DEMO

EXAMS DUMPS.co

Instant Download After Purchase

100% Money Back Guarantee

90 Days Free Updates

PDF FILE FORMAT

MICROSOFT AZ-203

Q&As

2020 Latest EXAMSDUMPS AZ-203 PDF Dumps Download. Following Questions and Answers are all new published by Microsoft Official Exam Center

Latest AZ-203 Dumps

AZ-203 Practice Test

AZ-203 Study Guide

Pass MICROSOFT AZ-203 Exam with 100% Guarantee

Free Download Real Questions & Answers PDF and VCE file from:

Microsoft AZ-203

Developing Solutions for Microsoft Azure

Topic 1 - Question Set 1

Question #1*Topic 1*

HOTSPOT -
You have an Azure Batch project that processes and converts files and stores the files in Azure storage. You are developing a function to start the batch job.
You add the following parameters to the function.

Parameter name	Description
`fileTasks`	a list of tasks to be run
`jobId`	the identifier that must be assigned to the job
`outputContainerSasUrl`	a storage SAS URL to store successfully converted files
`failedContainerSasUrl`	a storage SAS URL to store copies of files that failed to convert.

You must ensure that converted files are placed in the container referenced by the outputContainerSasUrl parameter. Files which fail to convert are places in the container referenced by the failedContainerSasUrl parameter.
You need to ensure the files are correctly processed.
How should you complete the code segment? To answer, select the appropriate options in the answer area.
NOTE: Each correct selection is worth one point.

Hot Area:

Answer Area

```
public list<CloudTasks> StartTasks(List<FileTask> fileTasks, string jobId,
 string outputContainerSasUrl, string failedContainerSasUrl)
{
  BatchSharedKeyCredentials sharedKeyCredentials =
   new BatchSharedKeyCredentials(batchAccountUrl, batchAccountName,
batchAccountKey);
 List<CloudTask> tasks = new List<CloudTask>();
 using (BatchClient batchClient = BatchClient.Open(sharedKeyCredentials))
{
   CloudJob = batchClient.JobOperations. [          ] ();
                                          GetJob
                                          GetTask
                                          EnableJob
                                          CreateJob
        job.Id = jobId,
        job.PoolInfromation = new PoolInformation { PoolId = poolId };
        job.Commit();
        fileTasks.ForEach((fileTask) =>
        {
          string taskId = $"Task{DateTime.Now.ToFileTimeUtc().ToString()}";
          CloudTask task = new CloudTask (taskId, fileTask.Command);
          List<OutputFile> outputFileList = new List<OutputFile>();
          OutputFileBlobContainerDestination outputContainer =
           new OutputFileBlobContainerDestination(outputContainerSasUrl);
        OutputFileBlobContainerDestination failedContainer =
           new OutputFileBlobContainerDestination (failedContainerSasUrl);
        outputFileList.Add(new OutputFile(fileTask.Output,
           new OutputFileDestination(outputContainer),
           new OutputFileUploadOptions(OutputFileUploadCondition. [          ] )));
                                                                  TaskSuccess
                                                                  TaskFailure
                                                                  TaskCompletion
          outputFileList.Add(new OutputFile(fileTask.Output,
           new OutputFileDestination(failedContainer),
           new OutputFileUploadOptions(OutputFileUploadCondition [          ] )));
```

Answer Area

```
public list<CloudTasks> StartTasks(List<FileTask> fileTasks, string jobId,
 string outputContainerSasUrl, string failedContainerSasUrl)
{
  BatchSharedKeyCredentials sharedKeyCredentials =
   new BatchSharedKeyCredentials(batchAccountUrl, batchAccountName,
batchAccountKey);
 List<CloudTask> tasks = new List<CloudTask>();
 using (BatchClient batchClient = BatchClient.Open(sharedKeyCredentials))
{
   CloudJob = batchClient.JobOperations. [ GetJob | GetTask | EnableJob | CreateJob ] ();
        job.Id = jobId,
        job.PoolInfromation = new PoolInformation { PoolId = poolId };
        job.Commit();
        fileTasks.ForEach((fileTask) =>
        {
          string taskId = $"Task{DateTime.Now.ToFileTimeUtc().ToString()}";
          CloudTask task = new CloudTask (taskId, fileTask.Command);
          List<OutputFile> outputFileList = new List<OutputFile>();
          OutputFileBlobContainerDestination outputContainer =
           new OutputFileBlobContainerDestination(outputContainerSasUrl);
        OutputFileBlobContainerDestination failedContainer =
           new OutputFileBlobContainerDestination (failedContainerSasUrl);
        outputFileList.Add(new OutputFile(fileTask.Output,
           new OutputFileDestination(outputContainer),
           new OutputFileUploadOptions(OutputFileUploadCondition. [ TaskSuccess | TaskFailure | TaskCompletion ] )));
          outputFileList.Add(new OutputFile(fileTask.Output,
           new OutputFileDestination(failedContainer),
           new OutputFileUploadOptions(OutputFileUploadCondition. [ ] )));
```

Correct Answer:

Box 1: CreateJob -

Box 2: TaskSuccess -

TaskSuccess: Upload the file(s) only after the task process exits with an exit code of 0.

Incorrect: TaskCompletion: Upload the file(s) after the task process exits, no matter what the exit code was.

Box 3: TaskFailure -

TaskFailure:Upload the file(s) only after the task process exits with a nonzero exit code.

Box 4: OutputFiles -

To specify output files for a task, create a collection of OutputFile objects and assign it to the CloudTask.OutputFiles property when you create the task.

References:

https://docs.microsoft.com/en-us/dotnet/api/microsoft.azure.batch.protocol.models.outputfileuploadcondition
https://docs.microsoft.com/en-us/azure/batch/batch-task-output-files