using System.Linq;

using System.Net;

using System.Net.Http;

using System.Threading.Tasks;

using Microsoft.Azure.WebJobs;

using Microsoft.Azure.WebJobs.Extensions.Http;

using Microsoft.Azure.WebJobs.Host;

using DocumentFormat.OpenXml.Packaging;

using System.Xml;

using Microsoft.Azure; // Namespace for Azure Configuration Manager

using Microsoft.WindowsAzure.Storage; // Namespace for Storage Client Library

using Microsoft.WindowsAzure.Storage.Blob; // Namespace for Azure Blobs

using Microsoft.WindowsAzure.Storage.File; // Namespace for Azure Files

using System;

namespace FunctionApp1

{

public static class Function1

{

[FunctionName("Function1")]

public static async Task<HttpResponseMessage> Run([HttpTrigger(AuthorizationLevel.Function, "get", "post", Route = null)]HttpRequestMessage req, TraceWriter log)

{

log.Info("C# HTTP trigger function processed a request.");

// parse query parameter

string name = req.GetQueryNameValuePairs()

.FirstOrDefault(q => string.Compare(q.Key, "name", true) == 0)

.Value;

// Get request body

dynamic data = await req.Content.ReadAsAsync<object>();

//GetFile();

try

{

log.Info("going to download the file");

GetFile();

log.Info("file downloaded");

var value = WDRetrieveCustomProperty(@"D:\home\site\wwwroot\Scopeexampleemail.docx", "firstcustomproperty");

log.Info("C# HTTP trigger value returned :"+ value);

}

catch(Exception ex)

{

log.Info("C# HTTP trigger function error in request."+ex.Message);

}

// Set name to query string or body data

name = name ?? data?.name;

return name == null

? req.CreateResponse(HttpStatusCode.BadRequest, "Please pass a name on the query string or in the request body")

: req.CreateResponse(HttpStatusCode.OK, "Hello " + name);

}

public static string WDRetrieveCustomProperty(string docName, string propertyName)

{

const string customPropertiesSchema = "http://schemas.openxmlformats.org/officeDocument/2006/custom-properties";

const string customVTypesSchema = "http://schemas.openxmlformats.org/officeDocument/2006/docPropsVTypes";

string propertyValue = string.Empty;

using (WordprocessingDocument wdPackage = WordprocessingDocument.Open(docName, true))

{

// Get the custom properties part (custom.xml).

CustomFilePropertiesPart customPropertiesPart = wdPackage.CustomFilePropertiesPart;

// There may not be a custom properties part.

if (customPropertiesPart != null)

{

// Manage namespaces to perform XML XPath queries.

NameTable nt = new NameTable();

XmlNamespaceManager nsManager = new XmlNamespaceManager(nt);

nsManager.AddNamespace("d", customPropertiesSchema);

nsManager.AddNamespace("vt", customVTypesSchema);

// Get the properties from the package.

XmlDocument xdoc = new XmlDocument(nt);

// Load the XML in the part into an XmlDocument instance.

xdoc.Load(customPropertiesPart.GetStream());

string searchString = string.Format("d:Properties/d:property[@name='{0}']", propertyName);

XmlNode xNode = xdoc.SelectSingleNode(searchString, nsManager);

if ((xNode != null))

{

propertyValue = xNode.InnerText;

}

else{

propertyValue = " property not found";

}

}

}

return propertyValue;

}

public static void GetFile()

{

CloudStorageAccount storageAccount = CloudStorageAccount.Parse("DefaultEndpointsProtocol=https;AccountName=wordfilestorage;AccountKey=aGFqITmemKL6q+5e9X10alYsSOxzd6VWrc1cLyy8hXI4fXJ4nEeYoBOPulio6ApBsyyPh+3PuyCO2UvSgAVWhQ==;EndpointSuffix=core.windows.net");

CloudFileClient fileClient = storageAccount.CreateCloudFileClient();

// Get a reference to the file share we created previously.

CloudFileShare share = fileClient.GetShareReference("worddocsfileshare");

// Ensure that the share exists.

if (share.Exists())

{

// Get a reference to the root directory for the share.

CloudFileDirectory rootDir = share.GetRootDirectoryReference();

// Get a reference to the directory we created previously.

//CloudFileDirectory sampleDir = rootDir.GetDirectoryReference("CustomLogs");

// Ensure that the directory exists.

if (rootDir.Exists())

{

// Get a reference to the file we created previously.

CloudFile file = rootDir.GetFileReference("Scopeexampleemail.docx");

// Ensure that the file exists.

if (file.Exists())

{

file.DownloadToFile(@"D:\home\site\wwwroot\Scopeexampleemail.docx", System.IO.FileMode.Create);

// var value = WDRetrieveCustomProperty(file.StorageUri.PrimaryUri.AbsoluteUri.ToString(), "firstcustomproperty");

// Write the contents of the file to the console window.

//Console.WriteLine(file.DownloadTextAsync().Result);

}

}

}

}

}

}

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

//Here is the method which will give us the property value on passing the property name and document path

//this will require to provide reference to couple of dlls :

// using DocumentFormat.OpenXml.Packaging; //by dll DocumentFormat.OpenXml

// using System.Xml;  //by System.Xml/dll

// System.IO.Compression.FileSystem

        public static string WDRetrieveCustomProperty(string docPathTillName, string propertyName)

        {

            const string customPropertiesSchema = "<http://schemas.openxmlformats.org/officeDocument/2006/custom-properties>";

            const string customVTypesSchema = "<http://schemas.openxmlformats.org/officeDocument/2006/docPropsVTypes>";

            string propertyValue = string.Empty;

            using (WordprocessingDocument wdPackage = WordprocessingDocument.Open(docPathTillName, true))

            {

                // Get the custom properties part (custom.xml).

                CustomFilePropertiesPart customPropertiesPart = wdPackage.CustomFilePropertiesPart;

                // There may not be a custom properties part.

                if (customPropertiesPart != null)

                {

                    // Manage namespaces to perform XML XPath queries.

                    NameTable nt = new NameTable();

                    XmlNamespaceManager nsManager = new XmlNamespaceManager(nt);

                    nsManager.AddNamespace("d", customPropertiesSchema);

                    nsManager.AddNamespace("vt", customVTypesSchema);

                    // Get the properties from the package.

                    XmlDocument xdoc = new XmlDocument(nt);

                    // Load the XML in the part into an XmlDocument instance.

                    xdoc.Load(customPropertiesPart.GetStream());

                    string searchString = string.Format("d:Properties/d:property[@name='{0}']", propertyName);

                    XmlNode xNode = xdoc.SelectSingleNode(searchString, nsManager);

                    if ((xNode != null))

                    {

                        propertyValue = xNode.InnerText;

                    }

                    else{

                        propertyValue = " property not found";

                    }

                }

            }

            return propertyValue;

        }

//we can call it like :

// var value = WDRetrieveCustomProperty(@"D:\home\site\wwwroot\Scopeexampleemail.docx", "firstcustomproperty");