Windows 8 开发 31 日

第 28 日

推送通知

译者：BeyondVincent(破船)

时间：2013.4.25

版本： 2.0

关于破船

程序猿砌墙于云南昆明!

长期扎根移动软件开发!

爱跑步爱打篮球爱运动!

命中无大富大贵之面相!

愿健康与平淡相随一生!

你可以发邮件与破船取得联系: BeyondVincent@gmail.com

还可以关注破船的微博: 腾讯微博和新浪微博。

这里是破船的个人博客，欢迎光临：破船之家

关于 Windows 8 开发 31 日翻译

Windows 8 开发 31 日是由 Jeff Blankenburg 和 Clark Sell 原创的。

官方站点：http://31daysofwindows8.com/

涉及到两个版本：

XAML/C#（由 Jeff Blankenburg 撰写）

HTML5/JS （由 Clark Sell 撰写）

其中涉及到的资源和相关代码请到这里下载：

https://github.com/csell5/31DaysOfWindows8

在这里，由于破船对 HTML5/JS 不熟悉，所以只翻译 XAML/C#相关主题。

建议大家前往看原创内容，如果看不明白，再来这里看我翻译的相关内容。

如果翻译不正确的地方，可以通过上面的联系方式告诉破船。

破船祝你阅读愉快！

目录

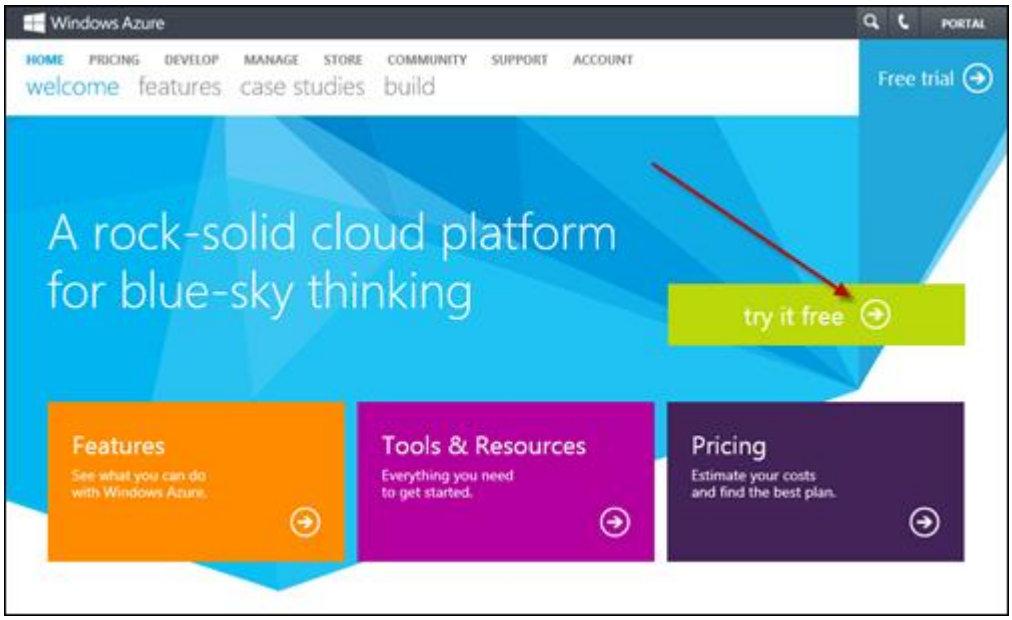
第 28 日推送通知

1.0. 介绍

今天，我来谈谈推送通知。在之前的 27 天里面，Clark Sell 和我都用相同的方法来解决每一个不同的技术问题。今天，Clark Sell 继续以传统的开发方式介绍，而我则先介绍一些来自 Windows Azure 非常棒的服务。下面我开始介绍 Windows Azure Mobile Services。

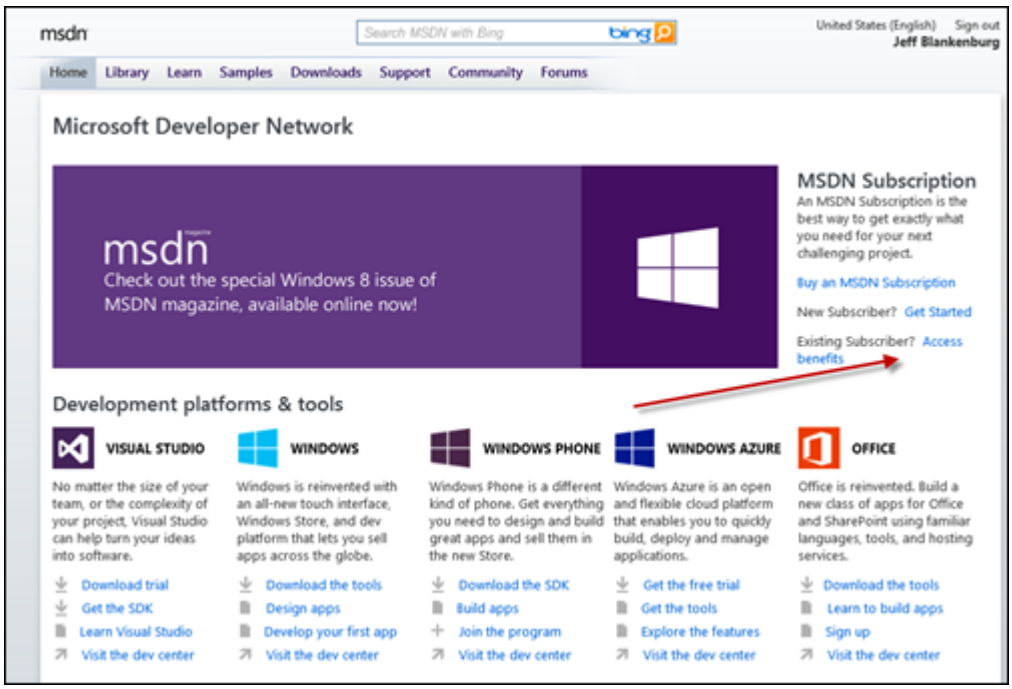
在使用 Windows Azure Mobile Services 之前，必须先做一些相关配置。首先用免费的 Windows Azure 帐号在 http://www.windowsazure.com 进行配置。

点击“try it free”按钮，然后注册一个帐号。



在这里我将介绍免费体验帐号的每一个使用步骤。

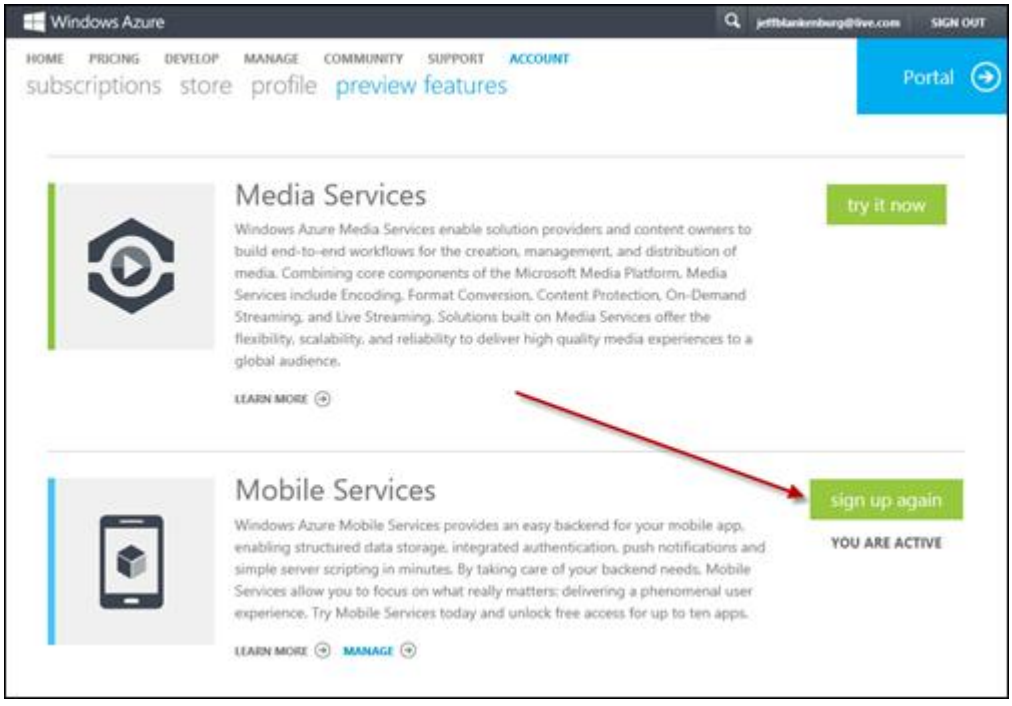
如果你是 MSDN 的订阅者，那么你已经获得了一个免费的 Azure！在 http://msdn.microsoft.com 点击”Access Benefits”链接。



在 Subscription Benefits 中，你会发现 Windows Store 开发者账号的访问，Windows Azure 访问和其它一些东西。点击“Activate Windows Azure”链接，然后跟着说明进行注册即可。

1.1. 现在你已经有一个 Windows Azure 帐号

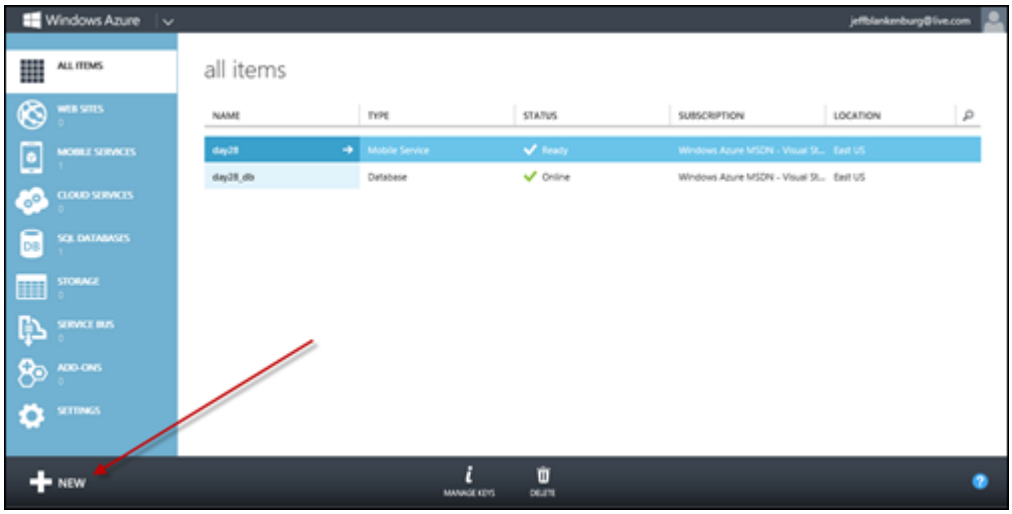
同样，你还得添加一个移动服务到你的账号中（免费的），访问 https://account.windowsazure.com/PreviewFeatures 来激活移动服务。如下图：



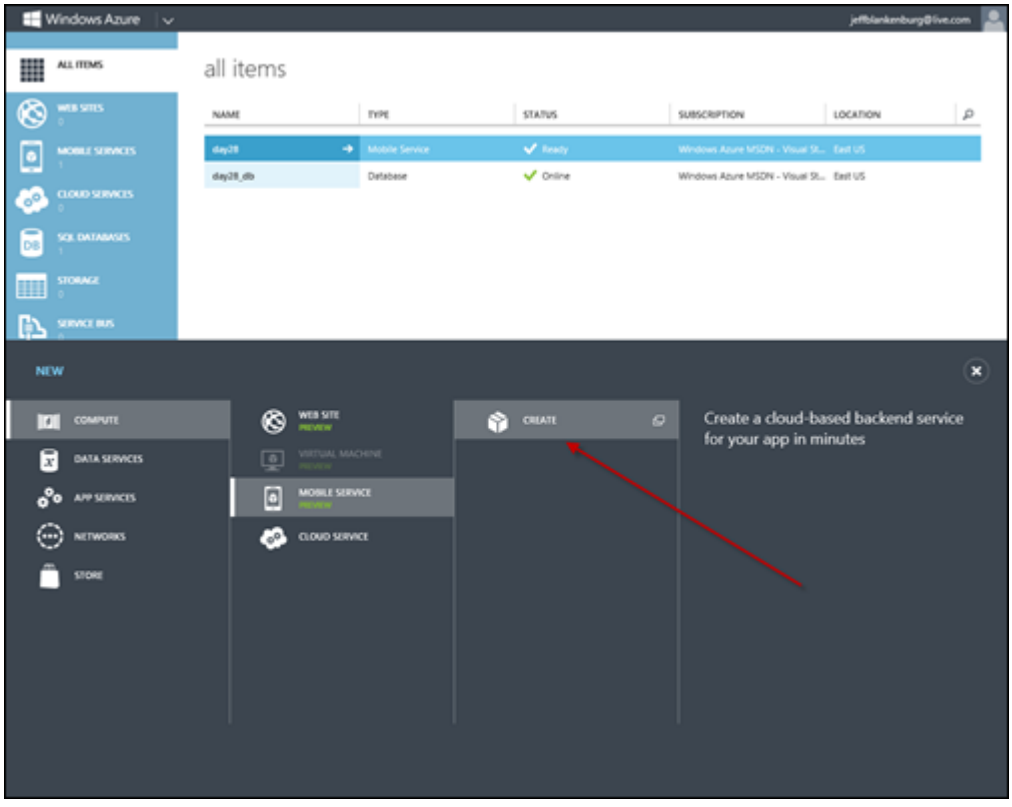
在上图中，我的是已经激活了，如果还没有的话，按钮会显示“try it now”。

1.2. 现在可以使用 Azure Mobile 服务

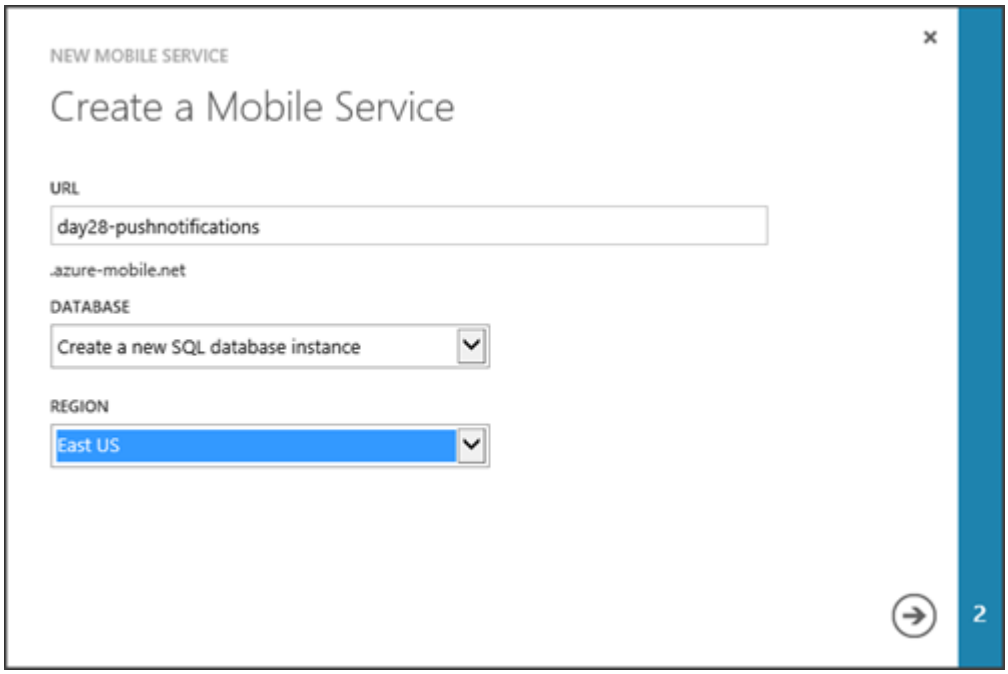
我们来创建一个新的移动服务。打开 http://manage.windowsazure.com,并登录进。点击屏幕底部的“New+”按钮。



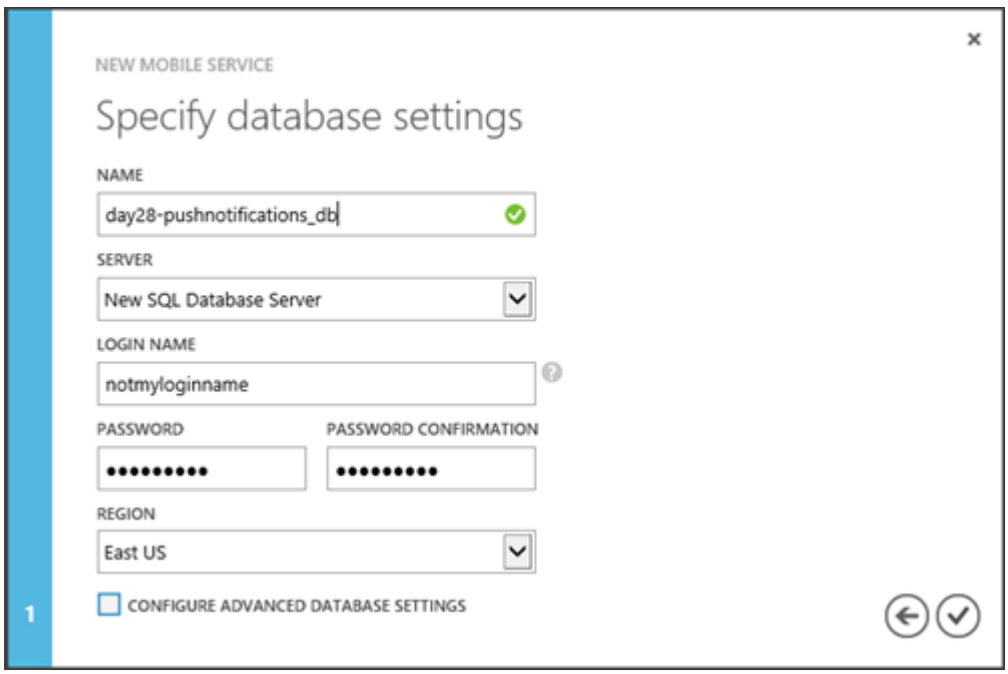
下面做一些选择设置（选择 Compute->Mobile Services）



在创建的第一步中，需要指定一个服务的名称。我在这里命名为“day28-pushnotifications”.然后指定我将使用一个 SQL 数据库实例。如果你之前已经用过 Windows Azure，也可以选择一个已经存在的数据实例。



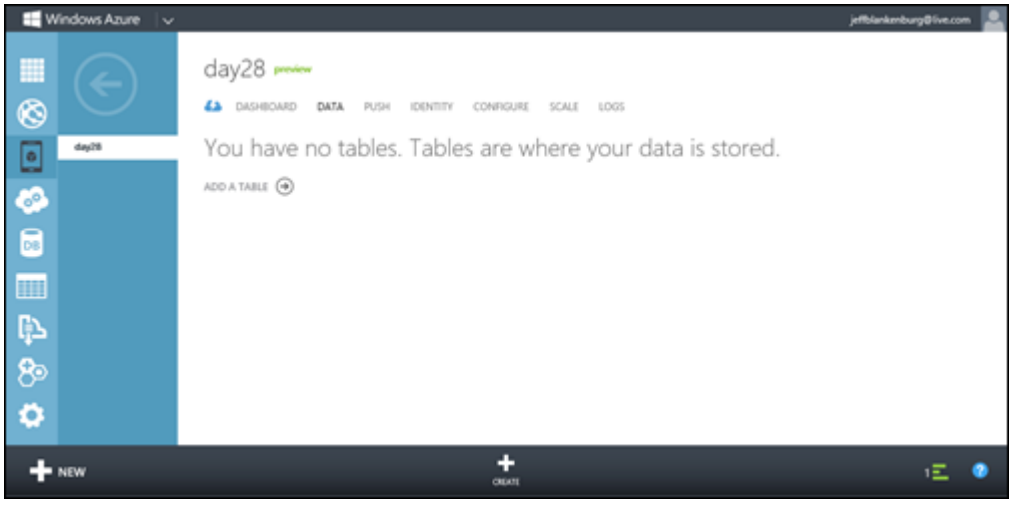
下一步，需要配置一下数据库，比如数据库 name，username 和密码。



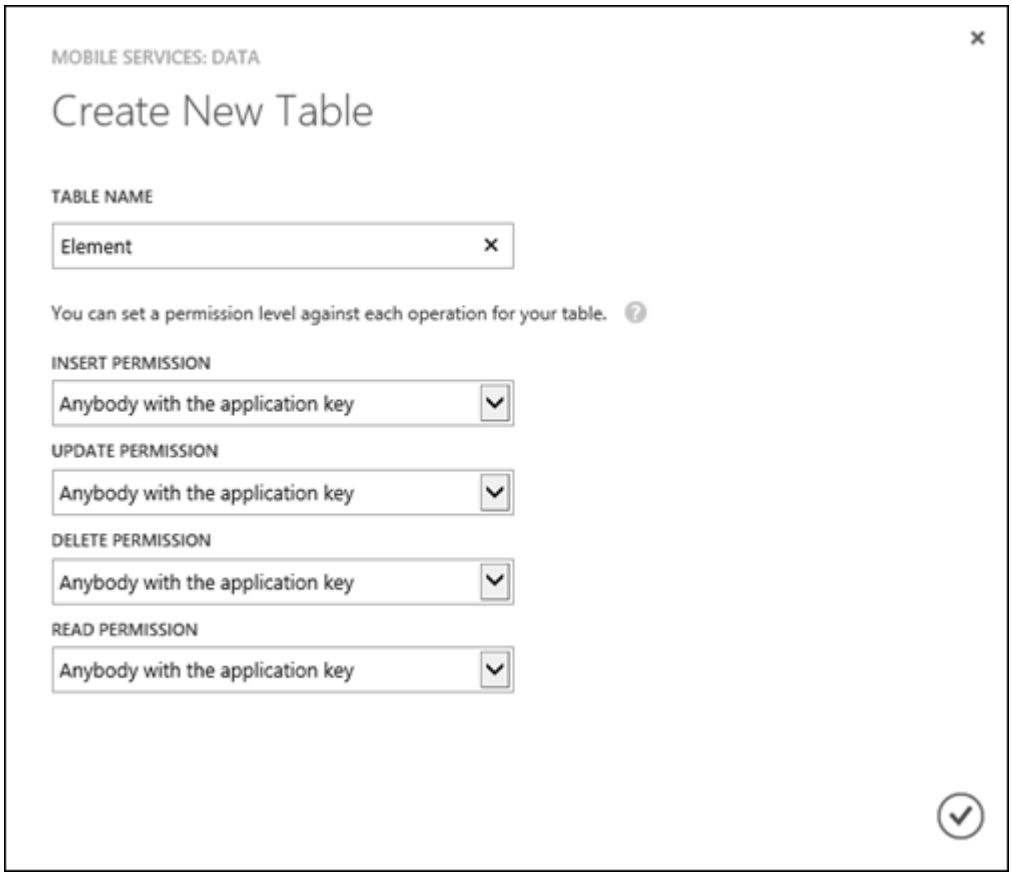
上面的步骤完成之后，我们需要对数据库做一些配置。

1.3. 现在已经有一个数据库&移动服务

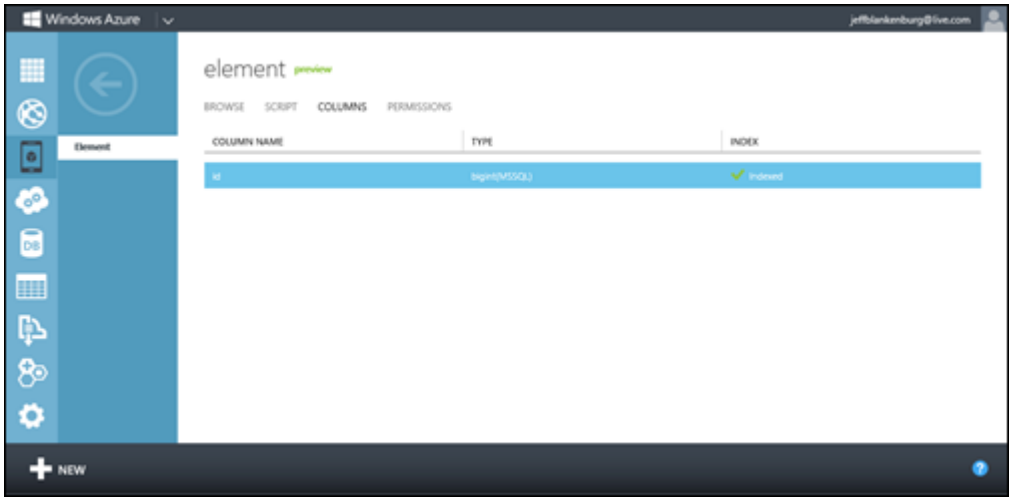
首先，来配置一下 data source，这样我们才有一个数据库表可以使用。在 Windows Azure Portal 中打开你的移动服务，选择 DATA 选项，如下：



点击“ADD A TABLE”链接，然后指定表名，这里，我命名为 Element：



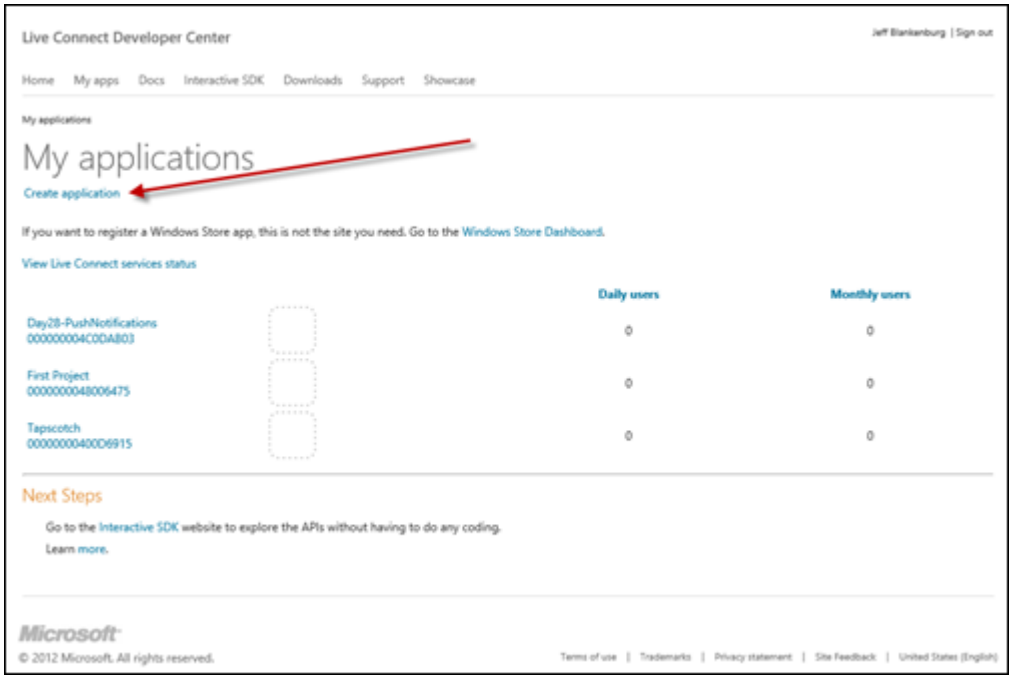
创建好表之后，打开该表，然后点击 COLUMNS 选项，如下：



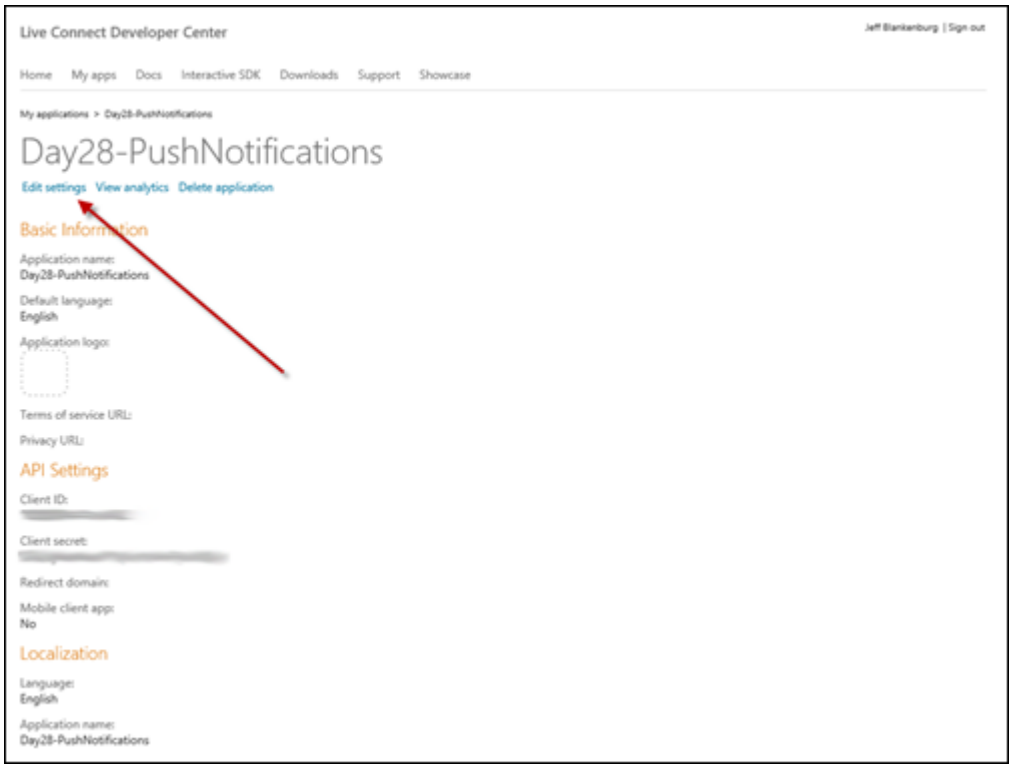
可以看到，此时表中只有 ID 列。稍后还会进行相关介绍。

1.4. 现在已经数据库表的配置

下面我先利用 Windows Live 注册我的程序（当然，这一步可以在我们程序创建完成之后进行，无所谓。），注册之后，就可以使用通知服务功能了（Windows Store 开发者账号在这里不是必须的），打开这个链接 http://manage.dev.live.com，在系统中创建一个新的程序，如下：



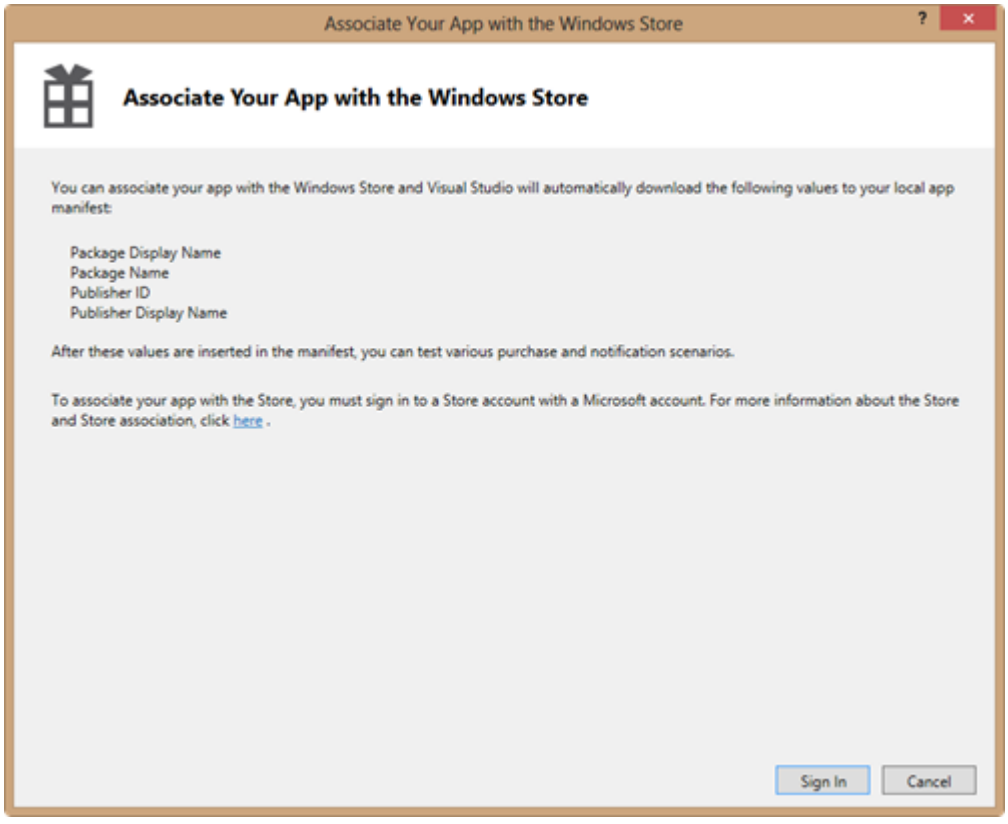
系统中的程序名字不是非常重要，不过最好还是有意义的名字，这样才方便你的记忆，创建好新的程序之后，在 dashboard 中点击程序，然后选择“编辑设置”。

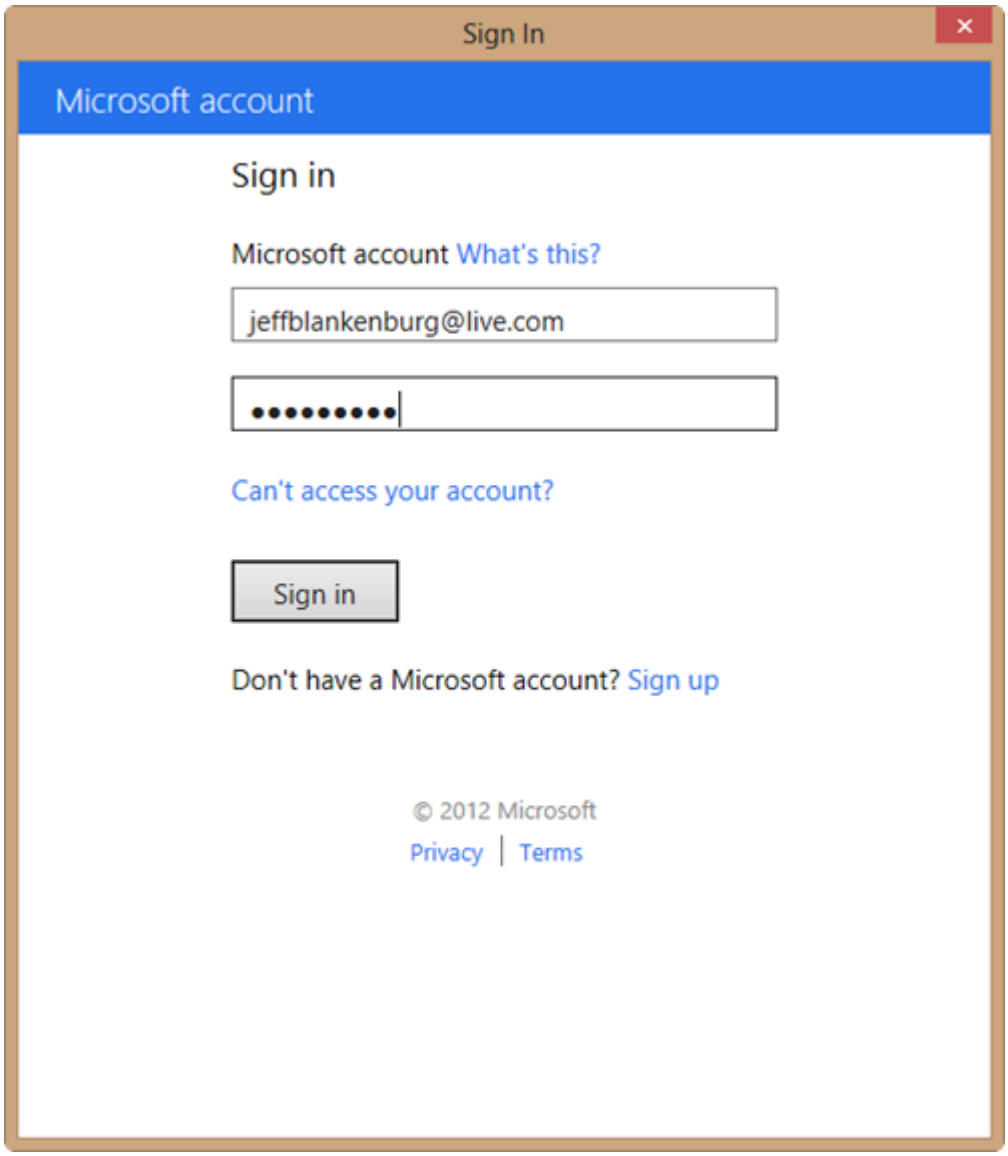


在设置中，选择 API Settings，将会显示两个重要的值："client secret"和"Package SID"。请记住这两个值，之后会用到。

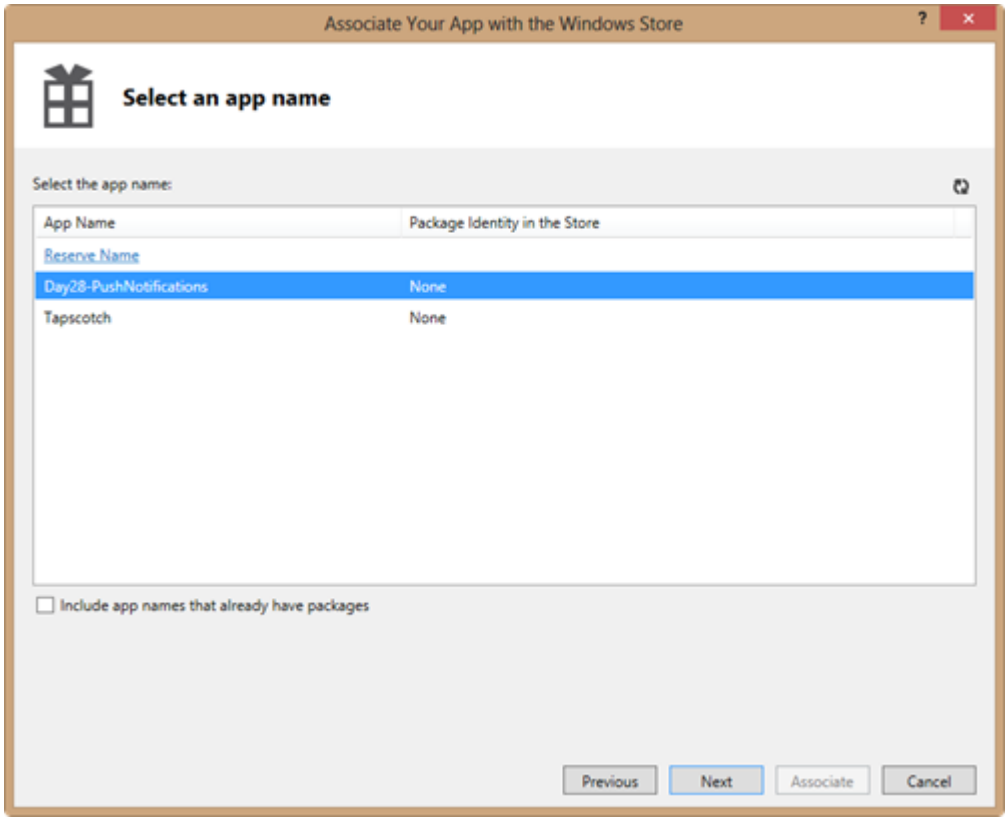
1.5. 现在已经有 Secret 和 SID 了

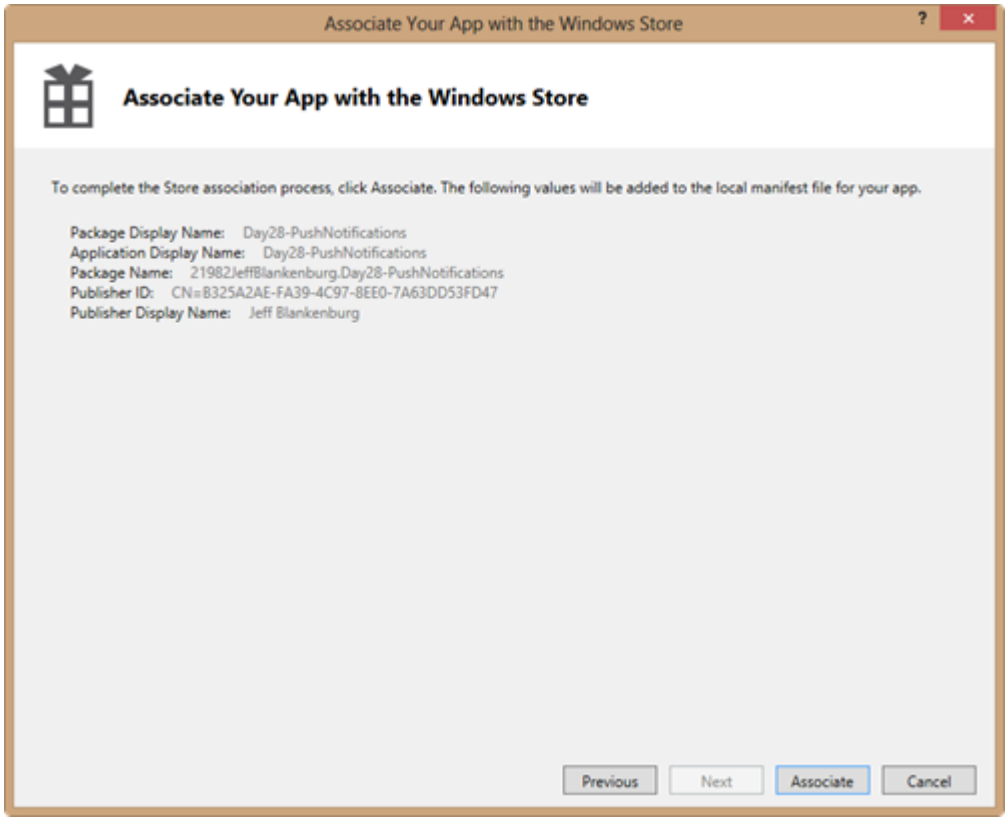
现在已经到开始创建我们的 Windows 8 程序了。在 Visual Studio 2012 中新建一个空白工程，然后在工程名字上右键单击，找到"Store"菜单选项，点击"Associate app with the Store…"选项,然后会看到如下信息:


Associate Your App with the Windows Store
Associate Your App with the Windows Store
You can associate your app with the Windows Store and Visual Studio will automatically download the following values to your local app manifest:
Package Display Name
Package Name
Publisher ID
Publisher Display Name
After these values are inserted in the manifest, you can test various purchase and notification scenarios.
To associate your app with the Store, you must sign in to a Store account with a Microsoft account. For more information about the Store and Store association, click here .
Sign In
Cancel



登录 Windows Live ID,会看到之前在 http://manage.dev.live.com 创建的程序列表,我选择创建的 Day28 这个.

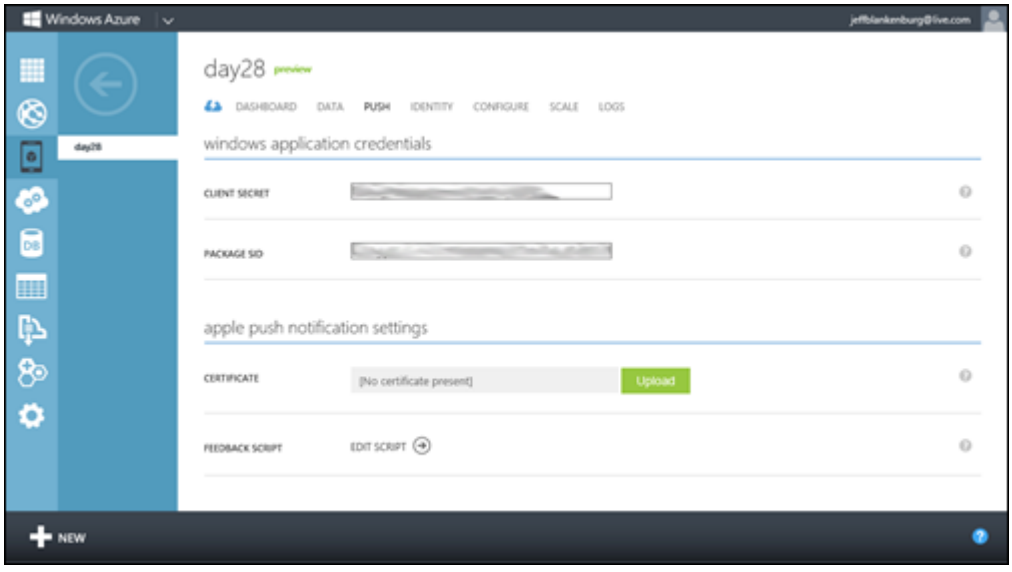

Associate Your App with the Windows Store
Select an app name
Select the app name:
App Name
Package Identity in the Store
Reserve Name
Day28-PushNotifications
None
Tapscotch
None
Include app names that already have packages
Previous
Next
Associate
Cancel



最后点击 Associate 按钮。现在你的 Windows 8 程序已经跟 http://manage.dev.live.com 中的相关程序关联上了。

1.6. 现在程序已经做好了关联处理

现在该更新一下我们的移动服务了，这样才能与我们的程序进行推送通知。返回到 http://manage.windowsazure.com 链接，打开移动服务，选择 PUSH 选项，如下：



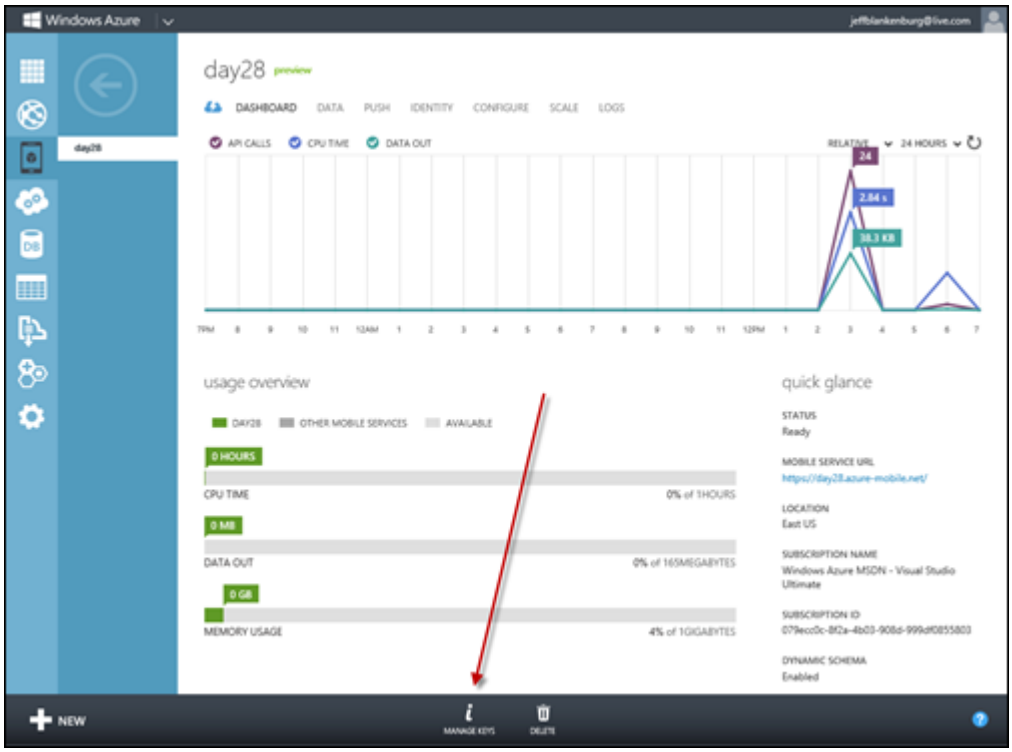
将之前的“client secret”和“package SID”值填写到对应框中。这样就无须其它步骤，就可以与我们的程序进行通信了。

1.7. 所有事情都配置完毕

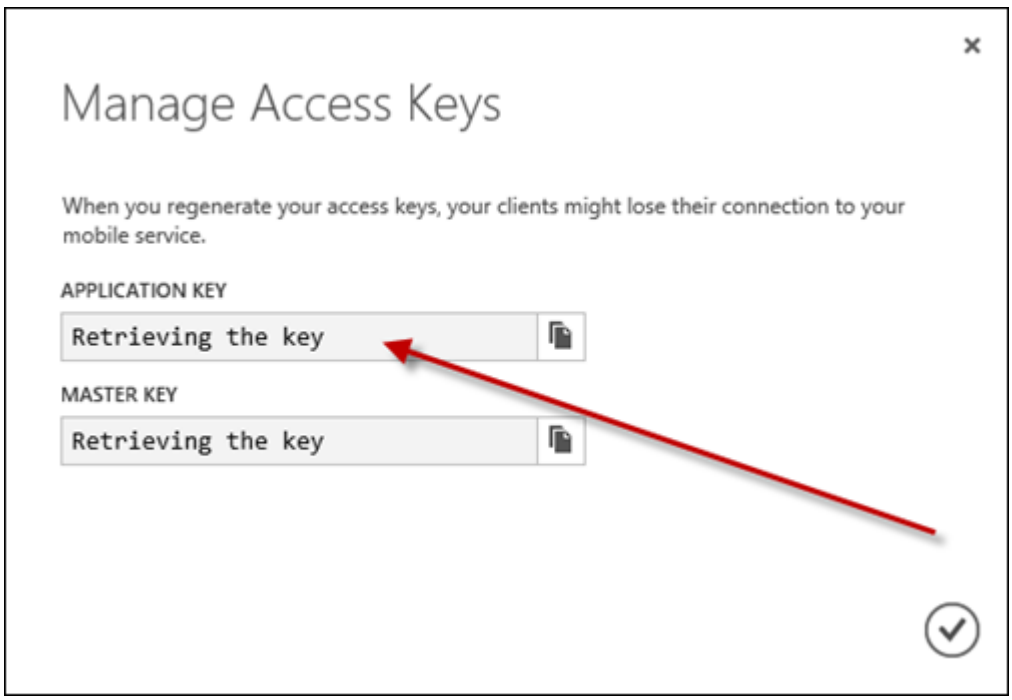
现在返回到我们的程序中，实际的开始写点代码吧。之前所做的使用步骤，就是让我们的程序可以使用 Azure。打开 App.xaml.cs 文件，添加如下几行代码：

```
publicstaticMobileServiceClientMobileService = newMobileServiceClient("https://day28.azure-mobile.net/",
"VbUeYjuNkkRSRRlKSRLqfqXQBmhVqq54");

publicstaticPushNotificationChannel channel { get; privateset; }

privateasyncvoidGetPushChannel()
{
channel = awaitPushNotificationChannelManager.CreatePushNotificationChannelForApplicationAsync();
}
```

第一行代码是使用相关的服务，同时需要提供你的“application key”，这个可以在 Azure portal 中找到，点击屏幕底部的“manage keys”按钮，如下：



然后从对话框中获取 Application Key：



上图中，很明显，我没有让你看到我的 keys，不过这个 key 是有字符和数据组成的。最后，在 OnLaunched 方法中调用的 GetPushChannel()方法。

1.8. 现在已经打开了一个推送通知通道

在本系列文章中好多地方都使用了“elements”，这里我将 Element 对象存储到我之前创建的数据中。首先，我新建一个 Element 类。代码如下：

```
using System;
usingSystem.Collections.Generic;
usingSystem.Linq;
usingSystem.Runtime.Serialization;
usingSystem.Text;
usingSystem.Threading.Tasks;

namespace Day28_PushNotifications
{
classElement
      {
publicint Id { get; set; }
publicdoubleAtomicWeight { get; set; }
publicintAtomicNumber { get; set; }
publicstring Symbol { get; set; }
publicstring Name { get; set; }
publicstring Category { get; set; }
publicstring State { get; set; }

            [DataMember(Name = "channel")]
publicstring Channel { get; set; }
      }
}
```

上面代码中，我添加了一些属性，其中 Id 和 Channel 是最重要的。ID 值用来匹配数据库中已经存在的 Id 字段，而 Channel 值则保存着之前在 App.xaml.cs 中获取到的 Channel URI。

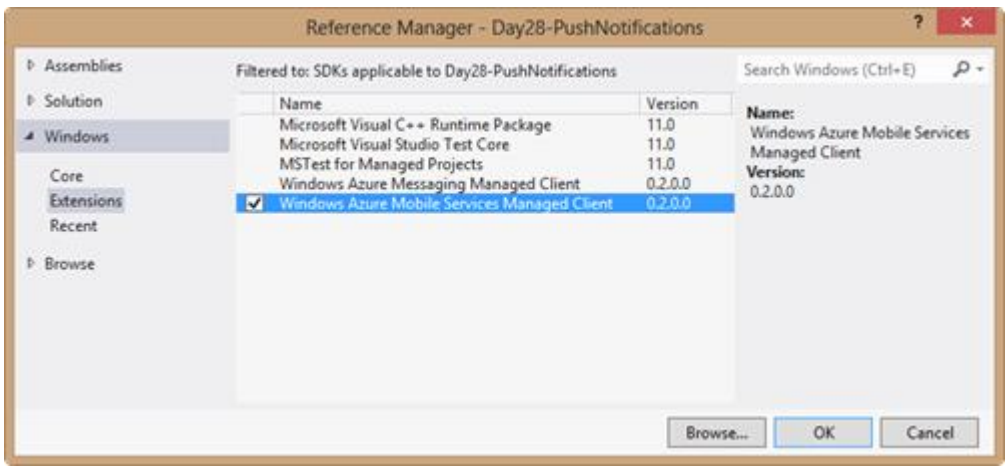
1.9. 现在已经配置了一个自定义的类

在写最后的代码之前，需要做的最后一个步骤是下载 Mobile Services SDK。点击这里下载 Mobile Services SDK。下载之后安装即可。

1.10. 现在已经安装了移动服务 SDK

为了让程序变得简单，我只是在 MainPage.xaml 文件中添加了一个 Button 控件 然后添加了一个 AddElementButton_Click()。当然 这里还需要将 MobileSerVices SDK 以引用的方式添加进来。

右键单击工程中的 Reference 文件夹，选择“Add Reference…”



在打开的窗口中，选择 Windows Azure Mobile Services Managed Client，然后点击 OK，就会将其添加到工程中，下面是在 App.xaml.cs 文件中需要添加的代码语句:

```
usingMicrosoft.WindowsAzure.MobileServices;
```

添加了名称控件之后，我们就可以与 Windows Azure 中的数据表进行直接交互。为了做一个简单的数据插入功能：将数据插入到 Element 表中，下面是相关的语法：

```
Elementelement = newElement
{
AtomicNumber = 1,
AtomicWeight = 1.01,
      Category = "Alkali Metals",
      Name = "Hydrogen",
      Symbol = "H",
      State = "Gas",
      Channel = App.channel.Uri
};
awaitApp.MobileService.GetTable<Element>().InsertAsync(element);
```

如上代码所示只需要两行代码即可将数据插入到数据库中。实际上，如果你打开 Windows Azure Portal 中的 Mobile Service 的 DATA 选项，可以看到数据记录的动态显示。

在这里，我们的程序已经可以将数据插入到 Azure 数据库中了。但是这篇文章的主题不是**推送通知**吗？我这是在做什么？在开始推送通知之前，我们来看看 MainPage.xaml.cs 文件的完整内容，如下：

```
usingSystem.Collections.Generic;
usingWindows.UI.Xaml;
usingWindows.UI.Xaml.Controls;
usingWindows.UI.Xaml.Navigation;

namespace Day28_PushNotifications
{
publicsealedpartialclassMainPage : Page
      {
int counter = 0;
List<Element> elements;

publicMainPage()
            {
this.InitializeComponent();
```

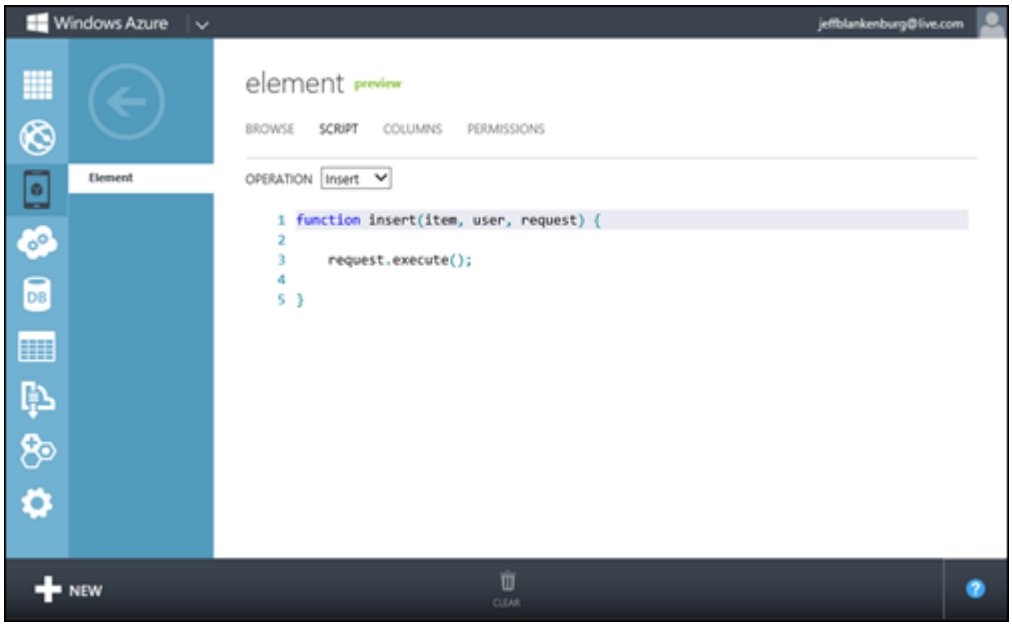
```
        }

protectedoverridevoidOnNavigatedTo(NavigationEventArgs e)
        {
CreateElementList();
        }

privateasyncvoidAddElementButton_Click(object sender, RoutedEventArgs e)
        {
Elementelement = elements[counter];
element.Channel = App.channel.Uri;
awaitApp.MobileService.GetTable<Element>().InsertAsync(element);
counter++;

if (counter >= elements.Count)
counter = 0;
        }

privatevoidCreateElementList()
        {
elements = newList<Element>();
elements.Add(newElement { AtomicNumber = 1, AtomicWeight = 1.01, Category = "Alkali Metals", Name =
"Hydrogen", Symbol = "H", State = "Gas" });
elements.Add(newElement { AtomicNumber = 2, AtomicWeight = 4.003, Category = "Noble Gases", Name =
"Helium", Symbol = "He", State = "Gas" });
elements.Add(newElement { AtomicNumber = 3, AtomicWeight = 6.94, Category = "Alkali Metals", Name =
"Lithium", Symbol = "Li", State = "Solid" });
elements.Add(newElement { AtomicNumber = 4, AtomicWeight = 9.01, Category = "Alkaline Earth Metals", Name
= "Beryllium", Symbol = "Be", State = "Solid" });
elements.Add(newElement { AtomicNumber = 5, AtomicWeight = 10.81, Category = "Non Metals", Name =
"Boron", Symbol = "B", State = "Solid" });
elements.Add(newElement { AtomicNumber = 6, AtomicWeight = 12.01, Category = "Non Metals", Name =
"Carbon", Symbol = "C", State = "Solid" });
elements.Add(newElement { AtomicNumber = 7, AtomicWeight = 14.01, Category = "Non Metals", Name =
"Nitrogen", Symbol = "N", State = "Gas" });
elements.Add(newElement { AtomicNumber = 8, AtomicWeight = 15.999, Category = "Non Metals", Name =
"Oxygen", Symbol = "O", State = "Gas" });
elements.Add(newElement { AtomicNumber = 9, AtomicWeight = 18.998, Category = "Non Metals", Name =
"Fluorine", Symbol = "F", State = "Gas" });
elements.Add(newElement { AtomicNumber = 10, AtomicWeight = 20.18, Category = "Noble Gases", Name =
"Neon", Symbol = "Ne", State = "Gas" });
elements.Add(newElement { AtomicNumber = 11, AtomicWeight = 22.99, Category = "Alkali Metals", Name =
"Sodium", Symbol = "Na", State = "Solid" });
elements.Add(newElement { AtomicNumber = 12, AtomicWeight = 24.31, Category = "Alkaline Earth Metals",
Name = "Magnesium", Symbol = "Mg", State = "Solid" });
elements.Add(newElement { AtomicNumber = 13, AtomicWeight = 26.98, Category = "Other Metals", Name =
"Aluminum", Symbol = "Al", State = "Solid" });
elements.Add(newElement { AtomicNumber = 14, AtomicWeight = 28.09, Category = "Non Metals", Name =
"Silicon", Symbol = "Si", State = "Solid" });
elements.Add(newElement { AtomicNumber = 15, AtomicWeight = 30.97, Category = "Non Metals", Name =
"Phosphorus", Symbol = "P", State = "Solid" });
```

```
elements.Add(newElement { AtomicNumber = 16, AtomicWeight = 32.06, Category = "Non Metals", Name =
"Sulfur", Symbol = "S", State = "Solid" });
elements.Add(newElement { AtomicNumber = 17, AtomicWeight = 35.45, Category = "Non Metals", Name =
"Chlorine", Symbol = "Cl", State = "Gas" });
elements.Add(newElement { AtomicNumber = 18, AtomicWeight = 39.95, Category = "Noble Gases", Name =
"Argon", Symbol = "Ar", State = "Gas" });
elements.Add(newElement { AtomicNumber = 19, AtomicWeight = 39.10, Category = "Alkali Metals", Name =
"Potassium", Symbol = "K", State = "Solid" });
elements.Add(newElement { AtomicNumber = 20, AtomicWeight = 40.08, Category = "Alkaline Earth Metals",
Name = "Calcium", Symbol = "Ca", State = "Solid" });
elements.Add(newElement { AtomicNumber = 21, AtomicWeight = 44.96, Category = "Transitional Metals", Name
= "Scandium", Symbol = "Sc", State = "Solid" });
elements.Add(newElement { AtomicNumber = 22, AtomicWeight = 47.90, Category = "Transitional Metals", Name
= "Titanium", Symbol = "Ti", State = "Solid" });
elements.Add(newElement { AtomicNumber = 23, AtomicWeight = 50.94, Category = "Transitional Metals", Name
= "Vanadium", Symbol = "V", State = "Solid" });
elements.Add(newElement { AtomicNumber = 24, AtomicWeight = 51.996, Category = "Transitional Metals", Name
= "Chromium", Symbol = "Cr", State = "Solid" });
elements.Add(newElement { AtomicNumber = 25, AtomicWeight = 54.94, Category = "Transitional Metals", Name
= "Manganese", Symbol = "Mn", State = "Solid" });
elements.Add(newElement { AtomicNumber = 26, AtomicWeight = 55.85, Category = "Transitional Metals", Name
= "Iron", Symbol = "Fe", State = "Solid" });
elements.Add(newElement { AtomicNumber = 27, AtomicWeight = 58.93, Category = "Transitional Metals", Name
= "Cobalt", Symbol = "Co", State = "Solid" });
elements.Add(newElement { AtomicNumber = 28, AtomicWeight = 58.70, Category = "Transitional Metals", Name
= "Nickel", Symbol = "Ni", State = "Solid" });
elements.Add(newElement { AtomicNumber = 29, AtomicWeight = 63.55, Category = "Transitional Metals", Name
= "Copper", Symbol = "Cu", State = "Solid" });
elements.Add(newElement { AtomicNumber = 30, AtomicWeight = 65.37, Category = "Transitional Metals", Name
= "Zinc", Symbol = "Zn", State = "Solid" });
elements.Add(newElement { AtomicNumber = 31, AtomicWeight = 69.72, Category = "Other Metals", Name =
"Gallium", Symbol = "Ga", State = "Solid" });
elements.Add(newElement { AtomicNumber = 32, AtomicWeight = 72.59, Category = "Other Metals", Name =
"Germanium", Symbol = "Ge", State = "Solid" });
elements.Add(newElement { AtomicNumber = 33, AtomicWeight = 74.92, Category = "Non Metals", Name =
"Arsenic", Symbol = "As", State = "Solid" });
elements.Add(newElement { AtomicNumber = 34, AtomicWeight = 78.96, Category = "Non Metals", Name =
"Selenium", Symbol = "Se", State = "Solid" });
elements.Add(newElement { AtomicNumber = 35, AtomicWeight = 79.90, Category = "Non Metals", Name =
"Bromine", Symbol = "Br", State = "Liquid" });
elements.Add(newElement { AtomicNumber = 36, AtomicWeight = 83.80, Category = "Noble Gases", Name =
"Krypton", Symbol = "Kr", State = "Gas" });
            }
    }
}
```

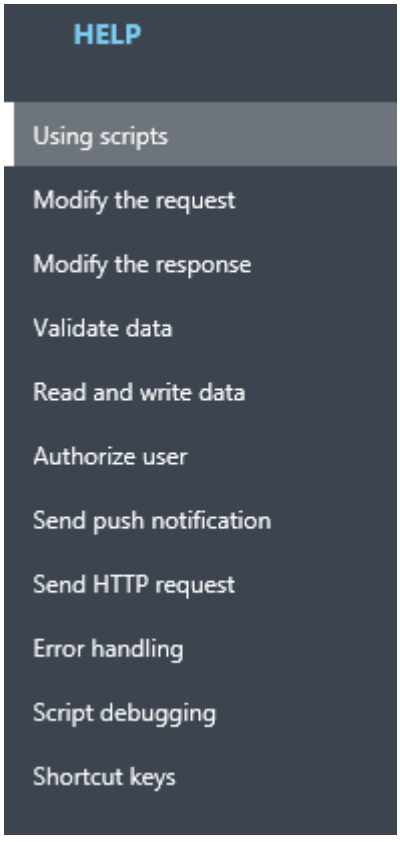
1.11. 现在已经有了数据库的相关事物

下面我们开始处理推送通知。在本例中，我将发送一个 toast 通知，用来显示插入的 element 名字。有许多中方法可以实现，但是，最基本的，我在这里创建了一个数据库事务触发器，当创建推送通知的时候，可以引发这个触发器。

记住，推送通知来自我们的服务，而不是程序本身，所以我们回到 Windows Azure Portal：http://manage.windowsazure.com。打开移动服务，点击 DATA 选项，然后选择你的数据表，我这里是 Element。看到表之后，选择 SCRIPT 选项，如下：



默认情况下，会看到一个 INSERT 脚本。当有记录发送到数据库中，这个 JavaScript 代码每次都会被执行。现在我们来修改这个脚本。如果你单击右下角的？按钮，会看到这里有不同事物的指南。如下图：



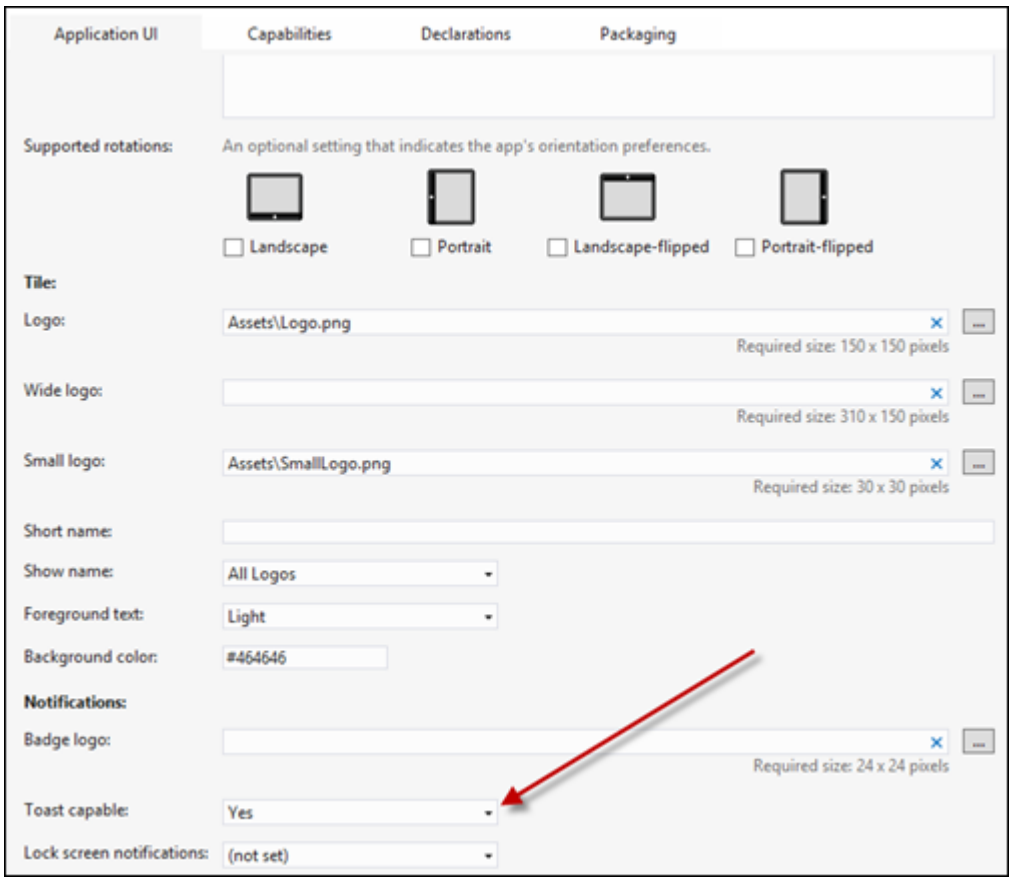
现在我通过修改这个 INSERT 脚本，如果记录插入成功，则发送一个推送通知。

```
function insert(item, user, request) {
request.execute({
success: function() {
request.respond();
push.wns.sendToastText02(item.channel, {
                        text1: "(" + item.AtomicNumber + ") " + item.Name + " - " + item.Category,
                        text2: item.Symbol + " | " + item.AtomicWeight + " | " + item.State + " | "
                    }, {
success: function(pushResponse) {
console.log("PUSH sent:", pushResponse);
                        }
                    });
            }
        });
}
```

在第 10 日中介绍的 Toast 通知:说了我们可以使用的 Toast 模板有许多种。这里的代码就是使用了其中的模板——ToastText02，我设置了 Text1 和 Text2，然后在我的日子文件中做了一个记录。

1.12. 现在就发送推送通知吧

最后一件事情 ,我们必须更新一下 package.appxmanifest 文件 :支持 toast 消息。打开 package.appxmanifest 文件，在支持 Toast 通知上选择“Yes”。如果不这样做的话，你永远都不会看到发送到你机器中的 toast 消息。



当做好这个修改之后，每当你点击程序中的按钮时，你不仅将数据插入到了 Azure 数据库中，还发送了一个 Toast 通知到你的机器上。很神奇吧。

1.13. 总结

今天，我们学习了如何利用 Windows Azure Mobile Services 来发送推送通知。如果你创建的程序涉及到在线数据存储，推送通知甚至是多个客户端，比如 iOS，Android 或者其它，这将是一个非常宝贵的工具。

点击下图，下载本文示例代码：

明天，我将介绍一个程序的生命周期，以及用户可以打开，返回到程序的许多方法。到时候见！

感谢你的阅读！

如果对这篇文章有什么想法，可以与破船联系，破船的联系方式在文章开头。

破船