How To... Call OData Services to Get Data Using the OData SDK (Windows)

Applicable Releases:

SAP Mobile Platform 3.0

Version 1.0

June 2014

|  |  |
| --- | --- |
| © Copyright 2014 SAP AG. All rights reserved.   1. No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice. 2. Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors. 3. Microsoft, Windows, Excel, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation. 4. IBM, DB2, DB2 Universal Database, System i, System i5, System p, System p5, System x, System z, System z10, System z9, z10, z9, iSeries, pSeries, xSeries, zSeries, eServer, z/VM, z/OS, i5/OS, S/390, OS/390, OS/400, AS/400, S/390 Parallel Enterprise Server, PowerVM, Power Architecture, POWER6+, POWER6, POWER5+, POWER5, POWER, OpenPower, PowerPC, BatchPipes, BladeCenter, System Storage, GPFS, HACMP, RETAIN, DB2 Connect, RACF, Redbooks, OS/2, Parallel Sysplex, MVS/ESA, AIX, Intelligent Miner, WebSphere, Netfinity, Tivoli and Informix are trademarks or registered trademarks of IBM Corporation. 5. Linux is the registered trademark of Linus Torvalds in the U.S. and other countries. 6. Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries. 7. Oracle is a registered trademark of Oracle Corporation. 8. UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group. 9. Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc. 10. HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology. 11. Java is a registered trademark of Sun Microsystems, Inc. 12. JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape. 13. SAP, R/3, SAP NetWeaver, Duet, PartnerEdge, ByDesign, SAP BusinessObjects Explorer, StreamWork, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries.   Business Objects and the Business Objects logo, BusinessObjects, Crystal Reports, Crystal Decisions, Web Intelligence, Xcelsius, and other Business Objects products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Business Objects Software Ltd. Business Objects is an SAP company.   1. Sybase and Adaptive Server, iAnywhere, Sybase 365, SQL Anywhere, and other Sybase products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Sybase, Inc. Sybase is an SAP company. | 1. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary. 2. The information in this document is proprietary to SAP. No part of this document may be reproduced, copied, or transmitted in any form or for any purpose without the express prior written permission of SAP AG. 3. This document is a preliminary version and not subject to your license agreement or any other agreement with SAP. This document contains only intended strategies, developments, and functionalities of the SAP® product and is not intended to be binding upon SAP to any particular course of business, product strategy, and/or development. Please note that this document is subject to change and may be changed by SAP at any time without notice. 4. SAP assumes no responsibility for errors or omissions in this document. SAP does not warrant the accuracy or completeness of the information, text, graphics, links, or other items contained within this material. This document is provided without a warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, or non-infringement. 5. SAP shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials. This limitation shall not apply in cases of intent or gross negligence. 6. The statutory liability for personal injury and defective products is not affected. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third-party Web pages nor provide any warranty whatsoever relating to third-party Web pages.   SAP “How-to” Guides are intended to simplify the product implement­tation. While specific product features and procedures typically are explained in a practical business context, it is not implied that those features and procedures are the only approach in solving a specific business problem using SAP NetWeaver. Should you wish to receive additional information, clarification or support, please refer to SAP Consulting.  Any software coding and/or code lines / strings (“Code”) included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, except if such damages were caused by SAP intentionally or grossly negligent.  Disclaimer  Some components of this product are based on Java™. Any code change in these components may cause unpredictable and severe malfunctions and is therefore expressively prohibited, as is any decompilation of these components.  Any Java™ Source Code delivered with this product is only to be used by SAP’s Support Services and may not be modified or altered in any way. |

Document History

|  |  |
| --- | --- |
| Document Version | Description |
| 1.00 | First official release of this guide |

Typographic Conventions

|  |  |
| --- | --- |
| Type Style | Description |
| Example Text | Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options.  Cross-references to other documentation |
| **Example text** | Emphasized words or phrases in body text, graphic titles, and table titles |
| Example text | File and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools. |
| Example text | User entry texts. These are words or characters that you enter in the system exactly as they appear in the documentation. |
| <Example text> | Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system. |
| EXAMPLE TEXT | Keys on the keyboard, for example, F2 or ENTER. |

Icons

|  |  |
| --- | --- |
| Icon | Description |
|  | Caution |
|  | Note or Important |
|  | Example |
|  | Recommendation or Tip |

Table of Contents

[1. Business Scenario 1](#_Toc393384191)

[2. Background Information 1](#_Toc393384192)

[3. Prerequisites 1](#_Toc393384193)

[4. Step-by-Step Procedure 2](#_Toc393384194)

[4.1 Windows Project 2](#_Toc393384195)

[4.2 Retrieving data from SMP Server 3](#_Toc393384196)

[4.2.1 Steps involved in submitting GET request 3](#_Toc393384197)

[4.2.2 GET request Process Flow 5](#_Toc393384198)

[4.2.3 Screen flow of the application 6](#_Toc393384199)

[4.2.4 GET CarrierCollection 7](#_Toc393384200)

[4.3 Running the application 9](#_Toc393384201)

[5. Appendix 11](#_Toc393384202)

[5.1 NuGet Package Manager 11](#_Toc393384203)

[5.1.1 Adding Windows SMP SDK package in Visual Studio 11](#_Toc393384204)

[5.1.2 Adding Windows SMP SDK references to the project 12](#_Toc393384205)

# Business Scenario

Travel Agency X would like to build an online mobile application for its customers, so they can book their flights anywhere, anytime from their devices. The **SAP Mobile Platform** provides a means for them to securely and efficiently access backend flight data exposed by their SAP NetWeaver Gateway system via OData-based REST services.

The application should retrieve data from outside the company network securely using open standards. Luckily the mobile platform client OData SDK provides easy-to-use libraries that can help app developers perform this task.

# Background Information

The goal of this exercise is not to show how to create a project from scratch and dissect every line of code. Instead, it shows the key pieces of code and information, along with a starter project template, so that developers understand how to leverage the OData SDK to on-board users in their own apps.

# Prerequisites

This exercise has the following prerequisites:

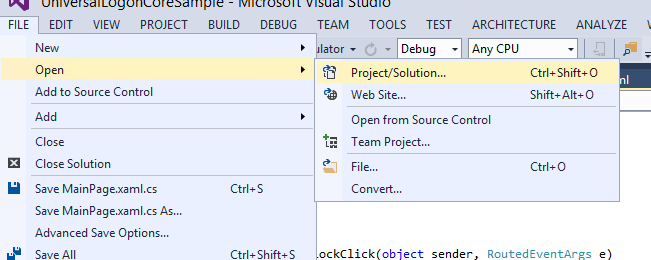
* Windows 8.1 operating system (for Windows Store applications)
* Visual Studio 2013 with Update 2 (for Windows Store and Desktop applications)
* .NET 4.5 (for Windows Desktop applications)
* To get the most out of this exercise, experience with Windows programming is recommended.

# Step-by-Step Procedure

In the previous exercise, we looked into the steps required to on-board a device. Now that the device is registered with the SMP Server, we will focus on retrieving data from the OData Service using SMP Server as a proxy. The following sections provide a detailed step-by-step procedure on how to retrieve data using the SAP Mobile Platform. Each exercise builds upon the previous exercise, so it is recommended that you complete each exercise before moving to the next.

## Windows Project

1. ...
   1. Open Visual Studio 2013 with Update 2 and open the solution RKT\_GetFlights.sln



* 1. Windows SMP SDK uses Microsoft OData Parser libraries to parse the OData responses. So in addition to adding Windows SMP SDK libraries as references, you should also add Microsoft OData Parser libraries as references.
  2. Windows SMP SDK libraries are packaged as NuGet packages. See appendix on how to add the libraries as reference.

## Retrieving data from SMP Server

### Steps involved in submitting GET request

The ODataStore library is used to interact with the SMP Server to submit HTTP requests to either GET data or perform CUD operations. Most applications defined in the SMP Server have a single endpoint url. A single ODataStore instance is created to make the HTTP requests. In case, the application has additional endpoint urls defined, then a separate ODataStore instance is created for each additional endpoint url. The ODataStore hides a lot of complexities and makes interacting with OData source fairly easy for the developer.

To create an instance of the ODataStore, the developer can use either one of the 2 constructors. The default value for the 2nd EntityFormat parameter is XML format. However, using JSON format considerably reduces the network traffic.

|  |
| --- |
| public ODataStore(string serviceUri, ODataStore.EntityFormat entityFormat = ODataStore.EntityFormat.XML);   1. public ODataStore(Uri serviceUri, ODataStore.EntityFormat entityFormat = ODataStore.EntityFormat.XML); 2. public enum EntityFormat 3. { 4. JSON = 0, 5. XML = 1, 6. } 7. var store = new ODataStore(uri); |

The uri is the application endpoint that is returned by the SMP Server as part of a successful registration. This value is stored in the data vault. Once an instance of ODataStore is created, the method OpenAsync is called. This method retrieves the service document and the metadata document. When making the OpenAsync call, it is also necessary to pass in the user credentials and the application connection id as header values. Creating an ODataStore instance and calling OpenAsync is only done once for a session.

|  |
| --- |
| 1. var client = new SAP.Net.Http.HttpClient( 2. new System.Net.Http.HttpClientHandler { Credentials = new NetworkCredential(“user", “password") }, true); 3. client.DefaultRequestHeaders.TryAddWithoutValidation("X-SMP-APPCID",appconnid); 4. await store.OpenAsync(client); |

The ScheduleReadEntitySet method is used to schedule an HTTP GET request. This method takes the collection name as a parameter. The Response object is then called asynchronously to submit the request.

|  |
| --- |
| var execution = store.ScheduleReadEntitySet(collectionName);  var response = await execution.Response; |

The response object is then cast as an ODataEntitySet and can be immediately bound to an UI control. The ODataEntitySet is an IObservableCollection which allows the UI controls to automatically update themselves when the collection is changed. There is no need to convert the ODataEntitySet into an intermediate class by iterating through all the properties to bind to the UI control, thereby eliminating the need to store the OData feed list in memory twice.

|  |
| --- |
| 1. var response = await execution.Response; 2. this.EntitySet = (SAP.Data.OData.Online.ODataEntitySet)((IODataResponseSingle)response).Payload; 3. } 4. **// Bind this.EntitySet directly to UI controls** |

### GET request Process Flow

Create an online ODataStore

1. At design time or runtime – Set the DataContext of the xaml page to the variable X
2. Bind properties to UI controls

Cast the Payload property to ODataEntitySet and store in variable X

await on the Response object

Call ScheduleReadEntitySet passing collection name as parameter

Call OpenAsync to initialize the ODataStore (retrieves service and metadata document)

### Screen flow of the application

This sample application submits 2 GET requests to the SMP Server. The first GET request is to retrieve the CarrierCollection. The second GET request is to retrieve all the flights for a selected Carrier. The application has 2 pages to display the CarrierCollection and the FlightCollection.

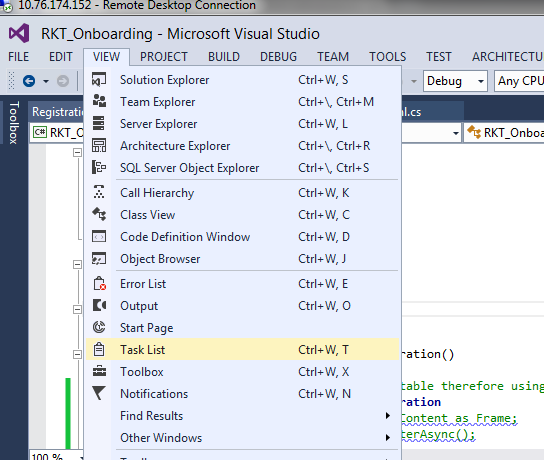
FlightCollection Page (flights.xaml) for a selected Carrier

CarrierCollection Page

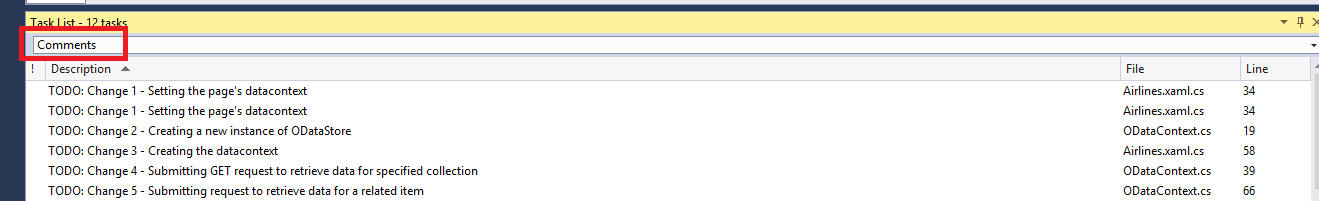
(Airlines.xaml)

### GET CarrierCollection

1. ...
   1. In Visual Studio 2013 + Update 2, click on View -> Task List



* 1. Sort the tasks alphabetically, to view all the tasks associated for this exercise. They should all begin with **// TODO: Change x.** There are 5 tasks for this exercise.



* 1. Open Airlines.xaml.cs in the Shared project.
  2. Go to the constructor of the Airlines class. In between the BEGIN and END //TODO Change1: markers enter the following code:

|  |
| --- |
| if (SharedContext.Context == null)  {  this.CreateContext();  }  else  {  this.DataContext = SharedContext.Context;  SharedContext.Context.IsStoreCreated = true;  } |

SharedContext.Context is set as the DataContext for the page programmatically in the constructor. The page is the highest level container, so all of the UI elements on the page will inherit SharedContext.Context as the DataContext. If SharedContext.Context is null, then we call CreateContext method to create the Context.

* 1. Open ODataContext.cs in the Shared project.
  2. Go to the constructor of the ODataContext class. In between the BEGIN and END //TODO Change2: markers enter the following code:

|  |
| --- |
| this.Store = new ODataStore(serviceUrl, ODataStore.EntityFormat.JSON); |

A new instance of the ODataStore is created in the constructor taking the application endpoint url and using the JSON format.

* 1. Open Airlines.xaml.cs in the Shared project.
  2. Go to the CreateContext method. In between the BEGIN and END //TODO Change3: markers enter the following code:

|  |
| --- |
| this.DataContext = SharedContext.Context = new ODataContext(serviceUrl);  SharedContext.Context.RingVisible = Visibility.Visible;  var client = new SAP.Net.Http.HttpClient(  new System.Net.Http.HttpClientHandler()  {  Credentials = new System.Net.NetworkCredential( Globals.LogonCore.LogonContext.RegistrationContext.BackendUserName, Globals.LogonCore.LogonContext.RegistrationContext.BackendPassword)  },  true); // will be disposed by the store!    client.DefaultRequestHeaders.TryAddWithoutValidation("X-SMP-APPCID", connectionId);  client.DefaultRequestHeaders.TryAddWithoutValidation("X-SUP-APPCID", connectionId);  client.ShouldHandleXcsrfToken = true;  await SharedContext.Context.Store.OpenAsync(client);  await SharedContext.Context.DownloadCollection("CarrierCollection"); |

The OpenAsync method is called with an instance of the SAP.Net.Http.HttpClient class. This instance of the SAP.Net.Http.HttpClient class passes in the credentials and also the application connection id as headers. Setting the ShouldHandleXcsrfToken property to true allows HTTP requests to modify backend SAP data. OpenAsync method retrieves the service document and metadata document from the SMP Server. After the call to OpenAsync method, the DownloadCollection method is called to retrieve the CarrierCollection.

* 1. Open ODataContext.cs in the Shared project.
  2. Go to the DownloadCollection method. In between the BEGIN and END //TODO Change4: markers enter the following code:

|  |
| --- |
| var execution = this.Store.ScheduleReadEntitySet(collectionName);  var response = await execution.Response;    this.EntitySet = (SAP.Data.OData.Online.ODataEntitySet)((IODataResponseSingle)response).Payload; |

The ScheduleReadEntitySet takes the collection name as a parameter. The Payload property of the response variable is then cast to an ODataEntitySet and saved in the EntitySet property of the Shared.Context class. The GridView in the Airlines.xaml page uses this.EntitySet as the ItemsSource and therefore the UI is bound directly. To verify that the GridView is being bound to the EntitySet property, open Airlines.xaml file. In the GridView element, you should see the following attribute…

|  |
| --- |
| <GridView  x:Name="ItemGridView"  Visibility="Visible"  ItemsSource="{Binding EntitySet}" |

* 1. Go to the DownloadRelatedItems method. In between the BEGIN and END //TODO Change5: markers enter the following code:

|  |
| --- |
| IODataNavigationProperty navigationProperty = this.entity.GetNavigationProperty(navigationPropertyName);  var execution = this.Store.ScheduleRequest(new ODataRequestParametersSingle(navigationProperty.AssociationResourcePath));  var response = await execution.Response;    if (response is IODataResponseSingle)  {  if (((IODataResponseSingle)response).PayloadType == ODataType.EntitySet)  {  this.RelatedEntitySet = (SAP.Data.OData.Online.ODataEntitySet)((IODataResponseSingle)response).Payload;  }  } |

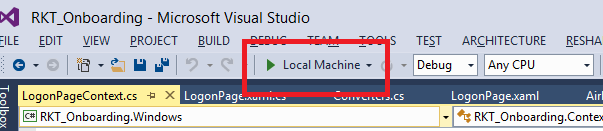
The second HTTP GET request to fetch the FlightCollection is done using the ScheduleRequest method. While we could have easily used the ScheduleReadEntitySet as in the previous GET request, we are using the ScheduleRequest method to show the usage of this method. The ScheduleRequest is the main method that all other methods call behind the covers to submit HTTP requests to the SMP Server. At a low level the developer can also call ScheduleRequest directly. It is recommended to call the specialized methods like ScheduleReadEntitySet to perform actions to read a collection.

The GridView in the Flights.xaml page uses this.RelatedEntitySet as the ItemsSource and therefore the UI is bound directly. To verify that the GridView is being bound to the RelatedEntitySet property, open Flights.xaml file. In the GridView element, you should see the following attribute…

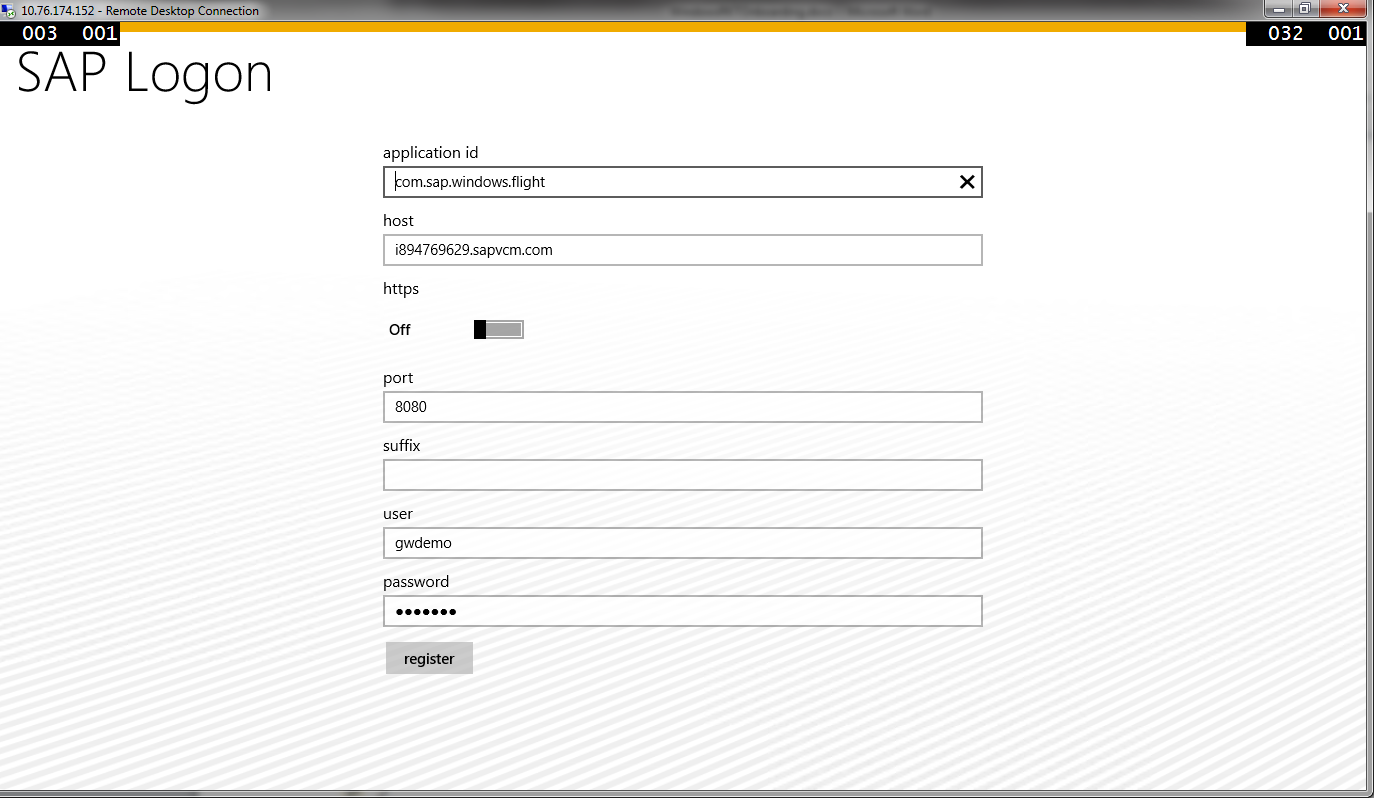
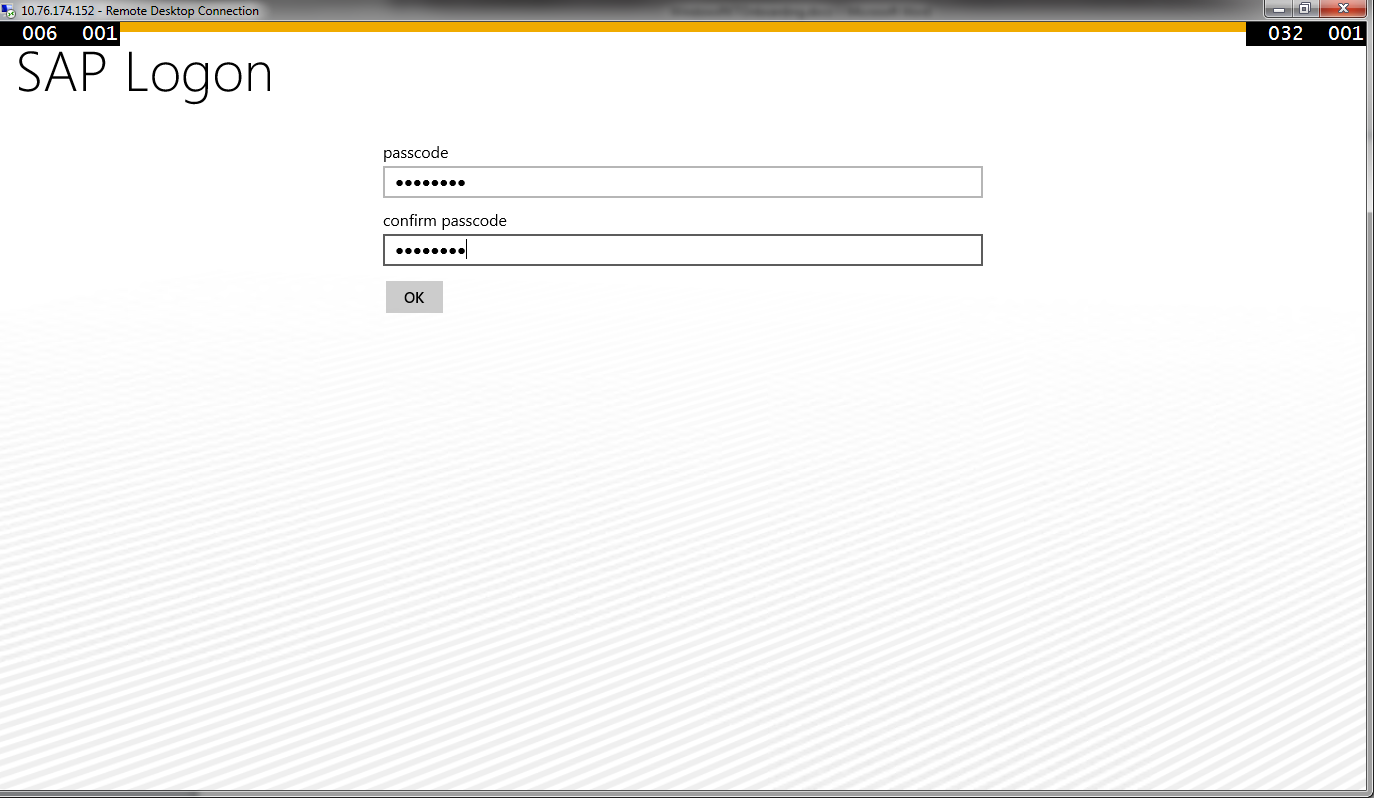
|  |
| --- |
| <GridView  x:Name="ItemGridView"  Visibility="Visible"  ItemsSource="{Binding RelatedEntitySet}" |

## Running the application

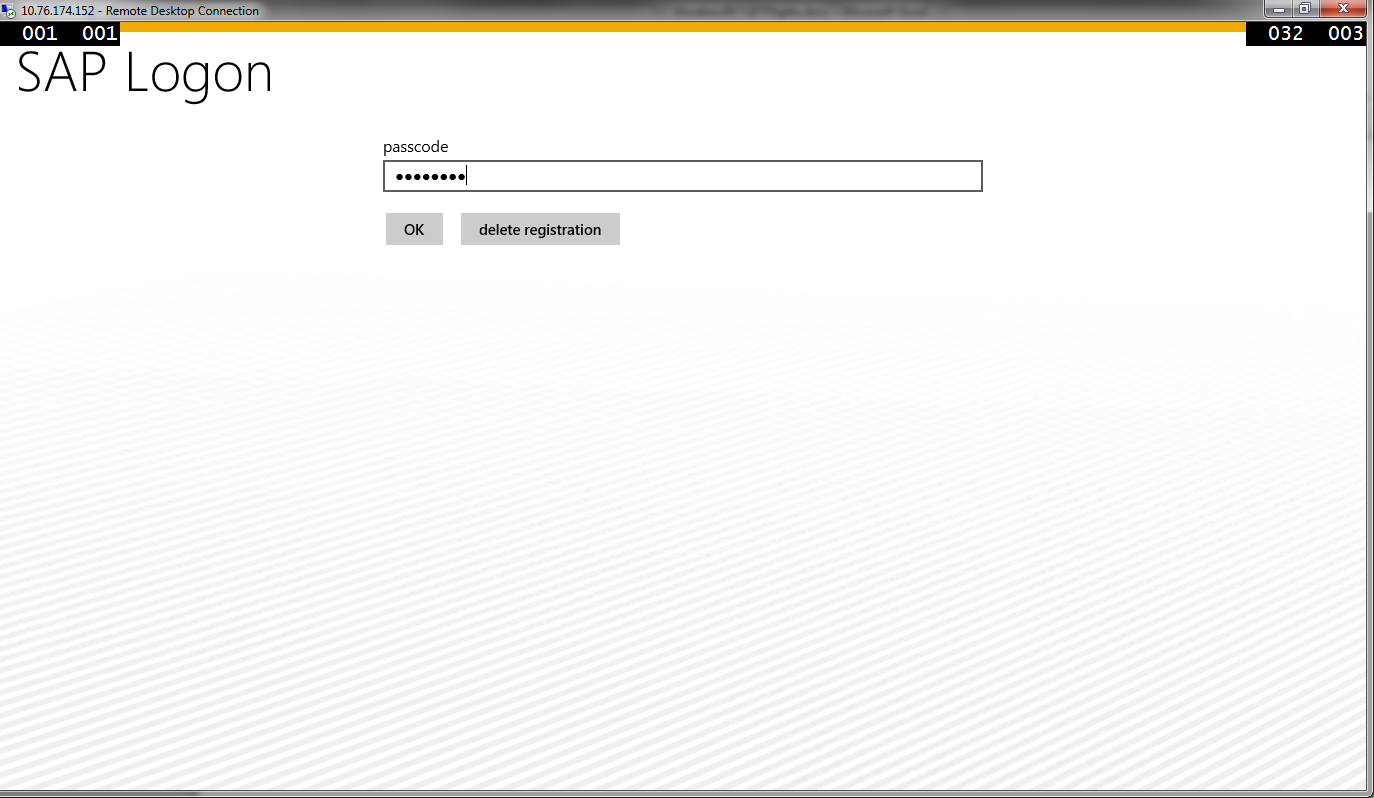
1. Run the application by clicking Run from within Visual Studio 2013 Update 2.



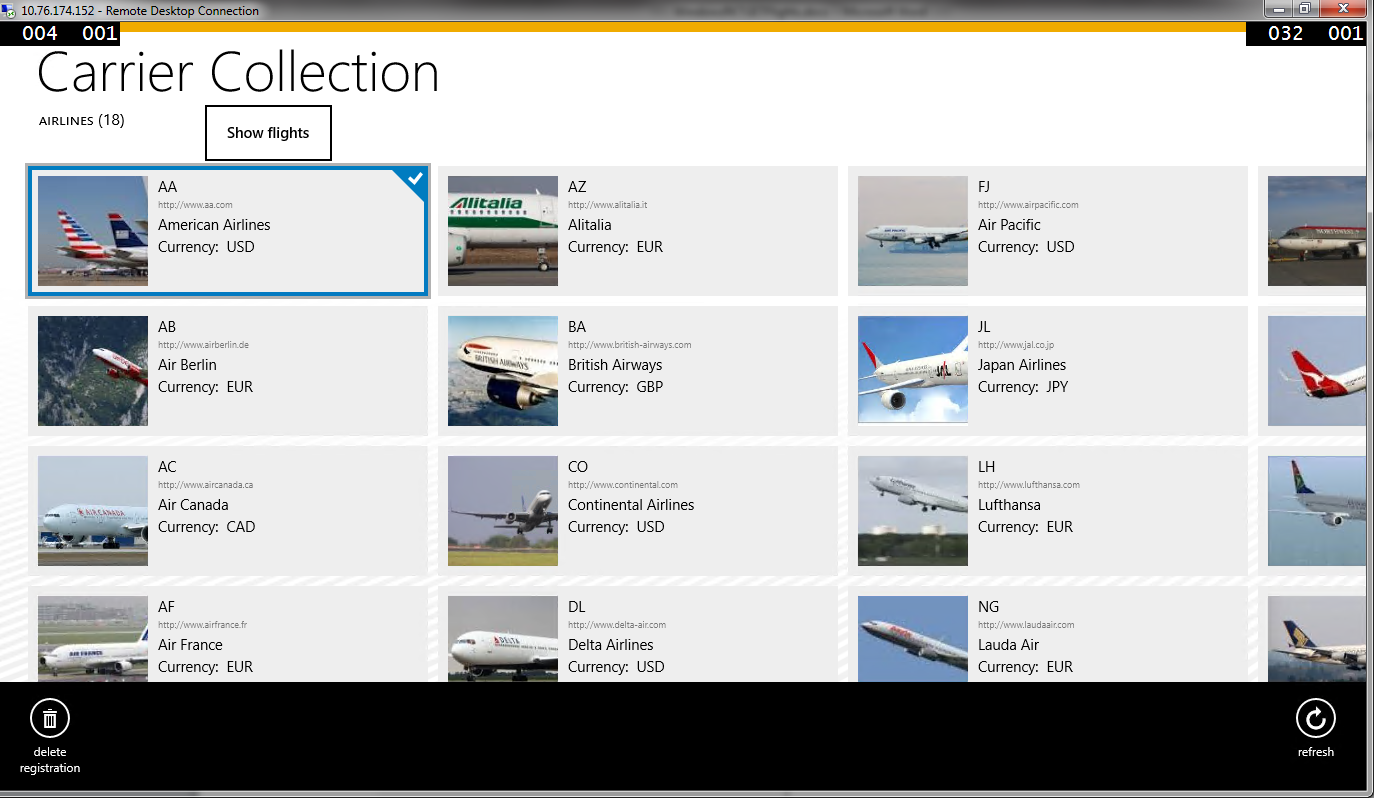
1. If you are not registered, click register. Enter passcode for data vault and click Ok.

If you are already registered, enter the passcode to unlock the data vault.



1. The application should now open up. Right click on any Carrier and click on Show flights.



1. You should now be taken to the Flights page.



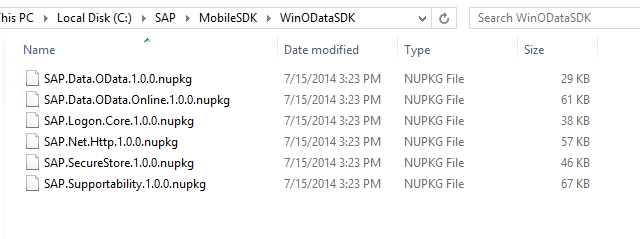
# Appendix

## NuGet Package Manager

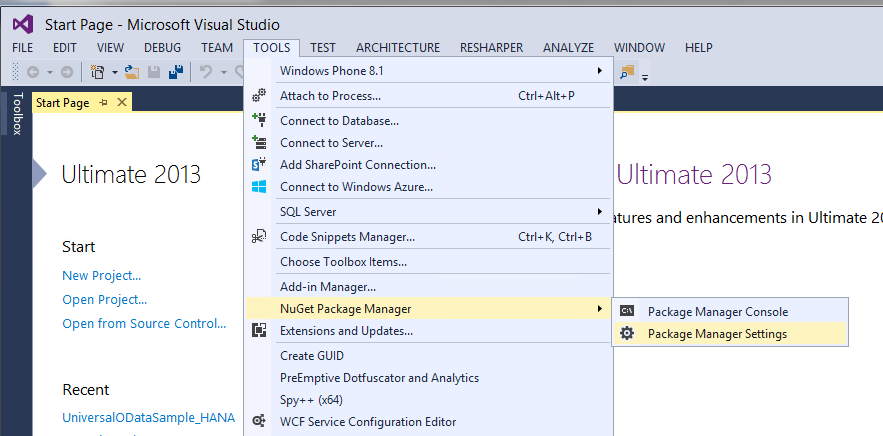
NuGet is the package manager for the Microsoft development platform including .NET. The NuGet client tools provide the ability to produce and consume packages. Starting with Visual Studio 2012, NuGet is included in every edition (except Team Foundation Server) by default. Updates to NuGet can be found through the Extension Manager.

### Adding Windows SMP SDK package in Visual Studio

1. Find the location of the Windows SMP SDK files in your local development machine (default location is C:\SAP\MobileSDK3\NativeSDK\ODataFramework\Windows). (The .nupkg file contains libraries for both Windows Store and Windows desktop)



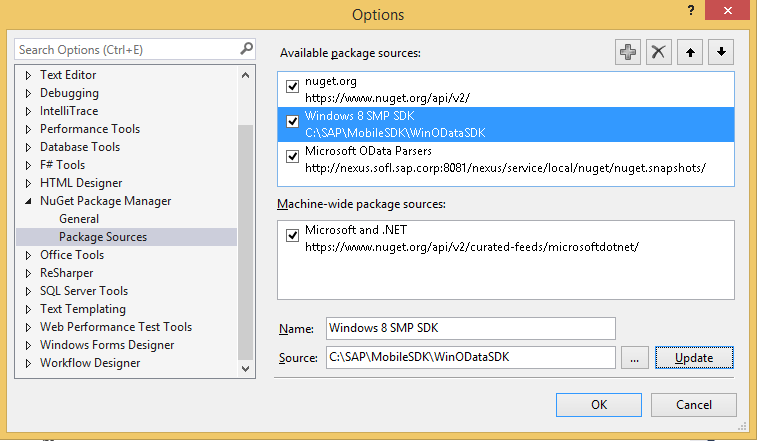
1. Open Visual Studio 2013 with Update 2. Click Tools -> NuGet Package Manager -> Package Manager Settings



1. Click on NuGet Package Manager -> Package Sources on the left pane. On the right pane, click the + sign on the right pane to add a new package source. Enter a name for the package and browse to the source of the unzipped Windows SMP SDK files. Click Update.

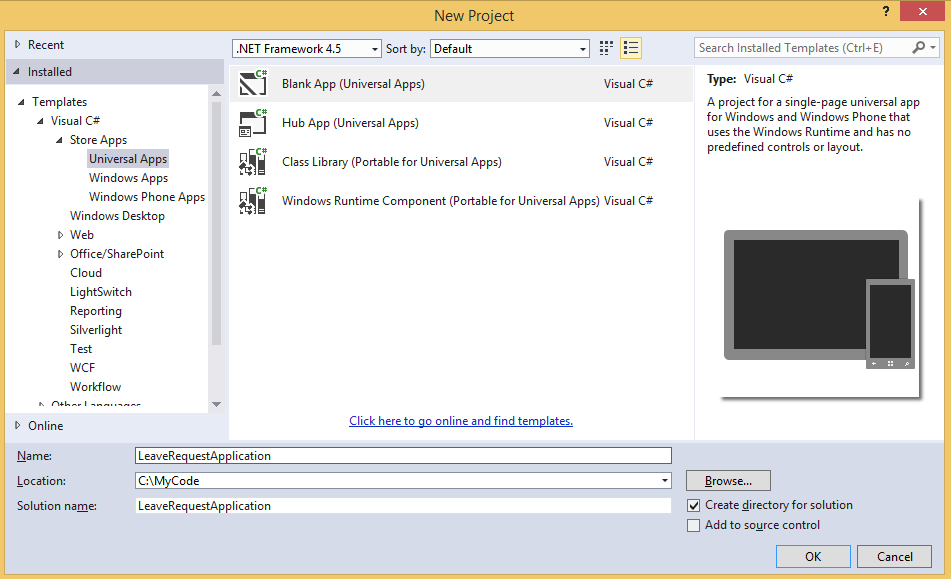
**Name:** Windows SMP SDK

**Source:** C:\SAP\MobileSDK3\NativeSDK\ODataFramework\Windows

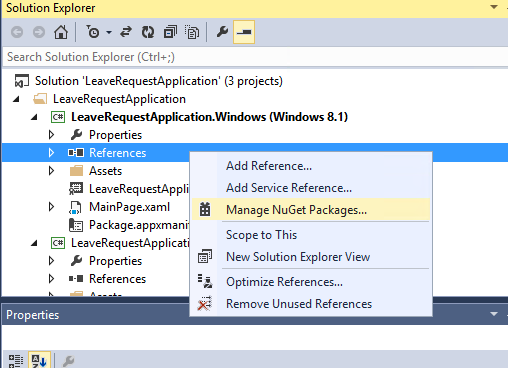


### Adding Windows SMP SDK references to the project

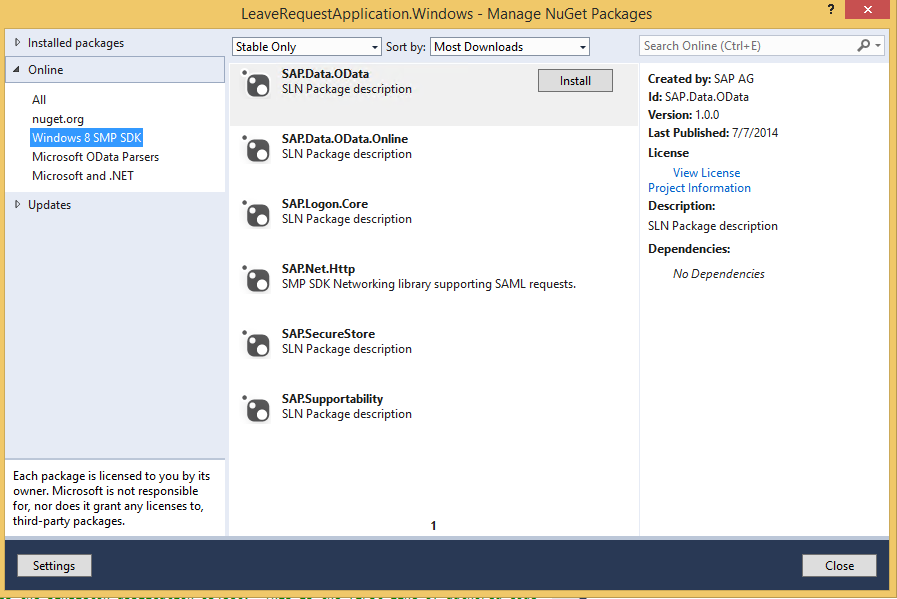
1. Create a new Windows Store Universal project by clicking on the New Project… link. Enter a name for the project.



1. To add references to the Windows project, right click on References and select Manage NuGet Packages…



1. Select the package source on the left pane that you created previously. Select the package that you need to add as a reference and click Install. NuGet Package Manager installs all dependent packages for you automatically. In addition, the proper package for the specified platform is installed.



1. In addition to the SAP NuGet packages, the developer also needs to add references to the Microsoft OData library packages. This can be done directly by adding the packages from the NuGet gallery. Click on nuget.org on the left pane and search for ODataLib. From the Search Results, install the package ODataLib for OData v1-3 (The version is 5.6.2). This will install all the dependent packages. Click I Accept to follow the prompts to install the packages.

Machine generated alternative text: .org
Each package is licensed to you by its
owner. Microsoft is not responsible
for, nor does it grant any licenses to,
third-party packages.
ODatalib for WIndows Phone
Classes to serialize, deserialize and validate OData payloads.
Enables construction of OData producers and consumers.Tar...
ODataLib
Classes to serialize, deserialize and validate OData JSON
payloads. Supports OData v4 only.
EdmLib for OData vi-3
Classes to represent construct, parse, serialize and validate
entity data models. Targets NET 4.0, Silverlight 4.0, or .NET P...
System.Spatial for OData vi-3
Contains classes and methods that facilitate geography and
geometry spatial operations. Targets .NET 4.0, Silverlight 4.0...
WCF Data Services Client for OData vi-3
LINQ-enabled client API for issuing OData queries and
consuming OData payloads. Supports OData v3. Targets .NE...
i 2 b
Created by: Microsoft Corpoi
Id: Microsoft.Data,OData
Version: 5.6.2
Last Published: 8/1/2014
Downloads: 1656106
View License
Project Information
Report Abuse
Description:
Classes to serialize, deserialize and
validate OData payloads. Enables
construction of OData producers and
consumers. Targets .NET 4.0, Silverlight
4.0 or NET Portable Lib with support
for NET 4.0, SL 5.0, Win Phone 8, Win
Phone 8.1, and Win 8. Localized for CHS,
CHT, DEU, ESN, FRA, ITA, JPN, KOR and
RUS.
Tags: wcf data services odata odatalib
edmlib spatial adonet S entity
framework open protocol wcfds
wcfdataservices dataservices
Dependencies:
System.Spatial (t 5.6.2)
Microsoft.DataEdm (= 5.62)
t...a.. :s_..... ...L..... . . LS..... _..L.
settings
L ‘.JL/ULO Parsers
Windows SMP SDK
Microsoft and .NET
ODatalib for OData vi-3
Classes to serialize,
OData JSON payloads.
Install
P Updates
Q
Q
QMachine generated alternative text: License Acceptance fl
The following package(s) require a click-to-accept license:
System.Spatial (Author Microsoft corporation)
I Vie’. LicEnzE
Mkrosoft.Data.Edm (Author Microsoft Corporation)
View License
MicrosoftData.OData (Author Microsoft Corporation)
View License
By clicking “I Accept” you agree to the license terms for the package
(s) listed above. If you do not agree to the license terms, click “I
Decline.”
I ecIine I Accept

1. Follow steps 3 and 4 to add references to the Windows 8.1 Phone project as well.