Module 8: Accès aux données distantes

# **Contenu:**

|  |  |
| --- | --- |
|  | [Présentation du module](https://www.skillpipe.com/api/2.1/content/urn:uuid:21ae3267-8517-4025-bf9a-2dd83635ed12@2020-12-11T20:16:12Z/OPS/html/20483C08.html#P1) |
| **Leçon 1:** | [Accès aux données sur le Web](https://www.skillpipe.com/api/2.1/content/urn:uuid:21ae3267-8517-4025-bf9a-2dd83635ed12@2020-12-11T20:16:12Z/OPS/html/20483C08.html#P2) |
| **Leçon 2:** | [Accès aux données à l'aide des services connectés OData](https://www.skillpipe.com/api/2.1/content/urn:uuid:21ae3267-8517-4025-bf9a-2dd83635ed12@2020-12-11T20:16:12Z/OPS/html/20483C08.html#P3) |
| **Laboratoire:** | [Récupération et modification des données de note à distance](https://www.skillpipe.com/api/2.1/content/urn:uuid:21ae3267-8517-4025-bf9a-2dd83635ed12@2020-12-11T20:16:12Z/OPS/html/20483C08.html#P4) |
|  | [Revue du module et points à retenir](https://www.skillpipe.com/api/2.1/content/urn:uuid:21ae3267-8517-4025-bf9a-2dd83635ed12@2020-12-11T20:16:12Z/OPS/html/20483C08.html#P5) |

# **Présentation du module**

Les systèmes se composent souvent de nombreux composants et services; certains peuvent être hébergés dans l'infrastructure de votre organisation, tandis que d'autres peuvent être hébergés dans des centres de données n'importe où dans le monde. La capacité des applications à interagir avec ces services est une exigence courante dans les applications modernes.

Dans ce module, vous apprendrez à utiliser les classes de demande et de réponse dans l'espace de noms System.Net pour manipuler directement les sources de données distantes. Vous apprendrez également à utiliser les services de données Windows® Communication Foundation (WCF) pour exposer et consommer un modèle de données d'entité (EDM) sur le Web.

### **Objectifs**

Après avoir terminé ce module, vous serez en mesure de:

|  |  |
| --- | --- |
| • | Envoyez et recevez des données depuis des services Web et d'autres sources de données distantes. |
| • | Accédez aux données à l'aide des services de données WCF. |

# Leçon 1: Accéder aux données sur le Web

Les données sont souvent exposées sur le Web via des services Web ou d'autres interfaces de programmation d'application (API). Pour pouvoir utiliser ces sources de données dans votre application, vous avez besoin d'un moyen d'envoyer et de recevoir des messages afin de pouvoir établir une connexion et, en fin de compte, envoyer et recevoir des données.

Dans cette leçon, vous apprendrez à consommer des sources de données distantes telles que les services Web et les sites FTP (File Transfer Protocol), qui comprendront comment créer une demande, fournir des informations d'identification pour l'authentification, regrouper les données dans la demande et traiter les données qui est renvoyé dans la réponse.

## **Objectifs de la leçon**

Après avoir terminé cette leçon, vous serez en mesure de:

|  |  |
| --- | --- |
| • | Décrivez comment Microsoft® .NET Framework utilise les demandes et les réponses. |
| • | Exposez les types de services Web à l'aide de contrats de données. |
| • | Créez une demande et traitez une réponse. |
| • | Fournissez des informations d'identification ou un jeton de sécurité pour permettre à la source de données distante d'effectuer l'authentification. |
| • | Envoyez et recevez des données. |

## **Vue d'ensemble de la connectivité Web dans le .NET Framework**

Le .NET Framework fournit l'infrastructure pour vous permettre d'intégrer vos applications à des sources de données distantes. La source de données distante peut être n'importe quoi, d'un site FTP à un service Web ASP.NET ou WCF.

Lors de la consommation d'une source de données distante dans votre application, le .NET Framework utilise des demandes et des réponses pour transmettre des messages entre les deux (ou plus) systèmes. Cela implique les étapes suivantes:

|  |  |
| --- | --- |
| 1. | Initiez une connexion à la source de données distante. Cela peut inclure la transmission d'un jeton de sécurité ou d'informations d'identification utilisateur afin que la source de données distante puisse vous authentifier. |
| 2. | Envoyez un message de demande à la source de données distante. Ce message peut également inclure toutes les données dont la source de données distante a besoin pour satisfaire la demande, telles que l'identifiant de l'enregistrement de vente que vous souhaitez récupérer. |
| 3. | Attendez que la source de données distante traite votre demande et émette une réponse. En tant que développeur, vous n'avez aucun contrôle sur le temps qu'il faut pour recevoir une réponse d'un service Web. |
| 4. | Traitez la réponse et extrayez toutes les données incluses dans la réponse. |

**Noter:**Toutes les communications ne doivent pas nécessairement inclure à la fois un message de demande et de réponse. Selon la nature de l'application, il peut ne s'appliquer qu'à envoyer un message. Par exemple, si votre application souhaite informer un service Web qu'il a terminé de traiter une tâche, il vous suffit d'envoyer une demande. Ceci est connu comme une opération à sens unique.

**Connectivité Web dans le .NET Framework**

Le .NET Framework fournit l'espace de noms System.Net, qui contient plusieurs classes de demande et de réponse qui vous permettent de cibler différentes sources de données. Le tableau suivant décrit certaines de ces classes de demande et de réponse.

| **Classer** | **La description** |
| --- | --- |
| **Demande Web** | Une classe abstraite qui fournit l'infrastructure de base pour toute demande adressée à un URI (Uniform Resource Identifier). |
| **WebResponse** | Une classe abstraite qui fournit l'infrastructure de base pour traiter toute réponse d'un URI. |
| **HttpWebRequest** | Un dérivé de la classe WebRequest qui fournit des fonctionnalités pour toute requête Web HTTP. |
| **HttpWebResponse** | Un dérivé de la classe WebResponse qui fournit des fonctionnalités pour traiter toute réponse Web HTTP. |
| **FtpWebRequest** | Un dérivé de la classe WebRequest qui fournit des fonctionnalités pour toute requête FTP. |
| **FtpWebResponse** | Un dérivé de la classe WebResponse qui fournit des fonctionnalités pour traiter toute réponse FTP. |
| **FileWebRequest** | Un dérivé de la classe WebRequest qui fournit des fonctionnalités pour demander des fichiers. |
| **FileWebResponse** | Un dérivé de la classe WebResponse qui fournit des fonctionnalités pour traiter une réponse de fichier. |

Selon que vous souhaitez envoyer une demande à un service Web à l'aide de HTTP ou que vous souhaitez télécharger un fichier à partir d'un site FTP, le .NET Framework fournit les classes nécessaires pour que vous utilisiez ces sources de données distantes dans vos applications.

**Lecture supplémentaire:**Pour plus d'informations sur l'espace de noms System.Net, reportez-vous à la page Espace de noms System.Net à l'adresse<https://aka.ms/moc-20483c-m8-pg1>.

## **Définition d'un contrat de données**

Une source de données distante peut exposer n'importe quel type de données. Par exemple, un service Web peut exposer des flux binaires, des valeurs scalaires ou des objets personnalisés. Le choix du type de données que vous exposez est déterminé par les exigences de votre application, mais la façon dont vous l'exposez est contrôlée par les contrats de données que vous définissez dans vos services Web.

Si vous souhaitez exposer un objet personnalisé à partir d'un service Web, vous devez fournir des métadonnées qui décrivent la structure de l'objet. Le processus de sérialisation utilise ces métadonnées pour convertir votre objet dans un format transportable, tel que XML ou JavaScript Object Notation (JSON). Ces métadonnées fournissent des instructions au sérialiseur qui vous permettent de contrôler quels types et membres sont sérialisés.

**Contrats de données dans le .NET Framework**

Le .NET Framework fournit l'espace de noms System.Runtime.Serialization, qui comprend les attributs DataContract et DataMember. Vous pouvez utiliser ces attributs pour définir des types et des membres sérialisables.

L'exemple de code suivant montre comment définir un type sérialisable à l'aide des attributs DataContract et DataMember.

**Définition d'un contrat de données**

[DataContract ()]

public class SalesPerson

{

[DataMember ()]

chaîne publique FirstName {get; ensemble; }

[DataMember ()]

chaîne publique LastName {get; ensemble; }

[DataMember ()]

chaîne publique Area {get; ensemble; }

[DataMember ()]

chaîne publique EmailAddress {get; ensemble; }

}

**Lecture supplémentaire:**Pour plus d'informations sur les attributs DataContract et DataMember, reportez-vous à la page Classe DataContractAttribute à l'adresse<https://aka.ms/moc-20483c-m8-pg2>.

## **Créer une demande et traiter une réponse**

Le protocole utilisé par votre source de données distante détermine les classes de demande et de réponse que vous devez utiliser. Quelles que soient les classes que vous utilisez, vous pouvez appliquer le même modèle pour envoyer une requête et recevoir une réponse.

**Les classes HttpWebRequest et HttpWebResponse**

Les étapes suivantes décrivent comment envoyer une requête HTTP à un service Web et traiter la réponse à l'aide des classes HttpWebRequest et HttpWebResponse:

|  |  |
| --- | --- |
| 1. | Obtenez un URI du service Web auquel vous souhaitez envoyer une demande. L'exemple de code suivant montre un URI HTTP qui traite la méthode GetSalesPerson dans le Fourth Coffee Sales Service.  var uri = "http://sales.fourthcoffee.com/SalesService.svc/GetSalesPerson"; |
| 2. | Créez un objet de requête pour configurer la requête et accéder à la réponse. L'exemple de code suivant montre comment créer un objet HttpWebRequest.  var request = WebRequest.Create (uri) as HttpWebRequest;  **Noter:**Quel que soit le type d'objet de requête dont vous avez besoin, vous utilisez toujours la méthode Create statique que la classe de base WebRequest expose, puis convertissez le type de requête dont vous avez besoin. |
| 3. | Obtenez la réponse de l'objet de requête. L'exemple de code suivant montre comment obtenir la réponse d'un objet HttpWebRequest.  var response = request.GetResponse () as HttpWebResponse;  **Noter:**Similaire à la création d'un objet de requête, vous créez un objet de réponse en appelant la méthode GetResponse sur l'objet WebRequest, puis vous effectuez un cast vers le type de réponse dont vous avez besoin. |
| 4. | Accédez et traitez la réponse à l'aide des différents membres fournis par l'objet WebResponse. L'exemple de code suivant montre comment utiliser et afficher l'état de la réponse à l'aide de la propriété StatusCode. |

var status = response.StatusCode; // Renvoie OK si une réponse est reçue.

Si la source de données distante utilise un protocole différent, tel que FTP, vous pouvez appliquer le même modèle mais utiliser les classes FtpWebRequest et FtpWebResponse à la place.

**Gestion des exceptions de réseau**

Lorsque vous consommez des ressources distantes, qu'il s'agisse d'un site FTP ou d'un service Web HTTP, vous ne pouvez pas garantir que la ressource est en ligne et à l'écoute de votre demande. Si vous essayez d'envoyer une demande à un service Web à l'aide de la classe HttpWebRequest et que le service Web n'est pas en ligne, l'appel de la méthode GetResponse lèvera une exception WebException avec le message suivant:

WebException - Le serveur distant a renvoyé une erreur: (404) Not Found.

De même, si vous essayez d'accéder à un service Web sécurisé avec des informations d'identification ou un jeton de sécurité incorrects, l'appel de la méthode GetResponse lèvera une exception WebException avec le message suivant:

WebException - Le serveur distant a renvoyé une erreur: 401 non autorisé

Si vous ne gérez pas ces exceptions dans votre code, elles entraîneront l'échec de votre application, offrant une expérience utilisateur médiocre. Par conséquent, au minimum, vous devez inclure toute logique qui communique avec une source de données distante dans une instruction try / catch, avec les exceptions de gestion de bloc catch de type WebException.

## **Authentification d'une requête Web**

Les sources de données distantes sont souvent protégées pour empêcher les utilisateurs non autorisés d'utiliser le service et d'accéder aux données. L'exposition d'une source de données non protégée sur le Web peut amener des utilisateurs non autorisés à envoyer des demandes et à augmenter la charge sur la source de données. Il existe de nombreuses façons de sécuriser les sources de données distantes. Une approche consiste à authentifier chaque utilisateur qui tente de se connecter à la source de données distante.

**Type d'authentification**

Le tableau suivant décrit certaines des techniques d'authentification courantes que vous pouvez utiliser pour sécuriser les sources de données distantes.

| **Mécanisme d'authentification** | **La description** |
| --- | --- |
| De base | Permet aux utilisateurs de s'authentifier à l'aide d'un nom d'utilisateur et d'un mot de passe. L'authentification de base ne crypte pas les informations d'identification pendant le transit, ce qui signifie que des utilisateurs non autorisés peuvent accéder aux informations d'identification. |
| Digérer | Permet aux utilisateurs de s'authentifier à l'aide d'un nom d'utilisateur et d'un mot de passe, mais contrairement à l'authentification de base, les informations d'identification sont chiffrées. |
| les fenêtres | Permet aux clients de s'authentifier à l'aide de leurs informations d'identification de domaine Windows. L'authentification Windows utilise le hachage ou un jeton Kerberos pour authentifier les utilisateurs en toute sécurité. L'authentification Windows est généralement utilisée pour fournir une expérience d'authentification unique (SSO) dans les organisations. |
| Certificat | Permet uniquement aux clients sur lesquels le certificat approprié est installé de s'authentifier auprès du service. |

La nature du service et son lieu d'hébergement sont susceptibles d'influencer le choix du mécanisme d'authentification par une organisation. Par exemple, un service exposé sur l'intranet d'une organisation peut utiliser l'authentification Windows afin que les utilisateurs puissent s'authentifier à l'aide de leurs informations d'identification des services de domaine Active Directory® (AD DS). L'utilisation de l'authentification Windows dans ce scénario fournira aux utilisateurs une expérience SSO et ne leur demandera pas de se souvenir des informations d'identification pour chaque service qu'ils consomment.

Le .NET Framework fournit un certain nombre de classes que vous pouvez utiliser pour vous authentifier auprès d'une source de données distante sécurisée.

**Authentification des utilisateurs à l'aide d'informations d'identification**

Lorsque vous communiquez avec une source de données distante qui nécessite un nom d'utilisateur et un mot de passe, vous utilisez la propriété Credentials exposée par toute classe dérivée de la classe de base WebRequest pour transmettre les informations d'identification à la source de données. Vous pouvez définir la propriété Credentials sur n'importe quel objet qui implémente l'interface ICredentials:

|  |  |
| --- | --- |
| • | La classe NetworkCredential implémente l'interface ICredentials et vous permet d'encapsuler un nom d'utilisateur et un mot de passe. L'exemple de code suivant montre comment instancier un objet NetworkCredential et transmettre les valeurs du nom d'utilisateur et du mot de passe au constructeur de classe.  var uri = "http://Sales.FourthCoffee.com/SalesService.svc/GetSalesPerson";  var request = WebRequest.Create (uri) as HttpWebRequest;  var username = "jespera";  var mot de passe = "Pa $$ w0rd";  request.Credentials = new NetworkCredential (nom d'utilisateur, mot de passe); |
| • | La classe CredentialCache fournit un certain nombre de membres qui vous permettent d'obtenir des informations d'identification sous la forme d'un objet ICredentials. Ces membres incluent la propriété DefaultCredentials, qui renvoie un objet ICredentials contenant les informations d'identification avec lesquelles l'utilisateur est actuellement connecté. L'exemple de code suivant montre comment utiliser la propriété DefaultCredentials pour obtenir les informations d'identification de l'utilisateur actuel. |

var uri = "http://Sales.FourthCoffee.com/SalesService.svc/GetSalesPerson";

var request = WebRequest.Create (uri) as HttpWebRequest;

request.Credentials = CredentialCache.DefaultCredentials;

L'authentification des utilisateurs à l'aide de l'authentification de base ou de Windows est courante, mais elle oblige les utilisateurs à se souvenir et à fournir des informations d'identification. Vous pouvez également choisir d'authentifier les clients à l'aide d'un certificat X509.

**Lecture supplémentaire:**Pour plus d'informations sur la classe NetworkCredential, reportez-vous à la page Classe NetworkCredential à l'adresse<https://aka.ms/moc-20483c-m8-pg3>.

**Authentification des utilisateurs à l'aide d'un certificat X509**

Vous pouvez utiliser un certificat X509 comme jeton de sécurité pour authentifier les utilisateurs. Au lieu que les utilisateurs doivent spécifier leurs informations d'identification lorsqu'ils accèdent à la source de données distante, ils auront automatiquement accès tant que la demande envoyée inclut le certificat X509 correct. Pour les utilisateurs, cela signifie qu'ils ont besoin du bon certificat X509 dans leur magasin de certificats sur l'ordinateur sur lequel l'application cliente s'exécute.

L'exemple de code suivant montre comment créer un objet HttpWebRequest et ajouter un certificat X509 à l'objet de demande à l'aide de la propriété ClientCertificates.

**Certificats HttpWebRequest et x509**

var uri = "http://Sales.FourthCoffee.com/SalesService.svc/GetSalesPerson";

var request = WebRequest.Create (uri) as HttpWebRequest;

certificat var = FourthCoffeeCertificateServices.GetCertificate ();

request.ClientCertificates.Add (certificat);

Lorsque la source de données distante reçoit la demande, elle doit extraire et traiter le certificat X509. Si le certificat X509 n'est pas valide, la source de données distante peut renvoyer une réponse appropriée, telle qu'un message «Le serveur distant a renvoyé une erreur: 401 non autorisé».

**Noter:**La méthode GetCertificate du service Web illustrée dans l'exemple précédent renvoie un objet X509Certficate2. Ce type fournit un accès par programme aux propriétés du certificat X509.

## **Envoi et réception de données**

Vous utilisez des demandes et des réponses pour envoyer ou récupérer des données à partir d'une source de données distante. Par exemple, vous souhaiterez peut-être ajouter un nouvel enregistrement à un service Web ou récupérer un fichier à partir d'un site FTP.

Chaque dérivé des classes WebRequest et WebResponse vous permet d'envoyer et de recevoir des données à l'aide du protocole spécifique implémenté par la classe. Par exemple, la classe HttpWebRequest vous permet d'envoyer des demandes à un service Web à l'aide du protocole HTTP.

La classe de base WebRequest comprend les membres suivants que vous pouvez utiliser pour envoyer des données à une source de données distante:

|  |  |
| --- | --- |
| • | **ContentType**. Cette propriété vous permet de définir le type de données que la demande enverra. Par exemple, si vous envoyez des données JSON à l'aide de la classe HttpWebRequest, vous utilisez le type de contenu application / json. |
| • | **Méthode**. Cette propriété vous permet de définir le type de méthode que l'objet WebRequest utilisera pour envoyer la requête. Par exemple, si vous téléchargez un fichier à l'aide de la classe FtpWebRequest, vous utilisez la méthode STOR. |
| • | **ContentLength**. Cette propriété vous permet de définir le nombre d'octets que la demande enverra. |
| • | **GetRequestStream**. Cette méthode vous permet d'accéder et d'écrire des données dans le flux de données sous-jacent dans l'objet de requête. |

La classe WebResponse fournit la méthode GetResponseStream, qui vous permet d'accéder aux données de la réponse et de les diffuser en continu.

**Envoi de données à une source de données distante**

Que vous envoyiez des données à un service Web ou que vous téléchargiez un fichier sur un site FTP, le processus de création de la demande reste le même:

|  |  |
| --- | --- |
| 1. | Obtenez l'URI de la source de données distante et les données que vous souhaitez envoyer. |
| 2. | Créez l'objet de requête. |
| 3. | Configurez l'objet de demande, qui inclut la définition de la méthode de demande et la longueur des données que la demande enverra. |
| 4. | Diffusez les données vers l'objet de requête. |

L'exemple de code suivant montre comment utiliser une opération POST pour envoyer une chaîne JSON à un service Web à l'aide de la classe HttpWebRequest.

**Envoi de données à un service Web**

// Récupère l'URI et les données.

var uri = "http://sales.fourthcoffee.com/SalesService.svc/GetSalesPerson";

var rawData = Encoding.Default.GetBytes ("{\" emailAddress \ ": \" jesper@fourthcoffee.com \ "}");

// Crée l'objet de requête.

var request = WebRequest.Create (uri) as HttpWebRequest;

// Configurez le type de données que la demande enverra.

request.Method = "POST";

request.ContentType = "application / json";

request.ContentLength = rawData.Length;

// Diffusez les données vers la requête.

var dataStream = request.GetRequestStream ();

dataStream.Write (rawData, 0, rawData.Length);

dataStream.Close ();

L'exemple de code suivant montre comment télécharger un fichier sur un site FTP à l'aide de la classe FtpWebRequest.

**Téléchargement d'un fichier sur un site FTP**

// Récupère l'URI et les données.

var uri = "ftp://sales.fourthcoffee.com/FileRepository/SalesForcast.xls";

var rawData = File.ReadAllBytes ("C: \\ FourthCoffee \\ Documents \\ SalesForecast.xls");

// Crée l'objet de requête.

var request = WebRequest.Create (uri) as FtpWebRequest;

// Configurez le type de données que la demande enverra.

request.Method = WebRequestMethods.Ftp.UploadFile;

request.ContentLength = rawData.Length;

// Diffusez les données vers la requête.

var dataStream = request.GetRequestStream ();

dataStream.Write (rawData, 0, rawData.Length);

dataStream.Close ();

**Réception de données à partir d'une source de données distante**

Pour obtenir les données qu'une réponse peut contenir, vous utilisez la méthode GetResponseStream de l'objet de réponse. La méthode GetResponseStream renvoie un flux que vous pouvez lire pour obtenir les données.

L'exemple de code suivant montre comment utiliser la méthode GetResponseStream pour accéder aux données de réponse.

**Lecture des données à partir de la réponse**

var request = WebRequest.Create (uri) as HttpWebRequest;

...

var response = request.GetResponse () as HttpWebResponse;

var stream = new StreamReader (response.GetResponseStream ());

// Code pour traiter le flux.

stream.Close ();

Lorsque vous avez acquis le flux de réponse, vous devez alors vérifier que les données sont au format correct. Par exemple, si la réponse renvoyée est au format JSON, vous pouvez utiliser la classe DataContractJsonSerializer pour sérialiser les données JSON brutes dans un objet que vous pouvez consommer dans votre code. Si les données sont au format XML, vous pouvez également utiliser la classe SoapFormatter pour désérialiser les données ou LINQ (Language-Integrated Query) en XML pour analyser manuellement le XML.

**Lecture supplémentaire:**Pour plus d'informations sur LINQ to XML, reportez-vous à la page LINQ to XML à l'adresse<https://aka.ms/moc-20483c-m8-pg4>.

## **Démonstration: utilisation d'un service Web**

Dans cette démonstration, vous utiliserez les classes HttpWebRequest et HttpWebResponse pour consommer un service Web via HTTP.

L'application utilisera les classes System.Net pour obtenir les enregistrements des vendeurs à partir du quatrième service de vente de café. Vous enverrez une demande qui contiendra l'adresse e-mail de l'enregistrement du vendeur que vous souhaitez obtenir au format JSON, traiter la réponse, puis afficher les détails de l'enregistrement dans l'interface utilisateur.

### **Démonstration étapes**

Vous trouverez les étapes dans la section Démonstration: Consommation d'un service Web «sur la page suivante: https://github.com/MicrosoftLearning/20483-Programming-in-C-Sharp/blob/master/Instructions/20483C\_MOD08\_DEMO.md

# Leçon 2: Accéder aux données à l'aide des services connectés OData

WCF Data Services suit le modèle architectural REST (Representational State Transfer) et utilise des normes Web ouvertes telles que l'Open Data Protocol (OData) pour exposer et consommer des données sur le Web. En suivant ces normes, vous pouvez créer des solutions basées sur les services de données WCF auxquelles une grande variété d'applications clientes peuvent facilement accéder, quelle que soit la technologie utilisée pour implémenter l'application cliente.

Dans cette leçon, vous apprendrez à créer un service de données WCF et à définir les entités et opérations que vous souhaitez exposer. Vous apprendrez également à créer une bibliothèque cliente et à consommer les entités et opérations que vous avez exposées.

## **Objectifs de la leçon**

Après avoir terminé cette leçon, vous serez en mesure de:

|  |  |
| --- | --- |
| • | Décrivez l'objectif d'un service de données WCF. |
| • | Définissez un service de données WCF. |
| • | Exposez un modèle de données à l'aide d'un service de données WCF. |
| • | Exposez les méthodes Web à l'aide d'un service de données WCF. |
| • | Créez une bibliothèque cliente et référencez un service de données WCF. |
| • | Récupérez et manipulez les entités exposées par un service de données WCF. |

## **Qu'est-ce que les services de données WCF?**

WCF Data Services vous permet de créer et d'accéder à des services de données sur le Web. Vous exposez vos données en tant que ressources auxquelles les applications peuvent accéder à l'aide d'un URI. Ces ressources sont exposées sous forme d'ensembles d'entités liées par des associations, les mêmes concepts que dans un GED. Cependant, vous pouvez exposer des données à partir de nombreux types de stockage, y compris des bases de données et des classes CLR (Common Language Runtime).

WCF Data Services utilise des URI pour adresser les données et des formats simples et bien connus pour représenter ces données, tels que XML et Atom. Il en résulte que les données sont servies comme une collection de ressources de style REST.

REST est devenu un modèle populaire pour la mise en œuvre de services Web qui ont besoin d'accéder aux données (d'autres modèles, tels que ceux basés sur le modèle WebRequest / WebResponse décrit dans la leçon précédente, sont plus adaptés à l'appel de méthodes distantes). REST décrit un schéma hiérarchique sans état pour représenter les ressources et les objets métier sur un réseau. Les ressources sont accessibles via des URI qui identifient les données à récupérer. Par exemple, Fourth Coffee peut choisir de rendre les données de tous ses commerciaux disponibles via l'URI suivant:

http://FourthCoffee.com/SalesService.svc/SalesPersons

Les données d'un commercial spécifique peuvent être récupérées en spécifiant l'identifiant (tel qu'un numéro de vendeur) pour ce commercial, comme ceci:

http://FourthCoffee.com/SalesService.svc/SalesPersons/99

De même, les détails des produits vendus peuvent être disponibles via l'URI suivant:

http://FourthCoffee.com/SalesService.svc/Products

Les détails d'un produit spécifique peuvent être récupérés en incluant l'ID de produit dans l'URI:

http://FourthCoffee.com/SalesService.svc/Products/25

**Noter:**Le schéma d'URI exact qu'un service Web utilise pour exposer des données est une décision de l'organisation qui implémente le service Web, mais les exemples présentés ici illustrent un modèle commun.

Le modèle REST exécute des requêtes HTTP GET pour récupérer des données, mais le modèle REST prend également en charge les opérations d'insertion, de mise à jour et de suppression à l'aide de requêtes HTTP PUT, POST et DELETE.

Le modèle REST permet à un service Web d'étendre les URI pour prendre en charge le filtrage, le tri et la pagination des données. Par exemple, l'URI suivant envoie une demande qui récupère uniquement les 10 premiers commerciaux:

http://FourthCoffee.com/SalesService.svc/SalesPersons?top=10

La liste des filtres et autres fonctionnalités dépend de la façon dont le service Web est implémenté, mais des fonctionnalités telles que celles-ci sont disponibles avec WCF Data Services. En outre, WCF Data Services vous permet d'étendre vos services Web en écrivant des opérations de service en tant que méthodes qui exécutent la logique métier sur le serveur. Ces méthodes sont ensuite accessibles en tant qu'URI d'une manière similaire aux ressources. Vous pouvez également définir des intercepteurs, qui sont appelés lorsque vous interrogez, insérez, mettez à jour ou supprimez des données et pouvez valider ou modifier les données, appliquer la sécurité ou rejeter la modification.

**Lecture supplémentaire:**Pour plus d'informations sur les services de données WCF, reportez-vous à la page Services de données WCF à l'adresse<http://go.microsoft.com/fwlink/?LinkID=267815>.

## **Définition d'un service de données WCF**

À l'aide de WCF Data Services, vous pouvez exposer des données à partir de sources de données relationnelles telles que Microsoft SQL Server® via un schéma conceptuel EDM créé à l'aide de ADO.NET Entity Framework, et vous pouvez permettre à une application client en utilisant ce schéma.

**Noter:**Les services de données WCF peuvent également exposer des données non relationnelles, mais cela nécessite la création de classes personnalisées. Les services de données WCF fonctionnent le plus naturellement avec le modèle présenté par ADO.NET Entity Framework.

Un service de données WCF est basé sur la classe générique System.Data.Services.DataService. Cette classe attend un paramètre de type qui est une collection contenant au moins une propriété qui implémente l'interface IQueryable, telle que la classe DbContext pour un jeu d'entités défini à l'aide de Entity Framework. Le type DataService implémente la fonctionnalité de base pour exposer les entités de cette collection sous la forme d'une série de ressources REST.

L'exemple de code suivant montre la définition d'un service de données WCF basé sur une classe DbContext appelée FourthCoffee qui est générée à l'aide de ADO.NET Entity Framework.

**Définition d'un service de données**

classe publique FourthCoffeeDataService: DataService <FourthCoffee>

{

...

}

Vous pouvez implémenter des méthodes dans un service de données WCF qui spécifient les entités à exposer à partir de l'EDM sous-jacent et qui configurent la taille des ensembles de données que le service de données présente. Vous pouvez également remplacer les méthodes héritées de la classe DataService pour personnaliser le mode de fonctionnement du service.

Par défaut, WCF Data Services utilise un schéma d'adressage simple qui expose les ensembles d'entités définis dans l'EDM spécifié. Lorsque vous utilisez un service de données WCF, vous adressez ces ressources d'entité comme un ensemble d'entités qui contient des instances d'un type d'entité. Par exemple, supposons que l'URI suivant (indiqué dans la rubrique précédente) renvoie toutes les entités SalesPerson qui ont été définies dans l'EDM qui a été utilisé pour construire un service de données WCF:

http://FourthCoffee.com/SalesService.svc/SalesPersons

L'élément "/ SalesPersons" de l'URI pointe vers l'ensemble d'entités SalesPersons, qui est le conteneur des instances SalesPerson.

**Lecture supplémentaire:**Pour plus d'informations sur la définition d'un service de données WCF, reportez-vous à la page Exposing Your Data as a Service (WCF Data Services) à l'adresse<http://go.microsoft.com/fwlink/?LinkID=267816>.

## **Exposition d'un modèle de données à l'aide des services de données WCF**

Pour des raisons de sécurité, WCF Data Services n'expose pas automatiquement les ressources, telles que les collections d'entités, implémentées par EDM. Vous spécifiez une stratégie qui active ou désactive l'accès aux ressources dans la méthode InitializeService de votre service de données. Cette méthode prend un objet DataServiceConfiguration, qui possède une propriété SetEntitySetAccessRule que vous utilisez pour définir la stratégie d'accès.

L'exemple de code suivant montre comment autoriser l'accès à toutes les ressources exposées par le service de données WCF.

**Accès au service de données à toutes les entités**

classe publique FourthCoffeeDataService: DataService <FourthCoffee>

{

public static void InitializeService (configuration DataServiceConfiguration)

{

config.SetEntitySetAccessRule ("\*", EntitySetRights.All);

}

}

Les paramètres de la méthode SetEntitySetAccessRule sont le nom d'une ressource et les droits d'accès à accorder sur cette ressource. Vous pouvez spécifier une ressource explicitement ou utiliser des caractères génériques. La valeur \* indiquée dans l'exemple de code est un moyen abrégé de spécifier toutes les ressources publiées par le service de données WCF. La valeur EntitySetRights.All accorde un accès illimité à ces ressources.

**Noter:**Les énumérations et les classes partielles implémentées dans un EDM ne se propagent pas via les services de données WCF. Si vous souhaitez exposer des énumérations et des extensions à des classes partielles, vous devez inclure ces types dans un assembly distinct que vous pouvez référencer directement dans le service et l'application cliente.

**Lecture supplémentaire:**Pour plus d'informations sur la définition des règles d'accès dans un service de données WCF, reportez-vous à la page Configuration du service de données (services de données WCF) à l'adresse<http://go.microsoft.com/fwlink/?LinkID=267817>.

## **Exposition de méthodes Web à l'aide des services de données WCF**

L'objectif principal d'un service de données WCF est de fournir un accès aux données. Cependant, vous pouvez également implémenter des opérations personnalisées qui manipulent les données. Une opération de service de données WCF est simplement une méthode du service de données visible par l'infrastructure REST et accessible en envoyant une requête HTTP GET, PUT, POST ou DELETE.

Les opérations des services de données WCF auxquelles on accède à l'aide d'une demande GET doivent être annotées avec l'attribut WebGet. Ces opérations renvoient généralement des données, bien qu'elles puissent exécuter une logique métier qui ne renvoie pas de valeur. Les opérations accessibles à l'aide de requêtes PUT, POST ou DELETE doivent être annotées avec l'attribut WebInvoke. Ces opérations modifient généralement les données que le service utilise d'une manière ou d'une autre.

À l'instar des ensembles d'entités, vous devez explicitement activer l'accès aux opérations exposées par un service de données WCF. Vous pouvez le faire en appelant la méthode SetServiceOperationAccessRule de l'objet DataServiceConfiguration lorsque vous initialisez le service. Vous spécifiez le nom de l'opération et les droits d'accès appropriés.

Une opération de service de données peut prendre des paramètres et renvoie l'un des types de données suivants:

|  |  |
| --- | --- |
| • | **IEnumerable <T>**ou IQueryable <T> (où T représente un type d'entité dans le service). Si une opération renvoie une collection énumérable basée sur un type d'entité que le service reconnaît, une application cliente peut effectuer des requêtes en spécifiant des URI HTTP de la manière indiquée dans les rubriques précédentes de cette leçon. L'implémentation d'une opération qui retourne une collection énumérable de cette manière vous donne un contrôle détaillé sur le contenu de la collection. Il est de la responsabilité de votre code de générer cette collection, éventuellement en fonction des informations fournies par l'application cliente. L'exemple de code suivant montre une opération qui récupère les commerciaux par zone.  classe publique FourthCoffeeDataService: DataService <FourthCoffee>  {  public static void InitializeService (  Configuration DataServiceConfiguration)  {  ...  config.SetServiceOperationAccessRule ("SalesPersonByArea",  ServiceOperationRights.ReadMultiple);  }  ...  [WebGet]  public IQueryable <SalesPerson> SalesPersonByArea (zone de chaîne)  {  if (! String.IsNullOrEmpty (zone))  {  retour de p dans this.CurrentDataSource.SalesPerson  où String.Equals (p.Area, area)  sélectionnez p;  }  autre  {  throw new ArgumentException ("La zone doit être spécifiée", "la zone");  }  }  }  Vous pouvez appeler cette opération à l'aide de l'URI suivant:  http: // <hostName> /FourthCoffee/FourthCoffeeDataService.svc/SalesPersonByArea?area='snacks ' |
| • | **T**(où T représente un type d'entité dans le service). Une opération peut renvoyer une seule instance d'une entité. L'exemple de code suivant montre une opération qui récupère un enregistrement de vendeur qui a une adresse e-mail spécifique. Notez que vous devez également annoter une opération qui renvoie une valeur scalaire avec l'attribut SingleResult.  classe publique FourthCoffeeDataService: DataService <FourthCoffee>  {  public static void InitializeService (  Configuration DataServiceConfiguration)  {  ...  config.SetServiceOperationAccessRule ("SalesPersonByEmail",  ServiceOperationRights.ReadMultiple);  }  ...  [WebGet]  [SingleResult]  public SalesPerson SalesPersonByEmail (chaîne emailAddress)  {  return (de p dans this.CurrentDataSource.SalesPerson  où String.Equals (p.Area, area)  sélectionnez p) .SingleOrDefault ();  }  } |
| • | Une valeur primitive. L'exemple de code suivant montre une opération qui récupère un nombre de tous les commerciaux et renvoie la valeur de comptage sous la forme d'un int.  classe publique FourthCoffeeDataService: DataService <FourthCoffee>  {  public static void InitializeService (  Configuration DataServiceConfiguration)  {  ...  config.SetServiceOperationAccessRule ("SalesPersonCount",  ServiceOperationRights.ReadSingle);  }  ...  [WebGet]  [SingleResult]  public int SalesPersonCount ()  {  return (de p dans this.CurrentDataSource.SalesPerson  sélectionnez p) .Count ();  }  } |
| • | **annuler**. Toutes les opérations ne doivent pas nécessairement renvoyer une valeur. |

**Lecture supplémentaire:**Pour plus d'informations sur les opérations de service, reportez-vous à la page Opérations de service (WCF Data Services) à l'adresse<http://go.microsoft.com/fwlink/?LinkID=267818>.

## **Référencer une source de données WCF**

La bibliothèque cliente d'un service de données WCF se compose d'une classe dérivée du type DataServiceContext qui expose un ou plusieurs objets DataServiceQuery en tant que propriétés. Le nom de cette classe est généralement le même que le nom de l'objet DbContext qui est utilisé par l'EDM sur lequel le service de données WCF est basé. Par exemple, le service de données WCF FourthCoffeeDataService utilise un objet DbContext appelé FourthCoffeeEntities pour se connecter à l'EDM sous-jacent, de sorte que le nom du type DataServiceContext qui est généré pour la bibliothèque cliente est également FourthCoffeeEntities.

La classe DataServiceContext joue un rôle similaire à la classe DbContext dans Entity Framework. Une application cliente se connecte à la source de données via un objet DataServiceContext et récupère les données pour les entités que le service de données expose à l'aide des propriétés DataServiceQuery. Chaque propriété DataServiceQuery est un objet de collection générique qui présente les données de l'une des entités sous-jacentes qui fournit les données pour le service de données WCF.

La bibliothèque cliente fournit également des définitions des types que contient chaque collection DataServiceQuery. Une application cliente peut effectuer des requêtes LINQ sur les propriétés de la collection DataServiceQuery, et la bibliothèque cliente construit la requête HTTP appropriée pour récupérer les données correspondantes. Le service de données WCF récupère les données correspondantes et remplit la collection DataServiceQuery. L'application cliente peut ensuite parcourir cette collection et récupérer les données pour chaque élément.

Vous pouvez générer la bibliothèque cliente pour un service de données WCF à l'aide de la boîte de dialogue Ajouter une référence de service dans Visual Studio ou à l'aide de l'utilitaire client du service de données WCF, DataSvcUtil, à partir de la ligne de commande.

**Ajout d'une référence de service**

Vous pouvez utiliser la boîte de dialogue Ajouter une référence de service dans une application cliente. Cette boîte de dialogue vous permet de spécifier l'URL du service de données WCF auquel se connecter. La boîte de dialogue envoie une requête de métadonnées à l'URL spécifiée et utilise la réponse pour générer la classe DataServiceContext appropriée qui contient les propriétés DataServiceQuery et les classes pour chacune des entités exposées par le service de données WCF. Les métadonnées renvoyées sont stockées dans le projet client sous forme de fichier .edmx. Ce n'est pas la même chose qu'un fichier .edmx qui est généré à l'aide du Concepteur de modèle de données d'entité ADO.NET (il a un format différent), mais vous pouvez afficher ce fichier de métadonnées à l'aide de l'éditeur XML ou de n'importe quel éditeur de texte.

Pour ajouter une référence de service de données, procédez comme suit:

|  |  |
| --- | --- |
| 1. | Si le service de données ne fait pas partie de la solution et n'est pas déjà en cours d'exécution, démarrez le service de données et notez son URI. |
| 2. | Dans l'Explorateur de solutions, cliquez avec le bouton droit sur le projet client, puis sélectionnez Ajouter une référence de service. |
| 3. | Si le service de données fait partie de la solution actuelle, cliquez sur Découvrir. |
| 4. | Sinon, si le service de données ne fait pas partie de la solution actuelle, dans la zone Adresse, tapez l'URL de base du service de données, puis cliquez sur OK. |
| 5. | Cliquez sur OK. |

Une fois que vous avez référencé le service de données WCF, vous pouvez consommer les entités et les opérations de service qu'il expose.

**Lecture supplémentaire:**Pour plus d'informations sur la création d'une référence de service à l'aide de la ligne de commande, reportez-vous à la page Utilitaire client du service de données WCF (DataSvcUtil.exe) à l'adresse<http://go.microsoft.com/fwlink/?LinkID=267819>.

## **Récupération et mise à jour de données dans un service de données WCF**

Une fois que vous avez référencé le service de données WCF en générant une bibliothèque cliente, vous pouvez ensuite consommer l'EDM et toutes les opérations de service exposées par le service.

**Récupération des données**

Pour récupérer des données à partir d'un service de données WCF à l'aide de la bibliothèque cliente, procédez comme suit:

|  |  |
| --- | --- |
| 1. | Créez une instance du type dérivé de la classe DataServiceContext dans la bibliothèque cliente, puis connectez-vous au service de données WCF. Le constructeur de cette classe attend un objet Uri qui contient l'adresse du service. |
| 2. | Récupérez des données en interrogeant la collection DataServiceQuery appropriée dans l'objet DataServiceContext. Lorsque vous interrogez une collection DataServiceQuery, la bibliothèque cliente construit une requête HTTP qui spécifie la ressource et tous les critères requis. La requête est transmise au service de données WCF et les données sont renvoyées et utilisées pour remplir l'objet DataServiceQuery. |
| 3. | Parcourez les éléments de la collection DataServiceQuery et traitez les objets renvoyés. |

L'exemple de code suivant se connecte au service de données WCF FourthCoffeeDataService à l'aide du type FourthCoffeeEntities dans la bibliothèque cliente (il s'agit de la classe dérivée de DataServiceContext). Le paramètre du constructeur est l'adresse du service. Le code interroge ensuite la propriété SalesPersonsDataServiceQuery pour récupérer tous les commerciaux et lit l'adresse e-mail de chaque enregistrement.

**Interrogation du service de données WCF FourthCoffeeDataService**

Contexte FourthCoffeeEntities = nouveau FourthCoffeeEntities (nouvel Uri

("http://FourthCoffee.com/FourthCoffeeDataService.svc"));

foreach (personne SalesPerson dans le contexte.SalesPersons)

{

var email = product.EmailAddress;

}

Vous pouvez également modifier les données en appelant les opérations de service personnalisées que vous avez peut-être exposées dans le service de données WCF. Vous pouvez appeler des opérations de service à l'aide de la méthode Execute de la classe DataServiceContext. La valeur renvoyée par la méthode Execute est une collection énumérable. Si l'opération de service renvoie une seule valeur scalaire, vous devez extraire cette valeur de la collection à l'aide d'une méthode telle que First.

L'exemple de code suivant montre comment appeler l'opération de service SalesPersonByArea et itérer les résultats.

**Interrogation du service de données WCF FourthCoffeeDataService à l'aide d'une opération de service**

Contexte FourthCoffeeEntities = nouveau FourthCoffeeEntities (nouvel Uri

("http://FourthCoffee.com/FourthCoffeeDataService.svc"));

foreach (personne SalesPerson dans le contexte.Execute <SalesPerson>

(new Uri ("/ SalesPersonByArea? area = 'snacks'", UriKind.Relative)))

{

var email = product.EmailAddress;

}

**Modifier les données**

Une fois que vous avez récupéré les données, vous pouvez modifier les entités comme vous le feriez lorsque vous travaillez avec un EDM, puis enregistrez vos modifications dans le service de données WCF.

La classe DataServiceContext fournit les méthodes suivantes qui vous permettent de travailler avec les entités de votre EDM.

|  |  |
| --- | --- |
| • | **Ajouter àXXXX**. Ces méthodes vous permettent d'ajouter une nouvelle entité à la collection d'entités. L'exemple de code suivant montre comment utiliser la méthode AddToSalesPersons pour ajouter une nouvelle entité SalesPerson.  Contexte FourthCoffeeEntities = nouveau FourthCoffeeEntities (nouvel Uri  ("http://FourthCoffee.com/FourthCoffeeDataService.svc"));  ...  var newSalesPerson = new SalesPerson  {  Zone = "thé",  EmailAddress = " roy@fourthcoffee.com ",  Prénom = "Roy",  LastName = "Antebi"  };  context.AddToSalesPersons (newSalesPerson);  context.SaveChanges (); |
| • | **DeleteObject**. Cette méthode vous permet de supprimer un objet existant de la collection d'entités. L'exemple de code suivant montre comment utiliser la méthode DeleteObject pour supprimer un commercial avec l'adresse e-mail roya@fourthcoffee.com.  Contexte FourthCoffeeEntities = nouveau FourthCoffeeEntities (nouvel Uri  ("http://FourthCoffee.com/FourthCoffeeDataService.svc"));  ...  var salesPerson = (de p dans le contexte.  où p.EmailAddress.Equals (" roy@fourthcoffee.com ")  sélectionnez p) .Single ();  context.DeleteObject (salesPerson);  context.SaveChanges (); |
| • | **UpdateObject**. Cette méthode vous permet de mettre à jour un objet existant dans la collection d'entités. L'exemple de code suivant montre comment utiliser la méthode UpdateObject pour modifier la zone à laquelle appartient un commercial. |

Contexte FourthCoffeeEntities = nouveau FourthCoffeeEntities (nouvel Uri

("http://FourthCoffee.com/FourthCoffeeDataService.svc"));

...

var salesPerson = (de p dans le contexte.

où p.EmailAddress.Equals (" roy@fourthcoffee.com ")

sélectionnez p) .Single ();

salesperson.Area = "boissons gazeuses";

context.UpdateObject (salesPerson);

context.SaveChanges ();

**Implémentation du chargement hâtif des entités**

Lors de la récupération de données à l'aide de WCF Data Services, par défaut, seule l'entité que vous avez demandée est renvoyée dans la réponse. Par exemple, l'entité SalesPerson dans l'objet FourthCoffeeEntities est liée à l'entité Sales. Lorsque vous demandez une entité SalesPerson, la réponse n'inclura pas l'entité Sales associée. Cependant, vous pouvez utiliser les méthodes Expand ou LoadProperty pour obtenir des entités associées.

L'utilisation de la stratégie de chargement hâtif implémentée par la méthode Expand entraîne l'extraction des données des entités associées spécifiées dans le cadre de la même demande qui récupère les données principales de la requête. Cette approche est utile si vous savez que vous aurez toujours besoin de ces données associées, mais cela peut être un gaspillage de bande passante dans les cas où vous n'utilisez pas réellement ces entités.

L'exemple de code suivant montre comment utiliser la méthode Expand pour récupérer les ventes associées à chaque objet SalesPerson.

**Chargement hâtif des entités**

Contexte FourthCoffeeEntities = nouveau FourthCoffeeEntities (nouvel Uri

("http://FourthCoffee.com/FourthCoffeeDataService.svc"));

var salesPersons = (de s dans context.SalesPersons.Expand ("Sales")

sélectionnez s) .ToList ();

Comme alternative, vous pouvez utiliser le chargement explicite. Cette stratégie envoie une requête supplémentaire au service de données WCF qui demande les données associées pour un objet spécifique, mais elle présente l'avantage de ne pas gaspiller de bande passante en récupérant automatiquement les données qui ne sont pas utilisées.

Vous pouvez implémenter le chargement explicite à l'aide de la méthode LoadProperty de l'objet DataServiceContext. Vous appelez la méthode LoadProperty chaque fois que vous avez besoin de données liées à une entité particulière; vous spécifiez l'entité et le nom de la propriété de collection DataServiceQuery qui contient les données associées.

L'exemple de code suivant montre comment utiliser la méthode LoadProperty pour récupérer les ventes associées à chaque objet SalesPerson.

**Chargement explicite des entités**

Contexte FourthCoffeeEntities = nouveau FourthCoffeeEntities (nouvel Uri

("http://FourthCoffee.com/FourthCoffeeDataService.svc"));

foreach (var salesPerson dans le contexte.SalesPersons)

{

...

context.LoadProperty (salesPerson, "Sales");

foreach (var vente dans salesPerson.Sales)

{

...

}

}

**Lecture supplémentaire:**Pour plus d'informations sur la modification des entités, reportez-vous à la page Comment: ajouter, modifier et supprimer des entités (services de données WCF) à l'adresse<http://go.microsoft.com/fwlink/?LinkID=267820>.

## **Démonstration: récupération et modification des données de note à distance**

Dans cette démonstration, vous découvrirez les tâches que vous effectuerez dans le laboratoire pour ce module.

### **Démonstration étapes**

Vous trouverez les étapes dans la section Démonstration: Récupération et modification à distance des données de note sur la page suivante:<https://github.com/MicrosoftLearning/20483-Programming-in-C-Sharp/blob/master/Instructions/20483C_MOD08_DEMO.md>.

# **Atelier: Récupération et modification des données de note à distance**

### **Scénario**

Actuellement, l'application récupère les données d'une base de données locale. Cependant, vous avez décidé de stocker les données dans le cloud et devez configurer l'application afin qu'elle puisse récupérer des données sur le Web.

Vous devez créer un service de données WCF pour la base de données SchoolGrades qui sera intégré à l'application pour permettre l'accès aux données.

Enfin, il vous a été demandé d'écrire du code qui affiche les images des élèves en les récupérant sur le Web.

### **Objectifs**

Après avoir terminé cet atelier, vous serez en mesure de:

|  |  |
| --- | --- |
| • | Créez un service de données WCF. |
| • | Utilisez un service connecté OData. |
| • | Récupérez des données sur le Web. |

##### **Configuration du laboratoire**

Durée estimée: 60 minutes

Assurez-vous d'avoir suivi toutes les étapes répertoriées dans le Guide de configuration de ce cours. Le Guide d'installation contient des instructions d'installation spécifiques au laboratoire de ce module. Le guide d'installation se trouve dans le dossier Instructions (https://github.com/MicrosoftLearning/20483-Programming-in-C-Sharp/tree/master/Instructions)

Les ressources d'installation spécifiques se trouvent dans le répertoire Allfiles:<https://github.com/MicrosoftLearning/20483-Programming-in-C-Sharp/tree/master/Allfiles/Assets>.

Vous trouverez les étapes de haut niveau sur la page suivante:<https://github.com/MicrosoftLearning/20483-Programming-in-C-Sharp/blob/master/Instructions/20483C_MOD08_LAB_MANUAL.md>.

Vous trouverez les étapes détaillées sur la page suivante:<https://github.com/MicrosoftLearning/20483-Programming-in-C-Sharp/blob/master/Instructions/20483C_MOD08_LAK.md>.

### **Exercice 1: Création d'un service de données WCF pour la base de données SchoolGrades**

##### **Scénario**

Dans cet exercice, vous allez créer un service de données WCF pour la base de données SchoolGrades afin que l'application cliente puisse se connecter à la base de données sur le Web.

Tout d'abord, vous allez ajouter un nouveau projet d'application Web ASP.NET à la solution et le configurer pour l'application cliente. Vous exposerez ensuite les entités dans l'EDM à partir d'un service de données dans le nouveau projet. Ensuite, vous allez spécifier le contexte de données pour le service de données et configurer les droits d'accès aux entités qu'il expose. Enfin, vous ajouterez une opération au service de données qui renvoie tous les étudiants d'une classe spécifiée.

**Résultat**: Après avoir terminé cet exercice, vous devez avoir ajouté un service de données WCF à l'application pour fournir un accès à distance à la base de données SchoolGrades.

### **Exercice 2: intégration du service de données dans l'application**

##### **Scénario**

Dans cet exercice, vous allez intégrer le service de données WCF dans l'application Grades Prototype.

Tout d'abord, vous allez ajouter une référence de service dans le projet GradesPrototype qui fait référence au service de données WCF en cours d'exécution. Vous modifierez ensuite le code qui accède à l'EDM local pour utiliser le service de données WCF à la place. Ensuite, vous modifierez le code qui enregistre les modifications dans la base de données pour le faire via le service de données. Enfin, vous testerez l'application pour vérifier qu'elle s'exécute de la même manière que si les données étaient appelées localement.

**Résultat**: Après avoir terminé cet exercice, vous devez avoir mis à jour l'application Grades Prototype pour utiliser le service de données WCF.

### **Exercice 3: Récupérer les photos des élèves sur le Web (si le temps le permet)**

##### **Scénario**

Dans cet exercice, vous allez écrire du code qui affiche les images des élèves en récupérant l'image sur le Web. Vous allez modifier la fenêtre StudentsPage (qui affiche la liste des étudiants dans une classe), la fenêtre StudentProfile (qui affiche les détails pour un étudiant individuel) et la fenêtre AssignStudentDialog (qui affiche une liste des étudiants non affectés) pour inclure les photographies des étudiants. . Les données de chaque élève contiennent une propriété ImageName qui spécifie le nom de fichier de la photo pour l'élève sur le serveur Web. Ces fichiers se trouvent dans le dossier Images \ Portraits sur le même serveur Web qui héberge le service de données (dans le projet Web.Grades. ) Vous allez créer une classe de convertisseur de valeur qui génère l'URL d'une image à partir de la propriété ImageName, puis utiliser un contrôle Image pour utiliser l'URL pour récupérer et restituer l'image dans chacune des fenêtres spécifiées. Enfin, vous exécuterez l'application pour vérifier que les images apparaissent.

**Résultat**: Une fois cet exercice terminé, la liste des étudiants, le profil des étudiants et la boîte de dialogue des étudiants non affectés afficheront les images des étudiants qui ont été récupérées sur le Web.

# **Revue du module et points à retenir**

Dans ce module, vous avez appris à utiliser les classes de demande et de réponse dans l'espace de noms System.Net pour manipuler directement des sources de données distantes et à utiliser WCF Data Services pour exposer et consommer un modèle de données d'entité sur le Web.

### **Question (s) de révision**