Contents

[Introduction 1](#_Toc366415820)

[Pre-requisites 1](#_Toc366415821)

[Create a mobile service 1](#_Toc366415822)

[Register app for push notifications 3](#_Toc366415823)

[Configuring the Visual Studio project 5](#_Toc366415824)

[Configure the mobile service for notifications 6](#_Toc366415825)

# Introduction

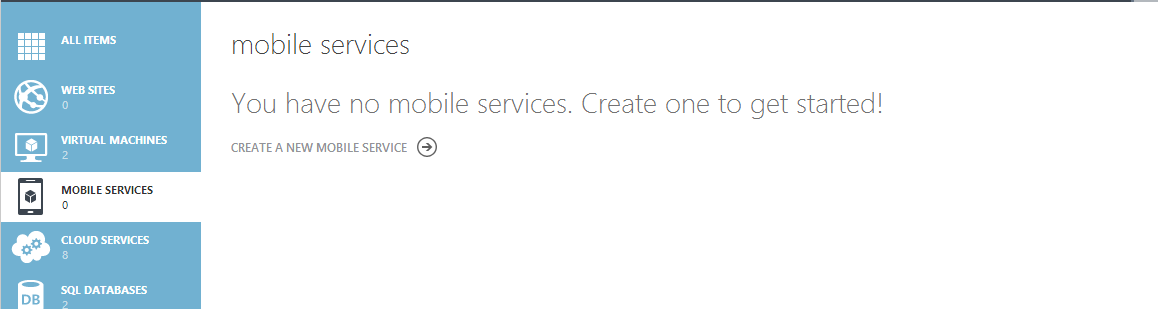
The purpose of this demo is to show how to use notifications on Windows 8 and reference back to Windows Mobile and/or iOS/Android.

# Pre-requisites

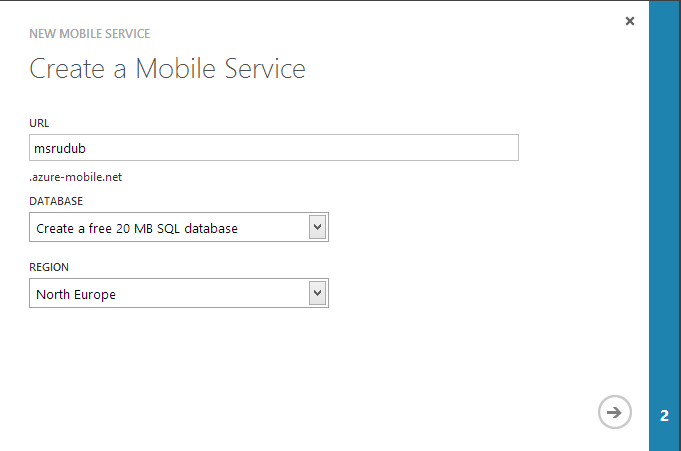
Visual Studio 2012 – any edition

# Create a mobile service

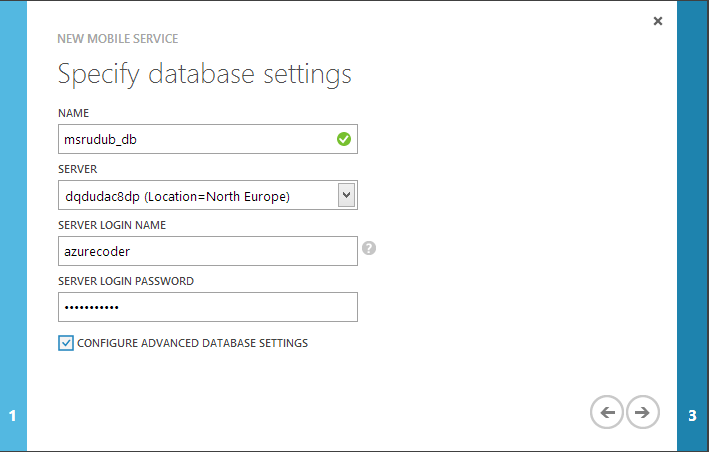
Login to the Windows Azure Management Portal and select to create a new mobile service.



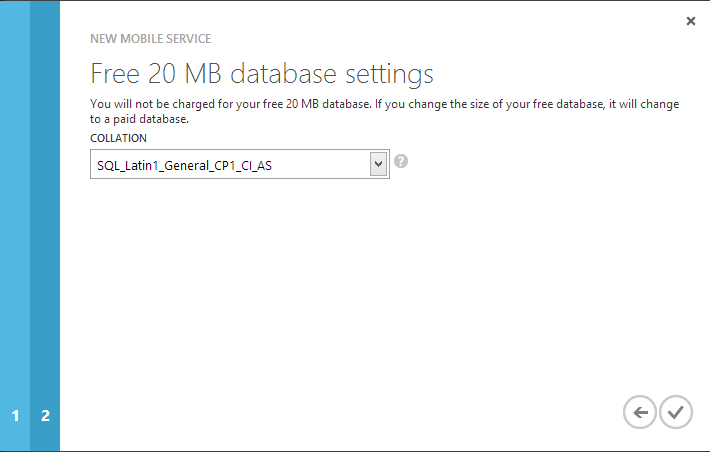
The first step of the wizard should appear as below.



Select the 20MB free Windows Azure Sql Database. You can select the database server we created in the previous sql demo or a new one.



Proceed to the next step and complete the wizard.



The mobile service has now been created.

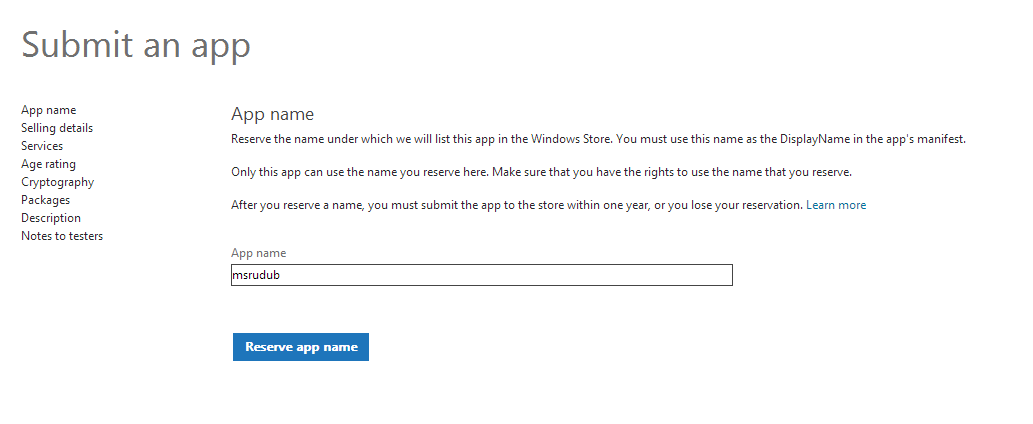


# Register app for push notifications

Go to the Windows Store registration link below

<http://go.microsoft.com/fwlink/p/?linkid=266582&clcid=0x409>

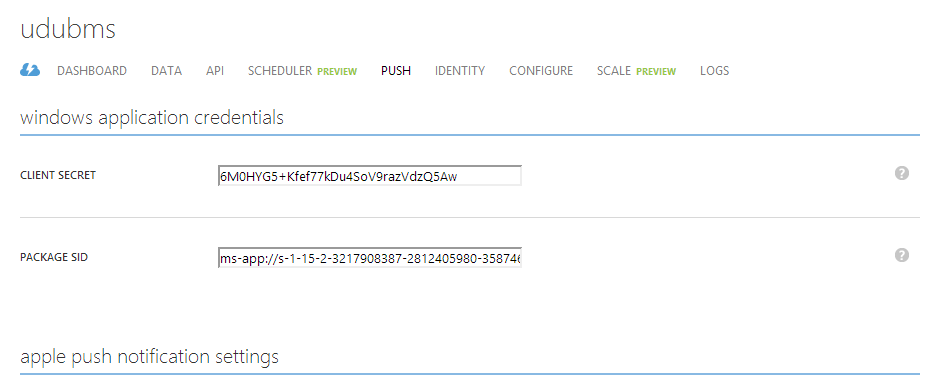
Enter the code that is sent to your mobile as this is protected by MFA.



Once this is done navigate to **services** and follow the **live services site** link.

Follow the **authenticating your service** link and make a note of the **client secret** and **package sid** values.

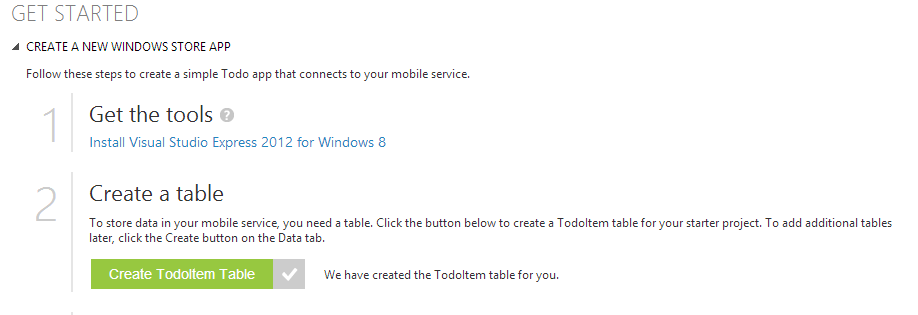
Navigate to the mobile service in the management portal and update the push notification settings through the push notifications tab.



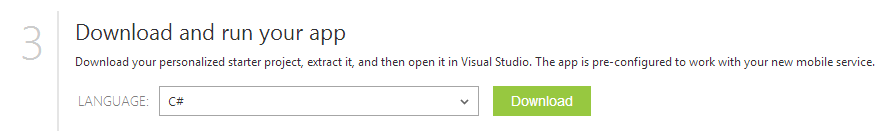
We’ll click on the lightning icon now selecting Windows Azure Store as the platform.



We can choose create a new Windows Store app from the page. We can skip the first of three steps and click on the Create TodoItem table button.

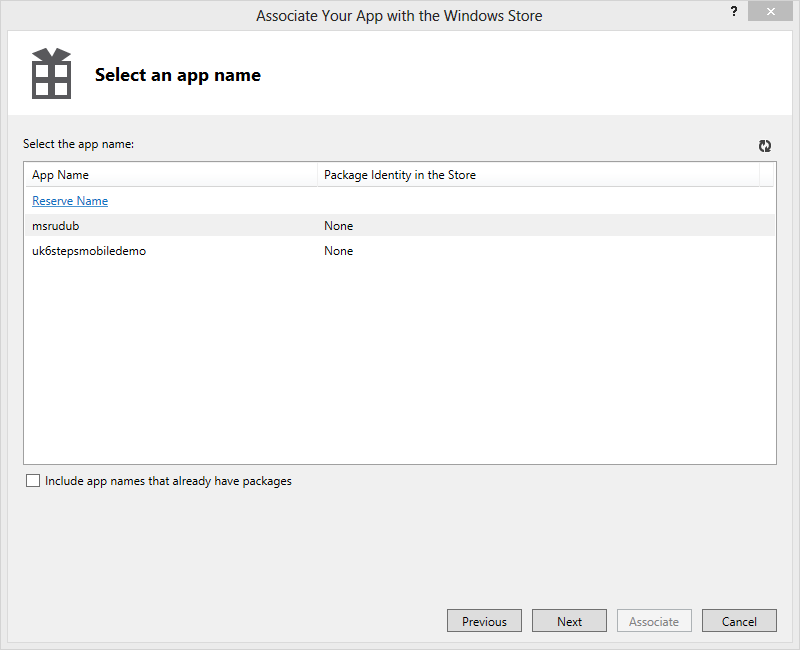


Download and extract the sample application.



We can now open this up in Visual Studio. We’ll click on the project file in visual studio and choose store -> associate app with windows store.

We can now associate the app with the app we created earlier in the list.



# Configuring the Visual Studio project

Open up app.xaml.cs and enter the following in the namespaces section.

using Windows.Networking.PushNotifications;

Add the following property and method in the class.

public static PushNotificationChannel CurrentChannel { get; private set; }

private async void AcquirePushChannel()

{

CurrentChannel = await PushNotificationChannelManager.CreatePushNotificationChannelForApplicationAsync();

}

Navigate to the **OnLaunched** event handler on the app.xaml.cs page and enter the following at the end of the method.

AcquirePushChannel();

Navigate to the **mainpage.xaml.cs** class and find the **TodoItem** type. Add the following property to it.

[JsonProperty(PropertyName = "channel")]

public string Channel { get; set; }

In the **ButtonSave\_Click** event handler we should add the following to the insert of the TodoItem class.

Channel = App.CurrentChannel.Uri

We now need to make the application toast capable. To do this we’ll double click on **Package.manifest** and select **All Image Assets**. We should see the **Toast Capable** appear on the screen. This should default to **yes** but if not select **yes** from the dropdown.

# Configure the mobile service for notifications

In the management portal for the mobile service click on **Data -> TodoItem -> script** and select insert from the **operations** list.

function insert(item, user, request) {

request.execute({

success: function() {

// Write to the response and then send the notification in the background

request.respond();

push.wns.sendToastText04(item.channel, {

text1: item.text

}, {

success: function(pushResponse) {

console.log("Sent push:", pushResponse);

}

});

}

});

}

You should be able to run the application now and generate a push notification based on the above. Simple add a todo item to the list and see the notification arrive to the Windows Store app.