

Forum:[Hack Forums](#)

Thread: [Using Cookies from WebBrowser control in WebClient Request.](#)

Suspect. - June 18, 2018, 5:44 pm

Have got it working now. Solution is below if anyone needs.

Snowfox - June 18, 2018, 11:13 pm

Yes. I've used it once. Get the cookies from WebBrowser control like below (If you are using winforms)

```
Code public CookieContainer GetCookieContainer()
{
    CookieContainer container = new CookieContainer();

    foreach (string cookie in webBrowser1.Document.Cookie.Split(';'))
    {
        string name = cookie.Split('=')[0];
        string value = cookie.Substring(name.Length + 1);
        string path = "/";
        string domain = ".google.com"; //change to your domain name
        container.Add(new Cookie(name.Trim(), value.Trim(), path, domain));
    }

    return container;
}
```

Next, serialize the cookiecontainer and save something like cookies.dat to reuse(deserialize) in HttpWebrequest

If you are trying to pass the cookies form native IE browser then

```
Code [DllImport("wininet.dll", SetLastError = true)]
public static extern bool InternetGetCookieEx(
    string url,
    string cookieName,
    StringBuilder cookieData,
    ref int size,
    Int32 dwFlags,
    IntPtr lpReserved);

private const Int32 InternetCookieHttponly = 0x2000;

/// <summary>
/// Gets the URI cookie container.
/// </summary>
/// <param name="uri">The URI.</param>
/// <returns></returns>
public static CookieContainer GetUriCookieContainer(Uri uri)
{
    CookieContainer cookies = null;
```

```

// Determine the size of the cookie
int datasize = 8192 * 16;
StringBuilder cookieData = new StringBuilder(datasize);
if (!InternetGetCookieEx(uri.ToString(), null, cookieData, ref datasize, InternetCookieHttponly,
IntPtr.Zero))
{
    if (datasize < 0)
        return null;
    // Allocate stringbuilder large enough to hold the cookie
    cookieData = new StringBuilder(datasize);
    if (!InternetGetCookieEx(
        uri.ToString(),
        null, cookieData,
        ref datasize,
        InternetCookieHttponly,
        IntPtr.Zero))
        return null;
}
if (cookieData.Length > 0)
{
    cookies = new CookieContainer();
    cookies.SetCookies(uri, cookieData.ToString().Replace(';', ','));
}
return cookies;
}

```

Source : [Stackoverflow](#)

And I believe you don't need to pass the cookies from browser to HttpWebRequest. You can do the same requests from HttpWebrequest to avoid above steps.

Hope this helps.

Suspect. - June 19, 2018, 10:00 pm

(Jun 18, 2018 11:13 PM)Snowfox Wrote: [\[CITE\]](#) Yes. I've used it once. Get the cookies from WebBrowser control like below (If you are using winforms)

```

Code public CookieContainer GetCookieContainer()
{
    CookieContainer container = new CookieContainer();

    foreach (string cookie in webBrowser1.Document.Cookie.Split(';'))
    {
        string name = cookie.Split('=')[0];
        string value = cookie.Substring(name.Length + 1);
        string path = "/";
        string domain = ".google.com"; //change to your domain name
        container.Add(new Cookie(name.Trim(), value.Trim(), path, domain));
    }

    return container;
}

```

Next, serialize the cookiecontainer and save something like cookies.dat to reuse(deserialize) in

HttpWebrequest

If you are trying to pass the cookies from native IE browser then

Code [DllImport("wininet.dll", SetLastError = true)]

```
public static extern bool InternetGetCookieEx(  
    string url,  
    string cookieName,  
    StringBuilder cookieData,  
    ref int size,  
    Int32 dwFlags,  
    IntPtr lpReserved);
```

```
private const Int32 InternetCookieHttponly = 0x2000;
```

```
/// <summary>
```

```
/// Gets the URI cookie container.
```

```
/// </summary>
```

```
/// <param name="uri">The URI.</param>
```

```
/// <returns></returns>
```

```
public static CookieContainer GetUriCookieContainer(Uri uri)
```

```
{  
    CookieContainer cookies = null;  
    // Determine the size of the cookie  
    int datasize = 8192 * 16;  
    StringBuilder cookieData = new StringBuilder(datasize);  
    if (!InternetGetCookieEx(uri.ToString(), null, cookieData, ref datasize, InternetCookieHttponly,  
        IntPtr.Zero))  
    {  
        if (datasize < 0)  
            return null;  
        // Allocate stringbuilder large enough to hold the cookie  
        cookieData = new StringBuilder(datasize);  
        if (!InternetGetCookieEx(  
            uri.ToString(),  
            null, cookieData,  
            ref datasize,  
            InternetCookieHttponly,  
            IntPtr.Zero))  
            return null;  
    }  
    if (cookieData.Length > 0)  
    {  
        cookies = new CookieContainer();  
        cookies.SetCookies(uri, cookieData.ToString().Replace(';', ','));  
    }  
    return cookies;  
}
```

Source : [Stackoverflow](#)

And I believe you don't need to pass the cookies from browser to HttpWebRequest. You can do the same requests from HttpWebrequest to avoid above steps.

Hope this helps.

Thanks for your reply. I got it working by Creating a new Class and inputing this code:

```
Code using System;
using System.ComponentModel;
using System.Net;
using System.Runtime.InteropServices;
using System.Security;
using System.Security.Permissions;
using System.Text;
using System.Windows.Forms;
```

```
internal sealed class NativeMethods
```

```
{
    #region enums

    public enum ErrorFlags
    {
        ERROR_INSUFFICIENT_BUFFER = 122,
        ERROR_INVALID_PARAMETER = 87,
        ERROR_NO_MORE_ITEMS = 259
    }

    public enum InternetFlags
    {
        INTERNET_COOKIE_HTTPONLY = 8192, //Requires IE 8 or higher
        INTERNET_COOKIE_THIRD_PARTY = 131072, INTERNET_FLAG_RESTRICTED_ZONE =
16
    }

    #endregion

    #region DLL Imports

    [SuppressUnmanagedCodeSecurity, SecurityCritical, DllImport("wininet.dll", EntryPoint =
"InternetGetCookieExW", CharSet = CharSet.Unicode, SetLastError = true, ExactSpelling = true)]
    internal static extern bool InternetGetCookieEx([In] string Url, [In] string cookieName, [Out]
StringBuilder cookieData, [In, Out] ref uint pchCookieData, uint flags, IntPtr reserved);

    #endregion
}
```

```
/// <SUMMARY></SUMMARY>
```

```
/// WebBrowserCookie?
```

```
/// webBrowser1.Document.CookieHttpOnlyCookie
```

```
///
```

```
public class FullWebBrowserCookie : WebBrowser
```

```
{

    [SecurityCritical]
    public static string GetCookieInternal(Uri uri, bool throwIfNoCookie)
```

```

    {
        uint pchCookieData = 0;
        string url = UriToString(uri);
        uint flag = (uint)NativeMethods.InternetFlags.INTERNET_COOKIE_HTTPONLY;

        //Gets the size of the string builder
        if (NativeMethods.InternetGetCookieEx(url, null, null, ref pchCookieData, flag, IntPtr.Zero))
        {
            pchCookieData++;
            StringBuilder cookieData = new StringBuilder((int)pchCookieData);

            //Read the cookie
            if (NativeMethods.InternetGetCookieEx(url, null, cookieData, ref pchCookieData, flag,
IntPtr.Zero))
            {
                DemandWebPermission(uri);
                return cookieData.ToString();
            }
        }

        int lastErrorCode = Marshal.GetLastWin32Error();

        if (throwIfNoCookie || (lastErrorCode !=
(int)NativeMethods.ErrorFlags.ERROR_NO_MORE_ITEMS))
        {
            throw new Win32Exception(lastErrorCode);
        }

        return null;
    }

    private static void DemandWebPermission(Uri uri)
    {
        string uriString = UriToString(uri);

        if (uri.IsFile)
        {
            string localPath = uri.LocalPath;
            new FileIOPermission(FileIOPermissionAccess.Read, localPath).Demand();
        }
        else
        {
            new WebPermission(NetworkAccess.Connect, uriString).Demand();
        }
    }

    private static string UriToString(Uri uri)
    {
        if (uri == null)
        {
            throw new ArgumentNullException("uri");
        }
    }

```

```
UriComponents components = (uri.IsAbsoluteUri ? UriComponents.AbsoluteUri :  
UriComponents.SerializationInfoString);  
return new StringBuilder(uri.GetComponents(components, UriFormat.SafeUnescaped),  
2083).ToString();  
}  
}
```

Then to use it in my WebRequest I used:

```
Code var cookies = FullWebBrowserCookie.GetCookieInternal(webBrowser1.Url, false);  
WebClient wc = new WebClient();  
wc.Headers.Add("Cookie: " + cookies);  
wc.Headers.Add("Content-Type", "application/x-www-form-urlencoded");  
byte[] result = wc.UploadData("<URL>", "POST",  
System.Text.Encoding.UTF8.GetBytes(postData));
```

Snowfox - June 19, 2018, 11:39 pm

Thanks for posting the fix as well. ?