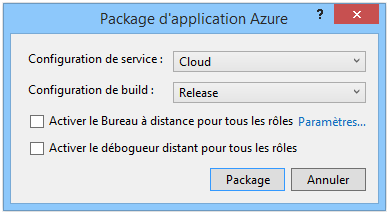
Vous venez de configurer la plateforme d’installation ODPI. Dans cette étape, vous allez procéder au déploiement effectif de la plateforme d’installation ODPI dans votre compte de service Azure associé. Pour cela, il est nécessaire de créer dans un premier temps un package de service qui sera ensuite déployé via le portail de gestion Azure.

Procédez comme suit :

1. Toujours dans Visual Studio 2013, faites un clic-droit sur le projet*ODPI.Cloud* et choisissez **Package**dans le menu contextuel.

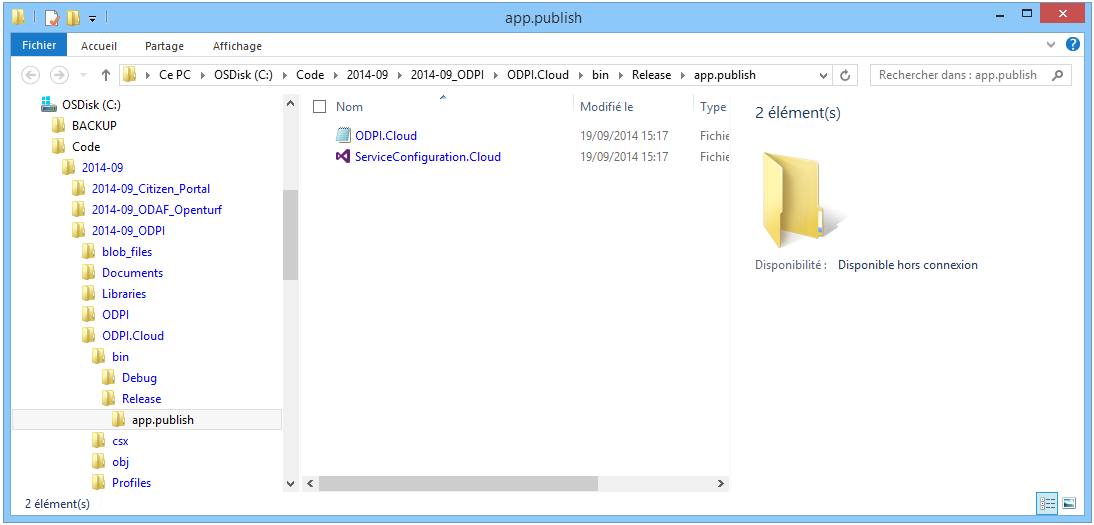


1. Dans la boite de dialogue **Package d’application Azure**, choisissez les options **Configuration de service** : **Cloud**, **Configuration de build** : **Release** et cliquez sur **Package** pour confirmer.



1. Une fois le package de service créé, une fenêtre Windows s’ouvre dans le dossier app.*publish* correspondant. Ce dossier contient le package *ODPI.Cloud.cspkg* et le fichier de configuration *ServiceConfiguration.Cloud.cscfg*associé.

Vous allez utiliser les fichiers correspondants dans les étapes suivantes. Mémorisez donc le chemin vers ce dossier.



1. Il ne vous reste plus qu’à déployer ce package de service via le portail de gestion Azure accessible depuis l’adresse <https://manage.windowsazure.com/>.

# Mise à jour des solutions Open Data dans ODPI

Vous allez à présent aborder les différentes étapes visant à mettre à jour votre solution de la plateforme d’installation ODPI dans votre souscription Azure.

Ceci permet de prendre en compte des modifications effectuées et/ou des évolutions introduites dans une ou plusieurs solutions Open Data déjà présentent dans la distribution par défaut de la plateforme ODPI.

## Etape 1 – Génération du package de service Azure d’une solution Open Data

Cette étape décrit comment réobtenir un package de service dans le cadre d’une mise à jour de la version d’une solution Open Data déjà présente dans la distribution par défaut de la plateforme ODPI.

Dans un premier temps, vous devez récupérer le code source de la solution à mettre à jour.

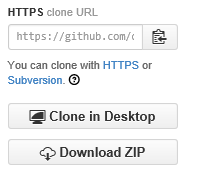
Par défaut, comme indiqué précédemment, la plateforme ODPI propose les 3 autres accélérateur de la solution Open Data Clé en main à installer dans les services cloud computing de Azure, à savoir :

* Le projet OGDI DataLab disponible sur la forge GitHub à l’adresse <https://github.com/openlab/OGDI-DataLab>,
* Le projet ODAF Openturf disponible également sur la forge GitHub à l’adresse  <https://github.com/openlab/ODAF-Openturf>,
* Et le projet Portail Citoyen Open Data disponible aussi sur la forge GitHub à l’adresse  <https://github.com/openlab/CitizenPortal>.

Les solutions communautaires évoluant avec l’investissement de la communauté, il peut s’avérer en effet souhaitable de disposer de la dernière version de ces projets, ne serait-ce que pour bénéficier, le cas échéant, de l’ensemble des correctifs.

[image](https://github.com/openlab/DataLab)

Pour télécharger les sources depuis la forge GitHub, le moyen le plus simple et rapide est de cliquer directement sur le bouton ZIP.



S’il constitue le moyen le plus simple, les autres options se révèlent plus adaptées pour mettre en place une gestion de versions.

Pour plus d’informations sur l’utilisation détaillée de solutions Open Data avec GitHub telles qu’OGDI DataLab ou ODAF Openturf, nous vous invitons à consulter les billets [OGDI DataLab sur GitHub ! Impliquez-vous dans la communauté](http://blogs.msdn.com/b/ogdifrance/archive/2012/05/14/ogdi-datalab-sur-github-impliquez-vous-dans-la-communaut-233.aspx)[[20]](#footnote-20) et [GitHub for Windows](http://blogs.msdn.com/b/ogdifrance/archive/2012/06/14/github-for-windows.aspx)[[21]](#footnote-21), tous deux présents sur le blog MSDN [OGDI France](http://blogs.msdn.com/b/ogdifrance/)[[22]](#footnote-22).

Les sources téléchargées, il vous faut à présent générer un package de service Azure qui servira de base lors d’une installation automatisée via la plateforme ODPI.

Selon la solution choisie, le processus de génération du package de service Azure est différent.

### Pour OGDI DataLab…

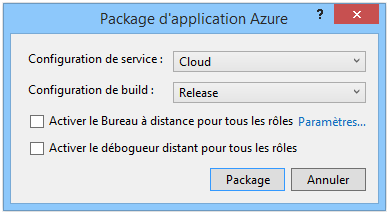
… rendez-vous dans le dossier *Documents/fr* depuis la racine du dossier récupéré de GitHub. Suivez les documentations *Démarrer avec OGDI DataLab.*docx et *Déployer OGDI DataLab dans Azure.docx* qui vous guideront dans la configuration, la compilation et la génération du package de service.

Pour aller plus vite, vous pouvez directement suivre les instructions de la section Résolution des dépendances avec NuGet du document précédemment ouvert.

Votre solution étant maintenant configurée, vous devez à présent créer le nouveau package de service Azure.

Pour cela, procédez comme suit :

1. Ouvrez Visual Studio 2013 en tant qu’administrateur.
2. Faîtes un clic droit sur *OGDI.Cloud* puis **Package.**
3. Dans la boite de dialogue **Package d’application Azure**, choisissez les options **Configuration de service** : **Cloud**, **Configuration de build** : **Release** et cliquez sur **Package** pour confirmer.



1. Une fois le package créé, une fenêtre Windows s’ouvre dans le dossier app.*publish* correspondant. Ce dossier contient donc le package *OGDI.Cloud.cspkg* qui nous intéresse. Mémorisez donc le chemin vers ce dossier.

### Pour ODAF Openturf…

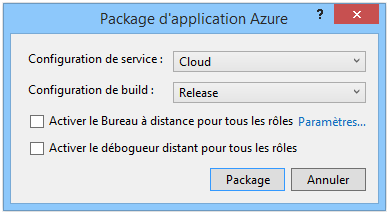
… rendez-vous dans le dossier *Documents/fr* depuis la racine du dossier récupéré de GitHub. Suivez les documentations *Démarrer avec ODAF Openturf.*docx et *Déployer ODAF Openturf dans Azure.docx* qui vous guideront dans la configuration, la compilation et la génération du package de service.

Pour aller plus vite, vous pouvez directement vous rendre à l’étape 6 de la section § Exécution de la solution ODAF Openturf en environnement local du document précédemment ouvert.

Votre solution étant maintenant configurée, vous devez à présent créer le nouveau package de service Azure.

Pour cela, procédez comme suit :

1. Ouvrez Visual Studio 2013 en tant qu’administrateur.
2. Faîtes un clic droit sur *ODAF.Azure* puis **Package**.
3. Dans la boite de dialogue **Package d’application Azure**, choisissez les options **Configuration de service** : **Cloud**, **Configuration de build** : **Release** et cliquez sur **Package** pour confirmer.



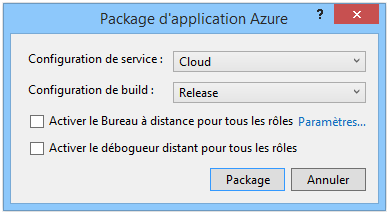
1. Une fois le package créé, une fenêtre Windows s’ouvre dans le dossier app.*publish* correspondant. Ce dossier contient donc le package *ODAF.Azure.cspkg* qui nous intéresse. Mémorisez donc le chemin vers ce dossier.

### Pour le Portail Citoyen Open Data…

… rendez-vous dans le dossier *Documents/fr* depuis la racine du dossier récupéré de GitHub. Suivez la documentation *Déployer le Portail Citoyen Open Data dans Azure.docx* qui vous guidera dans la configuration, la compilation et la génération du package de service.

Procédez comme suit :

1. Ouvrez Visual Studio 2013 en tant qu’administrateur
2. Faîtes un clic droit sur *CitizenPortal.Azure* puis **Package**.
3. Dans la boite de dialogue **Package d’application Azure**, choisissez les options **Configuration de service** : **Cloud**, **Configuration de build** : **Release** et cliquez sur **Package** pour confirmer.



1. Une fois le package créé, une fenêtre Windows s’ouvre dans le dossier app.*publish* correspondant. Ce dossier contient donc le package *CitizenPortal.Azure.cspkg* qui nous intéresse. Mémorisez donc le chemin vers ce dossier.

Le package généré est prêt à l’utilisation. Retenez son emplacement, vous en aurez besoin pour l’étape 3 (Cf. section Etape 3 – Ajout dans le compte de stockage) dans laquelle nous téléchargerons ce fichier vers le compte de stockage blob Azure.

## Etape 2 – Mise à jour des fichiers dans ODPI

L’étape 2 n’est pas requise si les paramètres de configuration présents dans le fichier *ServiceConfiguration.Cloud.cscfg* du projet *ODPI.Cloud* n’ont pas été modifiés. Si vous êtes dans ce cas, passez directement à la section suivante Etape 3 – Ajout dans le compte de stockage.

A contrario, si des paramètres ont été ajoutés/supprimés, vous devez alors suivre les étapes suivantes afin de répercuter les changements sur le site Web ODPI.

1. La première étape consiste à se rendre dans le dossier *Models/Config* et de rajouter ou supprimer les propriétés contenues dans le fichier éponyme à la solution à mettre à jour.
2. De même, veillez à modifier la variable *template* (gabarit) afin de répercuter les modifications dans les paramètres.

Illustrons ce premier point avec l’accélérateur OGDI DataLab. Imaginons que le paramètre de configuration *SolutionName* ait été intégré dans le projet OGDI DataLab afin de rendre le nom de la solution dynamique. Il convient à présent mettre à jour ODPI pour que la plateforme propose le nouveau paramètre lors de la configuration d’OGDI DataLab sur le site Web.

Pour cela, nous ouvrons le fichier *DataLabConfig.cs* et nous ajoutons la propriété suivante :

public string SolutionName { get; set; }

De même, la méthode *BuildSettingsString* est à mettre à jour pour refléter ce nouveau paramètre.

public string BuildSettingsString()

{

string template = @"

<Setting name=""RecaptchaPrivateKey"" value=""{0}""/>

              <Setting name=""RecaptchaPublicKey"" value=""{1}""/>

              <Setting name=""DataConnectionString""

value=""DefaultEndpointsProtocol=https;AccountName={2};AccountKey={3}""/>

              <Setting name=""DiagnosticsConnectionString""

value=""DefaultEndpointsProtocol=https;AccountName={2};AccountKey={3}""/>

              <Setting name=""serviceUri"" value=""http://{4}.cloudapp.net:8080/v1/""/>

              <Setting name=""IsAnalytics"" value=""0""/>

              <Setting name=""SolutionName"" value=""{6}"" />

           </ConfigurationSettings>

         </Role>

         <Role name=""DataBrowser.WorkerRole"">

          <Instances count=""1""/>

           <ConfigurationSettings>

            <Setting name=""serviceUri"" value=""http://{4}.cloudapp.net:8080/v1/"" />

              <Setting name=""DiagnosticsConnectionString""

value=""DefaultEndpointsProtocol=https;AccountName={2};AccountKey={3}"" />

            <Setting name=""DataConnectionString""

value=""DefaultEndpointsProtocol=https;AccountName={2};AccountKey={3}"" />

                                ";

  return string.Format(template, RecapPrivateKey, RecapPublicKey, BlobAccountName, BlobAccountKey, Dns,

SolutionName );

public string BuildSettingsString()

{

string template = @"

<Setting name=""Microsoft.WindowsAzure.Plugins.Caching.ConfigStoreConnectionString""

value=""DefaultEndpointsProtocol=https;AccountName={0};AccountKey={1}"" />

<Setting name=""Microsoft.WindowsAzure.Plugins.Diagnostics.ConnectionString""

value=""DefaultEndpointsProtocol=https;AccountName={0};AccountKey={1}"" />

<Setting name=""serviceUri"" value=""http://{2}.cloudapp.net:8080/v1/"" />

<Setting name=""DataConnectionString""

value=""DefaultEndpointsProtocol=https;AccountName={0};AccountKey={1}"" />

<Setting name=""DiagnosticsConnectionString""

value=""DefaultEndpointsProtocol=https;AccountName={0};AccountKey={1}"" />

<!--

Please replace the keys below with your private ones obtained from http://recaptcha.net/whyrecaptcha.html.

-->

<Setting name=""RecaptchaPrivateKey"" value=""{3}"" />

<Setting name=""RecaptchaPublicKey"" value=""{4}"" />

<Setting name=""bingCredential"" value=""{5}"" />

</ConfigurationSettings>

</Role>

<Role name=""DataBrowser.WorkerRole"">

<Instances count=""1"" />

<ConfigurationSettings>

<Setting name=""DiagnosticsConnectionString""

value=""DefaultEndpointsProtocol=https;AccountName={0};AccountKey={1}"" />

<Setting name=""DataConnectionString""

value=""DefaultEndpointsProtocol=https;AccountName={0};AccountKey={1}"" />

<Setting name=""serviceUri"" value=""http://{2}.cloudapp.net:8080/v1/"" />";

return string.Format(template, BlobAccountName, BlobAccountKey, Dns, RecapPublicKey, RecapPrivateKey, BingCredential);

}

}

Enfin, la méthode *BuildFromData* doit, elle aussi, être mise à jour afin de répercuter les données saisies depuis l’interface dans la propriété *SolutionName*.

public void BuildFromData(dynamic data)

{

Dns = data.dns;

   RecapPrivateKey = data.recappriv;

   RecapPublicKey = data.recappub;

   BlobAccountName = data.storagename;

   BlobAccountKey = data.storagekey;

SolutionName = data.solutionname ;\*

}

1. La seconde étape vise à permettre à l’utilisateur de remplir ce paramètre lors de la configuration de la solution à déployer dans l’interface Web de la plateforme ODPI.

Cette étape consiste à modifier la vue associée à la solution mise à jour dans le but de rajouter un champ de saisie. L’ensemble des vues relatives à la solution se trouve dans le dossier *Views/ConfTpl/*.

Ce qui donne dans le cadre de l’accélérateur OGDI DataLab la vue associée *Views/ConfTpl/DataLab.cshtml*.

Le code à rajouter pour un nouveau champ est le suivant :

<div class="control-group">

<label class="control-label">

    @ODI.Resources.Views.ConfTpl.DataLabResource.[NomDuParamètreDeTraductions]

</label>

<div class="controls">

    <input class="input-xlarge" type="text" id="solutionname" name="solutionname" />

</div>

</div>

1. La plateforme ODPI est à présent configurée pour prendre en compte les modifications apportées. Vous devez maintenant déployer la solution dans Azure. Pour déployer ODPI dans Azure, reportez-vous à la section § Publication d’ODPI dans Azure.

## Etape 3 – Mise à jour du paquet dans le compte de stockage

Comme vous avez déjà pu le voir, le fonctionnement interne de la plateforme ODPI est basé sur un compte de stockage contenant l’ensemble des packages de service Azure des solutions Open Data proposées à l’installation.

Une mise à jour de l’une des solutions proposées implique donc une mise à jour du package de service correspondant dans le compte de stockage qui sera directement répercutée dans la plateforme ODPI.

Pour cette étape, vous aurez besoin du paquet généré lors de l’étape 1. Ceci constitue la dernière modification pour que les changements soient effectifs. Il vous faut donc ajouter ce paquet dans le conteneur *components* du compte de stockage blob Azure.

Pour plus d’informations sur la manière d’envoyer un fichier vers un compte de stockage blob Azure, reportez-vous à la section § Etape 2 – Configuration du compte de stockage et notamment à l’outil utilisé dans ce contexte, à savoir CloudBerry Explorer for Azure.

# Ajouter une solution Open Data dans ODPI

Vous allez à présent aborder les différentes étapes visant à déployer totalement votre solution personnalisée de la plateforme ODPI dans votre souscription Azure.

## Etape 1 – Ajout du package de service dans le compte de stockage blob

La première étape est de vérifier que vous possédez bien un package de service Azure de votre solution Open Data destinée aux services cloud computing de Azure.

Si tel n’est pas le cas ou que vous ne savez pas comment obtenir, depuis votre solution Open Data personnelle, un package de service Azure, référez-vous à l’article [Publication d'une application Azure à l'aide de Windows Azure Tools](http://msdn.microsoft.com/library/azure/ff683672.aspx#PublishAzure)[[23]](#footnote-23).

Ce package est indispensable à l’utilisation de votre solution Open Data dans la plateforme ODPI car c’est à partir de celui-ci que les différents rôles de la solution seront générés dans les services cloud computing de Azure.

Le package créé, vous obtenez par la même occasion un fichier de configuration. Celui-ci possède l’extension *.cscfg*. Il contient les paramètres qui seront configurables depuis la plateforme ODPI.

Avant d’ajouter ce fichier dans le compte de stockage, vous devez le modifier afin d’en faire un gabarit spécifique à ODPI.

Remarque importante ODPI utilisera le gabarit avec la méthode *string.Format*. Il convient donc de l’adapter pour que celui-ci soit compatible.

### Création du gabarit de configuration

Le gabarit de configuration ressemble en beaucoup de points au fichier *.cscfg* généré à l’exception que celui-ci est prévu pour que les paramètres de configuration soient dynamiquement ajoutés par ODPI.

L’objectif, ici, consiste donc à remplacer l’ensemble des balises *Setting***,** qui devront être modifiées dynamiquement, par la chaîne de formatage *{0}.*

A titre d’exemple, le code suivant illustre les paramètres à modifier dans le fichier de configuration ODAF Openturf.

<ServiceConfiguration serviceName="ODAF.Azure"

xmlns="http://schemas.microsoft.com/ServiceHosting/2008/10/ServiceConfiguration">

<Role name="ODAF.Website.Mvc">

<Instances count="1" />

<ConfigurationSettings>

<Setting name="DiagnosticsConnectionString"

value="DefaultEndpointsProtocol=https;AccountName=XXXX;AccountKey=XXXXXX" />

<Setting name="ODAF"

value="Server=XXXX.database.windows.net;database=odaf;UserId=XXXXX;

Password=XXXXX;MultipleActiveResultSets=True;" />

<Setting name="AppName" value="ODAF Openturf" />

<Setting name="BitlyLogin" value="XXXX" />

<Setting name="BitlyAPIKey" value="XXXXXXX" />

<Setting name="tokenExpiryMinutes" value="10080" />

<Setting name="UseTokenExpiry" value="false" />

<Setting name="BlobStorageEndpoint"

value="DefaultEndpointsProtocol=https;AccountName=XXXXX;AccountKey=XXXX" />

<Setting name="AdminTwitterUser" value="XXXXXX" />

<Setting name="TwitterAppId" value="XXXX" />

<Setting name="DevMode" value="false" />

</ConfigurationSettings>

</Role>

<Role name="ODAF.SearchIndexRole">

<Instances count="1" />

<ConfigurationSettings>

<Setting name="BlobStorageEndpoint"

value="DefaultEndpointsProtocol=https;AccountName=XXXXX;AccountKey=XXXXXXXX" />

<Setting name="DiagnosticsConnectionString"

value="DefaultEndpointsProtocol=https;AccountName=XXXX;AccountKey=XXXXXX" />

</ConfigurationSettings>

</Role>

</ServiceConfiguration>

Le code suivant illustre le résultat dans le fichier de configuration ODAF Openturf, une fois les remplacements effectués.

<ServiceConfiguration serviceName="ODAF.Azure"

xmlns="http://schemas.microsoft.com/ServiceHosting/2008/10/ServiceConfiguration">

<Role name="ODAF.Website.Mvc">

<Instances count="1" />

<ConfigurationSettings>

{0}

<Setting name="tokenExpiryMinutes" value="10080" />

<Setting name="UseTokenExpiry" value="false" />

<Setting name="DevMode" value="false" />

</ConfigurationSettings>

</Role>

<Role name="ODAF.SearchIndexRole">

<Instances count="1" />

<ConfigurationSettings>

<Setting name="BlobStorageEndpoint"

value="DefaultEndpointsProtocol=https;AccountName=XXXXX;AccountKey=XXXXXXXX" />

<Setting name="DiagnosticsConnectionString"

value="DefaultEndpointsProtocol=https;AccountName=XXXX;AccountKey=XXXXXX" />

</ConfigurationSettings>

</Role>

</ServiceConfiguration>

### Envoi des fichiers vers le compte de stockage blobs

Vous disposez à présent un package de services *Azure* pour votre solution et un gabarit de configuration conforme aux attentes de la plateforme ODPI. Il vous faut à présent envoyer ces fichiers sur votre compte de stockage blobs Azure, plus précisément dans le conteneur *components*.

Pour cela, nous vous invitons à vous référer à la section § Etape 2 – Configuration du compte de stockage qui explique comment se connecter à un compte de stockage blob Azure et comment envoyer des fichiers dans le conteneur *components***.**

## Etape 2 – Création des nouveaux fichiers dans ODPI

Chaque solution Open Data présentée par ODPI est basée sur un ensemble de fichiers pour être déclarée, sélectionnée et ensuite, le cas échéant, installée.

Une solution Open Data dans ODPI se décompose de la manière suivante :

* Une classe de configuration située dans le dossier *Models/Config* ;
* Une vue permettant de saisir les informations et située dans le dossier *Views/ConfTpl* ;
* Les fichiers de traduction situés dans *Resources*/*Views/ConfTpl* ;
* Une classe de post-déploiement **optionnelle** située dans le dossier *Actions/PostDeploy*.

Afin d’illustrer la création d’une application Open Data, nous allons décrire dans la suite de cette étape l’ajout d’une solution nommée Open Intelligence (OpenIntel).

OpenIntel est un accélérateur pour les administrations, les collectivités territoriales et les entreprises souhaitant rapidement construire une application Open Data mettant en scène des données géo-localisées avec des fonctionnalités de Business Intelligence (BI).

Elle constitue une solution de visualisation et d’analyses des données qui peut servir aussi bien aux agents qu’à des utilisateurs finaux qui peuvent ainsi accéder à un annuaire riche et dynamique de données publiques.

Remarque Pour de plus amples informations, vous pouvez vous référer au billet [Accélérateur de solutions Open Intelligence](http://blogs.msdn.com/b/ogdifrance/archive/2011/10/07/acc-233-l-233-rateur-de-solutions-open-intelligence.aspx)[[24]](#footnote-24).

Pour ajouter cette solution dans ODPI, procédez comme suit :

1. Dans Visual Studio rendez-vous dans le dossier *Models/Config* puis ajoutez une classe nommée *OpenIntelConfig.cs*.

Remarque importante La convention de nommage pour les fichiers de configuration est la suivante *[NomDeLaSolution]Config.cs*.

1. Cette classe contient l’ensemble des propriétés qui seront exposées et donc configurées par l’utilisateur via la plateforme ODPI. De même, elle contient :
   1. Dans la méthode *BuildSettingsString* le gabarit permettant de générer le fichier de configuration Azure adéquat,
   2. Une méthode *BuildFromData* permettant de mettre à jour les propriétés en fonction des données saisies par l’utilisateur et un nom de gabarit utilisé pour récupérer la vue associée.

public class OpenIntelConfig : IOdiAppConfig

{

// Propriétés

   public string DbName { get; set; }

   public string DbHost { get; set; }

   public string DbUserName { get; set; }

   public string DbUserPassword { get; set; }

   public string BingServiceKey { get; set; }

   public string StorageName { get; set; }

   public string StorageKey { get; set; }

// Méthode contenant le gabarit de configuration Azure

   public string BuildSettingsString()

   {

    string template = @"<Setting name=""OIConnectionString""

value=""Server=tcp:{0};Database={1};User ID={2};Password={3};Trusted\_Connection=False;Encrypt=True;"" />

      <Setting name=""BingServiceKey"" value=""{4}"" />

      <Setting name=""mapdotnet.adminservicesettings.AzureStorageFactoryAccountName"" value=""{5}"" />

      <Setting name=""mapdotnet.adminservicesettings.AzureStorageFactoryAccessKey"" value=""{6}"" />

     ";

    return string.Format(template, DbHost, DbName, DbUserName, DbUserPassword, BingServiceKey,

StorageName, StorageKey);

}

// Méthode de liaison entre la vue et les propriétés

   public void BuildFromData(dynamic data)

   {

    DbName = data.dbname;

      DbHost = data.dbhost;

      DbUserName = data.dbusername;

      DbUserPassword = data.dbpassword;

      BingServiceKey = data.bingservicekey;

      StorageName = data.storagename;

      StorageKey = data.storagekey;

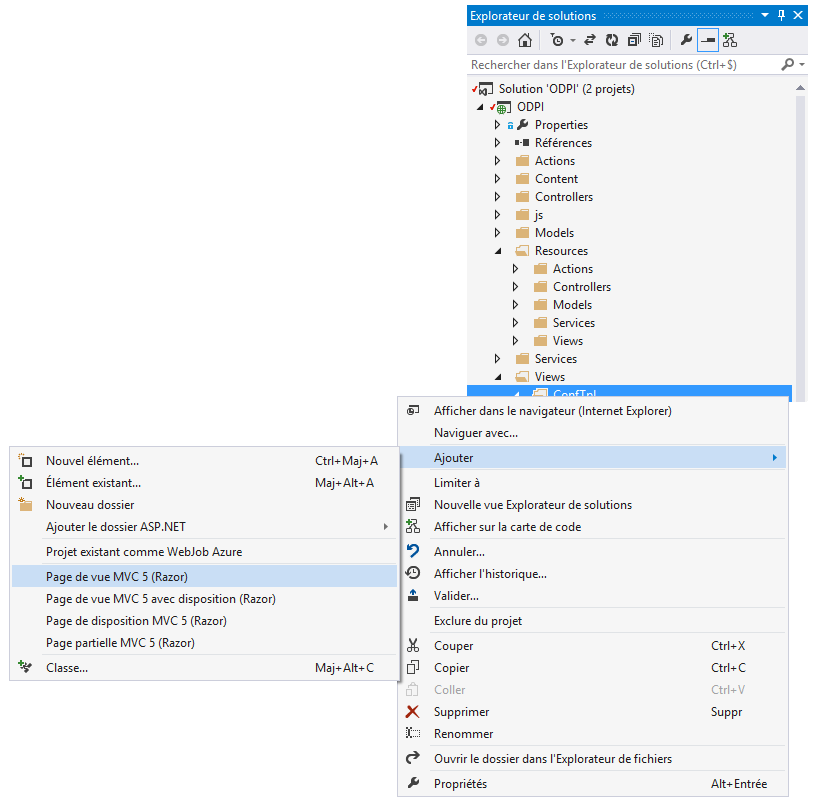
   }

// Nom permettant de retrouver la vue associée

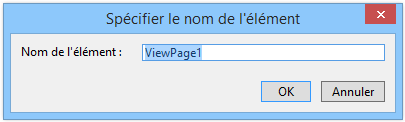
    public string Template { get { return "**OpenIntel**"; } }

}

1. La classe de configuration créée, il vous faut à présent créer la vue qui sera fournie à l’utilisateur via ODPI. Pour cela, faîtes un clic droit sur le dossier *Views/ConfTpl* -> **Ajouter** -> **Page de vue MVC 5 (Razor)**.



1. La fenêtre **Spécifier le nom de l’élément**.



1. Dans **Nom de l’élément**, entrez le nom de la vue en fonction du nom présent dans la propriété *Template* de la classe *OpenIntelConfig.cs*.

public string Template

{

get { return "**OpenIntel**"; }

}

Soit : *OpenIntel*Config

1. Cliquez sur **OK**.
2. Dans la vue correspondante (fichier *OpenIntelConfig.cshtml* nouvellement créé), supprimer le code suivant :

@{

Layout = null;

}

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width" />

<title></title>

</head>

<body>

<div>

</div>

</body>

</html>

1. Ajoutez les informations disponibles à la saisie dans cette même vue. Il s’agit typiquement l’ensemble des propriétés ajoutées dans la classe *OpenIntelConfig.cs*.

@{

   Layout = null;

}

<div class="page-header">

<h4>

@ODI.Resources.Views.ConfTpl.OpenIntelResource.ApplicationSettings

</h4>

   <p>

     @ODI.Resources.Views.ConfTpl.OpenIntelResource.ApplicationSettingsWillAllowDataLab

    </p>

</div>

<div class="control-group">

<label class="control-label">

        @ODI.Resources.Views.ConfTpl.OpenIntelResource.BingServiceKey

</label>

   <div class="controls">

    <input class="input-xlarge"

type="text"

id="bingservicekey"

name="bingservicekey"

placeholder="@ODI.Resources.Views.ConfTpl.OpenIntelResource.KeyProvidedByBingAPI" />

    </div>

</div>

<div class="page-header">

    <h4>

     @ODI.Resources.Views.ConfTpl.OpenIntelResource.SQLAzureInfo

</h4>

    <p>

       @Html.Raw(ODI.Resources.Views.ConfTpl.OpenIntelResource.IfYouDoNotHaveASqlAzure)

</p>

    <p>

        @ODI.Resources.Views.ConfTpl.OpenIntelResource.OnceTheDataBaseIsCreated

</p>

</div>

<div class="control-group">

    <label class="control-label">

     <a href="#"

data-toggle="modal"

data-target="#GetDbHostName"

rel="tooltip"

data-original-title="@ODI.Resources.Views.ConfTpl.OpenIntelResource.ClickToSeeHow">

        @ODI.Resources.Views.ConfTpl.OpenIntelResource.DatabaseHostname

</a>

</label>

    <div class="controls">

     <input class="input-xlarge"

type="text"

id="dbhost"

name="dbhost"

placeholder="@ODI.Resources.Views.ConfTpl.OpenIntelResource.YourServerDatabase" />

    </div>

</div>

…

Pour ajouter un champ de saisie, le gabarit type est le suivant :

<div class="control-group">

<label class="control-label">

    @ODI.Resources.Views.ConfTpl.OpenIntelResource.[NomDeLaVariableDeLocalisation]

</label>

   <div class="controls">

    <input class="input-xlarge"

type="text"

id="[iddeliaison]"

name="[iddeliaison]"

placeholder="@ODI.Resources.Views.ConfTpl.OpenIntelResource.[NomDeLaVariableDeLocalisation2]" />

   </div>

</div>

*[iddeliaison]* ci-dessus représente la variable saisie dans la méthode *BuildFromData* de la classe *OpenIntelConfig.cs* ; soit pour ajouter un champ lié à la propriété *StorageKey* :

StorageKey = data.storagekey;

*[iddeliaison]* sera remplacé par *storagekey*.

Ce qui nous donne :

<div class="control-group">

<label class="control-label">

    @ODI.Resources.Views.ConfTpl.OpenIntelResource.[NomDeLaVariableDeLocalisation]

</label>

   <div class="controls">

    <input class="input-xlarge"

type="text"

id="storagekey"

name="storagekey"

placeholder="@ODI.Resources.Views.ConfTpl.OpenIntelResource.[NomDeLaVariableDeLocalisation2]" />

</div>

</div>

1. Enfin, l’application étant entièrement localisée en fonction de la langue du navigateur (français et anglais dans l’état courant, avec l’anglais par défaut), il convient de créer les fichiers de ressources pour la traduction qui seront situés dans le dossier *Resources*/*Views/ConfTpl***.**

## Etape 3 – Mise à jour des fichiers existants

Les fichiers propres à la nouvelle solution créés, il est temps de configurer ODPI pour afficher cette solution.

La configuration se décompose en 2 parties :

1. La première partie consiste à référencer la vue dans ODPI afin que celle-ci soit automatiquement chargée en mémoire lors du premier lancement. Pour charger la vue associée, ouvrez le fichier *main.js* situé dans le dossier *js*.

Naviguez jusqu’à la fin du fichier. Vous trouverez une ligne commençant par *tpl.loadConfigTermplates*.

Cette ligne contient l’ensemble des vues qu’ODPI référence pour les accélérateurs de la solution Open Data à déployer.

En reprenant l’exemple précédent de la solution OpenIntel, il faut rajouter la valeur *"OpenIntel"* dans le tableau.

tpl.loadConfigTemplates(['SqlAzure',

'OpenIntel',

'BlobStorage',

'OpenTurf',

'DataLab',

'CitizenPortalOpenData'],

function () { });

1. La seconde et dernière étape de configuration permet d’ajouter la nouvelle solution à la liste des solutions Open Data proposées sur la page d’accueil.

* Pour cela, ouvrez le fichier *OdpiAppRepo* situé dans le dossier *Models*. Dans la méthode *Initialize()*, est défini une liste des solutions. Il convient donc de rajouter dans cette méthode votre solution à cette liste.
* A la fin de cette méthode, ajoutez donc les lignes suivantes pour la solution OpenIntel :

Apps.Add(new OdpiApp()

{

              Id = 4,

                DisplayOrder = 4,

                Name = ODI.Resources.Models.OdiAppRepoResource.OpenIntel,

                ConfigHelpText = ODI.Resources.Models.OdiAppRepoResource.OpenIntelConfigHelpText,

                Description = ODI.Resources.Models.OdiAppRepoResource.OpenIntelDescription,

                Configs = new IOdiAppConfig[] { new OpenIntelConfig() },

                Validations = new IValidateAction[] { new SqlAzureValidation(), new TableStorageValidation() },

                PostAction = new OpenIntelPostDeploy(),

PackageName = "openintel\_0.2.cspkg",

                ConfName = "openintel\_0.1.cscfg",

                RequiredFiles = new string[]

                {

                    "open\_intel\_create\_tables.0.1.sql",

                    "OI\_Sample.mapx",

                    "MapFiles.xml",

                    "ISC.MapDotNetServer.Common.dll",

                    "ISC.MapDotNetServer.Common.Maps.dll",

                    "ISC.MapDotNetServer.Common.Maps.xml",

                    "ISC.MapDotNetServer.Common.xml",

                    "MoveMap.exe",

                    "MoveMap.pdb"

                },

                SiteUrl = http://{0}.cloudapp.net

});

Remarque importante Vous noterez qu’ici aussi le texte est localisé. Pour ajouter ou modifier des informations, utilisez le fichier *OdpiAppRepoResource.resx* du dossier *Resources/Models*.

Vous trouverez ci-dessous une description des éléments de configuration de la classe *new OdpiApp()*.

| **Méthode** | **Description** |
| --- | --- |
| Id | Identifiant unique de la solution dans ODPI |
| DisplayOrder | Numéro définissant l’ordre d’affichage des solutions |
| Name | Nom de la solution dans ODPI |
| ConfigHelpText | Description dans les grandes lignes des informations nécessaires lors de la configuration |
| Description | Description de la solution apparaissant sur la page d’accueil |
| Configs | Information liant une ou plusieurs configurations à l’élément ajouté dans la liste. C’est ici que sera définie la solution correspondant à cette entrée dans la liste. |
| Validations | Ajoute des éléments de saisie et de validation lors de la configuration. Ceci est très utile afin d’inclure automatiquement les éléments de saisie et de valider les informations comme les informations d’une base de données SQL Azure ou encore d’un compte de stockage de type table Azure. |
| PostActions | Permet d’exécuter des actions post installation. Cette étape permet notamment d’exécuter des scripts SQL, de lancer des programmes automatisés, etc. |
| PackageName | Nom du package de service Azure présent dans le compte de stockage |
| ConfName | Nom du gabarit de configuration Azure présent dans le compte de stockage |
| RequiredFiles | Recense les fichiers qui doivent être obligatoirement dans le compte de stockage hors package de services Azure et gabarit de configuration. |
| SiteUrl | Url du site final déployé. Généralement http://{0}.cloudapp.net |

Les nouveaux fichiers étant créés, il vous faut à présent déployer ODPI dans Azure.

## Etape 4 – Déploiement d’ODPI dans Azure

Cette dernière étape permet de mettre en ligne la plateforme d’installation automatisée ODPI avec la nouvelle solution que vous avez ajoutée.

Pour déployer la solution dans Azure, référez-vous à la section § Publication d’ODPI dans Azure.

1. Solution Open Data clé en main : http://aka.ms/OpenDataCleEnMain [↑](#footnote-ref-1)
2. Sites Web: http://azure.microsoft.com/fr-fr/services/websites/ [↑](#footnote-ref-2)
3. Cloud Service: http://azure.microsoft.com/fr-fr/services/cloud-services/ [↑](#footnote-ref-3)
4. Microsoft Azure : http://azure.microsoft.com/fr-fr/ [↑](#footnote-ref-4)
5. Blog MSDN Open Data France : http://aka.ms/OpenDataFrance/ [↑](#footnote-ref-5)
6. Azure Readiness Content : http://www.microsoft.com/en-us/download/details.aspx?id=8396 [↑](#footnote-ref-6)
7. Microsoft Azure Blog : http://azure.microsoft.com/blog/ [↑](#footnote-ref-7)
8. Microsoft Public License (MS-PL): http://opensource.org/licenses/ms-pl.html [↑](#footnote-ref-8)
9. GitHub OpenLab : https://github.com/openlab [↑](#footnote-ref-9)
10. Blog MSDN OGDI France : http://blogs.msdn.com/b/ogdifrance/ [↑](#footnote-ref-10)
11. ODPI, la facilité de mise en œuvre des solutions Open Data : http://blogs.msdn.com/b/ogdifrance/archive/2012/07/13/odpi-la-facilit-233-de-mise-en-uvre-des-solutions-open-data.aspx [↑](#footnote-ref-11)
12. OGDI DataLab sur GitHub ! Impliquez-vous dans la communauté : http://blogs.msdn.com/b/ogdifrance/archive/2012/05/14/ogdi-datalab-sur-github-impliquez-vous-dans-la-communaut-233.aspx [↑](#footnote-ref-12)
13. GitHub for Windows: http://blogs.msdn.com/b/ogdifrance/archive/2012/06/14/github-for-windows.aspx [↑](#footnote-ref-13)
14. Blog MSDN Open Data France: http://aka.ms/OpenDataFrance/ [↑](#footnote-ref-14)
15. Visual Studio Tools for Git : http://visualstudiogallery.msdn.microsoft.com/abafc7d6-dcaa-40f4-8a5e-d6724bdb980c [↑](#footnote-ref-15)
16. Integrating and Using Github in Visual Studio 2012 : http://www.codeproject.com/Articles/581907/IntegratingplusandplusUsingplusGithubplusinplusVis [↑](#footnote-ref-16)
17. API du service de blobs Azure : http://msdn.microsoft.com/en-us/library/dd135733 [↑](#footnote-ref-17)
18. API REST de gestion des services : http://msdn.microsoft.com/fr-fr/library/ee460799.aspx [↑](#footnote-ref-18)
19. CloudBerry Explorer For Microsoft Azure : http://www.cloudberrylab.com/free-microsoft-azure-explorer.aspx [↑](#footnote-ref-19)
20. OGDI DataLab sur GitHub ! Impliquez-vous dans la communauté : http://blogs.msdn.com/b/ogdifrance/archive/2012/05/14/ogdi-datalab-sur-github-impliquez-vous-dans-la-communaut-233.aspx [↑](#footnote-ref-20)
21. GitHub for Windows: http://blogs.msdn.com/b/ogdifrance/archive/2012/06/14/github-for-windows.aspx [↑](#footnote-ref-21)
22. Blog MSDN OGDI France: http://blogs.msdn.com/b/ogdifrance/ [↑](#footnote-ref-22)
23. Publication d'une application Azure à l'aide de Windows Azure Tools : http://msdn.microsoft.com/library/azure/ff683672.aspx#PublishAzure [↑](#footnote-ref-23)
24. Accélérateur de solutions Open Intelligence: http://blogs.msdn.com/b/ogdifrance/archive/2011/10/07/acc-233-l-233-rateur-de-solutions-open-intelligence.aspx [↑](#footnote-ref-24)