

BK

[getburgerkingcode.zip](#)

[snippet.cs](#)

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Net;
using System.IO;
using System.Text.RegularExpressions;

namespace GetBurgerkingCode
{
    class Program
    {
        private CookieContainer cookieContainer = new
CookieContainer();
        private string mSurveyCode;
        int mStep = 1;
        string mLastContent;
        string mUrl = "https://kor.tellburgerking.com";
        string mInputCode = "4295028261123197";
        Dictionary<string, string> mParamDict = new Dictionary<string,
string>();

        static void Main(string[] args)
        {
            Program p = new Program();

            p.start();
        }

        private void start()
        {
            Console.WriteLine("Enter Request Code:"); // Prompt
            string rCode = Console.ReadLine(); // Get string from user

            mInputCode = rCode;
            Console.WriteLine("InputCode : {0}",rCode); // Report

            output

            Regex r = new Regex(@"^\d{16}$");
            if (r.IsMatch(mInputCode) == false) {
                Console.WriteLine("[ERROR] Write Valid Code");
                Environment.Exit(0);
            }
        }
    }
}
```

```
    }

    // step 1 connect
    printStep();
    step1();
    printStep();
    step2();

    //again
    while (hasNextStep()) {
        printStep();
        step3();
    }

    //Console.WriteLine(mLastContent);

    string myCodePattern = @"<p class=""ValCode"">          :
    (?<code>.+)</p>";
    Regex regex = new Regex(myCodePattern);
    MatchCollection mc = regex.Matches(mLastContent);

    String code;
    foreach (Match m in mc)
    {
        code = m.Groups["code"].ToString();
        Console.WriteLine("");
        Console.WriteLine("#####");
        Console.WriteLine("code is {0}", code);
    }

    string endSig = Console.ReadLine(); // Get string from user
}

private void printStep()
{
    Console.WriteLine("");
    Console.WriteLine("##### STEP {0} #####", mStep++);
}

private bool hasNextStep()
{
    if (mParamDict.ContainsKey("IoNF")) {
        return true;
    }

    return false;
}

private void step3(bool bPrint = false)
```

```
{
    string url = getUrl();
    string param = getParam();

    HttpWebResponse resp = GetResponse(url, param);

    if (resp.StatusCode == HttpStatusCode.OK)
    {
        string respContent = GetContent(resp);
        mLastContent = respContent;

        if (bPrint) {
            Console.WriteLine(respContent);
        }

        clearParam();
        addRadioParam(respContent);
        addHiddenParam(respContent);
    }
}

private void addRadioParam(string respContent)
{
    string pattern = @"<input type=""radio""
name=""(?<name>.+?)"" value=""(?<value>.+?)"" ";
    addNextParam(pattern, respContent);
}

private void addHiddenParam(string respContent)
{
    string pattern = @"<input type=""hidden""
name=""(?<name>.+?)"" value=""(?<value>.+?)"" ";
    addNextParam(pattern, respContent);
}

private void addNextParam(string pattern, string respContent)
{
    Regex regex = new Regex(pattern);
    MatchCollection mc = regex.Matches(respContent);

    String name, value;
    foreach (Match m in mc)
    {
        name = m.Groups["name"].ToString();
        Console.WriteLine("name is {0}", name);

        value = m.Groups["value"].ToString();
    }
}
```

```
        Console.WriteLine("value is {0}", value);

        addParam(name, value);
    }
}

private void step2()
{
    string cn1 = "CN1=" + mInputCode.Substring(0,3);
    string cn2 = "&CN2=" + mInputCode.Substring(3,3);
    string cn3 = "&CN3=" + mInputCode.Substring(6,3);
    string cn4 = "&CN4=" + mInputCode.Substring(9,3);
    string cn5 = "&CN5=" + mInputCode.Substring(12, 3);
    string cn6 = "&CN6=" + mInputCode.Substring(15, 1);
    string fip = "&FIP=" + "True";
    string jsEnabled = "&JavaScriptEnabled=" + "1";

    string step2Url = getUrl();
    string step2Data = cn1 + cn2 + cn3 + cn4 + cn5 + cn6 + fip
+ jsEnabled;

    HttpWebResponse resp = GetResponse(step2Url, step2Data);

    if (resp.StatusCode == HttpStatusCode.OK)
    {
        string respContent = GetContent(resp);

        clearParam();
        addRadioParam(respContent);
        addHiddenParam(respContent);
    }
}

private string getParam()
{
    string param = "";

    foreach (KeyValuePair<string, string> pair in mParamDict)
    {
        param += pair.Key.ToString() + "=" +
pair.Value.ToString() + "&";
    }

    param.Substring(0, param.Length -1);

    return param;
}
```

```
}

private string getUrl()
{
    return mUrl + "/Survey.aspx?" + mSurveyCode; ;
}

private void clearParam()
{
    mParamDict.Clear();
}

private void addParam(string name, string value)
{
    if (mParamDict.ContainsKey(name) == false) {
        mParamDict.Add(name, value);
    }
}

private void step1()
{
    HttpWebResponse resp = GetResponse(mUrl, "");

    if (resp.StatusCode == HttpStatusCode.OK)
    {
        string respContent = GetContent(resp);
        // Get the stream associated with the response.

        Regex qariRegex = new Regex(@"<form method=""post""
id=""surveyEntryForm"" action=""Survey.aspx\?(?<code>.+)"">");
        MatchCollection mc = qariRegex.Matches(respContent);

        foreach (Match m in mc)
        {
            Console.WriteLine("Request Code is {0}",
m.Groups["code"]);
            mSurveyCode = m.Groups["code"].ToString();
        }

        //Console.WriteLine("Response stream received.");
        //Console.WriteLine(content);
        resp.Close();
    }
}

private string GetContent(HttpWebResponse resp)
{
    Stream receiveStream = resp.GetResponseStream();

    // Pipes the stream to a higher level stream reader with
```

```
the required encoding format.
    StreamReader readStream = new StreamReader(receiveStream,
Encoding.UTF8);
    string content = readStream.ReadToEnd();
    readStream.Close();

    return content;
}

private HttpResponseMessage GetResponse(string url, string param)
{
    Console.WriteLine("STEP URL : {0}", url);
    Console.WriteLine("STEP DATA : {0}", param);

    HttpWebRequest req =
(HttpWebRequest)HttpWebRequest.Create(url);
    req.CookieContainer = cookieContainer; // <= HERE
    req.Method = "POST";
    req.KeepAlive = false;
    req.CookieContainer = cookieContainer; // <= HERE
    req.AllowAutoRedirect = true;
    req.UserAgent = "Mozilla/5.0 (Windows NT 6.1)
AppleWebKit/535.2 (KHTML, like Gecko) Chrome/15.0.874.121
Safari/535.2";
    req.ContentType = "application/x-www-form-urlencoded";

    byte[] byteArray = Encoding.UTF8.GetBytes(param);
    req.ContentLength = byteArray.Length;
    Stream datastream;
    datastream = req.GetRequestStream();
    datastream.Write(byteArray, 0, byteArray.Length);
    datastream.Close();

    HttpResponseMessage resp = (HttpResponseMessage)req.GetResponse();
    return resp;
}
}
```

- [C#](#)

From:

<http://moro.kr/> - **Various Ways**

Permanent link:

<http://moro.kr/open/burgerking-code>

Last update: **2020/06/02 09:25**

