AT&T Developer Program

AT&T API Platform Extensions for Microsoft ® Visual Studio®

Installation, Setup, and Instruction Guide

Revision **2.3.2**

Date **April, 2013**

Legal Disclaimer

This document and the information contained herein (collectively, the "**Information**") is provided to you (both the individual receiving this document and any legal entity on behalf of which such individual is acting) ("**You**" and "**Your**") by AT&T, on behalf of itself and its affiliates ("**AT&T**") for informational purposes only. AT&T is providing the Information to You because AT&T believes the Information may be useful to You. The Information is provided to You solely on the basis that You will be responsible for making Your own assessments of the Information and are advised to verify all representations, statements and information before using or relying upon any of the Information. Although AT&T has exercised reasonable care in providing the Information to You, AT&T does not warrant the accuracy of the Information and is not responsible for any damages arising from Your use of or reliance upon the Information. You further understand and agree that AT&T in no way represents, and You in no way rely on a belief, that AT&T is providing the Information in accordance with any standard or service (routine, customary or otherwise) related to the consulting, services, hardware or software industries.

AT&T DOES NOT WARRANT THAT THE INFORMATION IS ERROR-FREE. AT&T IS PROVIDING THE INFORMATION TO YOU "AS IS" AND "WITH ALL FAULTS." AT&T DOES NOT WARRANT, BY VIRTUE OF THIS DOCUMENT, OR BY ANY COURSE OF PERFORMANCE, COURSE OF DEALING, USAGE OF TRADE OR ANY COLLATERAL DOCUMENT HEREUNDER OR OTHERWISE, AND HEREBY EXPRESSLY DISCLAIMS, ANY REPRESENTATION OR WARRANTY OF ANY KIND WITH RESPECT TO THE INFORMATION, INCLUDING, WITHOUT LIMITATION, ANY REPRESENTATION OR WARRANTY OF DESIGN, PERFORMANCE, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, OR ANY REPRESENTATION OR WARRANTY THAT THE INFORMATION IS APPLICABLE TO OR INTEROPERABLE WITH ANY SYSTEM, DATA, HARDWARE OR SOFTWARE OF ANY KIND. AT&T DISCLAIMS AND IN NO EVENT SHALL BE LIABLE FOR ANY LOSSES OR DAMAGES OF ANY KIND, WHETHER DIRECT, INDIRECT, INCIDENTAL, CONSEQUENTIAL, PUNITIVE, SPECIAL OR EXEMPLARY, INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, LOSS OF GOODWILL, COVER, TORTIOUS CONDUCT OR OTHER PECUNIARY LOSS, ARISING OUT OF OR IN ANY WAY RELATED TO THE PROVISION, NON-PROVISION, USE OR NON-USE OF THE INFORMATION, EVEN IF AT&T HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH LOSSES OR DAMAGES.

Contents

Introduction

The AT&T API Platform extensions for Microsoft® Visual Studio® (Platform extensions for Visual Studio) combines the power of the Microsoft .NET® platform with the AT&T API Platform and RESTful APIs so that developers can quickly bring Microsoft .NET applications to the market.

The Platform extensions for Visual Studio go even further, delivering pre-packaged controls with a Windows 8 and Windows Phone SDK 8 look and feel that significantly reduces the complexity of building applications targeted to the AT&T API Platform for Windows Phone 8. These controls run in the Windows Phone 8 environment.

The Platform extensions for Visual Studio include the following controls:

- MmsButton
- SmsButton
- SpeechButton
- SpeechControl

Architectural Overview

The Platform extensions for Visual Studio provide simplified access to the AT&T API Platform RESTful APIs for .NET developers. This access is provided through the following components:

- **Windows Phone 8 Application** – The final developer code. A Windows Phone 8 application written in .NET programming language, such as Visual Basic .NET or C#, that makes use of the controls in this package.
- **Controls** – AT&T WP8 controls made available when AT&T WP8 SDK is installed as Visual Studio Extension.
    - The controls in this package can be embedded in Visual Studio applications. These controls handle all messaging back to the .Net Wrapper Library.
- **.NET Wrapper Library** – AT&T Wrapper Library made available when AT&T WP8 SDK is installed as Visual Studio Extension.
    - The .NET Wrapper Library are written in C# but are available to any .NET runtime language. This library provides wrapper around AT&T Platform APIs with a simplifying syntax that gives application developers a set of uniform and easy to use interfaces for addressing the AT&T REST APIs.
- **Communiation Layer** - This layer consists of APIs that communicate with AT&T services via REST.
- You can find more information about the AT&T API Platform on the AT&T Developer Program web site.

Installing the Platform extensions for Visual Studio

To use the Platform extensions for Visual Studio you need the following:

- A computer with Microsoft Windows 8 installed.
- Microsoft Visual Studio 2012 Professional, or a version with greater capabilities.
- Windows Phone 8 Software Development Kit
- An internet connection (required to download the necessary tools).

Please note the following about the controls in this package.

- They are designed to be used in Microsoft Windows Phone 8 applications and do not work with prior versions of the operating system or compilers.
- They are only capable within the AT&T network. Other networks are not supported.

In order to use the AT&T network, you and your organization must abide by the AT&T API agreement. To use the AT&T API Platform RESTful APIs on the AT&T network, a secret key and an API key are required. You can obtain these keys by joining the AT&T developer program.

There are two types of keys, ***Sandbox keys*** and ***Production keys***.

- ***Sandbox keys*** are available for free, and allow programmers to demonstrate proof-of-concept and perform testing on applications using the AT&T APIs.
- ***Production keys*** provide the same capabilities, but are designed to handle a much larger amount of traffic.

You can learn more about how to obtain these keys by creating an account and signing in to http://developer.att.com.

Installing the Components

To install the components of the Platform extensions for Visual Studio, perform the steps.

1. In Visual Studio, open the **Tools** menu and select **Extensions and Updates**. The **Extensions and Updates** dialog window appears.
2. In the **Extensions and Updates** dialog window, open **Online**.
3. Click **Visual Studio Gallery**, as shown in **Figure 1**. Wait a few seconds for the tool to retrieve the most popular extensions.

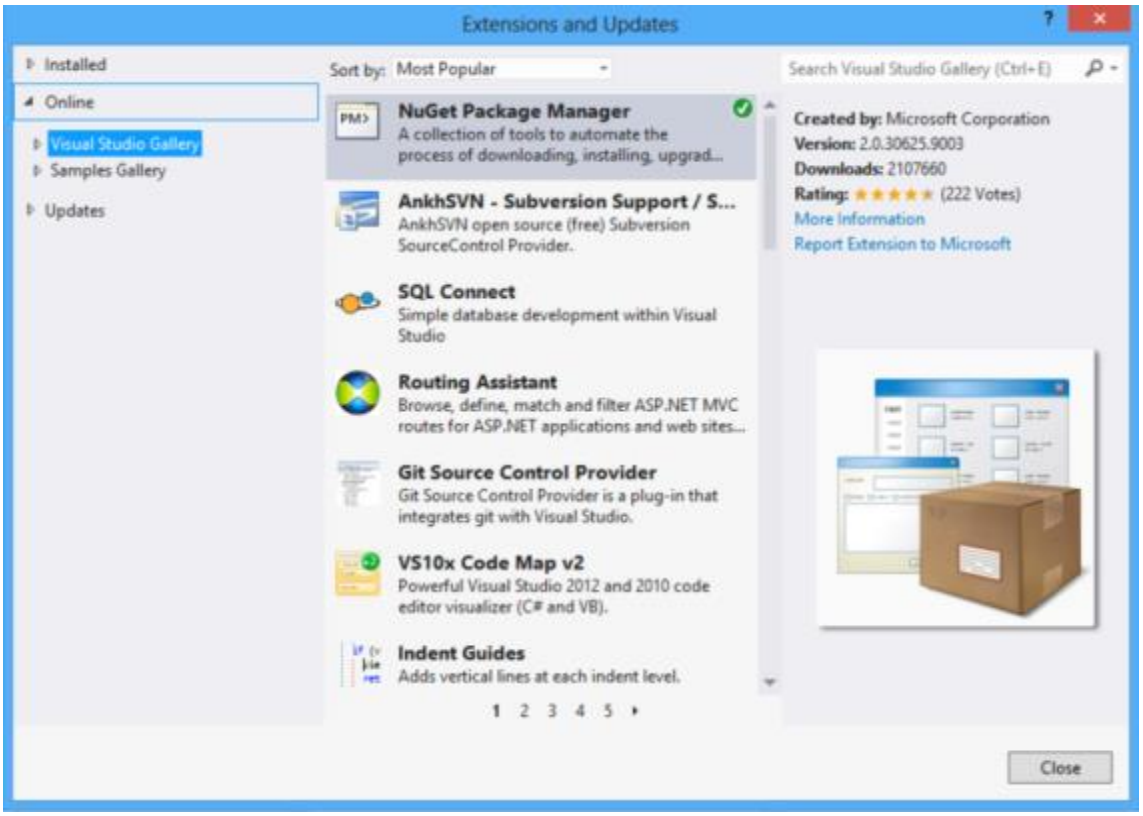

Figure 1: The Extensions and Updates dialog window.

4. When the gallery has finished loading, type `AT&T Extensions for Windows Phone 8` into the search box and hit **Enter** or **Return**. The extensions appear in your list of search results.
5. Click **AT&T Extensions** in the search list to highlight the selection.
6. Click **Download** to download and install the components.

When this process has completed, the Platform extensions for Visual Studio components are installed and ready for use.

Using the Components

To use the Platform extensions for Visual Studio components to create a Windows Phone application, perform the following steps.

1. Start Visual Studio.
2. In the **File** menu, select **New project**, **Templates**, **Windows Phone**.
3. For **Target Windows Phone OS Version**, select **Windows Phone OS 8.0**.
4. Enter an application name and path, and click **OK**.
5. Click on *MainPage.xaml*. An empty `Windows Phone` form should appear. On the left, before the File menu, a new, vertical toolbar should appear with the label: **Toolbox | Device | Document Outline | Data Source**.
6. Click to expand **Toolbox**.
7. Right-click on the toolbox and select **Add Tab** in the context menu. Create a new tab name, such as *AT&T Controls for Windows Phone 8*).
8. Right-click the new toolbox tab and select **Choose items…** in the context menu.
9. Select **Browse…** in the file dialog window.
10. Enter the fully-qualified path to the library, *ATT.WP8.Controls.dll*. The default path is C:\Program Files (x86)\Microsoft SDKs\Windows Phone\v8.0\ExtensionSDKs\AT&T Controls*Number*\References\CommonConfiguration\neutral\ATT.WP8.Controls.dll, where *Number* is the library version number.
11. Click **Open**.
12. Click **OK**.
13. The controls are displayed in the toolbox under **AT&T Controls for Windows Phone 8**, as shown in Figure 2.

T

Figure 2: Toolbox with the AT&T Control Category expanded showing the extensions for Windows Phone 8 Controls.

14. Once you see the list of controls, you can drag and drop a control onto the main page.

At this point, the application can be compiled and it can send messages on the AT&T network using a rotating, encrypted set of API keys to send messages. These keys are sufficient for proof of concept work, but they are not intended to be used for production deployments.

Instead, developers should obtain their own organization Secret Key and API Key at developer.att.com and configure the application to use those keys.

To change to production keys, perform the following steps.

1. Select the control.
2. If it's not open, open the Properties pane, as shown in Figure 3.

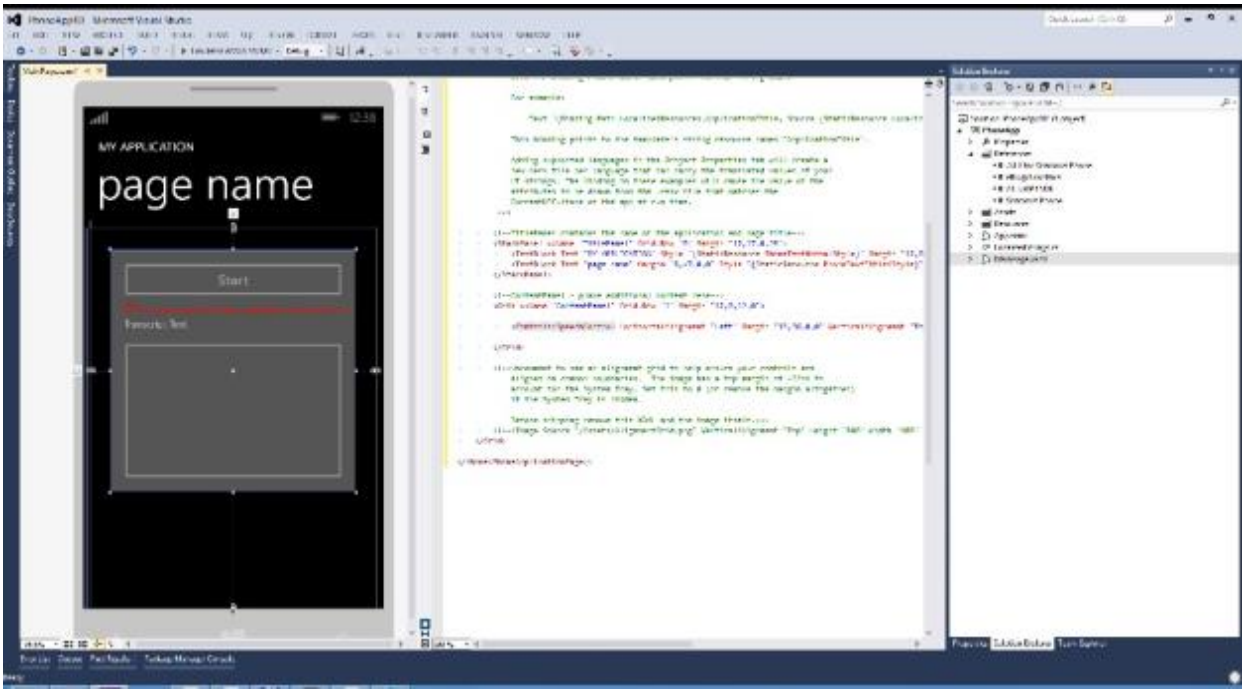

Figure 3: The Speech Control in Microsoft Visual Studio® 2012 Professional.

3. Scroll down the Properties pane to **Miscellaneous** and expand the category. The fields listed under **Miscellaneous** include both the Secret Key and the API Key.
4. Update the values of these fields with the new information.

Your application can now send speech and receive the transcript text on the AT&T Network.

You can also set the Secret Key and the API Key values in the controls xaml. Add the following properties for the Speech Button or Speech control elements.

```
SecretKey = "real value of secret key";
ApiKey = "real value of API key";
```

To test your application, perform the following steps

1. Open the **Build** menu and select **Build Solution**.
2. Open the **Debug** menu and select **Start Debugging on the Windows Phone 8 Emulator**.

About the Controls

This release of the Platform extensions for Visual Studio contains the following controls.

- MmsButton
- SmsButton
- SpeechButton
- SpeechControl

Note: For all of these controls, you must enter the Secret Key and the API Key in the miscellaneous properties of the control. For a description of how to set these values, see Using the Components.

The following sections describe these controls in more detail.

MmsButton

The MmsButton control interacts with the AT&T API Platform through the MMS API. It takes phone numbers, messages up to 600Kbytes, and files with specific type of content, sends them through the MMS API, and delivers status returned from the MMS API.

Figure 7: Mms Button Control

Properties

Property	Data Type	Description
ApiKey	string	Specifies the API key that is assigned to the application when the developer creates an application account at https://developer.api.att.com.
SecretKey	string	Specifies the secret key that is assigned to the application when the developer creates an application account at https://developer.api.att.com.
Endpoint	string	Specifies the endpoint, or base URL, for AT&T services. This value should be **https://api.att.com**.
PhoneNumbers	Observable-Collection-<string>	Specifies a collection of telephone numbers to which messages can be sent.
TextPhoneNumber	string	Specifies a single telephone number.
Message	string	Specifies the text message to send.
Attachments	Observable-Collection-<ContentInfo>	Specifies a collection of objects that contain media files, including their content and filename.
MessageStatus	Enumeration	Specifies the message to display as the status of the message changes: Initial, Sending, DeliveredToNetwork, DeliveredToTerminal, DeliveryImpossible, Error.
MessageStatusName	string	Specifies the name of the status message.
ErrorMessage	sting	Specifies the error message displayed when the speech transcription fails.
CompletedCommand	ICommand	Specifies commands to execute after the speech transcription is successful.
ErrorOccuredCommand	ICommand	Specifies the commands to execute after the speech transcription fails.

Events

Event	Description

ErrorOccurred	Specifies the event that is triggered when the speech to text transcription fails and returns exception details.
MessageStatusChanged	Specifies the event that is triggered when the status of the message changes.
Completed	Specifies the event that is triggered when the button command logic completes.

SmsButton

The SmsButton control interacts with the AT&T API Platform through the SMS API. It takes phone numbers as a collection string or a single string and a message, sends them through the SMS API, and delivers status returned from the SMS API.

Figure 6: Sms Button Control

Properties

Property	Data Type	Description
ApiKey	string	Specifies the API key that is assigned to the application when the developer creates an application account at https://developer.api.att.com.
SecretKey	string	Specifies the secret key that is assigned to the application when the developer creates an application account at https://developer.api.att.com.
Endpoint	string	Specifies the endpoint, or base URL, for AT&T services. This value should be **https://api.att.com**.
PhoneNumbers	Observable-Collection-<string>	Specifies a collection of telephone numbers to which messages can be sent.
TextPhoneNumber	string	Specifies a single telephone number.
Message	string	Specifies the text message to send.
MessageStatus	Enumeration	Specifies the message to display as the status of the message changes: Initial, Sending, DeliveredToNetwork, DeliveredToTerminal, DeliveryImpossible, Error.
MessageStatusName	string	Specifies the name of the status message.
ErrorMessage	sting	Specifies the error message displayed when the speech transcription fails.
CompletedCommand	ICommand	Specifies commands to execute after the speech transcription is successful.
ErrorOccuredCommand	ICommand	Specifies the commands to execute after the speech transcription fails.

Events

Event	Description
ErrorOccurred	Specifies the event that is triggered when the speech to text transcription fails and returns exception details.

MessageStatusChanged	Specifies the event that is triggered when the status of the message changes.
Completed	Specifies the event that is triggered when the button command logic completes.

SpeechButton

The SpeechButton control interacts with the AT&T API Platform through the Speech API. It takes an audio file, uploads the file to the Speech API, and returns the transcribed text of the audio.

Figure 5: Speech Button Control

Properties

Property	Data Type	Description
ApiKey	string	Specifies the API key that is assigned to the application when the developer creates an application account at https://developer.api.att.com.
SecretKey	string	Specifies the secret key that is assigned to the application when the developer creates an application account at https://developer.api.att.com.
Endpoint	string	Specifies the endpoint, or base URL, for AT&T services. This value should be **https://api.att.com**.
SpeechContext	string	Specifies a context to apply to the transcribed text. The user must select one of the following values from the dropdown list. • Generic • TV • BusinessSearch • Websearch • SMS • Voicemail • QuestionAndAnswer • Gaming • SocialMedia
XArgs	string	A set of name/value pairs providing additional informations to be used in an X-Args header. For example: “OrgId=MyOrganization,AppId=MyApplication”
AudioContent	byte[]	Specifies the content of the audio file, in byte [] format.
AudioContentName	string	The name, including extension, of the audio file containing the speech to transcribe.
TranscriptedText	string	Specifies the message displayed when the speech transcription is successful.

ErrorMessages	string	Specifies the error message displayed when the speech transcription fails.
CompletedCommand	ICommand	Specifies commands to execute after the speech transcription is successful.
ErrorOccuredCommand	ICommand	Specifies the commands to execute after the speech transcription fails.

Events

Event	Description
ErrorOccurred	Specifies the event that is triggered when the speech to text transcription fails and returns exception details.
MessageTranscripted	Specifies the event that is triggered when the speech to text transcription is successful.

SpeechControl

The Speech Control, as shown in Figure 4, interacts with the AT&T API Platform through the Speech API. It takes audio from your cell phone microphone, stores it in a file, uploads the file to the AT&T API Platform, and returns the text transcription. When you press **Start**, the control begins recording audio to a file. When you press **Stop**, the control stops recording and sends the file to the AT&T Speech API. The Speech API returns the transcribed text to the control, which displays the transcribed text.

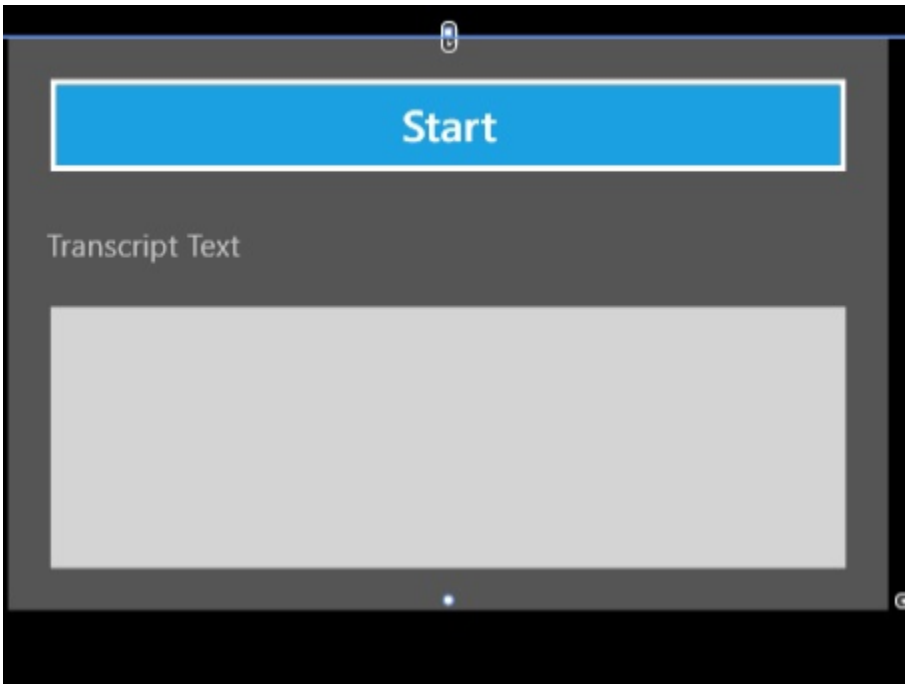

Figure 4: Speech Control

Properties

Property	Data Type	Description
ApiKey	string	Specifies the API key that is assigned to the application when the developer creates an application account at https://developer.api.att.com.
SecretKey	string	Specifies the secret key that is assigned to the application when the developer creates an application account at https://developer.api.att.com.
Endpoint	string	Specifies the endpoint, or base URL, for AT&T services. This value should be **https://api.att.com**.
TranscriptMessage	string	Specifies the message displayed when the speech transcription is successful.

About the Sample Application

In addition to the controls, the Platform extensions for Visual Studio include a sample application that demonstrates the functionality of each control.

To download the sample application, perform the following steps.

1. Go to the Platform extensions for Visual Studio github site at the following location:

   https://github.com/attdevsupport/ATT_APIPlatform_MS_W8_SDK
2. Click on the *ATT.WP8.SampleApp* directory.
3. Click the **ZIP** button to download the repository as a .zip file to your computer.
4. Extract the files from the .zip file to the desired directory.

To load the sample application in Visual Studio, perform the following steps.

1. Use Windows Explorer to browse to the location where you extracted the files.
2. Double-click Visual Studio project file to open the sample application in Visual Studio.

When the sample application opens, you should see a home screen like the one in Figure 6. You can select each of the individual controls and explore their capabilities.

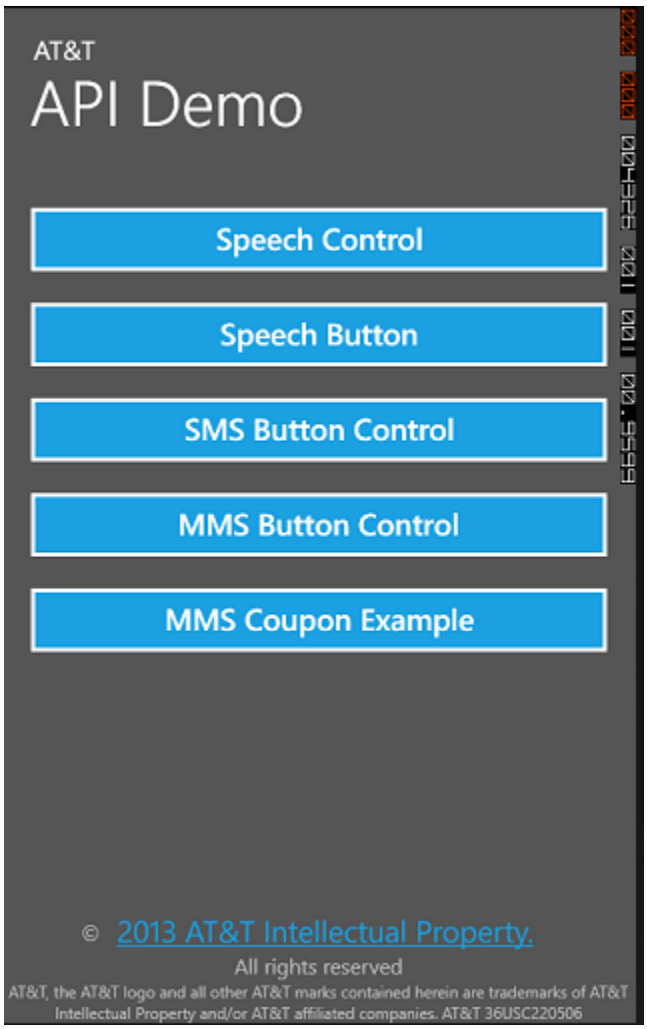

Figure 6: The home screen of the Platform extensions for Visual Studio sample application.