Silverlight 4 + RIA Services - Prêt pour les affaires

Exposer JSON Endpoint

par Brad Abrams (Blog) Deepin Prayag (Traduction) (Home)

Date de publication : 14/02/2012

Dernière mise à jour : 20/02/2012

Cet article fait partie d'une série de traductions d'articles de Brad Abrams sur le développement d'applications métier avec Silverlight 4 et .NET RIA Services.

Cette série se concentre uniquement sur la base des applications métier : l'interrogation, la mise à jour, la validation et la sécurisation de vos données métier importantes.

Elle sera également utilisée pour mettre à jour certains billets de la **série Silverlight 3**.

Traduction

Cet article est la traduction la plus fidèle possible de l'article original de **Brad Abrams**, **Silverlight 4 + RIA Services - Ready for Business: Ajax Endpoint**

Prérequis

La procédure pas à pas requiert :

- **Visual Studio 2010** (ou la **version express gratuite**)
- **Silverlight 4 Tools** (inclut **RIA Services**)

Vous pouvez **télécharger l'application complète**.

J'ai implémenté cela avec Silverlight 4 RC, mais je m'attends à ce que cela fonctionne avec Silverlight 4 RTM.

Ajax Endpoint

Poursuivant notre série, je voulais aborder la façon dont un service RIA peut être exposé à votre service dans **JSON**. Cela est très pratique pour les clients Ajax.

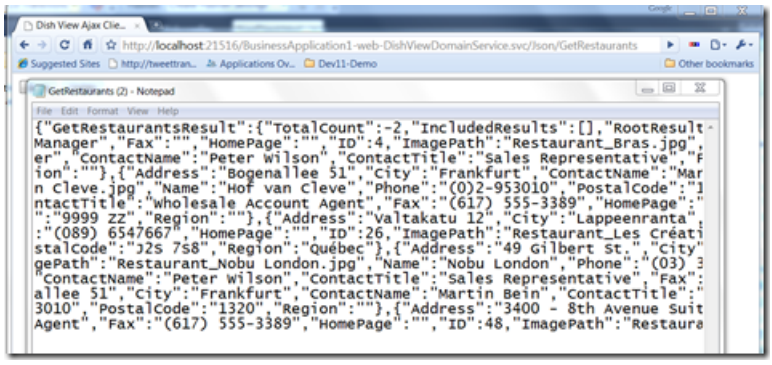
La bonne chose est qu'en activant l'endpoint JSON cela ne nécessite AUCUN changement au Domain Service. Tout ce que vous avez à faire pour l'activer est d'ajouter l'endpoint JSON dans web.config

```
 1: <system.serviceModel>
 2:  <domainServices>
 3:    <endpoints>
 4:      <add name="JSON"
 5:           type="Microsoft.ServiceModel.DomainServices.Hosting.JsonEndpointFactory, Microsoft.ServiceModel.Dom
    Version=4.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" />
 6:      <add name="OData"
 7:           type="System.ServiceModel.DomainServices.Hosting.ODataEndpointFactory, System.ServiceModel.DomainSe
    Version=4.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" />
 8:      <add name="Soap"
 9:           type="Microsoft.ServiceModel.DomainServices.Hosting.SoapXmlEndpointFactory, Microsoft.ServiceModel.
    Version=4.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" />
10:    </endpoints>
11:  </domainServices>
12:
```

Comme vous pouvez le constater, cet extrait ci-dessus montre l'ajout de l'endpoint JSON à partir du **toolkit RIA Services** aussi bien que ceux d'OData et SOAP.

Vous pouvez voir les résultats de l'endpoint en naviguant à l'adresse URL dans ce format :

http://localhost:21516/BusinessApplication1-web-DishViewDomainService.svc/Json/GetRestaurants




```
{"GetRestaurantsResult":{"TotalCount":-2,"IncludedResults":[],
"RootResults":[{"Address":"49 Gilbert St.","City":"London",
"ContactName":"Charlotte Cooper","ContactTitle":"Purchasing Manager",
"Fax":"(171) 555-2222","HomePage":"","ID":1,"ImagePath":
"Restaurant_Alinea.jpg","Name":"Alinea - Updated from Ajax",
"Phone":"(171) 555-2222","PostalCode":"EC1 4SD","Region":""},
{"Address":"P.O. Box 78934","City":"New Orleans","ContactName":
"Shelley Burke",
"ContactTitle":"Order
```

Comme vous pouvez le constater - du JSON agréable à voir. Maintenant pour écrire un client Ajax très simple.

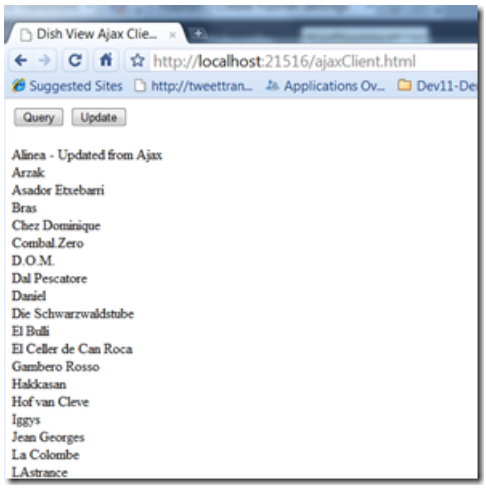
Ci-dessous un exemple de méthode de requête dans le client Ajax

```
         function query() {
            var xmlhttp = new XMLHttpRequest();
            xmlhttp.open("GET", "BusinessApplication1-web-DishViewDomainService.svc/Json/
GetRestaurants", false);
            xmlhttp.send();
            var rawResults = JSON.parse(xmlhttp.responseText);
            var results = rawResults.GetRestaurantsResult.RootResults;
            var entity
            for (var i = 0; i < results.length; i++)
            {
                entity = results[i];
                document.getElementById('results').innerHTML += ' <br> ' + entity.Name;
            }

        }
```

Ceci est relié à un bouton très simple

```
<button type="button" onclick="query()">
            Query</button>
```



La mise à jour est un peu plus compliquée, mais toujours basique :

```
        function update() {
            var operation = {};

 operation.Entity = { "__type": "Restaurant:#BusinessApplication1.Web", "ID": 1, "Name": "Alinea - Updated from A

 operation.OriginalEntity = { "__type": "Restaurant:#BusinessApplication1.Web", "ID": 1, "Name": "Alinea" };
            operation.Operation = 3; //update
            var csData = JSON.stringify({ "changeSet": [operation] });
            var xmlhttp = new XMLHttpRequest();
            xmlhttp.open('POST', 'BusinessApplication1-web-DishViewDomainService.svc/Json/
SubmitChanges', false);
            xmlhttp.setRequestHeader("Content-Type", "application/json");
            xmlhttp.send(csData);
            var results = xmlhttp.responseText;
            document.getElementById('results').innerHTML = results;
        }
```

Dans cette démo, nous avons montré comment activer le client Ajax/JSON pour RIA Services.

Conclusion

Ceci conclut la dixième partie de cette série. Dans la prochaine partie nous verrons comment déployer notre application construite en utilisant RIA Services.

Liens

- **Visual Studio 2010**
- **Visual Studio 2010 Express**
- **Silverlight 4 Tools**
- **RIA Services**
- **Télécharger l'application complète**

Remerciements

Je tiens ici à remercier **Brad Abrams** pour son aimable autorisation de traduire l'article.
Je remercie **Jean-Michel Ormes** pour sa relecture technique et ses propositions.
Je remercie également **xyz** pour sa relecture orthographique et ses propositions.