How To... Windows Generic Player Sample Application

Applicable Releases:

SAP Mobile Platform 3.0

Version 1.0

September 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. Background Information 1](#_Toc399413137)

[2. Prerequisites 1](#_Toc399413138)

[3. Step-by-Step Procedure 2](#_Toc399413139)

[3.1 Windows Project 2](#_Toc399413140)

[3.2 Retrieving data from SMP Server 3](#_Toc399413141)

[3.2.1 Steps involved in submitting GET request 3](#_Toc399413142)

[3.2.2 Windows Generic Player Sample Application – Code walkthrough 5](#_Toc399413143)

[3.3 Running the application 8](#_Toc399413144)

[4. Appendix 10](#_Toc399413145)

[4.1 NuGet Package Manager 10](#_Toc399413146)

[4.1.1 Adding Windows SMP SDK package in Visual Studio 10](#_Toc399413147)

[4.1.2 Adding Windows SMP SDK references to the project 11](#_Toc399413148)

# Background Information

The Windows Generic Player Sample Application illustrates how to dynamically consume any OData Service at runtime using the SAP Mobile Platform SDK SP05. This Windows Generic Player Sample Application is built as a Windows Store application that can run on any tablet or laptop running Windows 8.1. The entire application is built in about 200 lines of code (including UI and error handling).

An almost identical application is also built as a Windows Desktop application that can run on any Windows machine with .NET 4.5.

# Prerequisites

This exercise has the following prerequisites:

Windows Store Application

* Windows 8.1 operating system
* Visual Studio 2013 with Update 2
* To get the most out of this exercise, experience with Windows programming is recommended.

Windows Desktop Application

* Windows running .NET 4.5
* Visual Studio 2013 with Update 2
* To get the most out of this exercise, experience with Windows programming is recommended.

# Step-by-Step Procedure

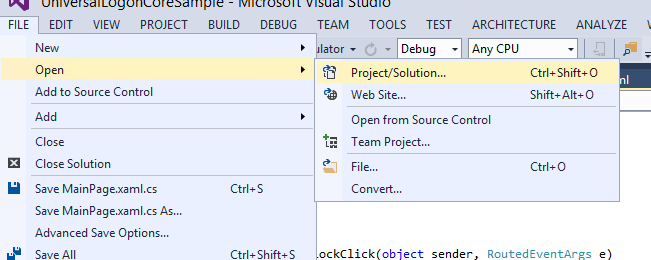
The Windows Generic Player Sample Application illustrates how to dynamically consume any OData Service at runtime using the SAP Mobile Platform SDK SP05 (hereinafter referred to as “SMP SDK” or “SDK”). This sample application is not integrated with the SAP Mobile Platform Runtime SP04 (hereinafter referred to as “SMP Server”) and therefore can be run as a stand-alone application.

This sample application receives an OData Service URL from the end user as input. Optionally, the end user can also provide the sample application with a username and password to connect to the OData Service URL. The sample application then connects to the OData Service URL and displays all the entity sets that are part of the OData Service. The end user can then select a particular entity set to view all the entities. Upon selecting a specific entity, the property and values belonging to that entity are displayed to the end user. The entire application is built in about 200 lines of code (including UI and error handling).

The application does not mimic any real world scenario, but merely illustrates the simplicity of consuming OData Services using the SMP SDK.

## Windows Project

1. ...
   1. Open Visual Studio 2013 with Update 2 and open the solution GPStoreApp.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 submit HTTP requests to either GET data or perform CUD operations. A single ODataStore instance is created for each OData source. A separate ODataStore instance is created for each additional OData source. 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. Example: 8. var store = new ODataStore(uri); |

The uri is the service document URL of the OData source. 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 (if required). 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. 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.

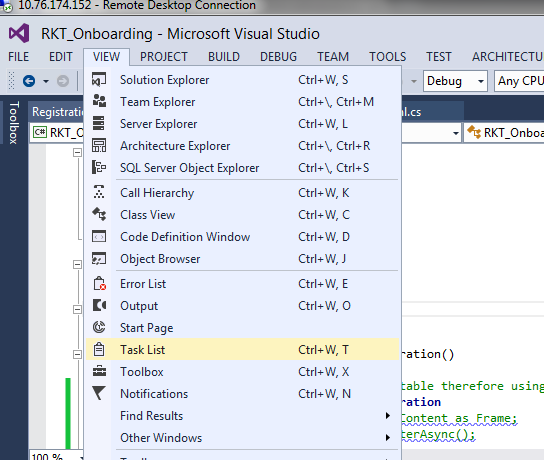
|  |
| --- |
| var execution = store.ScheduleReadEntitySet(collectionName);  var response = await execution.Response; |

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.

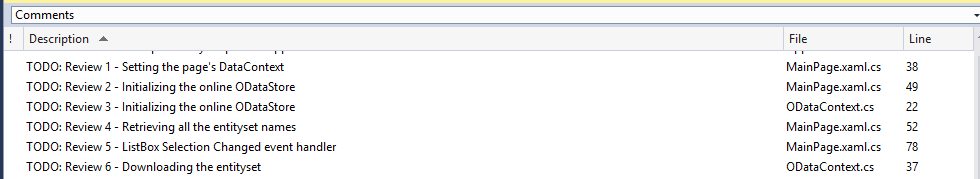
|  |
| --- |
| 1. var response = await execution.Response; 2. this.EntitySet = (SAP.Data.OData.Online.ODataEntitySet)((IODataResponseSingle)response).Payload; 3. } 4. **// Bind this.EntitySet directly to UI controls** |

### Windows Generic Player Sample Application – Code walkthrough

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 : Review x.** There are 6 tasks for this exercise.



* 1. Open MainPage.xaml.cs.
  2. Go to the constructor of the MainPage class. In between the BEGIN and END //TODO : Review 1 - markers, review the following code:

|  |
| --- |
| DataContext = SharedContext.Context = new ODataContext(); |

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.

* 1. Go to the InitializeContext method of the MainPage class. In between the BEGIN and END //TODO : Review 2 - markers, review the following code:

|  |
| --- |
| SharedContext.Context.InitializeStore(); |

The InitializeStore method of SharedContext.Context is called. The ODataStore is initialized within this method.

* 1. Open ODataContext.cs.
  2. Go to the InitializeStore method of the ODataContext class. In between the BEGIN and END //TODO : Review 3 - markers, review the following code:

|  |
| --- |
| this.Store = new SAP.Data.OData.Online.Store.ODataStore(this.ServiceUrl); |

A new instance of the ODataStore is created by passing the OData service document URL as the parameter. The ODataStore library is used to submit HTTP requests to either GET data or perform CUD operations.

* 1. Open MainPage.xaml.cs.
  2. Go to the InitializeContext method. In between the BEGIN and END //TODO : Review 4 – markers, review the following code:

|  |
| --- |
| var client = new SAP.Net.Http.HttpClient(  new System.Net.Http.HttpClientHandler  { Credentials = new System.Net.NetworkCredential(string.IsNullOrEmpty(TB\_User.Text) ? null : TB\_User.Text, string.IsNullOrEmpty(TB\_Password.Password) ? null : TB\_Password.Password)}, true);    await SharedContext.Context.Store.OpenAsync(client);  SharedContext.Context.IsStoreCreated = true;    SharedContext.Context.EntitySetNames = SharedContext.Context.Store.Metadata.GetMetaEntityContainer(SharedContext.Context.Store.Metadata.MetaEntityContainerNames.First()).EntitySetNames; |

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 (if required). OpenAsync method retrieves the service document and metadata document from the SMP Server. After the call to OpenAsync method, the names of all the entity sets are extracted from the metadata document using LINQ.

The ItemsSource property of the first list box in the XAML page is bound to the EntitySetNames property and so the UI is updated automatically with the names of all the entity sets.

|  |
| --- |
| <ListBox BorderThickness="0" BorderBrush="Transparent" ItemsSource="{Binding EntitySetNames}" |

* 1. Go to the LbCollectionsSelectionChanged event handler. In between the BEGIN and END //TODO : Review 5 – markers, review the following code:

|  |
| --- |
| var collectionName = (string)((ListBox)sender).SelectedItem;  if (collectionName == null)  {  SharedContext.Context.EntitySet = null;  }  else  {  await SharedContext.Context.DownloadCollection(collectionName);  } |

Upon selecting an entity set from the list box, the DownloadCollection method is called to retrieve all the entities for that specific entity set.

* 1. Open ODataContext.cs.
  2. Go to the DownloadCollection method. In between the BEGIN and END //TODO : Review 6 – markers, review the following code:

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

The ScheduleReadEntitySet method takes the name of the entity set as a parameter and reads all the entities for that specific entity set. The response from the server is then cast as an ODataEntitySet object. The ODataEntitySet is an IObservableCollection which allows the UI controls to automatically update themselves when the collection is changed.

The ItemsSource property of the second list box is bound to the EntitySet property. The DataTemplate consists of a StackPanel which is used to bind the EditResourcePath.

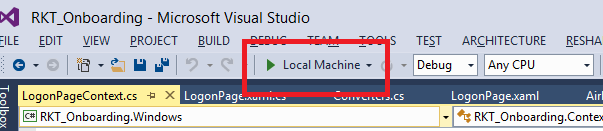
|  |
| --- |
| <ListBox ItemsSource="{Binding EntitySet}" Grid.Column="1" Grid.Row="1" BorderThickness="0" Name="LB\_Entities">  <ListBox.ItemTemplate>  <DataTemplate>  <StackPanel>  <TextBlock Text="{Binding EditResourcePath}" Visibility="{Binding EditResourcePath,Converter={StaticResource NullToVisibility}}"/>  <TextBlock Text="{Binding ResourcePath}" Visibility="{Binding ResourcePath,Converter={StaticResource NullToVisibility}}"/>  </StackPanel>  </DataTemplate>  </ListBox.ItemTemplate>  </ListBox> |

The ItemsSource property of the third list box is bound to the Properties of the selected item of the second list box. The third list box is then used to display the name and value of the properties.

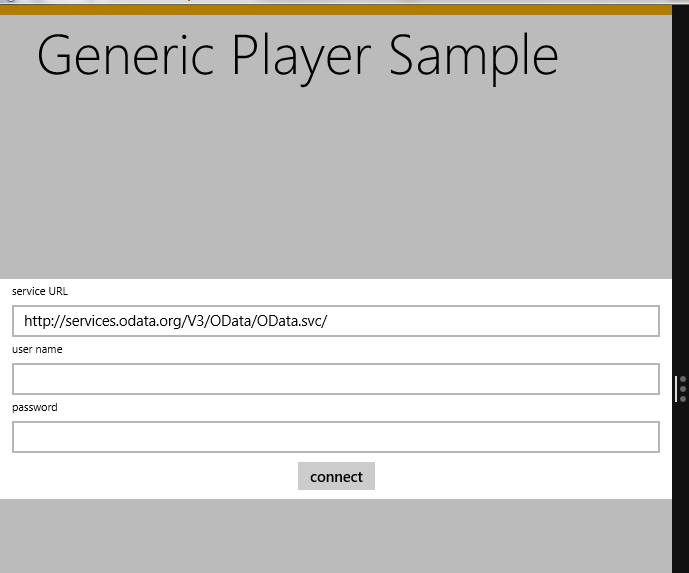
|  |
| --- |
| <ListBox Grid.Column="2" Grid.Row="1" BorderThickness="0" ItemsSource="{Binding ElementName=LB\_Entities, Path=SelectedItem.Properties}" ItemTemplateSelector="{StaticResource TemplateSelector}" IsEnabled="True">  <ListBox.ItemContainerStyle>  <Style TargetType="ListBoxItem">  <Setter Property="HorizontalContentAlignment" Value="Stretch" />  </Style>  </ListBox.ItemContainerStyle>  </ListBox> |

## Running the application

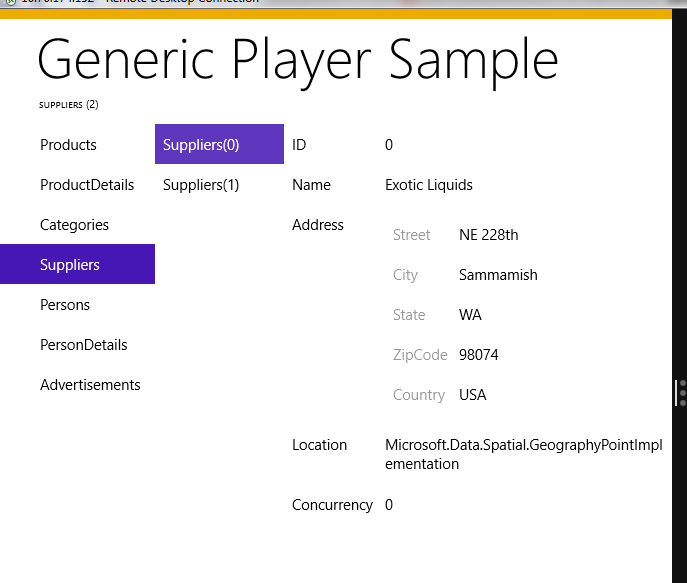
1. Run the application by clicking Run from within Visual Studio 2013 Update 2.



1. Enter any OData service document URL and click Connect. Optionally, you can also provide a username and password to connect to the OData source.



1. The entity sets belonging to the OData source is displayed. Clicking on any one of the entity set displays the entities that belong to that specific entity set. Clicking on an entity displays the name and value of all the properties that belong to that specific entity. Note that complex properties are also handled.



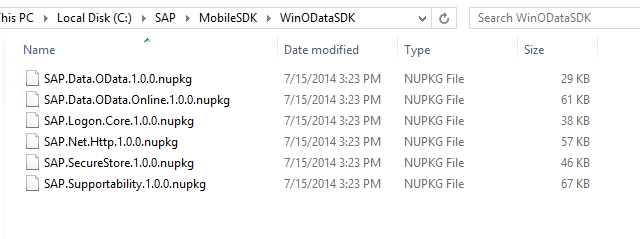
# Appendix

## NuGet Package Manager

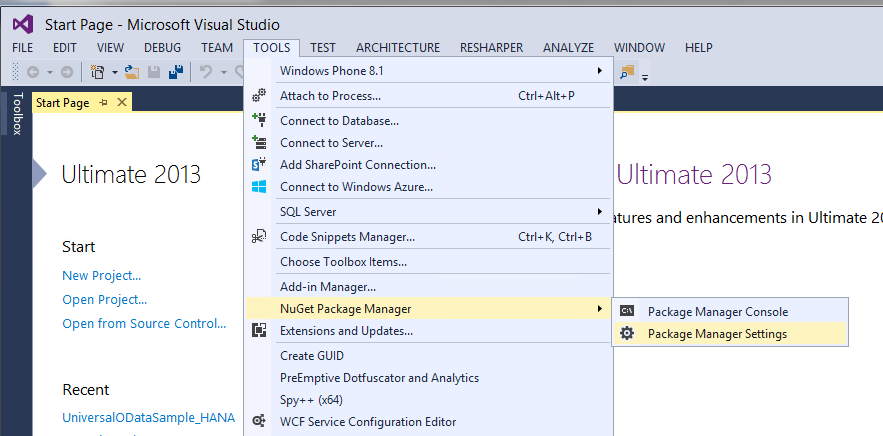
NuGet is the package manager for the Microsoft development platform including .NET. The NuGet client tools provide the ability to produce and consume packages. Starting with Visual Studio 2012, NuGet is included in every edition (except Team Foundation Server) by default. Updates to NuGet can be found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



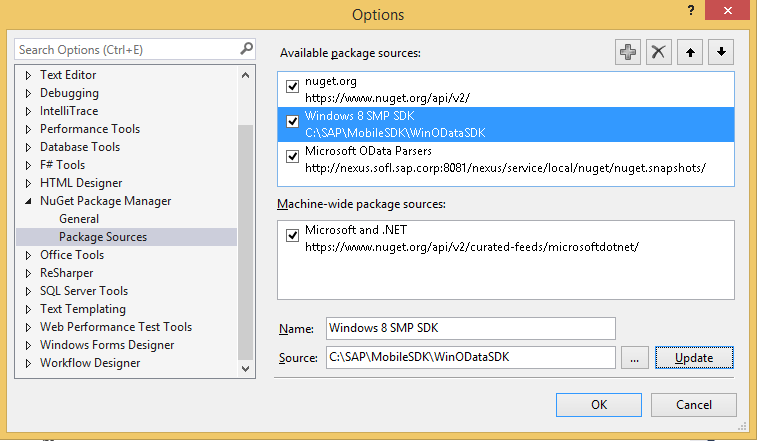
1. Open Visual Studio 2013 with Update 2. Click Tools -> NuGet Package Manager -> Package Manager Settings



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.

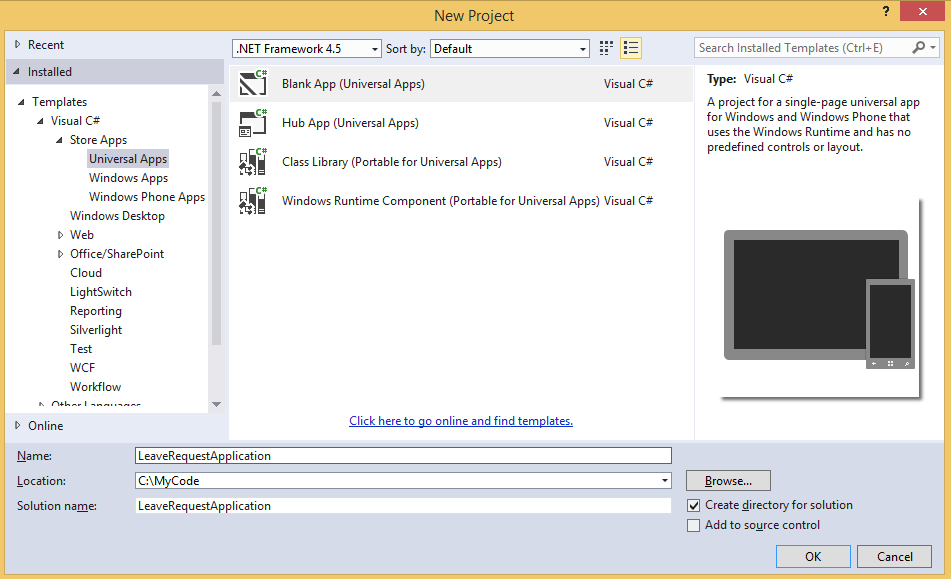
**Name:** Windows SMP SDK

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

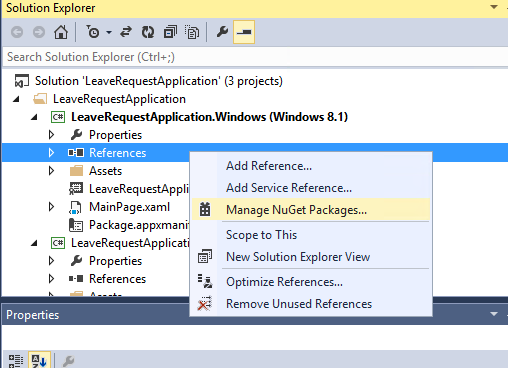


### Adding Windows SMP SDK references to the project

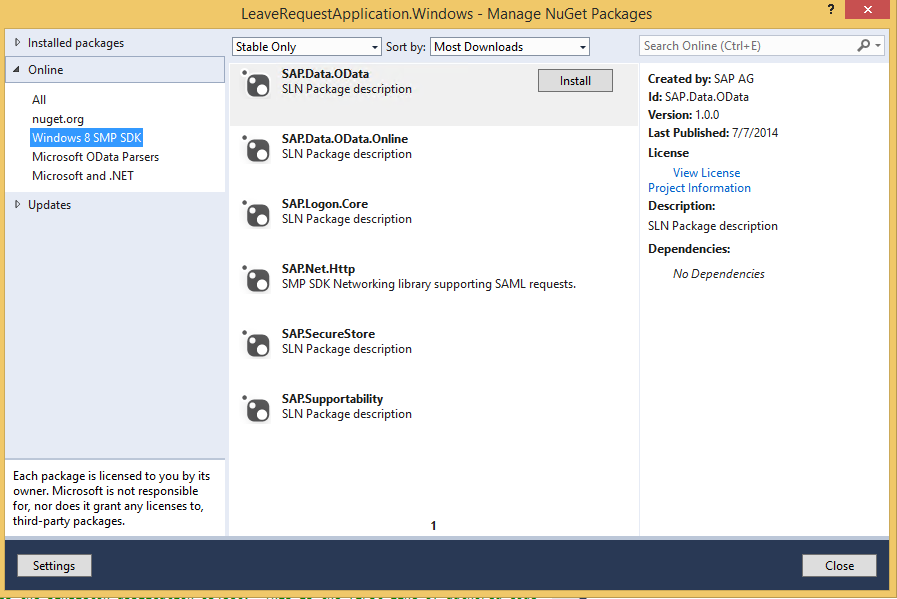
1. Create a new Windows Store Universal project by clicking on the New Project… link. Enter a name for the project.



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