Class OneDrive.xpp

1. Main properties & methods.
2. Usage examples.
3. Class content.
4. Additional Document: <https://docs.microsoft.com/en-us/onedrive/developer>
5. Main properties & methods:

* Properties:
* JsonResult (string): Raw json returned by OneDrive.
* Token (string): token used to authentication to OneDrive.
* Methods:
* void GetCurrentUserProfile(): Get currently logged in user information.
* void GetRootFolder(): Get root folder information.
* CreateFile(str \_parentId,str \_filename, Stream \_binaryStream): Upload file <4MB to OneDrive, \_parentId – the id of parent folder, \_filename – the file name of the file after being uploaded, \_binaryStream – the stream of the file.
* void Move(str \_movingItemId,str \_destinationId): Move file/folder to another folder, \_movingItemId – the id of the file/folder being moved, \_destinationId – the id of the destination folder.
* void ListChildren(str \_parentId): Get information of all files/folders inside a specific folder, \_parentId – the id of folder.
* void Rename(str \_itemId,str \_newName): Rename a file/folder, \_itemId – the id of the item being renamed, \_newName – the new name.
* Stream GetFileStream(str \_downloadLink): Get System.IO.Stream of a file, \_downloadLink – the download link of a file, the download link is in the file information after calling ListChildren for example.
* UpdateItemMetadata(str \_itemId,JObject \_metadata): Update metada of a file/folder, \_itemId – the id of the file/folder being updated, \_metadata is the json string of metadata. For the list of metadata, see: <https://docs.microsoft.com/en-us/onedrive/developer/rest-api/resources/drive> and <https://docs.microsoft.com/en-us/onedrive/developer/rest-api/resources/driveitem>

1. Usage Examples:

* Create OneDrive instance:

str token=”token\_of\_OneDrive”;

OneDrive oneDrive=new OneDrive(token);

Upload file:

System.IO.StreamReader sr=new System.IO.StreamReader("C:\\dddd\\test.txt");

OneDrive.CreateFile("014D3C2EN6Y2GOVW7725BZO354PWSELRRZ","postman.txt",sr.BaseStream);

info(OneDrive.JsonResult); //Get result

1. Class content:

using System.Net;

using System.IO;

using Newtonsoft.Json.Linq;

using Newtonsoft.Json;

public class OneDrive

{

public str Token;

public str JsonResult;

private WebRequest \_request;

private WebResponse \_response;

private StreamReader \_streamReader;

private Stream \_requestStream;

private Stream \_responseStream;

private WebHeaderCollection \_headers;

public void new(str \_token)

{

Token=\_token;

}

public void GetCurrentUserProfile()

{

this.\_SetRequestAddress("https://graph.microsoft.com/v1.0/me/drive");

this.\_SetRequestMethod("GET");

this.\_SendRequest();

}

public void GetRootFolder()

{

this.\_SetRequestAddress("https://graph.microsoft.com/v1.0/me/drive/root");

this.\_SetRequestMethod("GET");

this.\_SendRequest();

}

public void CreateFile(str \_parentId,str \_filename,Stream \_binaryStream)

{

this.\_SetRequestAddress(strFmt("https://graph.microsoft.com/v1.0/me/drive/items/%1:/%2:/content",\_parentId,\_filename));

this.\_SetRequestMethod("PUT");

this.\_SetContentType("binary");

this.\_SetBodyBinary(\_binaryStream);

this.\_SendRequest();

}

public void Move(str \_movingItemId,str \_destinationId)

{

JObject json=new JObject();

JObject parentJson=new JObject();

parentJson.Add("id",\_destinationId);

json.Add("parentReference",parentJson);

this.UpdateItemMetadata(\_movingItemId,json);

}

public void ListChildren(str \_parentId)

{

this.\_SetRequestAddress(strFmt("https://graph.microsoft.com/v1.0/me/drive/items/%1/children",\_parentId));

this.\_SetRequestMethod("GET");

this.\_SendRequest();

}

public void Rename(str \_itemId,str \_newName)

{

JObject json=new JObject();

json.Add("name",\_newName);

this.UpdateItemMetadata(\_itemId,json);

}

public Stream GetFileStream(str \_downloadLink)

{

\_request=WebRequest::Create(\_downloadLink);

this.\_SetRequestMethod("GET");

try

{

\_response=\_request.GetResponse();

}

catch(Exception::CLRError)

{

System.Exception ex=CLRInterop::getLastException() as System.Reflection.TargetInvocationException;

System.Net.WebException webEx=ex.InnerException as System.Net.WebException;

\_response=webEx.Response;

}

return \_response.GetResponseStream();

}

public void UpdateItemMetadata(str \_itemId,JObject \_metadata)

{

this.\_SetRequestAddress(strFmt("https://graph.microsoft.com/v1.0/me/drive/items/%1",\_itemId));

this.\_SetRequestMethod("PATCH");

this.\_SetContentType("application/json");

this.\_SetBodyText(\_metadata.ToString());

this.\_SendRequest();

}

private void \_SetRequestMethod(str \_method)

{

\_request.Method=\_method;

}

private void \_SetContentType(str \_contentType)

{

\_request.ContentType=\_contentType;

}

private void \_SetRequestAddress(str \_address)

{

\_request=WebRequest::Create(\_address);

\_headers=new WebHeaderCollection();

this.\_SetHeader("Authorization",Token);

\_request.Headers=\_headers;

}

private void \_SetBody(str \_bodyKey,str \_bodyValue)

{

\_requestStream=\_request.GetRequestStream();

System.Byte[] bodyBytes;

System.Text.Encoding encoding=System.Text.Encoding::get\_UTF8();

str body=strFmt("&%1=%2",\_bodyKey,\_bodyValue);

bodyBytes=encoding.GetBytes(body);

\_requestStream.Write(bodyBytes,0,bodyBytes.Length);

\_requestStream.Flush();

}

private void \_SetBodyText(str \_body)

{

\_requestStream=\_request.GetRequestStream();

System.Byte[] bodyBytes;

System.Text.Encoding encoding=System.Text.Encoding::get\_UTF8();

bodyBytes=encoding.GetBytes(\_body);

\_requestStream.Write(bodyBytes,0,bodyBytes.Length);

\_requestStream.Flush();

}

private void \_SetBodyBinary(System.IO.Stream inputStream)

{

\_requestStream=\_request.GetRequestStream();

int currentByte=inputStream.ReadByte();

while(currentByte!=-1)

{

\_requestStream.WriteByte(currentByte);

currentByte=inputStream.ReadByte();

}

inputStream.Close();

}

private void \_SetHeader(str \_headerKey,str \_headerValue)

{

\_headers.Add(\_headerKey,\_headerValue);

}

private void \_SendRequest()

{

if(\_requestStream!=null)

{

\_requestStream.Flush();

}

try

{

\_response=\_request.GetResponse();

}

catch(Exception::CLRError)

{

System.Exception ex=CLRInterop::getLastException() as System.Reflection.TargetInvocationException;

System.Net.WebException webEx=ex.InnerException as System.Net.WebException;

\_response=webEx.Response;

}

\_responseStream=\_response.GetResponseStream();

\_streamReader=new System.IO.StreamReader(\_responseStream);

JsonResult=\_streamReader.ReadToEnd();

if(\_requestStream!=null)

{

\_requestStream.Close();

}

\_streamReader.Close();

\_responseStream.Close();

}

}