

Forum:[Hack Forums](#)

Thread: [Do you know a way to obfuscate html string in c#](#)

Fhack - April 10, 2020, 12:41 pm

Hello i'm trying to obfuscate an HTML letter to increase his inbox rate !

For example i've this :

Code `<a href="mailto:someone@domain.com">Mail me</a>`
And i want this :

Code `<a href="mailto:someone@domain.com">Mail me</a>`

This is named character entitles obfuscation. It's like using `HtmlUtility.Encode(htmlString)` method in c# but only for character that are inside `<tags>` or double quotes.

Example :

Code `<p class="ENCODE THIS">AND I WANT TO ENCODE THIS</p>`
If you know how to do this in c# i'm here to follow your advise :) If you know other html obfuscation method that don't require javascript i'm also here to listen !

itcorp - April 10, 2020, 1:16 pm

if your goal is to make reading of Page Source a little harder I think you better use JavaScript or jquery for html obfuscation you can found on the internet solid solutions for what you want of course if i understand yours well.I wish you all the best buddy

Fhack - April 10, 2020, 2:32 pm

(Apr 10, 2020 01:16 PM)itcorp Wrote: [\[CITE\]](#) if your goal is to make reading of Page Source a little harder I think you better use JavaScript or jquery for html obfuscation you can found on the internet solid solutions for what you want of course if i understand yours well.I wish you all the best buddy
Javascript is not accepted inside email that's why i don't want to use it !

(Apr 10, 2020 02:32 PM)Fhack Wrote: [\[CITE\]](#) If you know how to do this in c# i'm here to follow your advise :) If you know other html obfuscation method that don't require javascript i'm also here to listen !

itcorp - April 10, 2020, 4:26 pm

Ok, I hope you asked for this i tried to find you solution for a problem as I understood it.I wish you all the best buddy

Code `@{
 ViewBag.Title = "Home Page";
}`

`<div class="jumbotron">
 <h1>ASP.NET</h1>`

```
@functions
```

```
{
    string Encode(string data)
    {
        string returnData = null;
        foreach (char c in data)
        {
            returnData += "&#" + Convert.ToInt32(c);
        }
        return returnData;
    }
}
```

```
<a href="@Encode("mailto:someone@domain.com")">@Encode("Mail me")</a><br />
```

```
<a
```

```
href="@Html.Raw(@HttpUtility.HtmlDecode(Encode("mailto:someone@domain.com"))) ">@Html.Ra
w(@HttpUtility.HtmlDecode(Encode("Mail me").ToString()))</a>
```

```
</div>
```

Page source

Code <div class="jumbotron">

```
<h1>ASP.NET</h1>
```

```
<a
```

```
href="&#109&#97&#105&#108&#116&#111&#58&#115&#111&#109&#101&#111&#110&#101&#64&#100&#111&#109&#97&#105&#110&#46&#99&#111&#109">&#77&#97&#105&#108&#32&#109&#101</a><br />
```

```
<a
```

```
href="&#109&#97&#105&#108&#116&#111&#58&#115&#111&#109&#101&#111&#110&#101&#64&#100&#111&#109&#97&#105&#110&#46&#99&#111&#109">&#77&#97&#105&#108&#32&#109&#101</a>
```

```
</div>
```

view

Fhack - April 11, 2020, 7:54 am

(Apr 10, 2020 04:26 PM)itcorp Wrote: [\[CITE\]](#) Ok, I hope you asked for this i tried to find you solution for a problem as I understood it.I wish you all the best buddy

Code @{

```
    ViewBag.Title = "Home Page";
```

```
}
```

```
<div class="jumbotron">
```

```
<h1>ASP.NET</h1>
```

```
@functions
```

```
{
```

```

string Encode(string data)
{
    string returnData = null;
    foreach (char c in data)
    {
        returnData += "&#" + Convert.ToInt32(c);
    }
    return returnData;
}
}

```

```

<a href="@Encode("mailto:someone@domain.com")">@Encode("Mail me")</a><br />

```

```

<a
href="@Html.Raw(@HttpUtility.HtmlDecode(Encode("mailto:someone@domain.com")))">@Html.Ra
w(@HttpUtility.HtmlDecode(Encode("Mail me").ToString()))</a>

```

```

</div>

```

Page source

Code <div class="jumbotron">

```

<h1>ASP.NET</h1>

```

```

<a

```

```

href="&#109&#97&#105&#108&#116&#111&#58&#115&#111&#109&#101&#111&#110&#101&#64&#100&#111&#109&#97&#105&#110&#46&#99&#111&#109">&#77&#97&#105&#108&#32&#109&#101</a><br />

```

```

<a

```

```

href="&#109&#97&#105&#108&#116&#111&#58&#115&#111&#109&#101&#111&#110&#101&#64&#100&#111&#109&#97&#105&#110&#46&#99&#111&#109">&#77&#97&#105&#108&#32&#109&#101</a>

```

```

</div>

```

view

Yes but it need to be generic for all html page not for just this example. And it need to be coded in c#

itcorp - April 11, 2020, 5:43 pm

here you go buddy, a quick and simple solution which you need to refine and perfect a little but I think you will have a clear picture.I wish you all the best buddy

Code namespace ConsoleApp20

```

{
    class Program
    {
        static void Main(string[] args)
        {
            Engine engine = new Engine();
            engine.SetHtml = "<html><a href=\"mailto:someone@domain.com\">Mail me</a><p

```

```

class=\"ENCODE THIS\">AND I WANT TO ENCODE THIS</p></html>";
    Console.ForegroundColor = ConsoleColor.Yellow;
    Console.WriteLine($"Number of Elements:{engine.CountTags()}");
    Console.ForegroundColor = ConsoleColor.White;
    engine.PrintTagNames();
    engine.PrintEncodedHtml();
    Console.ReadLine();
}
}
}
Code using HtmlAgilityPack;
using System;
using System.Linq;

namespace ConsoleApp20
{
    public class Engine
    {
        private string _html { get; set; }

        public Engine()
        {

        }

        public Engine(string _html)
        {
            this._html = _html;
        }

        public string SetHtml
        {
            set => _html = value;
        }

        public string GetHtml
        {
            get => _html;
        }

        public int CountTags()
        {
            if (!String.IsNullOrEmpty(_html))
            {
                var doc = new HtmlDocument();
                doc.LoadHtml(GetHtml);
                return doc.DocumentNode.Descendants().Where(e=>e.NodeType==
HtmlNodeType.Element).Count();
            }
            else
                return 0;
        }
    }
}

```

```

public void PrintTagNames()
{
    if (!String.IsNullOrEmpty(_html))
    {
        var doc = new HtmlDocument();
        doc.LoadHtml(GetHtml);
        var root = doc.DocumentNode.Descendants().ToList();

        foreach (var tag in root)
        {
            if (tag.NodeType != HtmlNodeType.Element)
                continue;
            else
            {
                Console.ForegroundColor = ConsoleColor.Green;
                Console.WriteLine("Name:" + tag.OriginalName + " " + "Type:" + tag.NodeType);
                Console.ForegroundColor = ConsoleColor.White;
            }
        }
    }
    else
    {
        Console.ForegroundColor = ConsoleColor.Red;
        Console.WriteLine("None");
        Console.ForegroundColor = ConsoleColor.White;
    }
}

```

```

public void PrintEncodedHtml()
{
    if (!String.IsNullOrEmpty(_html))
    {
        var doc = new HtmlDocument();
        doc.LoadHtml(GetHtml);
        var root = doc.DocumentNode.Descendants().ToList();

        foreach (var tag in root)
        {
            if (tag.NodeType != HtmlNodeType.Element)
                continue;
            else
            {
                if (tag.HasAttributes)
                {
                    if (tag.Attributes["href"]!=null)
                    {
                        var originalHref = tag.Attributes["href"].Value;
                        Console.ForegroundColor = ConsoleColor.DarkYellow;

                        Console.WriteLine($"Original href:{originalHref}");

                        Console.WriteLine($"Encoded href:{EncodeHtml(originalHref)}");

                        tag.SetAttributeValue("href", EncodeHtml(originalHref));
                    }
                }
            }
        }
    }
}

```

```

        var originalData = tag.InnerText;

        Console.WriteLine($"Original href data:{originalData}");

        Console.WriteLine($"Encoded href data:{EncodeHtml(originalData)}");
        Console.ForegroundColor = ConsoleColor.White;

        tag.InnerHtml=EncodeHtml(originalData);

    }

    if (tag.Attributes["class"]!=null)
    {
        var originalClass = tag.Attributes["class"].Value;
        Console.ForegroundColor = ConsoleColor.DarkYellow;

        Console.WriteLine($"Original class:{originalClass}");

        Console.WriteLine($"Encoded href:{EncodeHtml(originalClass)}");

        tag.SetAttributeValue("class", EncodeHtml(originalClass));

        var originalData = tag.InnerText;

        Console.WriteLine($"Original href data:{originalData}");

        Console.WriteLine($"Original href data:{EncodeHtml(originalData)}");
        Console.ForegroundColor = ConsoleColor.White;

        tag.InnerHtml = EncodeHtml(originalData);
    }
}
}
}
}
Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine(doc.DocumentNode.OuterHtml);

Console.ForegroundColor = ConsoleColor.White;

}
}

public string EncodeHtml(string data)
{
    string returnData = null;

    foreach (char c in data)
    {
        returnData += "&#" + Convert.ToInt32(c);
    }
    return returnData;
}

```

```

    }
}
}

```

Fhack - April 12, 2020, 3:56 am
 (Apr 11, 2020 05:43 PM)itcorp Wrote: [\[CITE\]](#) here you go buddy, a quick and simple solution which you need to refine and perfect a little but I think you will have a clear picture.I wish you all the best buddy

```

Code namespace ConsoleApp20
{
    class Program
    {
        static void Main(string[] args)
        {
            Engine engine = new Engine();
            engine.SetHtml = "<html><a href=\"mailto:someone@domain.com\">Mail me</a><p
class=\"ENCODE THIS\">AND I WANT TO ENCODE THIS</p></html>";
            Console.ForegroundColor = ConsoleColor.Yellow;
            Console.WriteLine($"Number of Elements:{engine.CountTags()}");
            Console.ForegroundColor = ConsoleColor.White;
            engine.PrintTagNames();
            engine.PrintEncodedHtml();
            Console.ReadLine();
        }
    }
}
Code using HtmlAgilityPack;
using System;
using System.Linq;

```

```

namespace ConsoleApp20
{
    public class Engine
    {
        private string _html { get; set; }

        public Engine()
        {

        }

        public Engine(string _html)
        {
            this._html = _html;
        }

        public string SetHtml
        {
            set => _html = value;
        }
    }
}

```

```

    }

    public string GetHtml
    {
        get => _html;
    }

    public int CountTags()
    {
        if (!String.IsNullOrEmpty(_html))
        {
            var doc = new HtmlDocument();
            doc.LoadHtml(GetHtml);
            return doc.DocumentNode.Descendants().Where(e=>e.NodeType==
HtmlNodeType.Element).Count();
        }
        else
            return 0;
    }

    public void PrintTagNames()
    {
        if (!String.IsNullOrEmpty(_html))
        {
            var doc = new HtmlDocument();
            doc.LoadHtml(GetHtml);
            var root = doc.DocumentNode.Descendants().ToList();

            foreach (var tag in root)
            {
                if (tag.NodeType != HtmlNodeType.Element)
                    continue;
                else
                {
                    Console.ForegroundColor = ConsoleColor.Green;
                    Console.WriteLine("Name:" + tag.OriginalName + " " + "Type:" + tag.NodeType);
                    Console.ForegroundColor = ConsoleColor.White;
                }
            }
        }
        else
        {
            Console.ForegroundColor = ConsoleColor.Red;
            Console.WriteLine("None");
            Console.ForegroundColor = ConsoleColor.White;
        }
    }

    public void PrintEncodedHtml()
    {
        if (!String.IsNullOrEmpty(_html))
        {
            var doc = new HtmlDocument();
            doc.LoadHtml(GetHtml);
            var root = doc.DocumentNode.Descendants().ToList();

```

```

foreach (var tag in root)
{
    if (tag.NodeType != HtmlNodeType.Element)
        continue;
    else
    {
        if (tag.HasAttributes)
        {
            if (tag.Attributes["href"]!=null)
            {
                var originalHref = tag.Attributes["href"].Value;
                Console.ForegroundColor = ConsoleColor.DarkYellow;

                Console.WriteLine($"Original href:{originalHref}");

                Console.WriteLine($"Encoded href:{EncodeHtml(originalHref)}");

                tag.SetAttributeValue("href", EncodeHtml(originalHref));

                var originalData = tag.InnerText;

                Console.WriteLine($"Original href data:{originalData}");

                Console.WriteLine($"Encoded href data:{EncodeHtml(originalData)}");
                Console.ForegroundColor = ConsoleColor.White;

                tag.InnerHtml=EncodeHtml(originalData);
            }

            if (tag.Attributes["class"]!=null)
            {
                var originalClass = tag.Attributes["class"].Value;
                Console.ForegroundColor = ConsoleColor.DarkYellow;

                Console.WriteLine($"Original class:{originalClass}");

                Console.WriteLine($"Encoded href:{EncodeHtml(originalClass)}");

                tag.SetAttributeValue("class", EncodeHtml(originalClass));

                var originalData = tag.InnerText;

                Console.WriteLine($"Original href data:{originalData}");

                Console.WriteLine($"Original href data:{EncodeHtml(originalData)}");
                Console.ForegroundColor = ConsoleColor.White;

                tag.InnerHtml = EncodeHtml(originalData);
            }
        }
    }
}

```

```

    }
    Console.ForegroundColor = ConsoleColor.Green;

    Console.WriteLine(doc.DocumentNode.OuterHtml);

    Console.ForegroundColor = ConsoleColor.White;

}
}

public string EncodeHtml(string data)
{
    string returnData = null;

    foreach (char c in data)
    {
        returnData += "&#" + Convert.ToInt32(c);
    }
    return returnData;
}
}
}

```

That's it big thanks let's add this to my software ! :)

itcorp - April 12, 2020, 6:39 am

(Apr 12, 2020 03:56 AM)Fhack Wrote: [\[CITE\]](#) That's it big thanks let's add this to my software ! :)
you're welcome buddy,keep up the hard work!