Get url parameters from a string in .NET

Ask Question

204

38

I've got a string in .NET which is actually a url. I want an easy way to get the value from a particular parameter.

Normally, I'd just use `Request.Params["theThingIWant"]` , but this string isn't from the request. I can create a new `Uri` item like so:

```
Uri myUri = new Uri(TheStringUrlIWantMyValueFrom);
```

I can use `myUri.Query` to get the query string...but then I apparently have to find some regexy way of splitting it up.

Am I missing something obvious, or is there no built in way to do this short of creating a regex of some kind, etc?

c# .net url parsing parameters

edited Jul 11 '14 at 19:31
Kevin Panko
6,999 9 44 55

asked Mar 18 '09 at 20:05
Beska
8,402 14 70 105

12 Answers

430

Use static `ParseQueryString` method of `System.Web.HttpUtility` class that returns `NameValueCollection`.

```
Uri myUri = new Uri("http://www.example.com?param1=good&param2=bad")
string param1 = HttpUtility.ParseQueryString(myUri.Query).Get("param
```

Check documentation at http://msdn.microsoft.com/en-us/library/ms150046.aspx

edited Mar 2 '15 at 19:10
Matt Bishop
920 5 15

answered Mar 18 '09 at 20:19
CZFox
6,632 3 21 12

13 This doesn't seem to detect the first parameter. eg parsing "google.com/ ..." doesn't detect the parameter q – Andrew Shepherd Jun 30 '09 at 4:26

@Andrew I confirm. It's strange (bug?). You coul still use `HttpUtility.ParseQueryString(myUri.Query).Get(0)` though and it will extract first parameter. ` – Mariusz Pawelski Aug 2 '11 at 15:03

13 Doesn't work with a relative url... – Stefan Steiger Dec 21 '12 at 1:46

1 You can't parse full query URLs with `HttpUtility.ParseQueryString(string)`! As it's name says, it's to parse Query Strings, not URLs with query parameters. If you want to do it, you must first split it by `?` like this: `Url.Split('?')` and get last element using (depending on situation and what you need) `[0]` or LINQ's `Last()` / `LastOrDefault()`. – Kosiek Jan 16 '18 at 11:50

1 When trialling this myself, the signature appears to have changed to this: HttpUtility.ParseQueryString(uri.Query).GetValues("param1").First() – The Senator Feb 21 '18 at 18:48

0

This is actually very simple, and that worked for me :)

```
        if (id == "DK")
        {
            string longurl = "selectServer.aspx?country=";
            var uriBuilder = new UriBuilder(longurl);
            var query = HttpUtility.ParseQueryString(uriBuilder.Quer
            query["country"] = "DK";

            uriBuilder.Query = query.ToString();
            longurl = uriBuilder.ToString();
        }
```

answered Feb 28 at 15:19

Ralle12

1 1

2

Use .NET Reflector to view the `FillFromString` method of `System.Web.HttpValueCollection`. That gives you the code that ASP.NET is using to fill the `Request.QueryString` collection.

edited Nov 1 '16 at 17:33

SharpC

3,480 2 25 30

answered Mar 18 '09 at 20:20

David

30.7k 3 55 73

@Andrew and @CZFox

10

I had the same bug and found the cause to be that parameter one is in fact: `http://www.example.com?param1` and not `param1` which is what one would expect.

By removing all characters before and including the question mark fixes this problem. So in essence the `HttpUtility.ParseQueryString` function only requires a valid query string parameter containing only characters after the question mark as in:

```
HttpUtility.ParseQueryString ( "param1=good&param2=bad" )
```

My workaround:

```
string RawUrl = "http://www.example.com?param1=good&param2=bad";
int index = RawUrl.IndexOf ( "?" );
if ( index > 0 )
    RawUrl = RawUrl.Substring ( index ).Remove ( 0, 1 );

Uri myUri = new Uri( RawUrl, UriKind.RelativeOrAbsolute);
string param1 = HttpUtility.ParseQueryString( myUri.Query ).Get( "pa
```

edited Nov 1 '16 at 17:31

SharpC
3,480 2 25 30

answered Sep 27 '11 at 17:14

Mo Gauvin
127 2 4

When the URI is instantiated I get the error "Invalid URI: The format of the URI could not be determined." I don't think this solution works as intended. – Paul Matthews Nov 11 '13 at 23:17

@PaulMatthews, you are correct. At the time of this given solution, I was using the older .net framework 2.0. To confirm, your statement, I copied and pasted this solution into LINQPad v2 by Joseph Albahara and received the same error you mentioned. – Mo Gauvin Nov 12 '13 at 22:18

@PaulMatthews, To fix, remove the line that reads Uri myUri = new Uri(

RawUrl); and merely pass RawUrl to the last statement as in: string param1 = HttpUtility.ParseQueryString(RawUrl).Get("param2"); – Mo Gauvin Nov 12 '13 at 22:25

Yes it works if you do it that way :) – Paul Matthews Nov 12 '13 at 23:00

11

Looks like you should loop over the values of `myUri.Query` and parse it from there.

```
string desiredValue;
foreach(string item in myUri.Query.Split('&'))
{
    string[] parts = item.Replace('?', '').Split('=');
    if(parts[0] == "desiredKey")
    {
        desiredValue = parts[1];
        break;
    }
}
```

I wouldn't use this code without testing it on a bunch of malformed URLs however. It might break on some/all of these:

- `hello.html?`
- `hello.html?valuelesskey`
- `hello.html?key=value=hi`
- `hello.html?hi=value?&b=c`
- etc

edited Nov 1 '16 at 17:30
SharpC
3,480 2 25 30

answered Mar 18 '09 at 20:11
Tom Ritter

83.8k 27 125 163

26

Here's another alternative if, for any reason, you can't or don't want to use `HttpUtility.ParseQueryString()` .

This is built to be somewhat tolerant to "malformed" query strings, i.e. `http://test/test.html?empty=` becomes a parameter with an empty value. The caller can verify the parameters if needed.

```
public static class UriHelper
{
    public static Dictionary<string, string> DecodeQueryParameters(t
    {
        if (uri == null)
            throw new ArgumentNullException("uri");

        if (uri.Query.Length == 0)
            return new Dictionary<string, string>();

        return uri.Query.TrimStart('?')
                        .Split(new[] { '&', ';' },
StringSplitOptions.RemoveEmptyEntries)
                        .Select(parameter => parameter.Split(new[] {
StringSplitOptions.RemoveEmptyEntries))
                        .GroupBy(parts => parts[0],
                                 parts => parts.Length > 2 ? string.
parts.Length - 1) : (parts.Length > 1 ? parts[1] : ""))
                        .ToDictionary(grouping => grouping.Key,
                                      grouping => string.Join(",", g
    }
}
```

Test

```
[TestClass]
public class UriHelperTest
{
    [TestMethod]
    public void DecodeQueryParameters()
    {
        DecodeQueryParametersTest("http://test/test.html", new Dicti
string>());
        DecodeQueryParametersTest("http://test/test.html?", new Dict
string>());
        DecodeQueryParametersTest("http://test/test.html?key=bla/blu
Dictionary<string, string> { { "key", "bla/blub.xml" } });
        DecodeQueryParametersTest("http://test/test.html?eins=1&zwei
Dictionary<string, string> { { "eins", "1" }, { "zwei", "2" } });
        DecodeQueryParametersTest("http://test/test.html?empty", new
string> { { "empty", "" } });
        DecodeQueryParametersTest("http://test/test.html?empty=", ne
string> { { "empty", "" } });
        DecodeQueryParametersTest("http://test/test.html?key=1&", ne
string> { { "key", "1" } });
        DecodeQueryParametersTest("http://test/test.html?key=value?&
Dictionary<string, string> { { "key", "value?" }, { "b", "c" } });
        DecodeQueryParametersTest("http://test/test.html?key=value=w
Dictionary<string, string> { { "key", "value=what" } });
        DecodeQueryParametersTest("http://www.google.com/search?
q=energy+edge&rls=com.microsoft:en-au&ie=UTF-8&oe=UTF-8&startIndex=&
            new Dictionary<string, string>
            {
                { "q", "energy+edge" },
                { "rls", "com.microsoft:en-au" },
                { "ie", "UTF-8" },
                { "oe", "UTF-8" },
                { "startIndex", "" },
                { "startPage", "1%22" },
            });
        DecodeQueryParametersTest("http://test/test.html?key=value;k
new Dictionary<string, string> { { "key", "value,anotherValue" } });
    }

    private static void DecodeQueryParametersTest(string uri, Dictio
expected)
    {
        Dictionary<string, string> parameters = new Uri(uri).DecodeQ
        Assert.AreEqual(expected.Count, parameters.Count, "Wrong par
{0}", uri);
        foreach (var key in expected.Keys)
        {
            Assert.IsTrue(parameters.ContainsKey(key), "Missing para
{1}", key, uri);
            Assert.AreEqual(expected[key], parameters[key], "Wrong p
{0}. Uri: {1}", parameters[key], uri);
        }
        }
    }
```

edited Oct 10 '16 at 10:23

answered Nov 22 '13 at 0:56

alsed42

973 6 9

-2

I used it and it run perfectly

```
<%=Request.QueryString["id"] %>
```

answered May 17 '16 at 14:29

Hoang Huynh Nhat

7 1

1 from a string not query string – Arun Prasad E S Jul 9 '16 at 5:38

1

Or if you don't know the URL (so as to avoid hardcoding, use the `AbsoluteUri`

Example ...

```
        //get the full URL
        Uri myUri = new Uri(Request.Url.AbsoluteUri);
        //get any parameters
        string strStatus = HttpUtility.ParseQueryString(myUri.Query)
```

```
string strMsg = HttpUtility.ParseQueryString(myUri.Query).Ge
switch (strStatus.ToUpper())
{
    case "OK":
        webMessageBox.Show("EMAILS SENT!");
        break;
    case "ER":
        webMessageBox.Show("EMAILS SENT, BUT ... " + strMsg)
        break;
}
```

answered Dec 1 '15 at 3:58

Fandango68
2,086 2 20 38

1

```
HttpContext.Current.Request.QueryString.Get("id");
```

answered Jan 6 '15 at 11:43

Hallgeir Engen
709 7 10

how to use this with a string – Arun Prasad E S Jul 9 '16 at 5:37

0

if you want in get your QueryString on Default page .Default page means your current page url . you can try this code :

```
string paramIl = HttpUtility.ParseQueryString(this.ClientQueryString
```

answered Dec 6 '13 at 11:27

Erhan Demirci

3,178 3 29 40

2

You can use the following workaround for it to work with the first parameter too:

```
var param1 =
    HttpUtility.ParseQueryString(url.Substring(
        new []{0, url.IndexOf('?')}.Max()
    )).Get("param1");
```

answered Jun 15 '12 at 12:40

tomsv
4,949 4 42 78

45

This is probably what you want

```
var uri = new Uri("http://domain.test/Default.aspx?var1=true&var2=te
var query = HttpUtility.ParseQueryString(uri.Query);

var var2 = query.Get("var2");
```

answered Mar 18 '09 at 20:27

Sergej Andrejev
5,637 10 56 102

This worked out great to catch the first parameter – Daniel S Jan 17 at 8:22