Déployer la plateforme d’installation automatisée ODPI dans Windows Azure

Solution Open Data Clé en main

Publication : Juin 2013

Version: 2.0

Auteurs : Philippe Beraud (Microsoft France), Rémi Olivier (Microsoft France), Alexandre Fournier (Microsoft France)

Pour les dernières informations, consultez

aka.ms/OpenDataFrance

Sommaire

[Commentaires 3](#_Toc358205773)

[Résumé 4](#_Toc358205774)

[Prérequis 4](#_Toc358205775)

[Connaissances et attendu 5](#_Toc358205776)

[Présentation rapide de la plateforme ODPI 6](#_Toc358205777)

[Organisation du code source d’ODPI 8](#_Toc358205778)

[Dossier ODI (Open Data Platform Installer) 8](#_Toc358205779)

[Dossier *ODI.Azure* 9](#_Toc358205780)

[Dossier *blob files* 10](#_Toc358205781)

[Mise en place de la configuration du compte Windows Azure 11](#_Toc358205782)

[Etape 1 – Création du compte de stockage 11](#_Toc358205783)

[Etape 2 – Configuration du compte de stockage 14](#_Toc358205784)

[Publication d’ODPI dans Windows Azure 17](#_Toc358205785)

[Etape 1 – Configuration des paramètres d’ODPI 17](#_Toc358205786)

[Etape 2 – Déploiement d’ODPI 18](#_Toc358205787)

[Mise à jour des solutions Open Data dans ODPI 20](#_Toc358205788)

[Etape 1 – Génération du package de service Windows Azure d’une solution Open Data 20](#_Toc358205789)

[Pour OGDI DataLab… 21](#_Toc358205790)

[Pour ODAF Openturf… 21](#_Toc358205791)

[Pour le Portail Citoyen Open Data… 22](#_Toc358205792)

[Etape 2 – Mise à jour des fichiers dans ODPI 22](#_Toc358205793)

[Etape 3 – Mise à jour du paquet dans le compte de stockage 24](#_Toc358205794)

[Ajouter une solution Open Data dans ODPI 25](#_Toc358205795)

[Etape 1 – Ajout du package de service dans le compte de stockage blob 25](#_Toc358205796)

[Création du template de configuration 25](#_Toc358205797)

[Envoi des fichiers vers le compte de stockage blobs 26](#_Toc358205798)

[Etape 2 – Création des nouveaux fichiers dans ODPI 26](#_Toc358205799)

[Etape 3 – Mise à jour des fichiers existants 30](#_Toc358205800)

[Etape 4 – Déploiement d’ODPI dans Windows Azure 31](#_Toc358205801)

# 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 Open Data Clé en main](https://mspartner.microsoft.com/fr/fr/Pages/Solutions/Open-Data-Cle-en-main.aspx)[[1]](#footnote-1) disponible sur le site Microsoft Partner Network et ses différents accélérateurs.

Le [blog MSDN Open Data France](http://blogs.msdn.com/b/ogdifrance/)[[2]](#footnote-2) 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.

# Résumé

Ce document détaille l’ensemble du processus de mise en œuvre et de publication sur [Windows Azure](http://www.microsoft.com/france/windows-azure/)[[3]](#footnote-3) de l’environnement ODPI (*Open Data Platform Installer*), une plateforme d’installation automatisée de solutions Open Data destinées à s’exécuter dans les [services de cloud computing (*Cloud Services*)](http://www.windowsazure.com/fr-fr/home/features/cloud-services/)[[4]](#footnote-4) de Windows Azure.

En ce sens, elle vise à offrir une installation simplifiée dans l’esprit de ce qui est proposé pour les sites Web (*Web Sites*, nom de code Antares)[[5]](#footnote-5) ASP.NET, PHP et Node.js de Windows Azure, mais 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 OGDI (*Open Governement Data Initiative*) DataLab et ODAF (*Open Data Application Framework*) OpenTurf disponibles tous deux sur la [forge GitHub](https://github.com/openlab)[[6]](#footnote-6). Vous pouvez tester une instance de cette plateforme à l’adresse <http://odpifrance.cloudapp.net>.

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 Windows Azure, etc.

Ce document aborde ainsi comment configurer et publier la plateforme ODPI dans votre compte Windows 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 solutions Open Data 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.

# Prérequis

La plateforme ODPI à destination de l’environnement d’exécution Windows Azure nécessite un ordinateur sous Windows Server 2003 (R2) SP1, Windows 7, Windows 8, Windows Server (R2) 2008, Windows Server 2012 ou ultérieur.

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

# Connaissances et attendu

La mise en œuvre de la plateforme ODPI requiert une expérience préalable :

* De l’environnement Windows Azure dans sa dimension PaaS (Platform as a Service),
* De l’environnement de développement Microsoft Visual Studio 2010,
* De la programmation avec le Microsoft Framework .NET,
* 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 les kits de formation suivants qui comprennent des présentations, des vidéos de formation, des démonstrations accompagnées de leur script ainsi que des ateliers didactiques complets :

* [Microsoft Windows Azure Training Kit (mise à jour de mai 2013)](http://www.microsoft.com/downloads/en/details.aspx?FamilyID=413E88F8-5966-4A83-B309-53B7B77EDF78&displaylang=en)[[7]](#footnote-7) couvrant la plateforme Windows Azure et l’environnement de développement Microsoft Visual Studio en version 2010 et 2012.
* [Microsoft Web Camps Training Kit](http://trainingkit.webcamps.ms/Default.htm)[[8]](#footnote-8) couvrant notamment dans le contexte de ce kit de démarrage les technologies ASP.NET MVC, OData et jQuery.

Remarque : Le [blog du groupe Produit de Windows Azure](http://blogs.msdn.com/b/windowsazure/)[[9]](#footnote-9) 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 Windows Azure en termes de services à la demande.

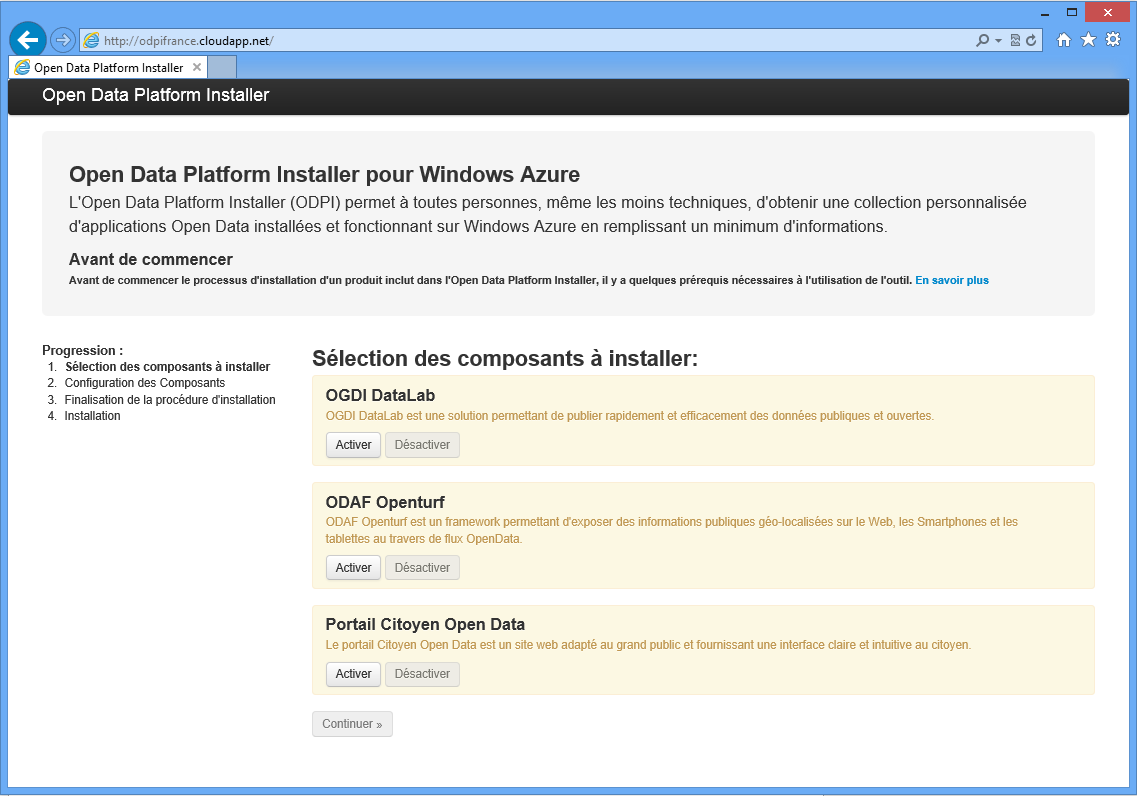
# Présentation rapide de la plateforme ODPI

La plateforme ODPI est une solution dont l’objectif est de permettre à toutes personnes, même les moins techniques, d’installer simplement et rapidement des solutions Open Data complexes conçues pour les [services de cloud computing (*Cloud Services*)](http://www.windowsazure.com/fr-fr/home/features/cloud-services/)[[10]](#footnote-10) de Windows Azure. Vous pouvez tester une instance de cette plateforme 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 Windows 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)[[11]](#footnote-11) sur la [forge GitHub](https://github.com/openlab)[[12]](#footnote-12).

Pour plus de renseignements sur la plateforme ODPI, nous vous invitons à visiter le blog MSDN [OGDI France](http://blogs.msdn.com/b/ogdifrance/)[[13]](#footnote-13) 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)[[14]](#footnote-14).

# 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 *ODI* 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 MVC3 destinée à l’environnement Windows Azure et qui permet de visualiser les données provenant d’un service RESTful OData;

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

Par défaut, ces solutions Open Data à destination des services de cloud computing (PaaS) de Windows Azure sont au nombre de 3, en l’occurrence OGDI (*Open Governement Data Initiative*) DataLab, ODAF (*Open Data Application Framework*) 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 *ODI.sln* pour Visual Studio 2010 et ainsi vous l’approprier plus facilement en vue par exemple de sa personnalisation (légère ou avancée).

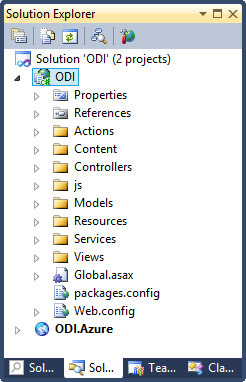
Tous les sites, services et composants afférents sont écrits en C#. Ils sont disponibles pour la version v4.0.30319 du Microsoft Framework .NET. Les fichiers de solution associés et présents dans ces dossiers sont destinés à l’environnement de développement Visual Studio 2010.

## Dossier ODI (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 MVC3 destinée aux services de cloud computing de l’environnement Windows Azure.

Remarque importante : La publication requiert un compte et une souscription active dans Windows Azure. Vous pouvez accéder à Windows Azure avec un paiement à l’usage, sans engagement, ou bien des forfaits comme décrit à l’adresse <https://www.windowsazure.com/fr-fr/pricing/purchase-options/>.

Le dossier *ODI* 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 3, il contient les images et feuilles de styles.* |
| ***Controllers*** | *Ce dossier est un dossier classique d’un site Web ASP.NET MVC 3.* |
| ***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 3, 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 Windows Azure et qui permet également de configurer les packages en fonction d’un modèle (template) précis puis de déployer dans Windows Azure.* |
| ***Views*** | *Ce dossier est un dossier classique d’un site Web ASP.NET MVC 3. Il contient l’ensemble des pages accessibles en fonction de la route demandée.* |

## Dossier *ODI.Azure*

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

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

## Dossier *blob files*

CL dossier *blob files* contient l’ensemble des packages de service Windows Azure relatifs aux solutions Open Data préconfigurées et proposées à l’installation par le site Web ODPI. Nous y revenons dans la suite de ce document.

# Mise en place de la configuration du compte Windows Azure

La mise en œuvre de la plateforme ODPI requiert un compte Windows Azure. Cette section décrit les éléments de configuration relatifs à ce compte Windows Azure à mettre en place préalablement à la publication de la solution ODPI en tant que telle dans les services de cloud computing de Windows Azure.

Remarque importante : Les étapes suivantes requièrent un compte et une souscription active avec Windows Azure. Vous pouvez accéder à Windows Azure avec un paiement à l’usage, sans engagement, ou bien des forfaits comme décrit à l’adresse [hhttps://www.windowsazure.com/fr-fr/pricing/purchase-options/](http://www.microsoft.com/france/windows-azure/offres.aspx).

A la date de publication de ce document, une offre relative à une version d’évaluation gratuite est proposée également comme décrit à l’adresse <https://www.windowsazure.com/fr-fr/pricing/free-trial/>.

## Etape 1 – Création du compte de stockage

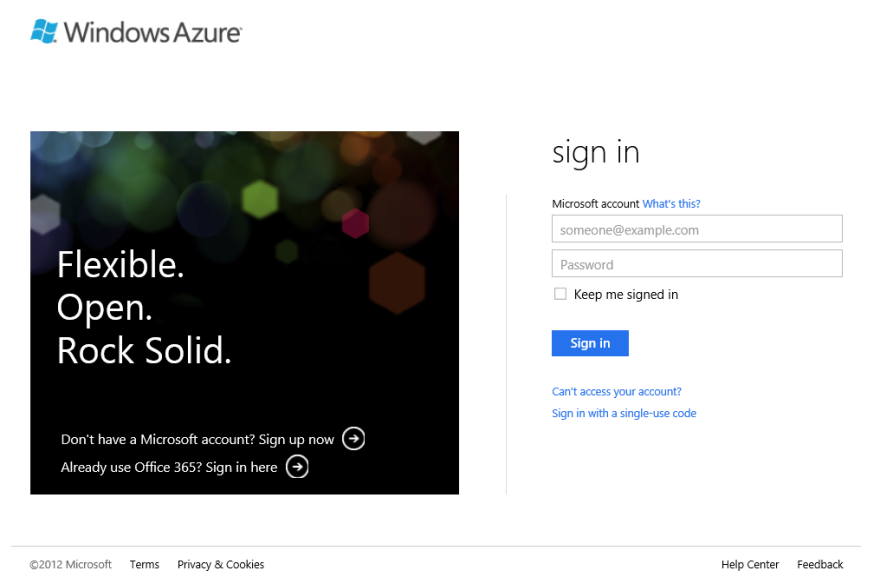
La plateforme ODPI nécessite 1 compte de stockage Windows Azure pour son fonctionnement.

Ce compte de stockage à travers le [service de blobs Windows Azure](http://msdn.microsoft.com/en-us/library/dd135733)[[15]](#footnote-15) permet de stocker les packages de service Windows Azure contenant les solutions Open Data proposées à l’installation ainsi que leur modèles (*template*) associés.

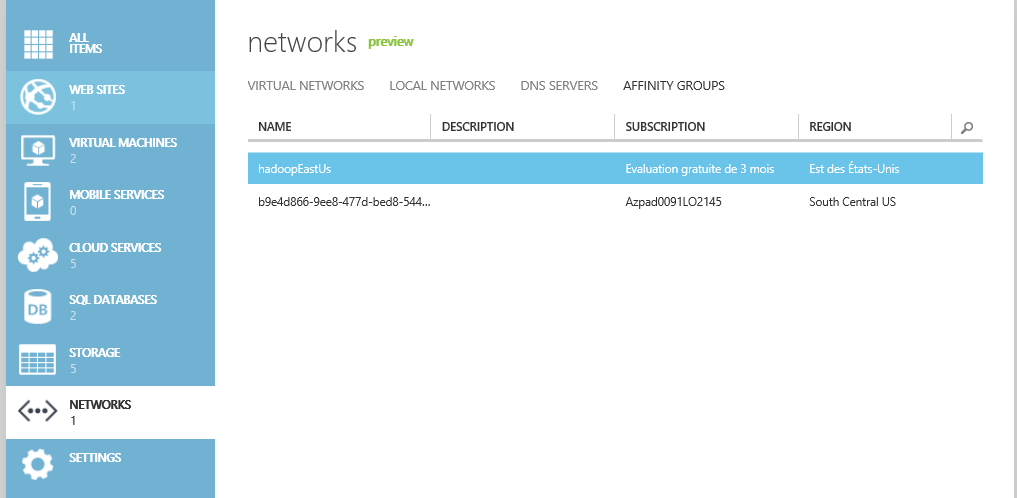
Lors de cette première étape, vous allez créer le compte de stockage utilisé par la plateforme ODPI depuis le portail de gestion Windows Azure à l’adresse <https://manage.windowsazure.com/>.

Pour créer le compte de stockage, procédez comme suit :

1. Ouvrez votre navigateur, naviguez vers <http://manage.windowsazure.com/> et connectez-vous en utilisant votre compte Microsoft associé à votre compte Windows Azure.



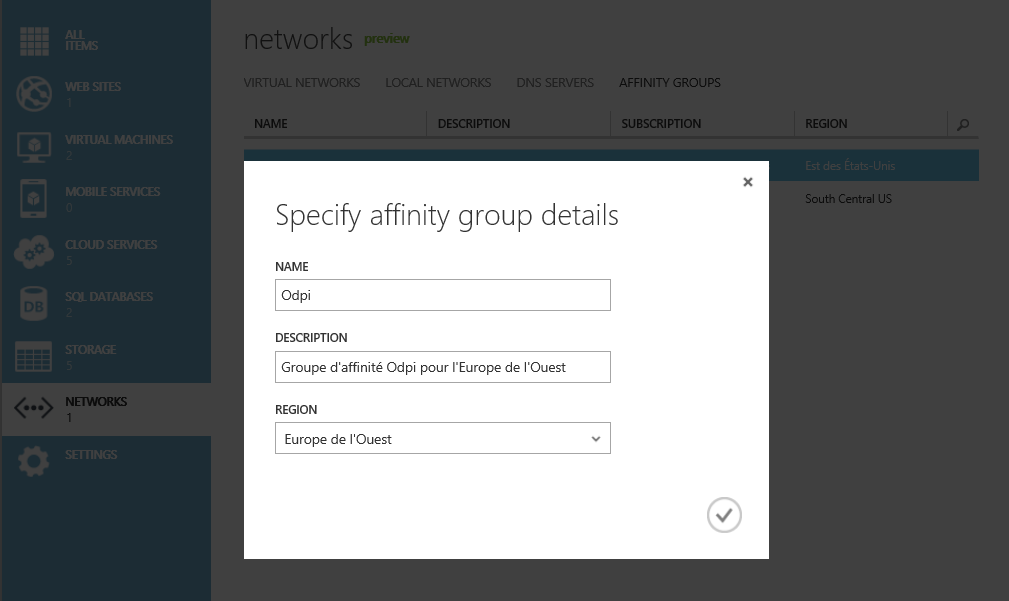
1. La première chose à faire est de créer un groupe d’affinité. Pour cela, sélectionnez l’onglet **NETWORKS** puis le menu **AFFINITY GROUPS** et choisissez ensuite**CREATE*.***



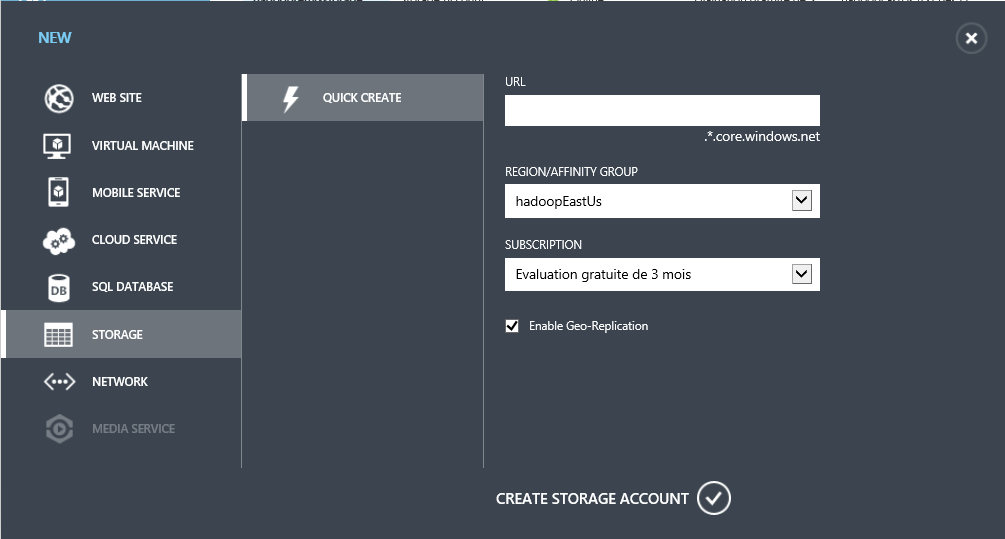
Remarque : Vous créez un groupe d’affinités ici de façon à pouvoir ensuite déployer la solution ODPI dans la même localisation géographique, ceci afin d’optimiser la bande passante et de réduire la latence entre les applications et les données dont elles dépendent.

**Remarque** : L’ensemble des captures d’écran sont réalisés avec la nouvelle version du portail de gestion Windows Azure.

1. Dans la boite de dialogue **Specify affinity group details**, choisissez un nom pour votre groupe, par exemple *Odpi* ainsi qu’une localisation géographique, **Europe de l’Ouest** étant conseillé pour un accès depuis la France métropolitaine.

****

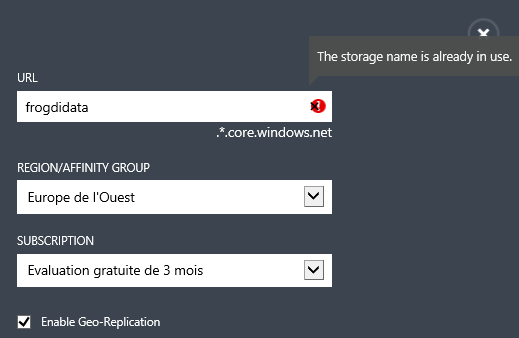
1. Cliquez sur le bouton coche pour valider la création du groupe d’affinité. Après un temps de chargement pouvant durer quelques minutes, la création du groupe devient effective.
2. Maintenant, pour créer le compte de stockage, cliquez sur le bouton **New** en bas à gauche puis sur **Storage**.



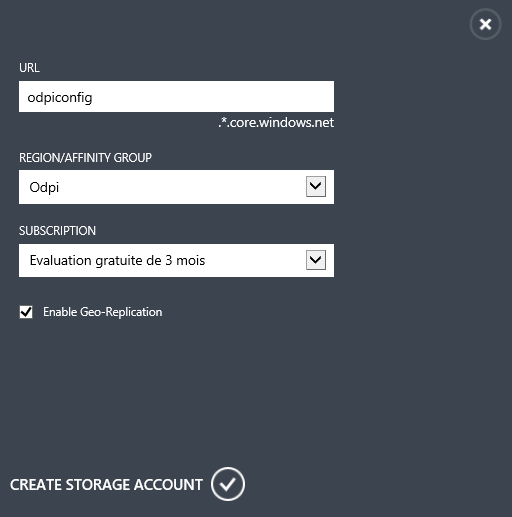
1. Dans le volet qui apparait, sélectionnez **QUICK CREATE** puis dans la zone de texte **URL**, entrez le nom de votre compte de stockage, par exemple *odpiconfig* dans ce cas.

Windows 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.



1. Sélectionnez la région créée précédemment dans la liste **REGION/AFINITY GROUP*.***

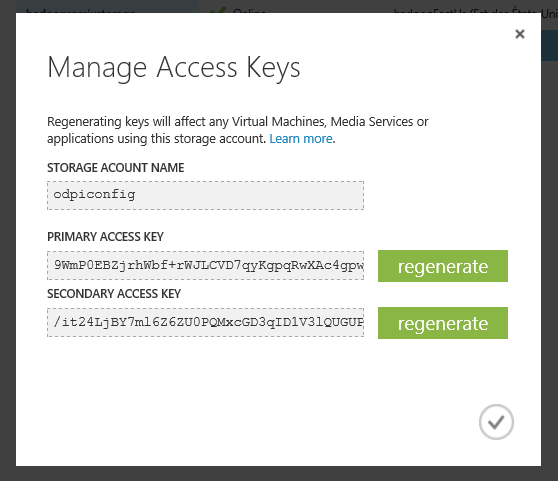


1. Cliquez sur **CREATE STORAGE ACCOUNT** pour valider la création du compte de stockage. Après un temps de chargement pouvant durer quelques minutes, la création du compte devient effective.

C:\Users\t-rolivi\Desktop\odpiconfigcreated.png

Notez que le bouton **MANAGE KEYS** donne accès à la clé primaire du compte nécessaire pour pouvoir accéder à ces services via les [interfaces programmatiques (*API*) de gestion Windows Azure](http://msdn.microsoft.com/fr-fr/library/ee460799.aspx)[[16]](#footnote-16).

1. Cliquez sur le bouton **MANAGE KEYS** de la zone **PRIMARY ACCESS KEY** pour voir les clés associées à ce compte.



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

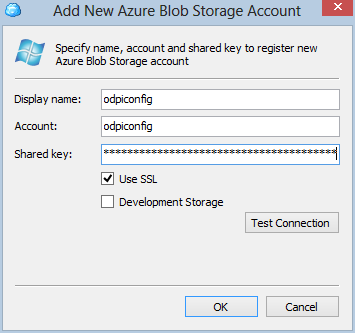
La plateforme ODPI nécessite 1 seul compte de service en plus d’un compte de stockage pour s’exécuter. Le compte de stockage sera utilisé pour ses fonctionnalités blobs destinées à héberger les packages de service des solutions Open Data ainsi que leurs templates.

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

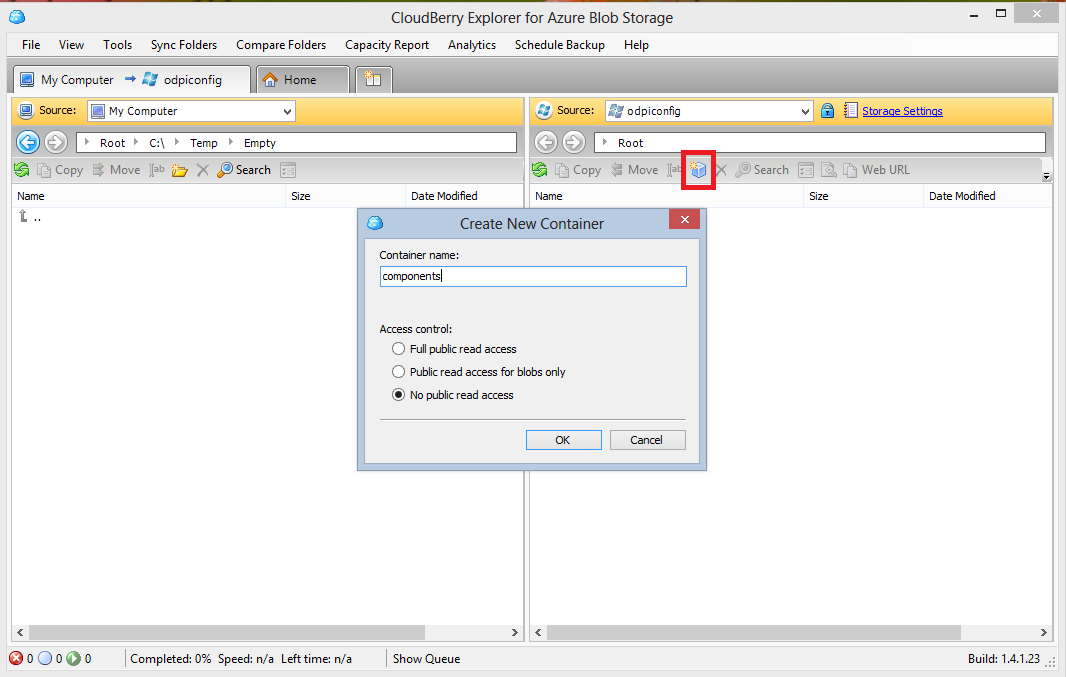
Nous utiliserons ici l’application [CloudBerry Explorer For Windows Azure](http://www.cloudberrylab.com/) qui a l’avantage d’être gratuite en plus d’être parfaitement adaptée à notre besoin.

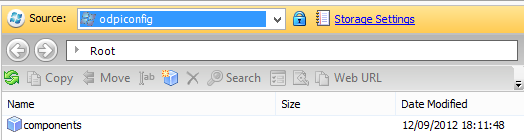
Procédez comme suit :

1. Dans CloudBerry Explorer, dans la liste déroulante **Source**, cliquez sur **<New Azure Blob Storage Account>** pour ajouter votre compte de stockage précédemment créé.
2. Dans la fenêtre **Add New Azure Blob Storage Account**, remplissez les informations nécessaires. **Shared key** correspond à la clé d’accès primaire visualisé dans l’étape 9 de la section précédente (champs **PRIMARY ACCESS KEY**).

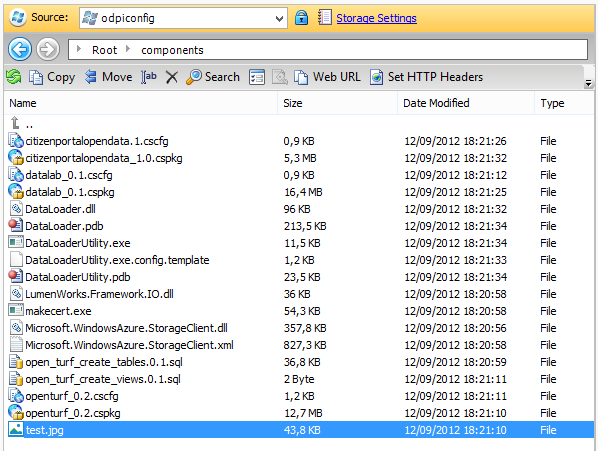


1. Cliquez sur **OK** pour vous connecter et explorer votre compte de stockage qui pour l’instant est vide.
2. Vous allez à présent créer un conteneur pour héberger les packages de service des services cloud computing de Azure contenant les solutions Open Data à proposer. Pour cela, créez un conteneur nommé *components* en cliquant sur le bouton en forme de cube.





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 solutions Open Data ainsi proposées au travers des packages de service copiés.

# Publication d’ODPI dans Windows Azure

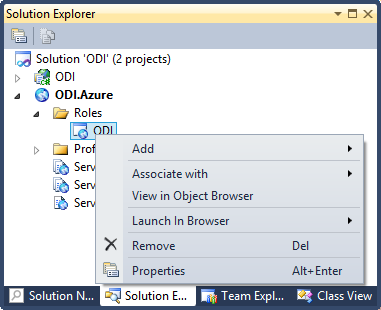
Vous allez à présent aborder les différentes étapes visant à déployer totalement votre solution ODPI dans votre compte Windows Azure.

## Etape 1 – Configuration des paramètres d’ODPI

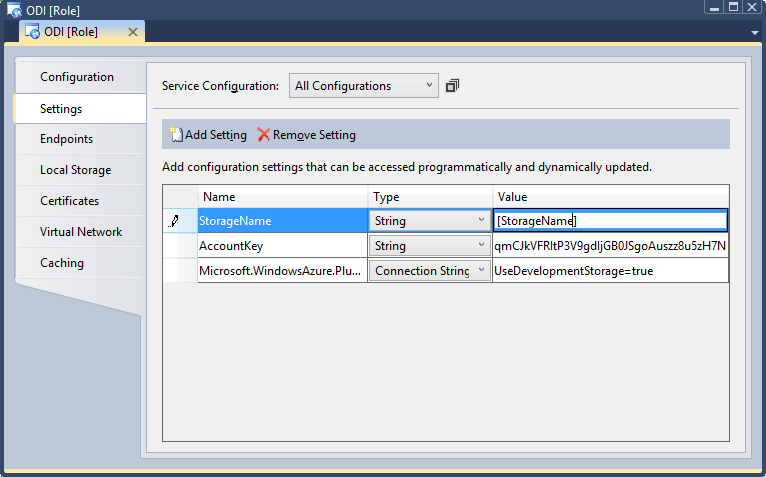
La configuration centralisée de la plateforme ODPI regroupe les éléments nécessaires au fonctionnement du site Web et notamment au bon déploiement des solutions.

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

1. Dans Visual Studio 2010, dans **Solution Explorer**, avec la solution *ODI.sln* ouverte, développez le dossier **Roles** du projet *ODI.Azure*. Ce projet de type Windows Azure vous sert à déployer le projet *ODI*.
2. Faites un clic droit sur *ODI* sous le dossier **Roles**, puis sélectionnez **Properties.**

****

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



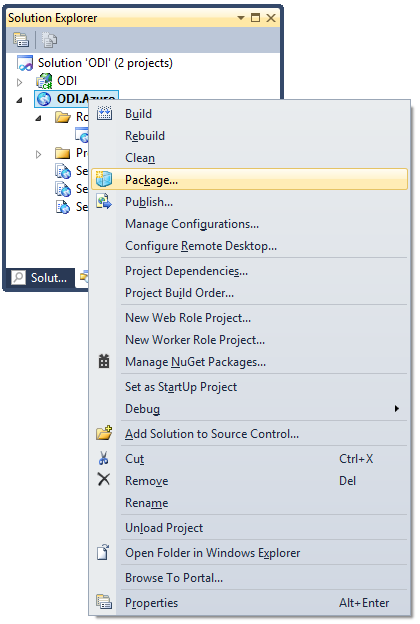
1. Saisissez les informations du compte de stockage créé à la section Etape 1 – Création du compte de stockage en page 11 et contenant à présent vos packages de service pour les solutions Open Data à installer de façon automatisée via la plateforme.

## Etape 2 – Déploiement d’ODPI

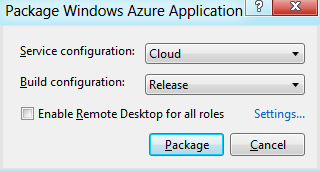
Vous venez de configurer la plateforme d’installation ODPI. Dans cette étape, vous allez procéder au déploiement effectif de la plateforme d’installation ODPI dans votre compte de service Windows Azure associé. Pour cela, il est nécessaire de créer dans un premier temps un package de service qui sera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

Procédez comme suit :

1. Toujours dans Visual Studio 2010, faites un clic-droit sur le projet*ODI.Azure* et choisissez **Package**dans le menu contextuel.

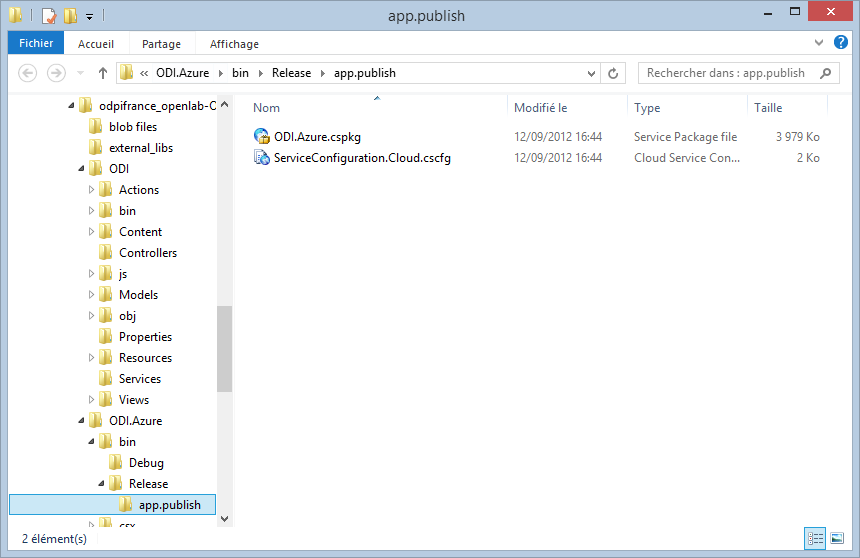


1. Dans la boite de dialogue **Package Windows Azure Application**, choisissez les options **Service Configuration : Cloud**, **Local Configuration : Release** et cliquez sur **Package** pour confirmer.



1. Une fois le package de service créé, une fenêtre Windows s’ouvre dans le dossier app.*publish* correspondant. Ce dossier contient le package *ODI.Azure.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 Windows 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 compte Windows 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 Windows 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.

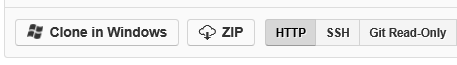
Par défaut, comme indiqué précédemment, la plateforme ODPI propose 3 solutions Open Data à installer dans les services cloud computing de Windows Azure, à savoir :

* La solution communautaire OGDI DataLab disponible sur la forge GitHub à l’adresse <https://github.com/openlab/DataLab> ;
* La solution communautaire ODAF Openturf disponible également sur la forge GitHub à l’adresse : <https://github.com/openlab/Openturf> ;
* Et un exemple de portail Citoyen Open Data simple.

Les solutions communautaires évoluant avec l’investissement de la communauté, il peut s’avérer en effet souhaitable de disposer 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 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)[[17]](#footnote-17) et [GitHub for Windows](http://blogs.msdn.com/b/ogdifrance/archive/2012/06/14/github-for-windows.aspx)[[18]](#footnote-18), tous deux présents sur le blog MSDN [OGDI France](http://blogs.msdn.com/b/ogdifrance/)[[19]](#footnote-19).

Les sources téléchargées, il vous faut à présent générer un package de service Windows 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 Windows Azure est différent.

### Pour OGDI DataLab…

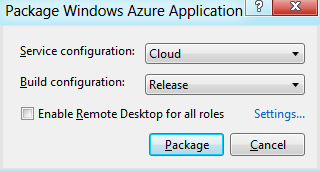
… rendez-vous dans le dossier *Documents* situé à la racine du dossier récupéré de GitHub. Suivez la documentation *Démarrez avec le Kit de démarrage OGDI DataLab.*docx qui vous guidera dans la configuration, la compilation et la génération du package de service.

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 Windows Azure.

Pour cela, procédez comme suit :

1. Ouvrez Visual Studio 2010 en tant qu’administrateur.
2. Faîtes un clic droit sur *ODAF.Azure* puis **Package.**
3. Dans la boite de dialogue **Package Windows Azure Application**, choisissez les options **Service Configuration : Cloud**, **Local Configuration : 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 donc le package *ODAF.Azure.cspkg* qui nous intéresse. Mémorisez donc le chemin vers ce dossier.

### Pour ODAF Openturf…

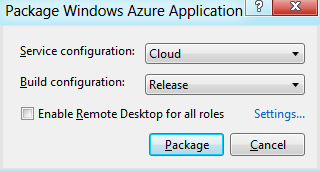
… rendez-vous dans le dossier *Documents* situé à la racine du dossier récupéré de GitHub. Suivez la documentation ***Déployer le Kit de démarrage ODAF Openturf dans Windows Azure.docx*** qui vous guidera dans la configuration, la compilation et la génération du package de service.

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 Windows Azure.

Pour cela, procédez comme suit :

1. Ouvrez Visual Studio 2010 en tant qu’administrateur.
2. Faîtes un clic droit sur *DataBrowser.Cloud* puis **Package**.
3. Dans la boite de dialogue **Package Windows Azure Application**, choisissez les options **Service Configuration : Cloud**, **Local Configuration : Release** et cliquez sur **Package** pour confirmer.

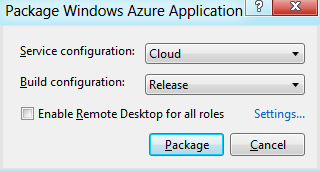


1. Une fois le package créé, une fenêtre Windows s’ouvre dans le dossier app.*publish* correspondant. Ce dossier contient donc le package *DataBrowser.Cloud.cspkg* qui nous intéresse. Mémorisez donc le chemin vers ce dossier.

### Pour le Portail Citoyen Open Data…

…procédez comme suit :

1. Ouvrez Visual Studio 2010 en tant qu’administrateur
2. Faîtes un clic droit sur *DataBrowser.Cloud* puis **Package**.
3. Dans la boite de dialogue **Package Windows Azure Application**, choisissez les options **Service Configuration : Cloud**, **Local Configuration : 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 *DataBrowser.Cloud.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 Windows 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 *ODI.Azure* 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* afin de répercuter les modifications dans les paramètres.

Illustrons ce premier point avec la solution OGDI DataLab. Le paramètre de configuration *SolutionName* a été intégré dans le kit de démarrage interactif 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 );

}

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 la solution 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 Windows Azure. Pour déployer ODPI dans Windows Azure, reportez-vous à la section Publication d’ODPI dans Windows Azure en page 17.

## 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 Windows 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 Windows Azure.

Pour plus d’informations sur la manière d’envoyer un fichier vers un compte de stockage blob Windows Azure, reportez-vous à la section Etape 2 – Configuration du compte de stockage en page 14 et notamment à l’outil utilisé dans ce contexte, à savoir CloudBerry Explorer for Windows 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 compte Windows 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 Windows Azure de votre solution Open Data destinée aux services cloud computing de Windows 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 Windows Azure, référez-vous à l’article [Publication d'une application Windows Azure à l'aide de Windows Azure Tools](http://msdn.microsoft.com/fr-fr/library/windowsazure/ff683672.aspx#PublishAzure)[[20]](#footnote-20).

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 Windows 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 template spécifique à ODPI.

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

### Création du template de configuration

Le template 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 *Windows Azure* pour votre solution et un template de configuration conforme aux attentes de la plateforme ODPI. Il vous faut à présent envoyer ces fichiers sur votre compte de stockage blobs Windows 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 en page 14 qui explique comment se connecter à un compte de stockage blob Windows 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 *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. Une démonstration en ligne est proposée ici : <http://openintel.cloudapp.net/OI.aspx>.

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)[[21]](#footnote-21).

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 template permettant de générer le fichier de configuration Windows 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 template 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 template de configuration Windows 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* -> **Add** -> **View**.

Dans la fenêtre **Add View** qui s’ouvre, entrez le nom de la vue en fonction du nom présent dans la propriété *Template* de la classe *OpenIntelConfig.cs*.

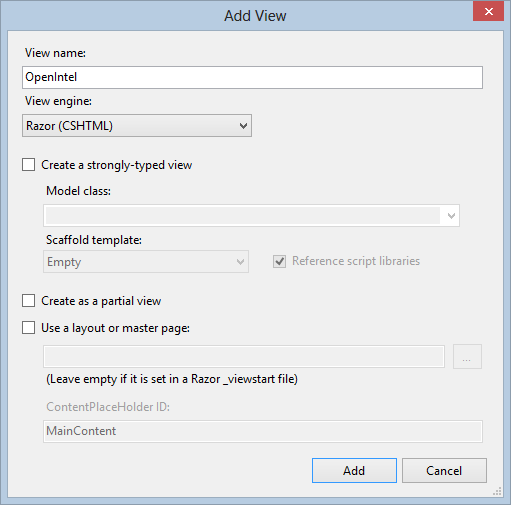
public string Template

{

get { return "**OpenIntel**"; }

}

Soit :



1. Cliquez sur **Add**. Dans la vue, ajoutez les informations disponibles à la saisie. 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 template 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>

Remarque importante : *[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 *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 solutions Open Data.

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 *OdiAppRepo* 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 OdiApp()

{

              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 *OdiAppRepoResource.resx* du dossier *Resources/Models*.

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

| **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 Windows 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 Windows Azure présent dans le compte de stockage |
| **ConfName** | Nom du template de configuration Windows Azure présent dans le compte de stockage |
| **RequiredFiles** | Recense les fichiers qui doivent être obligatoirement dans le compte de stockage hors package de services Windows Azure et template 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 Windows Azure.

## Etape 4 – Déploiement d’ODPI dans Windows Azure

Cette dernière étape permet de mettre en ligne l’installeur automatique de solution Open Data (ODPI) avec la nouvelle solution que vous avez ajouté.

Pour déployer la solution dans Windows Azure, référez-vous à la section Publication d’ODPI dans Windows Azure en page 17.

1. Solution Open Data clé en main: https://mspartner.microsoft.com/fr/fr/Pages/Solutions/Open-Data-Cle-en-main.aspx [↑](#footnote-ref-1)
2. Blog MSDN Open Data France : http://aka.ms/OpenDataFrance/ [↑](#footnote-ref-2)
3. Microsoft Windows Azure : http://www.microsoft.com/france/windows-azure/ [↑](#footnote-ref-3)
4. Services de cloud computing Windows Azure: http://www.windowsazure.com/fr-fr/home/features/cloud-services/ [↑](#footnote-ref-4)
5. Sites Web Windows Azure: http://www.windowsazure.com/fr-fr/home/features/web-sites/ [↑](#footnote-ref-5)
6. GitHub OpenLab: https://github.com/openlab [↑](#footnote-ref-6)
7. Microsoft Windows Azure Training Kit : http://www.microsoft.com/downloads/en/details.aspx?FamilyID=413E88F8-5966-4A83-B309-53B7B77EDF78&displaylang=en [↑](#footnote-ref-7)
8. Microsoft Web Camps Training Kit : http://trainingkit.webcamps.ms/Default.htm [↑](#footnote-ref-8)
9. Windows Azure Team Blog : http://blogs.msdn.com/b/windowsazure/ [↑](#footnote-ref-9)
10. Services de cloud computing Windows Azure: http://www.windowsazure.com/fr-fr/home/features/cloud-services/ [↑](#footnote-ref-10)
11. Microsoft Public License (MS-PL): http://opensource.org/licenses/ms-pl.html [↑](#footnote-ref-11)
12. GitHub OpenLab : https://github.com/openlab [↑](#footnote-ref-12)
13. Blog MSDN OGDI France : http://blogs.msdn.com/b/ogdifrance/ [↑](#footnote-ref-13)
14. 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-14)
15. API du service de blobs Windows Azure : http://msdn.microsoft.com/en-us/library/dd135733 [↑](#footnote-ref-15)
16. Windows Azure Service Management API : http://msdn.microsoft.com/fr-fr/library/ee460799.aspx [↑](#footnote-ref-16)
17. 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-17)
18. GitHub for Windows: http://blogs.msdn.com/b/ogdifrance/archive/2012/06/14/github-for-windows.aspx [↑](#footnote-ref-18)
19. Blog MSDN OGDI France: http://blogs.msdn.com/b/ogdifrance/ [↑](#footnote-ref-19)
20. Publication d'une application Windows Azure à l'aide de Windows Azure Tools : http://msdn.microsoft.com/fr-fr/library/windowsazure/ff683672.aspx#PublishAzure [↑](#footnote-ref-20)
21. 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-21)