Key-Value Service Programming Guide

Version 2.1 – September 13, 2012

Abstract

The Key-Value Service (KVS) provides a simple key-value store for mobile applications. By using the KVS, an application can save and manage application-wide state information. The KVS is provided as part of the Project Hawaii SDK.

Contents

[Introduction 2](#_Toc335306940)

[Prerequisites 2](#_Toc335306941)

[Overview of the Key-Value Service 2](#_Toc335306942)

[KeyValueItem Class 2](#_Toc335306943)

[KeyValueService Class 3](#_Toc335306944)

[Walkthrough: KeyValueSample 4](#_Toc335306945)

[Source Files 5](#_Toc335306946)

[Initialization 6](#_Toc335306947)

[Key Search 6](#_Toc335306948)

[Key Creation 7](#_Toc335306949)

[Key Deletion 8](#_Toc335306950)

[Using the Key-Value Service in an Application 9](#_Toc335306951)

[Add Required Assemblies 9](#_Toc335306952)

[Reference the Namespace 9](#_Toc335306953)

[Use Your ADM Credentials 9](#_Toc335306954)

[Error Handling 10](#_Toc335306955)

[Resources 10](#_Toc335306956)

**Disclaimer:** This document is provided “as-is”. Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

© 2012 Microsoft Corporation. All rights reserved.

Microsoft, Visual Studio, Silverlight, Windows Azure, and Windows are trademarks of the Microsoft group of companies. All other trademarks are property of their respective owners.

# Introduction

The Project Hawaii Key-Value Service (KVS) provides a simple key-value store for mobile applications. By using the KVS, an application can store and retrieve application-wide state information as text using key-value pairs. The KVS is part of the Project Hawaii SDK.

This document provides an introduction to the managed-code interface for KVS and walks you through a simple example of a Windows Phone application that uses the service.

# Prerequisites

Before you can build an application that uses the Key-Value service, you must:

* Install the Project Hawaii SDK.
* Build the Project Hawaii SDK.
* Obtain Project Hawaii authentication credentials.

For information about installation, build procedures, and credentials, see “Hawaii Installation Guide,” which is installed with the SDK and is available on the web, as listed in “Resources” at the end of this document.

In addition, you should be familiar with the following:

* Windows Communication Foundation (WCF)
* Microsoft Silverlight®
* Windows Phone 7 SDK

# Overview of the Key-Value Service

The Key-Value Service implements a REST interface that supports requests to save, search for, and delete key-value pairs from a data store in the cloud. The body of all POST and PUT requests must be in JavaScript Object Notation (JSON).

The **Microsoft.Hawaii.KeyValue.Client** namespace defines the interface to the Key-Value service. Applications use the **KeyValueService** helper class, which generates calls to the REST interface in the correct JSON format to simplify programming. This document describes how to use the helper class. In addition, the namespace includes the **KeyValueItem** class, which defines a key and its value.

The following sections provide more information about the **KeyValueItem** and **KeyValueService** classes.

## KeyValueItem Class

The **KeyValueItem** class encapsulates a key and its value. The class exposes a constructor and the following properties:

|  |  |
| --- | --- |
| Property | Description |
| **Key** | Gets or sets the key. |
| **LastModifiedDate** | Gets or sets the last modified date. |
| **Value** | Gets or sets the value of the key. |

A **Key** is a string of 1-255 characters. It can contain any characters except the following:

* The forward slash (/) character
* The backslash (\) character
* The number sign (#) character
* The question mark (?) character

A **Value** is a string that contains less than 64K characters.

The **LastModifiedDate** is a [DateTime](http://msdn.microsoft.com/library/03ybds8y) structure that indicates the time at which the key-value pair was most recently changed.

The class also inherits the following methods from the [Object](http://msdn2.microsoft.com/en-us/library/e5kfa45b) class:

* **Equals**
* **Finalize**
* **GetHashCode**
* **GetType**
* **MemberwiseClone**
* **ToString**

## KeyValueService Class

To create, delete, modify, and search for keys, applications can use members of the **KeyValueService** helper class. Methods in this class initiate calls to the Key-Value service on behalf of client applications. This class exposes the following members:

Constructors

|  |  |
| --- | --- |
| Name | Description |
| **KeyValueService** | Initializes a new instance of the **KeyValueService** class. |

Methods

|  |  |
| --- | --- |
| Name | Description |
| **CreateAsync** | Initiates a call to create a key-value item. The item must not already exist. |
| **DeleteAsync** | Initiates a call to delete key-value items. |
| **DeleteByKeysAsync** | Initiates a call to delete key-value items by keys. |
| **GetAsync** | Initiates a call to get key-value items. |
| **GetByKeyAsync** | Initiates a call to get key-value items by key. |
| **GetByKeysAsync** | Initiates a call to get key-value items by keys. |
| **SetAsync** | Initiates a call to set one or more key-value items. If the key does not exist, the method creates it. |

Fields

|  |  |
| --- | --- |
| Name | Description |
| **BeforeKey** | Specifies a timestamp. The query returns items whose **LastModifiedDate** precedes the **BeforeKey**. |
| **ContinuationTokenKey** | Specifies a continuation token to return if the query returns more results than the method can handle in a single request. |
| **DefaultHostName** | Specifies the default service host name, which is used to create the service URL. |
| **GetBatchKey** | Specifies batch operation for the REST methods underlying the client library; not used by application code |
| **KeyValueSignature** | Specifies a signature for the REST methods that manage the KeyValue item; not used by application code. |
| **PrefixKey** | Specifies the beginning of the key name to search for. |
| **SizeKey** | Specifies the number of results to return. |

Properties

|  |  |
| --- | --- |
| Name | Description |
| **HostName** | Returns the host name to use when accessing the service. This property is read-only. |

The class also inherits the following methods from the [Object](http://msdn2.microsoft.com/en-us/library/e5kfa45b) class.

* **Equals**
* **Finalize**
* **GetHashCode**
* **GetType**
* **MemberwiseClone**
* **ToString**

# Walkthrough: KeyValueSample

KeyValueSample is installed with the Project Hawaii SDK. This application implements a simple interface with which a user can create, delete, and search for key-value pairs. This brief walkthrough introduces the components of the sample and shows you how the sample calls methods in the **KeyValueService** class to manipulate key-value pairs.

To compile and run KeyValueSample

1. In Visual Studio, open KeyValueSampleApp.csproj.

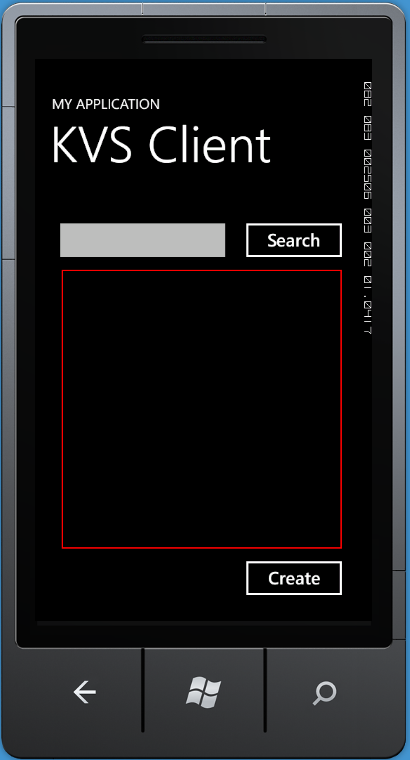
2. Open the HawaiiClient.cs file and set the **AdmClientId** and **AdmClientSecret** strings to your ADM credentials.

3. Save the HawaiiClient.cs file.

4. Build the solution.

5. Run the sample with or without the debugger, as you prefer.

The following figure shows the initial Windows Phone emulator window for the sample:



To use KeyValueSample

* To create a key-value pair, tap **Create** and type the name of the key and its value.
* To search for a key, tap in the text box, type the beginning of the key name to search for, and then tap **Search**. The program displays all matching key names in the listbox.
* To change the value of a key, tap its name in the listbox and then type the new value.
* To delete a key, hold your finger over the name of the key (in the emulator, hold down the left mouse button over its name in the list box), and then tap **Delete** in the pop-up menu.

## Source Files

The following table lists the C# source files for the sample:

|  |  |
| --- | --- |
| Filename | Description |
| App.xaml.cs | Implements App as the **Application** class object and the PhoneApplicationFrame class. |
| HawaiiClient.cs | Implements the HawaiiClient class, which stores the Hawaii Application ID. |
| KeyvalueDetails.xaml.cs | Implements the KeyValueDetails class, which enables a user to search for keys, create keys, and set key values. |
| MainPage.xaml.cs | Implements the initial UI for the application and manages user input. |

Because this document assumes that you are familiar with Windows Phone application development, the walkthrough focuses on the aspects of the sample that are unique to the Hawaii Key-Value service.

## Initialization

When you start the application, the constructor for the Application object sets the root frame for the phone application, establishes an exception handler, initializes Silverlight, and initializes the application. It also sets up profiling information for debugging. This code appears in App.xaml and its code-behind file.

After application initialization is complete, Silverlight makes the main phone application page active and calls the **OnNavigatedTo** method for the page. In the MainPage.xaml.cs file, the sample overrides this method as follows:

protected override void OnNavigatedTo(System.Windows.Navigation.NavigationEventArgs e)

{

base.OnNavigatedTo(e);

this.SearchByKeyPrefix();

}

## Key Search

SearchByKeyPrefix is a local method that calls the Key-Value service to get one or more keys. At startup, the sample gets a list of all the keys, with which it populates the initial frame. The following shows the code for the SearchByKeyPrefix method:

private void SearchByKeyPrefix()

{

this.StartAsync();

KeyValueService.GetAsync(

HawaiiClient.AdmClientId,

HawaiiClient.AdmClientSecret,

this.tbKeysSearch.Text.Trim(),

1000,

string.Empty,

this.OnGetComplete,

null);

}

The local StartAsync method dims the screen and disables the buttons while the service call is in progress. The method then calls **KeyValueService.GetAsync** with the following parameters:

* The AdmClientId and AdmClientSecret, which are set in the HawaiiClient.cs file.
* A prefix string, which identifies the keys to return. The **KeyValueService.GetAsync** method returns all the keys that start with the prefix string. In this case, the prefix string is null, which causes the **GetAsync** method to return all the keys that are defined for this Hawaii Application ID.
* The maximum number of keys to return.
* A continuation token, which the application does not use in this call.
* A pointer to a delegate for callback when this method has completed.
* A user-defined object, which the application does not use in this call.

When the method is complete, the OnGetComplete callback reenables the user interface (UI) and binds the resulting list to the lbkeys ListBox for display.

private void OnGetComplete(GetResult result)

{

Dispatcher.BeginInvoke(() =>

{

this.EndAsync();

if (result.Status != Hawaii.Services.Client.Status.Success)

{

MessageBox.Show(result.Exception.Message);

}

else

{

this.lbKeys.ItemsSource = result.KeyValueCollection;

}

});

}

## Key Creation

When the user clicks **Save**, the sample calls **KeyValueService.SetAsync**, which appears in the Keyvaluedetails.xaml.cs file:

private void BtnSave\_Click(object sender, RoutedEventArgs e)

{

this.StartAsync();

KeyValueService.SetAsync (

HawaiiClient.AdmClientId,

HawaiiClient.AdmClientSecret,

new KeyValueItem[1] { new KeyValueItem() { Key = this.Key,   
 Value = this.Value } },

this.OnSetComplete,

null);

}

The call to **KeyValueService.SetAsync** takes the following parameters:

* The ADM client ID and ADM client secret, which are set in the HawaiiClient.cs file.
* The key and its value.
* A pointer to a delegate for callback when this method has completed.
* A user-defined object, which the application does not use in this call.

When the **SetAsync** operation is complete, the following callback checks status and indicates whether the operation succeeded:

private void OnSetComplete(SetResult result)

{

Dispatcher.BeginInvoke(() =>

{

this.EndAsync();

if (result.Status != Hawaii.Services.Client.Status.Success)

{

MessageBox.Show(result.Exception.Message);

}

else

{

MessageBox.Show("Operation succeeded!");

NavigationService.GoBack();

}

});

}

## Key Deletion

When the user clicks **Delete**, the sample calls **KeyValueService.DeleteByKeysAsync**. The following shows this code from MainPage.xaml.cs:

private void BtnDelete\_Click(object sender, RoutedEventArgs e)

{

this.StartAsync();

KeyValueService.DeleteByKeysAsync(

HawaiiClient.AdmClientId,

HawaiiClient.AdmClientSecret,

new string[1] { (string)(sender as MenuItem).Tag },

this.OnDeleteComplete,

null);

}

The call to **KeyValueService.DeleteByKeysAsync** takes the following parameters:

* The AdmClientId and AdmClientSecret, which are set in the HawaiiClient.cs file.
* A string that contains the keys to delete. The application allows a user to delete only one key at a time.
* A pointer to a delegate for callback when this method has completed.
* A user-defined object, which the application does not use in this call.

When the **DeleteByKeysAsync** operation is complete, the following callback checks status, indicates whether the operation succeeded, and calls SearchByKeyPrefix, described earlier in “Key Search,” to display the currently defined keys:

private void OnDeleteComplete(DeleteResult result)

{

Dispatcher.BeginInvoke(() =>

{

if (result.Status != Hawaii.Services.Client.Status.Success)

{

this.EndAsync();

MessageBox.Show(result.Exception.Message);

}

else

{

MessageBox.Show("Operation succeed!");

this.SearchByKeyPrefix();

}

});

}

# Using the Key-Value Service in an Application

To use the Key-Value service in your own application, you must:

* Add required assemblies to the Visual Studio project.
* Reference the namespace in your source code.
* Set up your Azure Data Market (ADM) credentials.

## Add Required Assemblies

Applications that use the Key-Value service depend on the following libraries, which are built as part of the Project Hawaii SDK:

* Microsoft.Hawaii.ClientBase.dll
* Microsoft.Hawaii.KeyValue.Client.dll

To add the libraries to your application

* Build the Hawaii SDK, as described in “Getting Started with the Project Hawaii SDK.”
* Add references to the following DLLs to your Visual Studio project:
* Microsoft.Hawaii.ClientBase.dll
* Microsoft.Hawaii.KeyValue.Client.dll

## Reference the Namespace

The Key-Value service is defined in the **Microsoft.Hawaii.KeyValue.Client** namespace. For ease of reference, include the following in your code:

using Microsoft.Hawaii.KeyValue.Client;

## Use Your ADM Credentials

Your application authenticates itself with the Key-Value service by using an ADM client ID and client secret. If you do not already have ADM credentials, obtain them as described in “Getting Started with the Project Hawaii SDK.”

The easiest way to use the ADM credentials in your code is to copy the HawaiiClient.cs file from one of the sample applications, set the **AdmClientId** and AdmClientSecretstrings to your ID and secret, and add the source file to your project. You can then use **HawaiiClient.AdmClientId** and **HawaiiClient.AdmClientSecret** wherever the service requires the credentials.

## Error Handling

Methods in the **KeyValueService** return status as one of the following values of the **Microsoft.Hawaii.Status** enumeration:

|  |  |
| --- | --- |
| Enumeration Value | Meaning |
| **Success** | The call completed successfully. |
| **InternalServerError** | An internal server error occurred. |
| **Error** | The call failed. |

# Resources

This section provides links to additional information about Project Hawaii and related topics.

Microsoft Research Project Hawaii

<http://research.microsoft.com/en-us/projects/hawaii/default.aspx>

Getting Started with the Project Hawaii SDK

<http://research.microsoft.com/en-US/projects/hawaii/docs.aspx>

Microsoft Research Project Hawaii on Facebook

<http://www.facebook.com/pages/Microsoft-Research-Project-Hawaii/164295863611699>

MSDN

Programming Windows Phone 7  
<http://blogs.msdn.com/b/microsoft_press/archive/2010/10/28/free-ebook-programming-windows-phone-7-by-charles-petzold.aspx>

How to: Create Your First Silverlight Application for Windows Phone  
<http://msdn.microsoft.com/library/ff402526(v=VS.92).aspx>

How to: Create Your First XNA Framework Application for Windows Phone  
<http://msdn.microsoft.com/en-us/library/ff472340(v=vs.92).aspx>