# Events App Template

The events app template is a completely functional WP8 application that sources the eventful.com api for event information. As provided the template app is an events app for Austin Texas. With a few changes and your own azure account you can customize the template to create event apps for other cities, or use the template as a guide to add events to your apps. The server side portion may also be retrofitted to proxy to other API providers.

Customizing for a different city requires to major steps:

1. Create an Azure Website that proxies to eventful.
2. Customize the app so that it links to your azure website.

There are some finishing touches left undone, such as notifications and live tiles, that you may want to add to your <% city %> events app.

## Create Your Azure Website

The simplest way to provide an API proxy on azure is with an Azure Web Site. They are easy to create and manage and virtually free for low utilization scenarios.

### Step 1: Create Azure Website

1. Visit github.com and fork the project at <https://github.com/tacowan/eventful-proxy>
   1. Create an account on github if you do not already have one.
   2. Navigate to <https://github.com/tacowan/eventful-proxy> and click the “fork” button.
2. From within http://manage.windowsazure.com select New -> Compute -> Web Site -> Quick Create. Henceforth I will refer to the site’s name (the subdomain) as YOURSITENAME. For example, if your website is foo.azurewebsites.net, then YOURSITENAME = foo.
3. Navigate to you newly created site in the azure console. From within the Quick start page select “Set up deployment from source control”. Choose “github”, allow it to access your github account, and connect it to the eventful-proxy fork created in 1.

### Step 2: Customize your new website with your own secret keys

The eventful api proxy needs an evenful.com api key. It also needs to access an azure storage account where we’ll be caching some fairly static result sets to improve your apps responsiveness.

1. Visit <http://api.eventful.com> and create and account and get an api key.
2. If you don’t have one, create a storage account in azure.
3. Navigate to the “CONFIGURE” tab for your new web site (within the azure console). You’ll create 3 connection strings of type custom. Connection strings are managed about half way down within the CONFIGURE screen
4. Create custom connection string called **EVENTFUL\_KEY**. Set its value to your eventful api key.
5. Create custom connection string called **ACCOUNT\_NAME**. Set its value to your storage account name.
6. Create custom connection string called **ACCOUNT\_KEY**. Set its value to your storage account secret key.

### Step 3: Verify Things are working

You can verify things are working by issuing some api requests. Anything sent to <http://YOURSITENAME.azurewebsites.net/eventful> will be proxied to eventful with your key added. Anything sent to <http://YOURSITENAME.azurewebsites.net/cache> will be saved as a BLOB in your azure storage account. The following should return results to your browser:

<http://YOURSITENAME.azurewebsites.net/eventful/rest/events/search>? (returns error from eventful)

<http://YOURSITENAME.azurewebsites.net/eventful/rest/events/search?location=austin> (returns a result set)

Now you can merely use the api.eventful.com documentation to send requests and play with the api. You might also like to take a look at server.js in the gihub project to see how it’s done and see how you could easily add proxies to other services. Let’s also verify your storage account settings are correct. Issue the following request:

<http://YOURSITENAME.azurewebsites.net/cache/rest/events/search?location=austin&sort_order=popularity&page_size=8&blobname=popular>

Now navigate to you azure storage account. There you will find a new container called “events” and within it a file called “popular”. If you are not finding a new container or blob you may have a mistake in your ACOUNT\_NAME or ACCOUNT\_KEY values.

## Modify the WP8 Visual Studio Project

1. Change App.mawssubdomain to your Windows Azure Web Site name. (not the full domain, just the first part before azurewebsites.net. This is in the file called App.xaml.cs.
2. Change App.MY\_BLOB\_CONTAINER to the full public path to your events blob container. It’ll be in the format of <http://youraccount.blob.core.windows.net/events/>.
3. Now we are going to create 5 Windows Azure Scheduled events using the scheduler. Each of these should be configured to run at least once a day. You can easily see within the MainPage.xaml.cs file how these blobs will provide data for the panorama page. You can also verify you are getting data by changing “cache” to “eventful” to see the results returned to your browser if you like.
   1. Create the music feed. Create an HTTP GET event using the scheduler service using your own azure website but following this format. Make sure you change “location=austin” to the city you are targeting for your app.

<http://YOURSITENAME.azurewebsites.net/cache/rest/events/search?location=austin&sort_order=popularity&page_size=50&category=music&blob_name=music>

* 1. Create the family feed

<http://YOURSITENAME.azurewebsites.net/cache/rest/events/search?location=austin&sort_order=popularity&page_size=50&category=family_fun_kids&blobname=family_fun_kids>

* 1. Create the sports feed

<http://YOURSITENAME.azurewebsites.net/cache/rest/events/search?location=austin&sort_order=popularity&page_size=50&category=sports&blobname=sports>

* 1. Create the performing arts feed.

<http://YOURSITENAME.azurewebsites.net/cache/rest/events/search?location=austin&sort_order=popularity&page_size=50&category=performing_arts&blobname=performing_arts>

* 1. Create the venues feed. Notice that this one is slightly different and uses a different eventful rest call for the venues. As before change the city name. You may also modify the page size to get more or less items to show on the panorama.

<http://YOURSITENAME.azurewebsites.net/cache/rest/venues/search?sort_order=popularity&location=austin&page_size=20&blobname=topvenues>