**Lesson12 develop Azure Batch Service Job through .Net SDK 9.0**

**Notes: -**

**1-we see that we install 2 NUGET packages**

**(the other packages installed automatically when the 2 packages installed)**

**Microsoft.Azure.Batch 9.0**

**Microsoft.Azure.Storage 9.0**

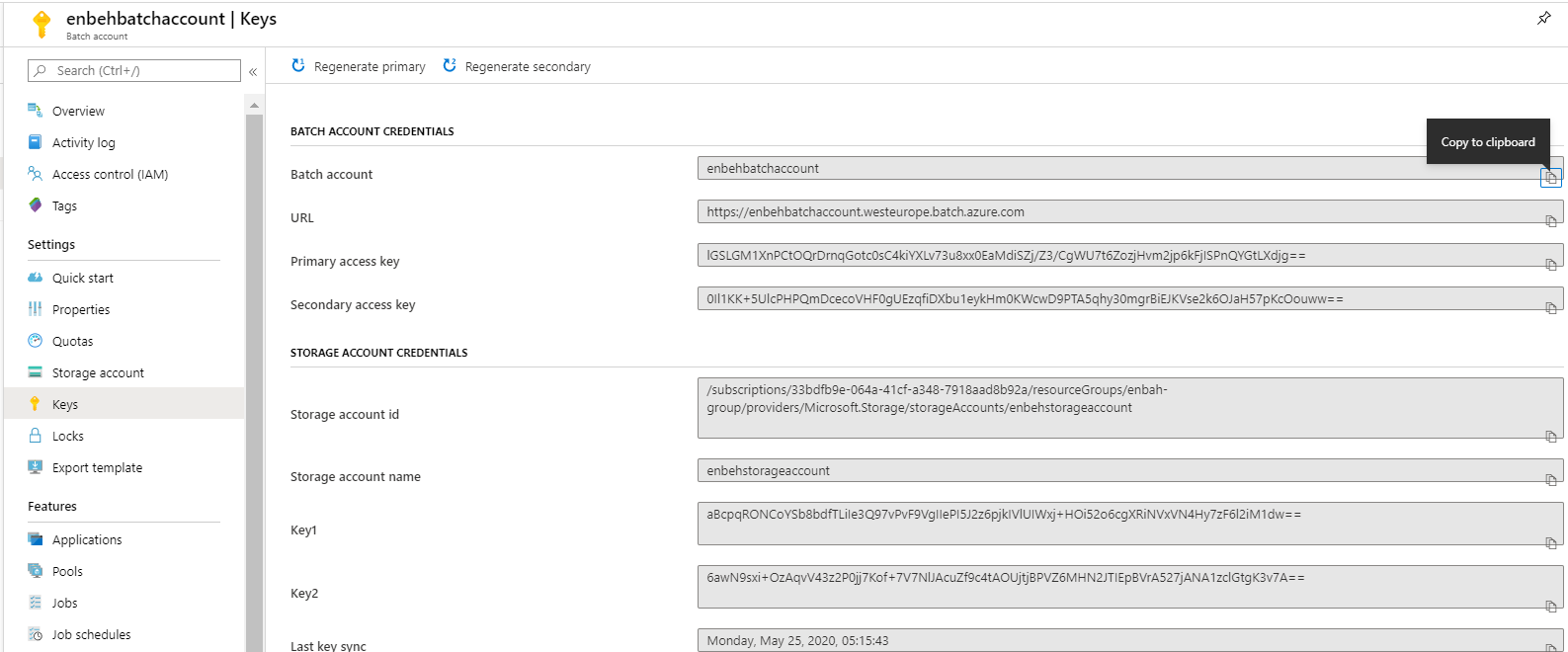
**2-this sample project it will make the following :-**

**A-create Pool**

**B-create job**

**C-create task assigned for this pool under this job**

**(to get the keys for storage and batch account ,on batch account > keys)**



**Example:-**

**using Microsoft.Azure.Batch;**

**using Microsoft.Azure.Batch.Auth;**

**using Microsoft.Azure.Batch.Common;**

**using Microsoft.Azure.Storage;**

**using Microsoft.Azure.Storage.Blob;**

**using System;**

**using System.Collections.Generic;**

**using System.IO;**

**using System.Linq;**

**using System.Text;**

**using System.Threading.Tasks;**

**namespace batchdemo{**

**class Program{**

**// Add the batch account crdentials here**

**private const string demo\_batchAccountName = "enbehbatchaccount";**

**private const string demo\_batchAccountKey = "lGSLGM1XnPCtOQrDrnqGotc0sC4kiYXLv73u8xx0EaMdiSZj/Z3/CgWU7t6ZozjHvm2jp6kFjISPnQYGtLXdjg==";**

**private const string demo\_batchAccountUrl = "https://enbehbatchaccount.westeurope.batch.azure.com";**

**// Here add the storage account details**

**private const string demo\_storageAccountName = "enbehstorageaccount";**

**private const string demo\_storageAccountKey = "aBcpqRONCoYSb8bdfTLiIe3Q97vPvF9VgIIePI5J2z6pjkIVlUIWxj+HOi52o6cgXRiNVxVN4Hy7zF6l2iM1dw==";**

**// These are general values required for the batch service**

**private const string PoolId = "ffmpegpool";**

**private const string jobID = "video\_processor";**

**private const string demo\_packageid = "ffmpeg";**

**private const string demo\_packageversion = "3";**

**static void Main(string[] args){**

**try{CoreAsync().Wait();}**

**finally{**

**Console.WriteLine();**

**Console.WriteLine("Program complete");**

**Console.ReadLine();}}**

**private static async Task CoreAsync(){**

**//we create batch account credential to batch url , batch account name , batch account access key**

**BatchSharedKeyCredentials demo\_sharedKeyCredentials = new BatchSharedKeyCredentials(demo\_batchAccountUrl, demo\_batchAccountName, demo\_batchAccountKey);**

**//we assign the batch credential to batch client to access to batch account**

**using (BatchClient demo\_batchClient = BatchClient.Open(demo\_sharedKeyCredentials)){**

**// This method is used to create the pool of virtual machines**

**//await PoolCreation(demo\_batchClient, PoolId);**

**//await JobCreation(demo\_batchClient, jobID, PoolId);**

**await TaskCreation(demo\_batchClient, jobID);}}**

**private static async Task PoolCreation(BatchClient p\_batchClient, string p\_poolId){**

**Console.WriteLine("Creating the pool of virtual machines");**

**try{**

**//we define the image o.s**

**ImageReference demo\_image = new ImageReference(**

**publisher: "MicrosoftWindowsServer",**

**offer: "WindowsServer",**

**sku: "2016-Datacenter",**

**version: "latest");**

**VirtualMachineConfiguration demo\_configuration =**

**new VirtualMachineConfiguration(**

**imageReference: demo\_image,**

**nodeAgentSkuId: "batch.node.windows amd64");**

**CloudPool demo\_pool = null;**

**//it will create 1 dedicated pool of type Standard\_A1\_V2**

**demo\_pool = p\_batchClient.PoolOperations.CreatePool(**

**poolId: p\_poolId,**

**targetDedicatedComputeNodes: 1,**

**targetLowPriorityComputeNodes: 0,**

**virtualMachineSize: "STANDARD\_D1\_v2",**

**virtualMachineConfiguration: demo\_configuration);**

**//we assign the pool with package defined with id and version**

**demo\_pool.ApplicationPackageReferences = new List<ApplicationPackageReference>{**

**new ApplicationPackageReference{**

**ApplicationId = demo\_packageid,**

**Version = demo\_packageversion}};**

**//it will apply and create the pool**

**await demo\_pool.CommitAsync();}**

**catch (BatchException pool\_error){Console.WriteLine(pool\_error.Message);}}**

**private static async Task JobCreation(BatchClient p\_batchClient, string p\_jobId, string p\_poolId){**

**//we get all jobs and check if the job exist or not to delete if exist and create new one**

**var lst = p\_batchClient.JobOperations.ListJobs().ToList();**

**var isExist = lst.Where(x => x.Id == p\_jobId).FirstOrDefault();**

**if (isExist != null){p\_batchClient.JobOperations.DeleteJob(p\_jobId);}**

**Console.WriteLine("Creating the job");**

**//it will create job with id and assign pool to it**

**CloudJob demo\_job = p\_batchClient.JobOperations.CreateJob();**

**demo\_job.Id = p\_jobId;**

**demo\_job.PoolInformation = new PoolInformation { PoolId = p\_poolId };**

**//it will apply the configuration**

**await demo\_job.CommitAsync();}**

**private static async Task TaskCreation(BatchClient p\_batchClient, string p\_jobId){**

**Console.WriteLine("Creating the Task");**

**string taskId = "demotask";**

**string container\_name = "inputs";**

**string l\_blobName = "1280.mp4";**

**string storageConnectionString = String.Format("DefaultEndpointsProtocol=https;AccountName={0};AccountKey={1}",**

**demo\_storageAccountName, demo\_storageAccountKey);**

**CloudStorageAccount l\_storageAccount = CloudStorageAccount.Parse(storageConnectionString);**

**//it will create cloud blob client to access to blob**

**CloudBlobClient l\_blobClient = l\_storageAccount.CreateCloudBlobClient();**

**//its reference from azure batch account,**

**List<ResourceFile> resourceFiles = new List<ResourceFile>();**

**//it will get the container reference**

**CloudBlobContainer l\_container = l\_blobClient.GetContainerReference(container\_name);**

**//it will get the blob block reference to get the URL and apply SASS**

**CloudBlockBlob l\_blobData = l\_container.GetBlockBlobReference(l\_blobName);**

**//it will define the credential for blob expire time and permission to applied for blob called 1280.mp4**

**SharedAccessBlobPolicy l\_sasConstraints = new SharedAccessBlobPolicy{**

**SharedAccessExpiryTime = DateTime.UtcNow.AddHours(2),**

**Permissions = SharedAccessBlobPermissions.Read | SharedAccessBlobPermissions.List};**

**//it will generate sass token and create sass url = sass token + container url**

**string sasToken = l\_container.GetSharedAccessSignature(l\_sasConstraints);**

**string l\_blobSasUrl = String.Format("{0}{1}", l\_blobData.Uri, sasToken);**

**//whith the SASS we can add blob SASS URL to resource files to inform the azure batch service where to get files**

**resourceFiles.Add(new ResourceFile(l\_blobSasUrl, l\_blobName));**

**//we design the command for AZ\_BATCH\_APP\_PACKAGE**

**string appPath = String.Format("%AZ\_BATCH\_APP\_PACKAGE\_{0}#{1}%", demo\_packageid, demo\_packageversion);**

**string taskCommandLine = String.Format("cmd /c {0}\\ffmpeg.exe -i {1} -vn -acodec copy audio.aac", appPath, l\_blobName);**

**//we define AZ Batch task with define task Id and command line and resoruce files**

**CloudTask task = new CloudTask(taskId, taskCommandLine);**

**task.ResourceFiles = resourceFiles;**

**//it will add task for batch service**

**await p\_batchClient.JobOperations.AddTaskAsync(p\_jobId, task);}}}**