**Features**

1. Supports HTTP request
2. Support HTTP response

**Requirements**

Microsoft ***.NET Framework 4.6.2*** or higher, Microsoft ***Visual C++ 2017 Redistributable (x86).*** It is yet to be tested with lower versions of the above.

**Basic**

1. Create a ***Webrequest*** instance by calling ***Create*** with the URI of the resource.

|  |
| --- |
| String a = "https://sandbox.safaricom.co.ke/oauth/v1/generate?grant\_type=client\_credentials";  String baseUrl = a;  HttpWebRequest request = (HttpWebRequest)WebRequest.Create(baseUrl); |
|  |  |

**Note**

The .NET Framework provides protocol-specific classes derived from ***WebRequest*** and ***WebResponse*** for URIs that begin with "http:" or "https:'' To access resources using other protocols, you must implement protocol-specific classes that derive from ***WebRequest*** and ***WebResponse***.

1. Set any property values that you need in the ***WebRequest***. For example,

* to enable authentication,

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | String token = "ACCESS\_TOKEN";  request.Headers.Add("authorization", "Bearer " + token); | |  |  | |

* Set the method to be used mostly GET or POST.

In daraja .When generating the access token the GET method is used while the POST is used with almost the rest of the other APIs.

request.Method = "POST"; or request.Method = "GET";

* To set the content type to be used in daraja json is used.

request.ContentType = "application/json";

Generate the API Body depending on the Api parameter requirements eg for b2c we will use :

string json = "{\"InitiatorName\":\" here\"," +

"\"SecurityCredential\":\" here \"," +

"\"CommandID\":\" here \"," +

"\"Amount\":\" here \"," +

"\"PartyA\":\" here \"," +

"\"PartyB\":\" here \"," +

"\"Remarks\":\" here \"," +

"\"QueueTimeOutURL\":\"http://your\_timeout\_url\"," +

"\"ResultURL\":\"http://your\_result\_url\"," +

"\"Occasion\":\" here \"}";

Replace “here ” with actual data ,avoid leaving spaces before and after inserting the actual data .

Write the json string to a Stream which will be sent along with your request.

using (var streamWriter = new StreamWriter(request.GetRequestStream()))

{

//the string (json) should be here

|  |
| --- |
| streamWriter.Write(json);  streamWriter.Flush();  streamWriter.Close(); |
|  |  |
|  |  |

}

1. Call ***GetResponse*** . The actual type of the returned ***WebResponse*** object is determined by the scheme of the requested URI.

HttpWebResponse response = (HttpWebResponse)request.GetResponse();

**Note**

After you are finished with a ***WebResponse*** object, you must close it by calling the Close method.

response.Close();

Alternatively, if you have gotten the response stream from the response object, you can close the stream by calling the Stream.close method. If you do not close either the response or the stream, your application can run out of connections to the server and become unable to process additional requests.

1. To get the stream containing response data sent by the server, use the [GetResponseStream](https://docs.microsoft.com/en-us/dotnet/api/system.net.httpwebresponse.getresponsestream) method of the ***WebResponse****.*

Stream receiveStream = response.GetResponseStream();

After reading the data from the response, you must either close the response stream using the ***Stream. Close*** readStream.Close(); method or close the response using the ***WebResponse.Close*** method response.Close();. It is not necessary to call the **Close** method on both the response stream and the ***WebResponse****,* but doing so is not harmful.

**How to use ?**

1. Put data to the API request body as stated above example.
2. Put the actual token to "ACCESS\_TOKEN";
3. Use your application consumer key for this field "YOUR\_APP\_CONSUMER\_KEY"
4. Use your application consumer secret for this field "YOUR\_APP\_CONSUMER\_SECRET".

**Development**

1. Visual Studio 2017 & .NET Framework 4.6.2 are required.
2. It is recommended to share your idea on the Issue Board before you start to work, especially for feature development.