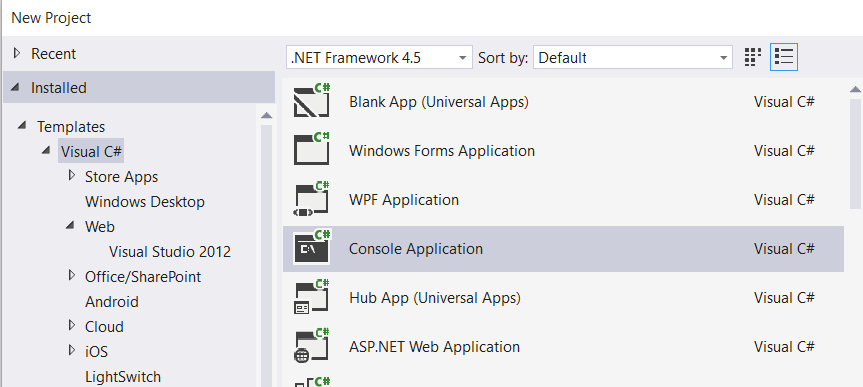
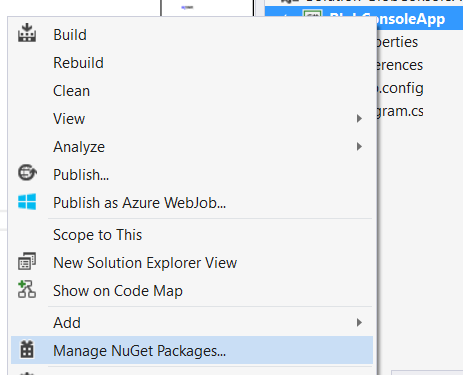
**Azure Storage Queue – Console App**

The Azure Storage Queue service provides a mechanism for reliable inter-application mes­saging to support asynchronous distributed application workflows. This section covers a few fundamental features of the Queue service for adding messages to a queue, processing those messages individually or in a batch, and scaling the service.

Step 1: Create New Console App from Visual Studio Template

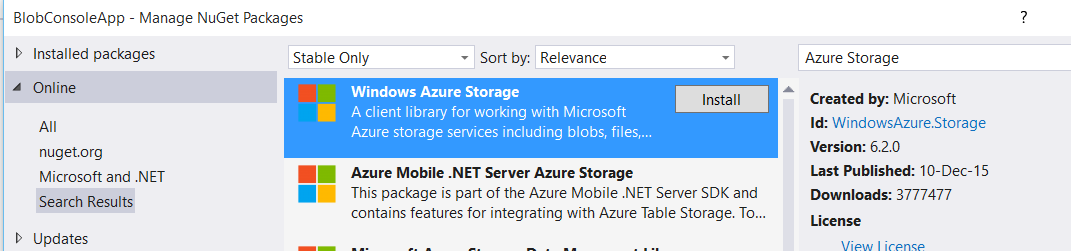


Step 2: Right click on Project name & select “Manage NuGet Packages…”



Step 3: Manage NuGet Packages dialog box will open & search for “Azure Storage”

Install “Windows Azure Storage”



Step 4: Open app.config file, add an entry under the Configuration element, replacing the account name and key with your own storage account details:

<?xml version="1.0" encoding="utf-8" ?>

<configuration>

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />

</startup>

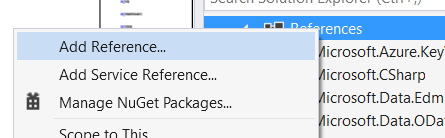
<appSettings>

<add key="StorageConnectionString" value="DefaultEndpointsProtocol=https;AccountName=storageaccountname;AccountKey=storagekey" />

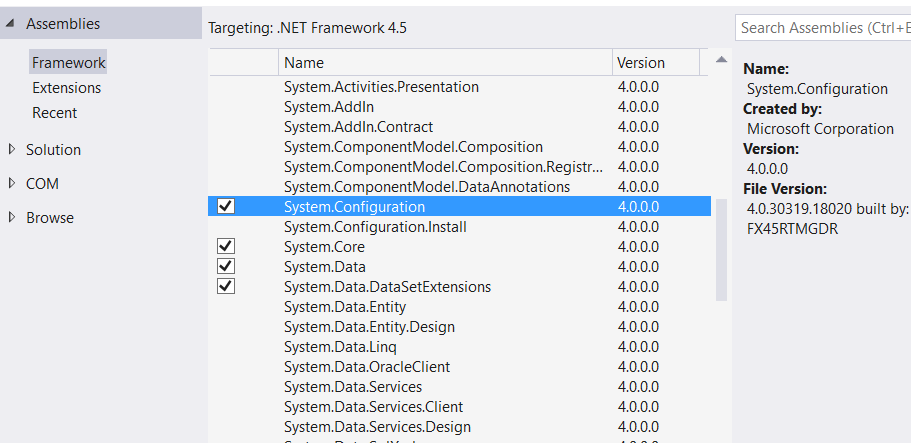
</appSettings>

</configuration>

Step 5: Add one reference



Select “System.Configuration”



Step 6: Open Program.cs file

Add references of Azure Storage

using System.Configuration;

using Microsoft.WindowsAzure.Storage;

using Microsoft.WindowsAzure.Storage.Auth;

using Microsoft.WindowsAzure.Storage.Queue;

static void Main(string[] args)

{

//Retrieve Storage account from connection string

CloudStorageAccount storageAccount = CloudStorageAccount.Parse(ConfigurationManager.AppSettings["StorageConnectionString"]);

//Get a reference to the Queue client

CloudQueueClient queueClient = storageAccount.CreateCloudQueueClient();

//Get a reference to a Queue object

CloudQueue queue = queueClient.GetQueueReference("storagequeue");

queue.CreateIfNotExists();

}

Output : storagequeue will generate

**Adding messages to a queue**

You can access your storage queues and add messages to a queue using many storage browsing tools; however, it is more likely you will add messages programmatically as part of your application workflow.

static void Main(string[] args)

{

//Retrieve Storage account from connection string

CloudStorageAccount storageAccount = CloudStorageAccount.Parse(ConfigurationManager.AppSettings["StorageConnectionString"]);

//Get a reference to the Queue client

CloudQueueClient queueClient = storageAccount.CreateCloudQueueClient();

//Get a reference to a Queue object

CloudQueue queue = queueClient.GetQueueReference("storagequeue");

queue.CreateIfNotExists();

queue.AddMessage(new CloudQueueMessage("Message #1"));

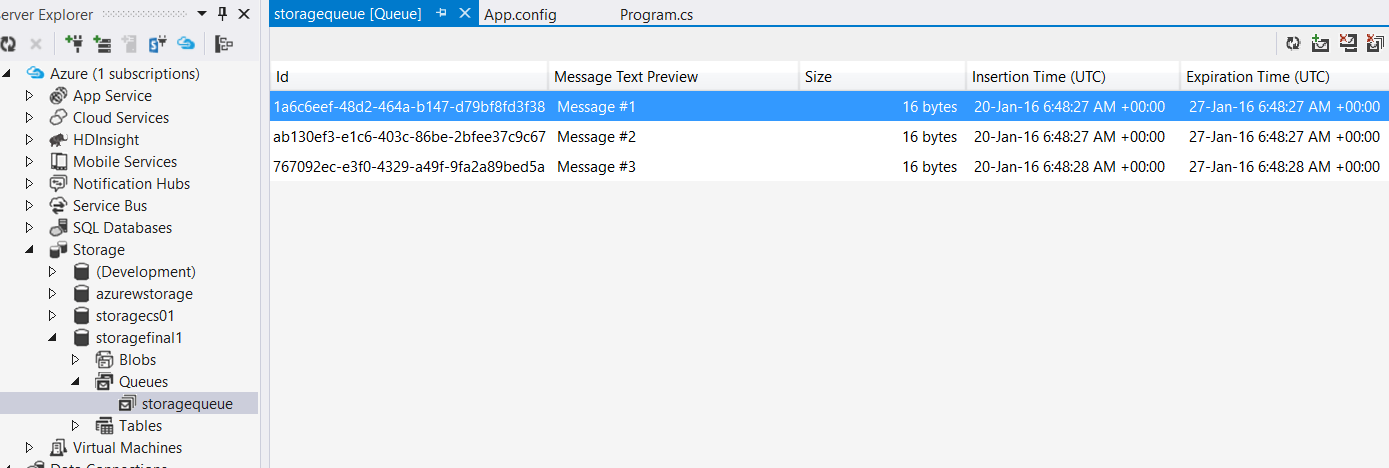
queue.AddMessage(new CloudQueueMessage("Message #2"));

queue.AddMessage(new CloudQueueMessage("Message #3"));

}

Run the project

Server Explorer -> Azure Subscription -> Storage -> StorageName -> Queues



**Peek messages**

Add one more reference

using System.Diagnostics;

static void Main(string[] args)

{

//Retrieve Storage account from connection string

CloudStorageAccount storageAccount = CloudStorageAccount.Parse(ConfigurationManager.AppSettings["StorageConnectionString"]);

//Get a reference to the Queue client

CloudQueueClient queueClient = storageAccount.CreateCloudQueueClient();

//Get a reference to a Queue object

CloudQueue queue = queueClient.GetQueueReference("storagequeue");

queue.CreateIfNotExists();

//queue.AddMessage(new CloudQueueMessage("Message #1"));

//queue.AddMessage(new CloudQueueMessage("Message #2"));

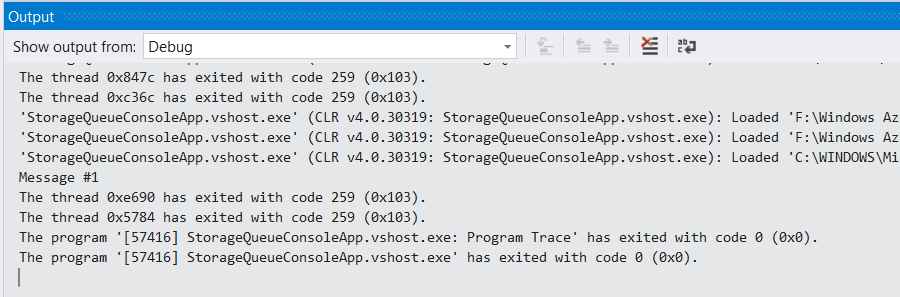
//queue.AddMessage(new CloudQueueMessage("Message #3"));

CloudQueueMessage peekedMessage = queue.PeekMessage();

Debug.WriteLine(peekedMessage.AsString);

}

Run the Project



**Peek messages – multiple messages**

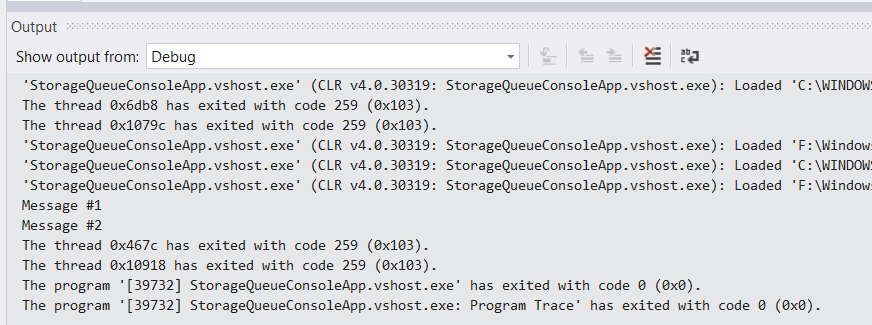
CloudQueueMessage message1 = queue.GetMessage();

CloudQueueMessage message2 = queue.GetMessage();

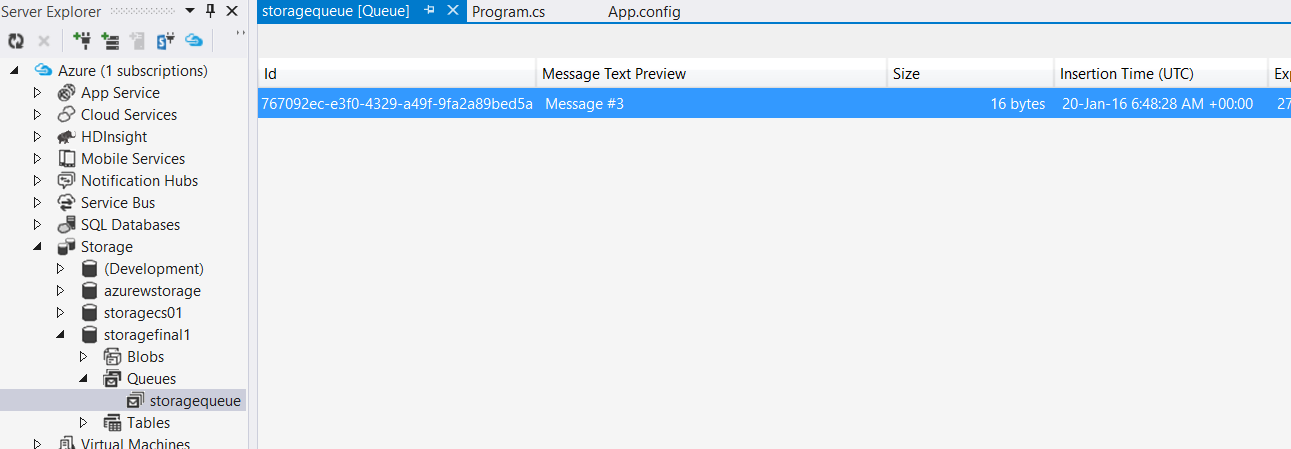
Debug.WriteLine(message1.AsString);

Debug.WriteLine(message2.AsString);

again run the project

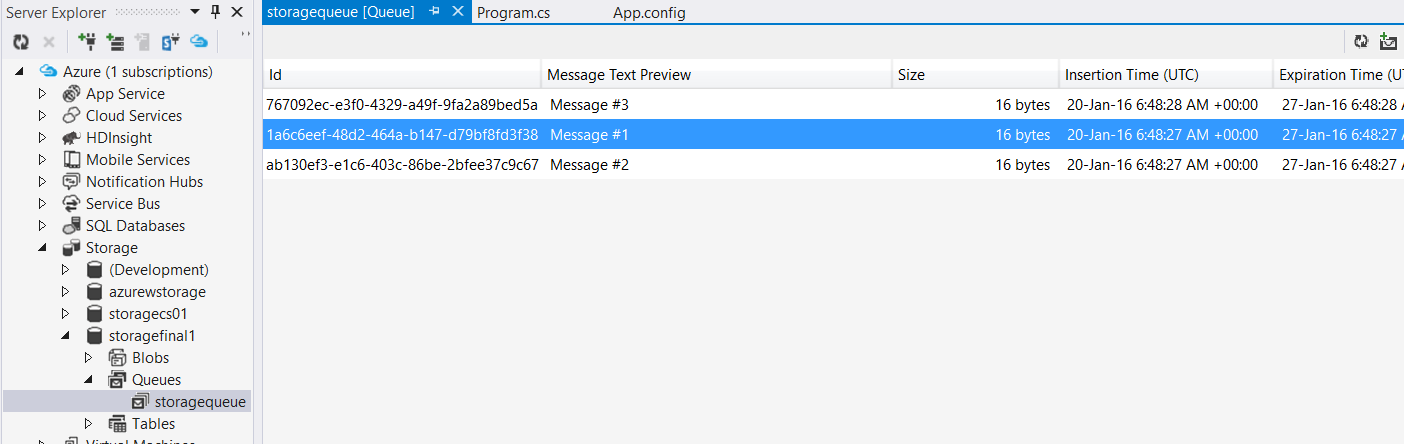


Open Queues from Azure Storage Explorer. So only Message #3 available



Click on refresh button.

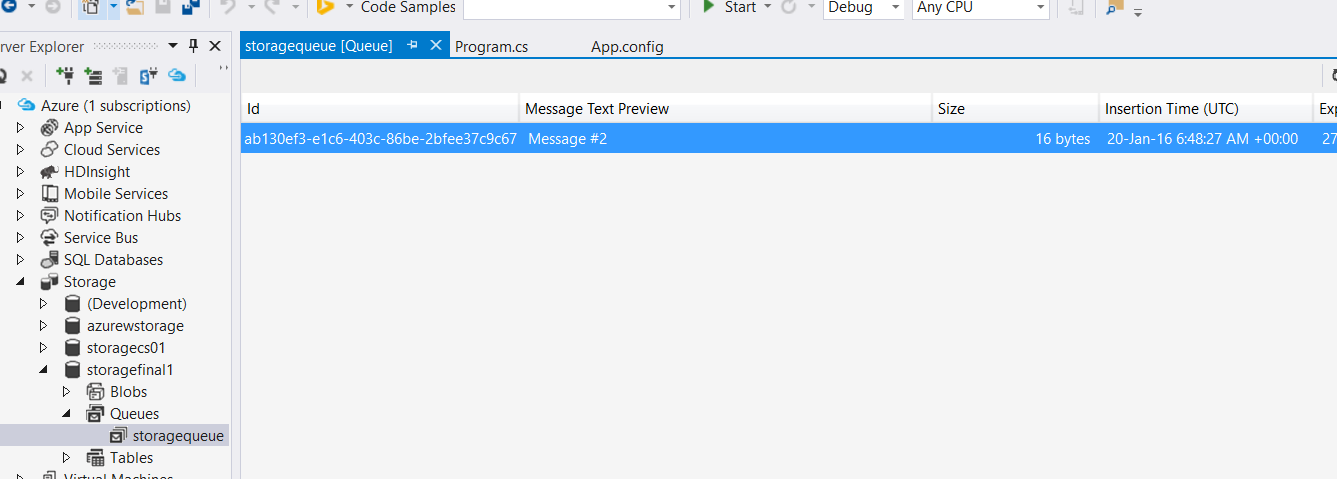
Message #1, Message #2 available in the queue



**Deleting Message**

queue.DeleteMessage(message1);

Again click on refresh button – Message #2



Again click on refresh button – Message #2, Message #1

