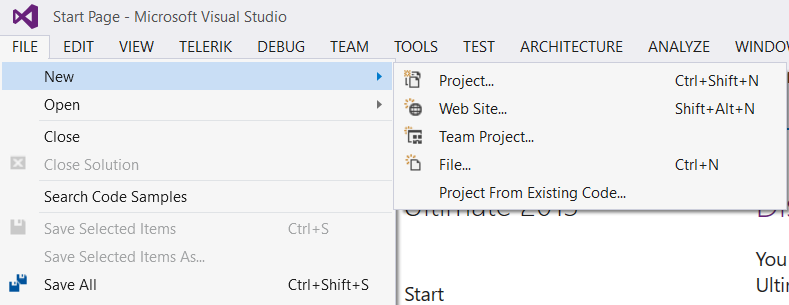
**Writing WebJobs using the SDK**

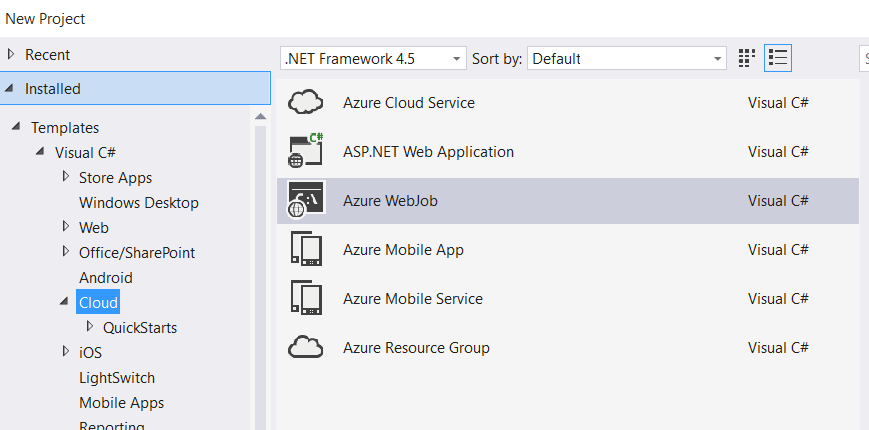
With the WebJobs SDK, the work of connecting the logic of the console application to Azure blobs and queues becomes little more than a matter of decorating your public static meth­ods in your console application with parameter level attributes (the attributes precede the parameter name and type in the method signature). With these attributes, the WebJobs SDK knows to invoke your methods so they run based on the appearance of a blob in Blob storage or a message in a queue and then just as easily output a new blob or message as a result of the invocation. WebJobs SDK handles the triggering of your decorated methods, and binding takes care of passing you a reference to the blob or message as desired.

|  |  |
| --- | --- |
| **Example Attribute** | **Meaning** |
| [BlobTrigger(“input/{blob­Name}”)] | Triggers the method to run when a new blob is detected within the input container. The name of the blob is captured (much like for MVC routes) into blobName and can be reused later by the Blob attribute as well as by adding a parameter to the method named blobName. |
| [QueueTrigger(“queueName”)] | Triggers the method to run when a new message appears in the queue indicated. |
| [Blob(“output/{blobName}”, FileAccess.Write)] | Provides access to the blob identified by the parameter. Often used with the optional FileAccess.Write parameter to output the results of the method to the indicated blob. |
| [Queue(“queueName”)] | Provides access to the queue identified by the queue name, particularly to add a new message to the queue. |

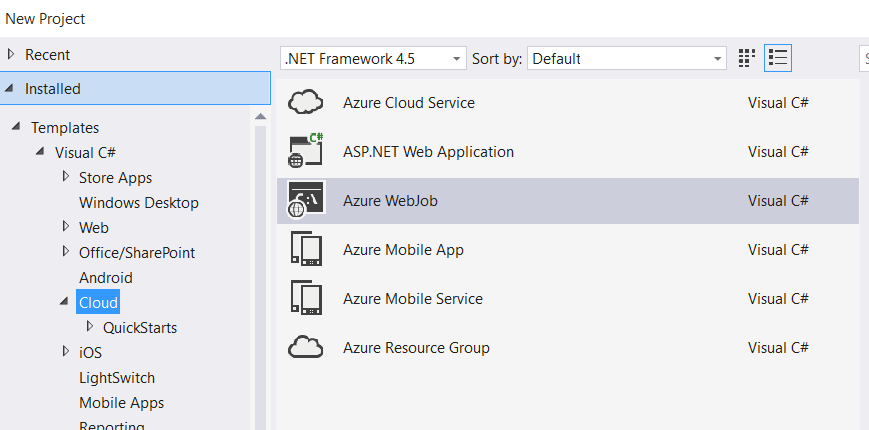
1. Within Visual Studio, click File, New Project.



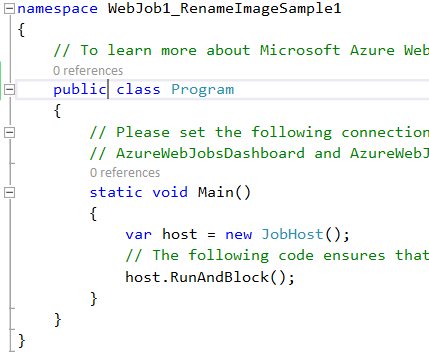
1. In the New Project dialog box navigation pane, expand Installed, Templates, Visual C#, and then click Cloud.



1. In the list of templates, select Microsoft Azure WebJob.

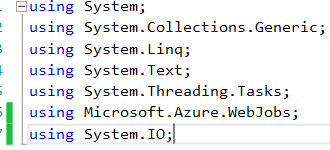


1. Name your project, and click OK.
2. Open Program.cs.
3. Modify the class Program so that it is “public”. If you do not make the class public, WebJobs will not detect and run your operations.



1. Inside Program.cs, add a reference

*using System.IO;*



1. Next, add a method to be invoked by WebJobs as shown below. In this case, it “listens” for files with the .png extension being added in the “input” container of Blob storage and then re-writes the file with a new file name to the “ouput” container, where the new file name is the original .png file name without the extension, prefixed with “processed\_” and suffixed with “\_final.png”.

public static void RenameFile(

[BlobTrigger("input/{blobName}.png")] Stream input,

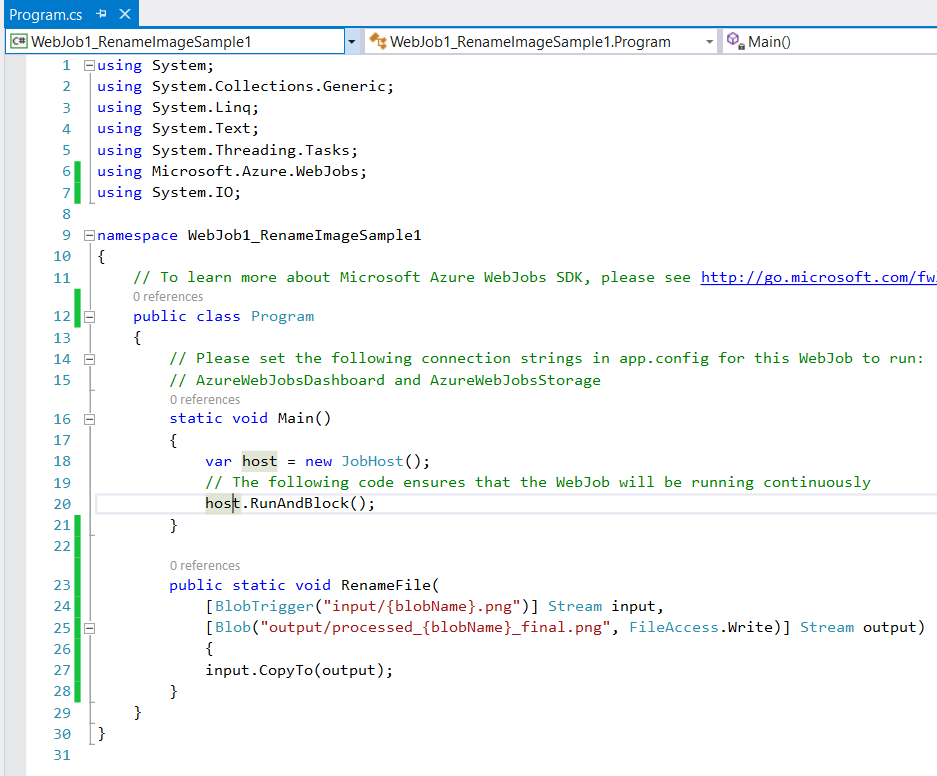
[Blob("output/processed\_{blobName}\_final.png", FileAccess.Write)] Stream output)

{

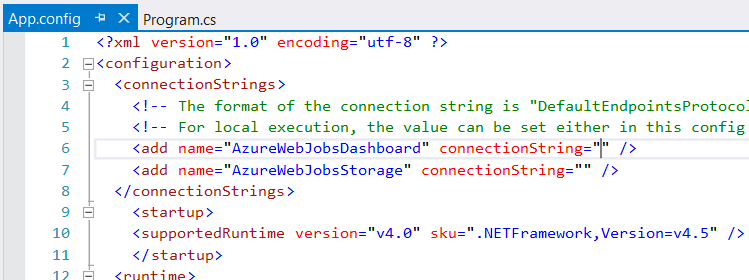
input.CopyTo(output);

}

Final Program.cs file

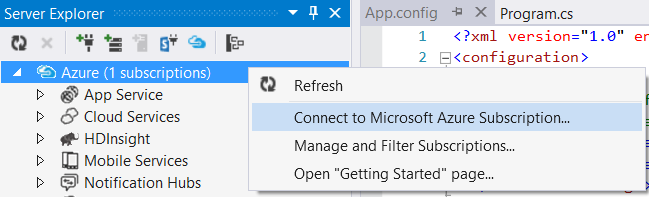


9. To run this WebJob locally, you need to apply some local configuration. Open the App.config file and add connection strings for the canonical storage account connection strings expected by WebJobs SDK, AzureWebJobsDashboard, and AzureWebJobsStorage as shown below. Be sure to replace name and key values with the values from your own storage account.



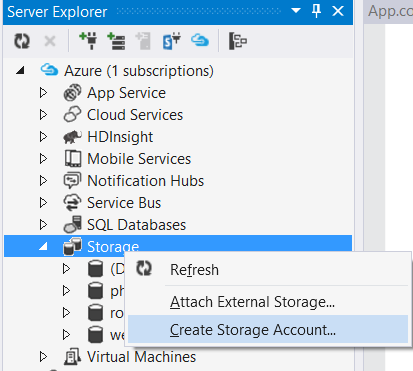
10. Create Storage Container from visual Studio

Start “Server Explorer” from View menu

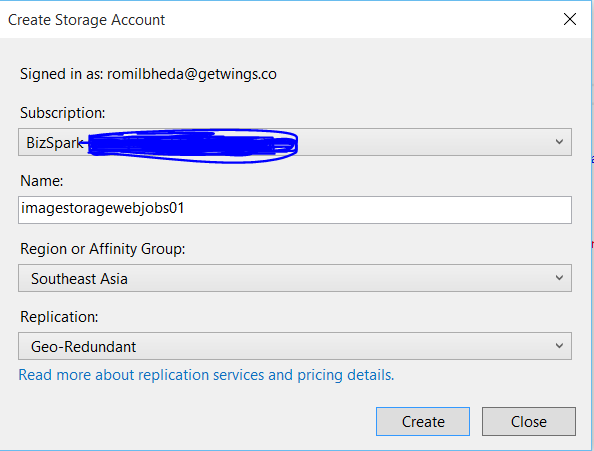


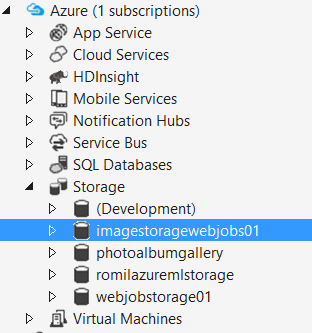
11. Right-click on “Azure Subscription” & select “Connect to Microsoft Azure Subscription…” option. With the help of this step Azure account connect with Visual Studio.

Right-click on “Storage” -> “Create Storage Account…”

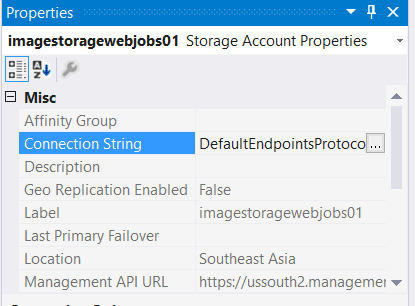


12. Enter Storage Name & choose region from the list

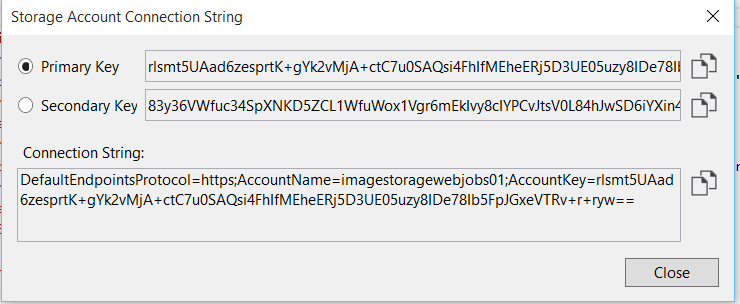




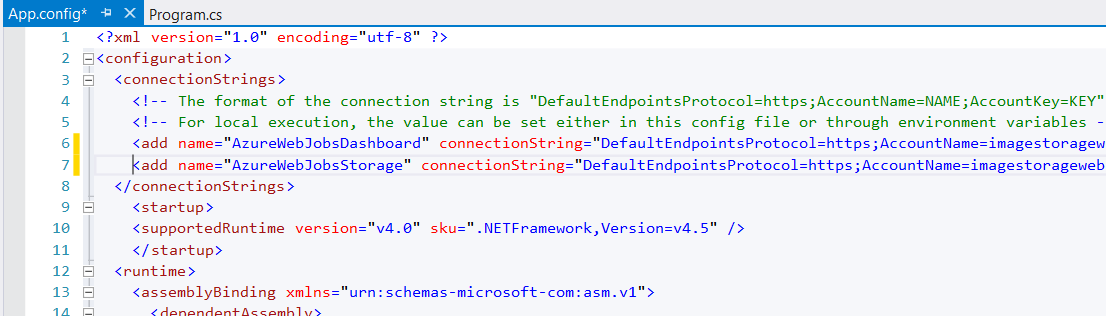
13. Select new create azure storage & open properties window



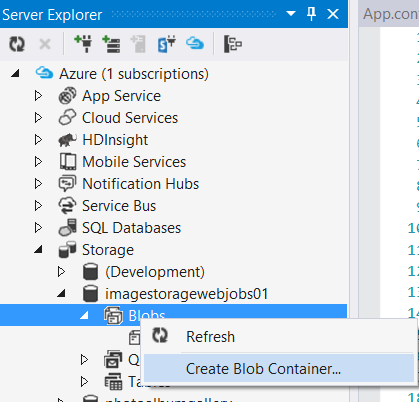
Copy full Connection String



Paste connection string in both line.

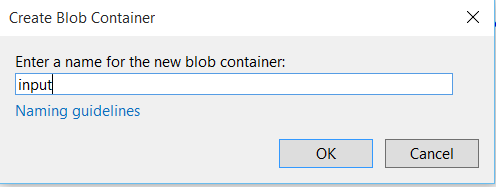


14. Open New storage -> select Blobs -> Create Blob Container

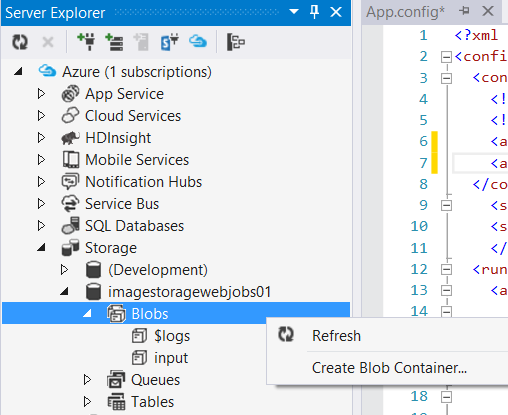
’

Enter blob container name.

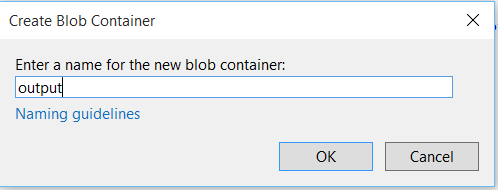
Example. input



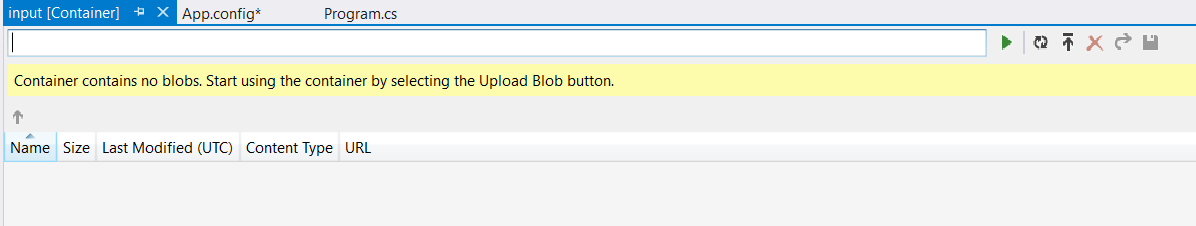
Again click on Blob & create new blob



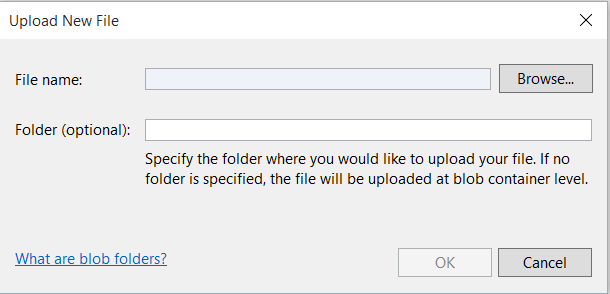
Enter another blob storage. Such as output

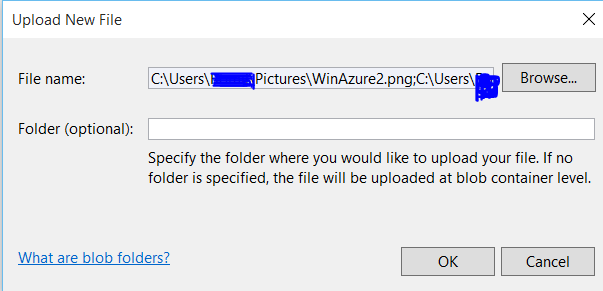


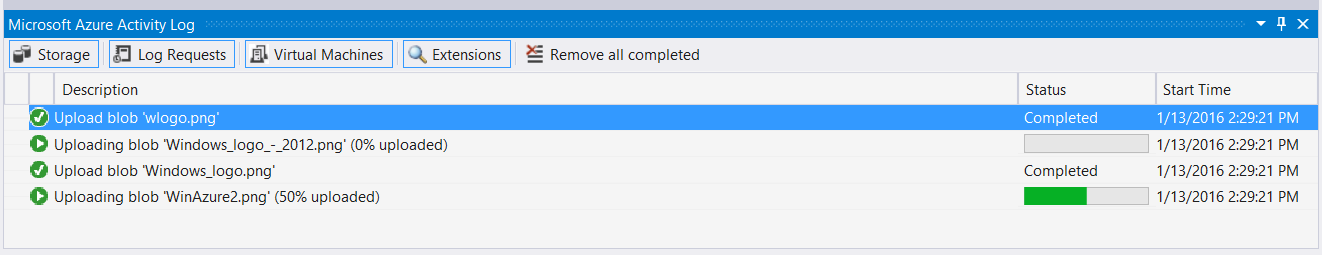
Upload a few .png files to process the input blob container. One way to do this is to double-click the container in Server Explorer, and in the document window, click the Upload Blob button (which is the third button from the left in the toolbar at the top of the document window), and select the .png files to upload.

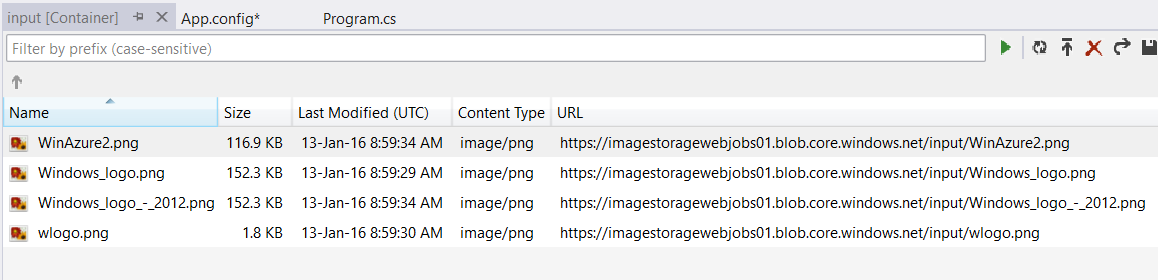


Browse the .png files









Finally, press F5 or click the Debug menu, Start Debugging. The console window should appear with an indicator of “Job host started.” About 10 to 15 seconds later, you should see diagnostic traces indicating “Executing: ‘Program.RenameFile because New blob detected.”

**15.** To confirm the files were copied to the output container, open the container in the tool of your choice (you can use the Server Explorer approach for this also).

