CustomURIParser User Guide

Version: 1.0

Document Date: 08/04/2018

Author: Tansel Arif

1. Introduction

The Custom URI Parser is a library which can be embedded into larger projects to parse presented URI's.

2. Example Usage

a. Simple Usage

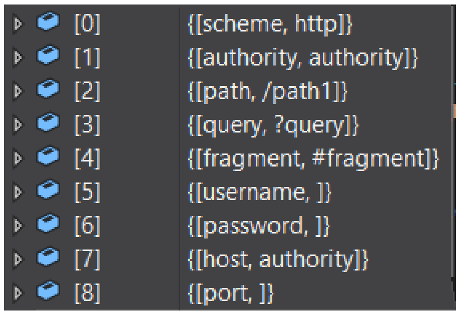
Instantiate as a URIParser object:

URIParser MyParser = new URIParser();

This sets the parser up as an absolute URI parser. This means that any parsing functionality will assume the URI passed in is in absolute format.

The following will then return a Dictionary with string key-value pairs depicting the components.

MyParser.parseUri("http://authority/path1?query#fragment");



```
[0]    {[scheme, http]}
[1]    {[authority, authority]}
[2]    {[path, /path1]}
[3]    {[query, ?query]}
[4]    {[fragment, #fragment]}
[5]    {[username, ]}
[6]    {[password, ]}
[7]    {[host, authority]}
[8]    {[port, ]}
```

b. Making use of relative URI input

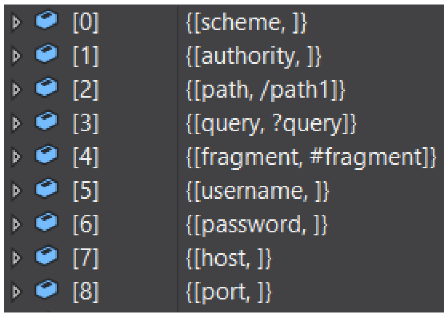
If we want to use the parser for parsing relative URIs, we create an instance using the isAbsolute flag:

URIParser MyParser = new URIParser(false);

This sets the parser up as relative URI parser. This means that any parsing functionality will assume the URI passed in is in relative format. Since no absolute reference is given in this case, a default absolute reference is chosen as “http://authority”.

The following will then return a Dictionary with string key-value pairs depicting the components without consideration of the default absolute URI.

MyParser.parseUri(“/path1?query#fragment”);

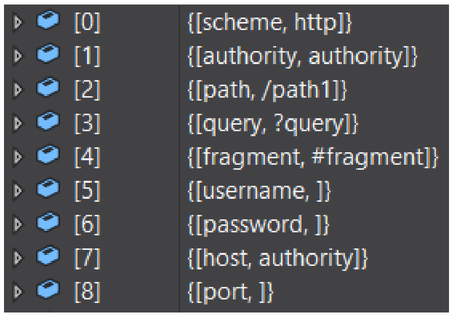


```
[0]    {[scheme, ]}
[1]    {[authority, ]}
[2]    {[path, /path1]}
[3]    {[query, ?query]}
[4]    {[fragment, #fragment]}
[5]    {[username, ]}
[6]    {[password, ]}
[7]    {[host, ]}
[8]    {[port, ]}
```

We can specify an absolute URI for reference during instantiation:

URIParser MyParser = new URIParser(false, “http://authority”);

`The above usage will then consider the absolute path as well in the response:`



```
[0]    {[scheme, http]}
[1]    {[authority, authority]}
[2]    {[path, /path1]}
[3]    {[query, ?query]}
[4]    {[fragment, #fragment]}
[5]    {[username, ]}
[6]    {[password, ]}
[7]    {[host, authority]}
[8]    {[port, ]}
```

c. Validating a URI

The user may want to simply validate whether a URI is valid or not. This method can be used to carry out the validation according to how the parser was set up.

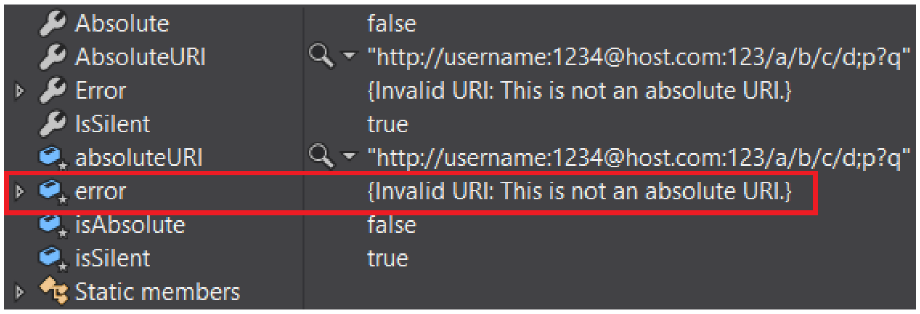
For example, set the parser up as a relative parser with an absolute reference URI of “http://`username:1234@host.com:123/path2?Query#fragment`”:

URIParser MyParser = new URIParser(false, “http://username:1234@host.com:123/path2?Query#fragment”);

And then validating a URI of the form “path?query#fragment” while specifying as an absolute validation type:

MyParser.validateUri("path?query#fragment", URIParser.validationType.Absolute);

Will return as false. The embedded error message will then give more information (highlighted in red below):



```
Absolute          false
AbsoluteURI       "http://username:1234@host.com:123/a/b/c/d;p?q"
Error             {Invalid URI: This is not an absolute URI.}
IsSilent          true
absoluteURI       "http://username:1234@host.com:123/a/b/c/d;p?q"
error             {Invalid URI: This is not an absolute URI.}
isAbsolute        false
isSilent          true
Static members
```

However, validating as a relative URI returns true:

MyParser.validateUri("path?query#fragment", URIParser.validationType.Relative);

In some cases, regardless of the parser set up. A user may want to determine whether a URI is valid in general, i.e. either as an absolute URI or a relative URI. To do this we simply indicate that the validation is to not assume absolute or relative using the 'URIParser.validationType' enum of 'Either'. Both the below examples return "true":

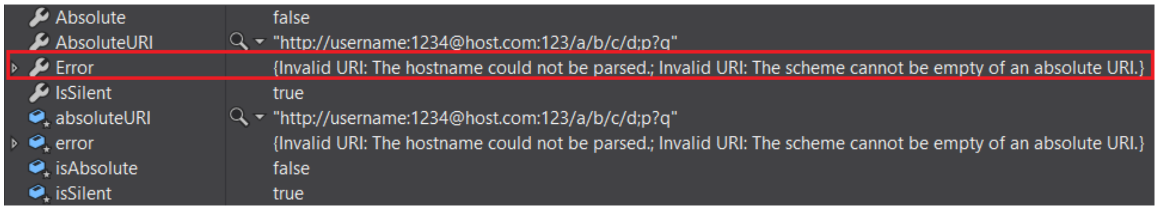
MyParser.validateUri("path?query#fragment", URIParser.validationType.Either);

MyParser.validateUri("http://a/path?query#fragment", URIParser.validationType.Either);

`But if we check whether the URI '///a/path?query#fragment'is valid as either an absolute URI or a relative URI:`

MyParser.validateUri("///a/path?query#fragment", URIParser.validationType.Either);

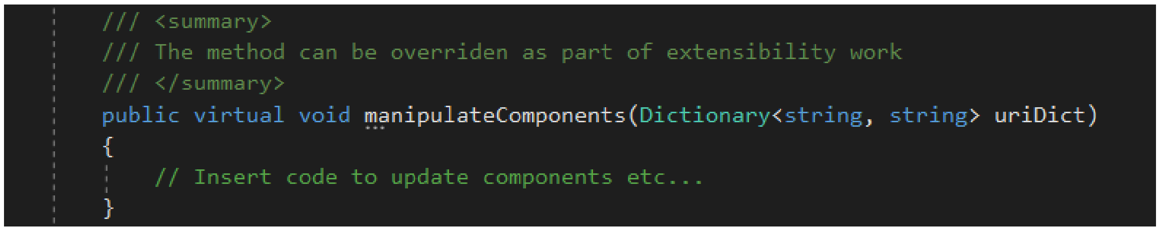
`The error message will contain both the reason why it is not an absolute URI and the reason it is not a relative URI separated by a semi-colon:`



```
Absolute          false
AbsoluteURI       "http://username:1234@host.com:123/a/b/c/d;p?q"
Error             {Invalid URI: The hostname could not be parsed.; Invalid URI: The scheme cannot be empty of an absolute URI.}
IsSilent          true
absoluteURI       "http://username:1234@host.com:123/a/b/c/d;p?q"
error             {Invalid URI: The hostname could not be parsed.; Invalid URI: The scheme cannot be empty of an absolute URI.}
isAbsolute        false
isSilent          true
```

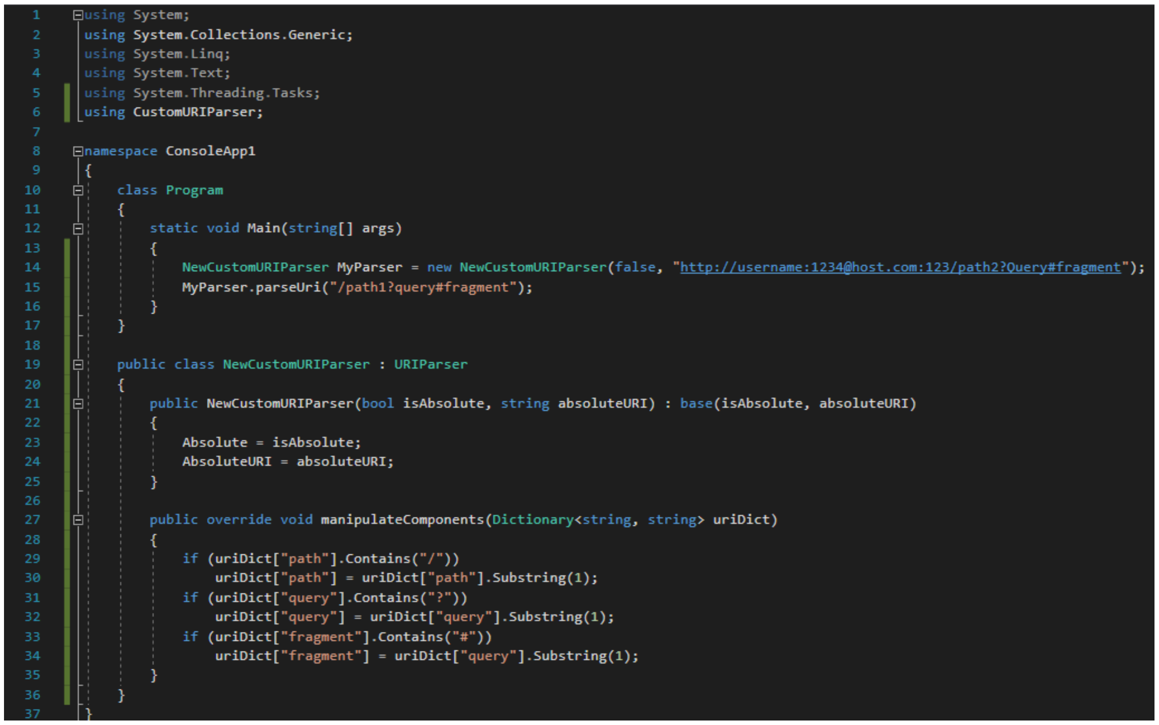
3. Extensibility

The CustomURIParser returns the 'path' including the first slash '/', the 'query' including the first question mark '?', and the fragment including the first hash '#'. The following method can be overridden in order to manipulate these components before presenting it to the user:



```
/// <summary>
/// The method can be overriden as part of extensibility work
/// </summary>
public virtual void manipulateComponents(Dictionary<string, string> uriDict)
{
    // Insert code to update components etc...
}
```

An example implementation is as follows: We inherit from URIParser in the CustomURIParser namespace once we’ve included the CustomURIParser.dll as a reference. Then we define our own component handling method:



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using CustomURIParser;

namespace ConsoleApp1
{
    class Program
    {
        static void Main(string[] args)
        {
            NewCustomURIParser MyParser = new NewCustomURIParser(false, "http://username:1234@host.com:123/path2?Query#fragment");
            MyParser.parseUri("/path1?query#fragment");
        }
    }

    public class NewCustomURIParser : URIParser
    {
        public NewCustomURIParser(bool isAbsolute, string absoluteURI) : base(isAbsolute, absoluteURI)
        {
            Absolute = isAbsolute;
            AbsoluteURI = absoluteURI;
        }

        public override void manipulateComponents(Dictionary<string, string> uriDict)
        {
            if (uriDict["path"].Contains("/"))
                uriDict["path"] = uriDict["path"].Substring(1);
            if (uriDict["query"].Contains("?"))
                uriDict["query"] = uriDict["query"].Substring(1);
            if (uriDict["fragment"].Contains("#"))
                uriDict["fragment"] = uriDict["query"].Substring(1);
        }
    }
}
```