High-performance .NET C# HTMLparser Library
Majestic12 Namespace

Namespace hierarchy

Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTMLchunk</td>
<td>Parsed HTML token that is either text, comment, script, open or closed tag as indicated by the oType variable.</td>
</tr>
<tr>
<td>HTMLparser</td>
<td>Allows to parse HTML by splitting it into small token (HTMLchunks) such as tags, text, comments etc. Do NOT create multiple instances of this class - REUSE single instance Do NOT call same instance from multiple threads - it is NOT thread safe</td>
</tr>
</tbody>
</table>

Enumerations

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTMLchunkType</td>
<td>Type of parsed HTML chunk (token), each non-null returned chunk from HTMLparser will have oType set to one of these values</td>
</tr>
</tbody>
</table>
High-performance .NET C# HTMLparser Library
**HTMLchunk Class**

Parsed HTML token that is either text, comment, script, open or closed tag as indicated by the oType variable.

For a list of all members of this type, see [HTMLchunk Members](#).

```csharp
public class HTMLchunk : IDisposable
```

**Thread Safety**

Public static ([Shared](#) in Visual Basic) members of this type are safe for multithreaded operations. Instance members are **not** guaranteed to be thread-safe.

**Requirements**

- **Namespace:** [Majestic12](#)
- **Assembly:** HTMLparserDotNet11 (in HTMLparserDotNet11.exe)

**See Also**

- [HTMLchunk Members](#) | [Majestic12 Namespace](#)

---

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTML parser Library
**HTMLchunk Members**

**HTMLchunk overview**

**Public Static Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MakeSafeParamValue</td>
<td>Makes parameter value safe to be used in param - this will check for any conflicting quote chars, but not full entity-encoding</td>
</tr>
</tbody>
</table>

**Public Instance Constructors**

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTMLchunk Constructor</td>
<td>Initialises new HTMLchunk</td>
</tr>
</tbody>
</table>

**Public Instance Fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bHashMode</td>
<td>If true then tag params will be kept in a hash rather than in a fixed size arrays. This will be slow down parsing, but make it easier to use.</td>
</tr>
<tr>
<td>cParamChars</td>
<td>Character used to quote param's value: it is taken actually from parsed HTML</td>
</tr>
<tr>
<td>iChunkLength</td>
<td>Length of the chunk in bHTML data array</td>
</tr>
<tr>
<td>iChunkOffset</td>
<td>Offset in bHTML data array at which this chunk starts</td>
</tr>
<tr>
<td>iParams</td>
<td>Number of parameters and values stored in sParams array, OR in oParams hashtable if bHashMode is true</td>
</tr>
<tr>
<td>oEnc</td>
<td>Encoder to be used for conversion of binary data into strings, Encoding.Default is used by default, but it can be</td>
</tr>
</tbody>
</table>
changed if top level user of the parser detects that encoding was different

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>oHTML</td>
<td>For TAGS: it stores raw HTML that was parsed to generate thus chunk will be here UNLESS HTML parser was configured not to store it there as it can improve performance For TEXT or COMMENTS: actual text or comments - you MUST call Finalise(); first.</td>
</tr>
<tr>
<td>oParams</td>
<td>Hashtable with tag parameters: keys are param names and values are param values. ONLY used if bHashMode is set to TRUE.</td>
</tr>
<tr>
<td>oType</td>
<td>Chunk type showing whether its text, open or close tag, comments or script. WARNING: if type is comments or script then you have to manually call Finalise(); method in order to have actual text of comments/scripts in oHTML variable</td>
</tr>
<tr>
<td>sParams</td>
<td>Param names will be stored here - actual number is in iParams. ONLY used if bHashMode is set to FALSE.</td>
</tr>
<tr>
<td>sTag</td>
<td>If its open/close tag type then this is where lowercased Tag will be kept</td>
</tr>
<tr>
<td>sValues</td>
<td>Param values will be stored here - actual number is in iParams. ONLY used if</td>
</tr>
</tbody>
</table>
bHashMode is set to FALSE.

## Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AddParam</strong></td>
<td>Adds tag parameter to the chunk</td>
</tr>
<tr>
<td><strong>Clear</strong></td>
<td>Clears chunk preparing it for</td>
</tr>
<tr>
<td><strong>ConvertParamsToHash</strong></td>
<td>This function will convert parameters stored in sParams/sValues arrays into oParams hash Useful if generally parsing is done when bHashMode is FALSE. Hash operations are not the fastest, so its best not to use this function.</td>
</tr>
<tr>
<td><strong>Dispose</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Equals</strong> (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>GenerateHTML</strong></td>
<td>Generates HTML based on current chunk's data Note: this is not a high performance method and if you want ORIGINAL HTML that was parsed to create this chunk then use relevant HTMLparser method to obtain such HTML</td>
</tr>
<tr>
<td><strong>GenerateParamHTML</strong></td>
<td>Generates HTML for param/value pair</td>
</tr>
<tr>
<td><strong>GenerateParamsHTML</strong></td>
<td>Generates HTML for params in this chunk</td>
</tr>
<tr>
<td><strong>GetHashCode</strong> (inherited from Object)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data</td>
</tr>
</tbody>
</table>
structures like a hash table.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>GetParamValue</code></td>
<td>Returns value of a parameter</td>
</tr>
<tr>
<td><code>GetType</code> (inherited from Object)</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td><code>SetEncoding</code></td>
<td>Sets encoding to be used for conversion of binary data into string</td>
</tr>
<tr>
<td><code>ToString</code> (inherited from Object)</td>
<td>Returns a String that represents the current Object.</td>
</tr>
</tbody>
</table>

### Protected Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Finalize</code> (inherited from Object)</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><code>MemberwiseClone</code> (inherited from Object)</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
</tbody>
</table>

### See Also

- [HTMLchunk Class](#) | [Majestic12 Namespace](#)  

---

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLchunk Constructor

Initialises new HTMLchunk

```csharp
public HTMLchunk(
    bool p_bHashMode
);
```

Parameters

- `p_bHashMode`
  Sets

See Also

- [HTMLchunk Class](#)
- [Majestic12 Namespace](#)
- [bHashMode](#)

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
### HTMLchunk Fields

The fields of the **HTMLchunk** class are listed below. For a complete list of **HTMLchunk** class members, see the [HTMLchunk Members](#) topic.

#### Public Instance Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bHashMode</code></td>
<td>If true then tag params will be kept in a hash rather than in a fixed size arrays. This will be slow down parsing, but make it easier to use.</td>
</tr>
<tr>
<td><code>cParamChars</code></td>
<td>Character used to quote param's value: it is taken actually from parsed HTML</td>
</tr>
<tr>
<td><code>iChunkLength</code></td>
<td>Length of the chunk in bHTML data array</td>
</tr>
<tr>
<td><code>iChunkOffset</code></td>
<td>Offset in bHTML data array at which this chunk starts</td>
</tr>
<tr>
<td><code>iParams</code></td>
<td>Number of parameters and values stored in sParams array, OR in oParams hashtable if bHashMode is true</td>
</tr>
<tr>
<td><code>oEnc</code></td>
<td>Encoder to be used for conversion of binary data into strings, Encoding.Default is used by default, but it can be changed if top level user of the parser detects that encoding was different</td>
</tr>
<tr>
<td><code>oHTML</code></td>
<td>For TAGS: it stores raw HTML that was parsed to generate thus chunk will be here UNLESS HTMLparser was configured not to store it there</td>
</tr>
</tbody>
</table>
as it can improve performance
For TEXT or COMMENTS: actual text or comments - you MUST call Finalise(); first.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>oParams</strong> Hashtable with tag parameters: keys are param names and values are param values. ONLY used if bHashMode is set to TRUE.</td>
</tr>
<tr>
<td></td>
<td><strong>oType</strong> Chunk type showing whether its text, open or close tag, comments or script. WARNING: if type is comments or script then you have to manually call Finalise(); method in order to have actual text of comments/scripts in oHTML variable</td>
</tr>
<tr>
<td></td>
<td><strong>sParams</strong> Param names will be stored here - actual number is in iP Rams. ONLY used if bHashMode is set to FALSE.</td>
</tr>
<tr>
<td></td>
<td><strong>sTag</strong> If its open/close tag type then this is where lowercased Tag will be kept</td>
</tr>
<tr>
<td></td>
<td><strong>sValues</strong> Param values will be stored here - actual number is in iP Rams. ONLY used if bHashMode is set to FALSE.</td>
</tr>
</tbody>
</table>

See Also

[HTMLchunk Class](#) | [Majestic12 Namespace](#)

---

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLchunk.bHashMode Field

If true then tag params will be kept in a hash rather than in a fixed size arrays. This will be slow down parsing, but make it easier to use.

```java
public bool bHashMode;
```

See Also

[HTMLchunk Class](#) | [Majestic12 Namespace](#)
---

[Send comments on this topic.](#)

[Copyright (c) Alex Chudnovsky, Majestic-12 (UK)](#)
High-performance .NET C# HTMLparser Library
HTMLchunk.cParamChars Field

Character used to quote param's value: it is taken actually from parsed HTML

```java
public byte[] cParamChars;
```

See Also

- HTMLchunk Class
- Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLchunk.iChunkLength Field

Length of the chunk in bHTML data array

```java
public int iChunkLength;
```

See Also

HTMLchunk Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTML parser Library
Offset in bHTML data array at which this chunk starts

```java
public int iChunkOffset;
```

See Also

- [HTMLchunk Class](#) | [Majestic12 Namespace](#)

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTML parser Library
HTMLchunk.iparams Field

Number of parameters and values stored in sParams array, OR in oParams hashtable if bHashMode is true

```csharp
public int iparams;
```

See Also

HTMLchunk Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLchunk.oEnc Field

Encoder to be used for conversion of binary data into strings, Encoding.Default is used by default, but it can be changed if top level user of the parser detects that encoding was different

```java
public Encoding oEnc;
```

See Also

[HTMLchunk Class](#) | [Majestic12 Namespace](#)

---

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
**HTMLchunk.oHTML Field**

For TAGS: it stores raw HTML that was parsed to generate thus chunk will be here UNLESS HTMLparser was configured not to store it there as it can improve performance.

For TEXT or COMMENTS: actual text or comments - you MUST call Finalise(); first.

```csharp
public string oHTML;
```

**See Also**

[HTMLchunk Class](#) | [Majestic12 Namespace](#)

[Send comments on this topic.](#)

[Copyright (c) Alex Chudnovsky, Majestic-12 (UK)](#)
High-performance .NET C# HTMLparser Library
**HTMLchunk.oParams Field**

Hashtable with tag parameters: keys are param names and values are param values. ONLY used if bHashMode is set to TRUE.

```java
public Hashtable oParams;
```

See Also

HTMLchunk Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTML parser Library
**HTMLchunk.oType Field**

Chunk type showing whether its text, open or close tag, comments or script. WARNING: if type is comments or script then you have to manually call Finalise(); method in order to have actual text of comments/scripts in oHTML variable

```java
public HTMLchunkType oType;
```

**See Also**

- [HTMLchunk Class](#) | [Majestic12 Namespace](#)

---

[Send comments on this topic.](#)

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
**Param names will be stored here - actual number is in iParams. ONLY used if bHashMode is set to FALSE.**

```java
public string[] sParams;
```

**See Also**

[HTMLchunk Class](#) | [Majestic12 Namespace](#)

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
**HTMLchunk.sTag Field**

If its open/close tag type then this is where lowercased Tag will be kept

```java
public string sTag;
```

**See Also**

[HTMLchunk Class](#) | [Majestic12 Namespace](#)

---

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLchunk.sValues Field

Param values will be stored here - actual number is in iParams. ONLY used if bHashMode is set to FALSE.

```java
public string[] sValues;
```

See Also

[HTMLchunk Class](#) | [Majestic12 Namespace](#)

Send comments on this topic.

[Copyright (c) Alex Chudnovsky, Majestic-12 (UK)](#)
High-performance .NET C# HTML parser Library
# HTMLchunk Methods

The methods of the **HTMLchunk** class are listed below. For a complete list of **HTMLchunk** class members, see the **HTMLchunk Members** topic.

## Public Static Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$MakeSafeParamValue</td>
<td>Makes parameter value safe to be used in param - this will check for any conflicting quote chars, but not full entity-encoding</td>
</tr>
</tbody>
</table>

## Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddParam</td>
<td>Adds tag parameter to the chunk</td>
</tr>
<tr>
<td>Clear</td>
<td>Clears chunk preparing it for</td>
</tr>
<tr>
<td>ConvertParamsToHash</td>
<td>This function will convert parameters stored in sParams/sValues arrays into oParams hash Useful if generally parsing is done when bHashMode is FALSE. Hash operations are not the fastest, so its best not to use this function.</td>
</tr>
<tr>
<td>Dispose</td>
<td></td>
</tr>
<tr>
<td>Equals (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td>GenerateHTML</td>
<td>Generates HTML based on current chunk's data Note: this is not a high performance method and if you want ORIGINAL HTML that was</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>GenerateParamHTML</strong></td>
<td>Generates HTML for param/value pair</td>
</tr>
<tr>
<td><strong>GenerateParamsHTML</strong></td>
<td>Generates HTML for params in this chunk</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td><strong>GetParamValue</strong></td>
<td>Returns value of a parameter</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>SetEncoding</strong></td>
<td>Sets encoding to be used for conversion of binary data into string</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from <strong>Object</strong>)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from <strong>Object</strong>)</td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from <strong>Object</strong>)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**See Also**

[HTMLchunk Class] | [Majestic12 Namespace]

Send comments on this topic.
Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
**HTMLchunk.AddParam Method**

Adds tag parameter to the chunk

```csharp
public void AddParam(
    string sParam,
    string sValue,
    byte cParamChar
);
```

**Parameters**

- **sParam**
  - Parameter name (ie color)

- **sValue**
  - Value of the parameter (ie white)

- **cParamChar**

**See Also**

- [HTMLchunk Class](#) | [Majestic12 Namespace](#)

---

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTML parser Library
HTMLchunk.Clear Method

Clears chunk preparing it for

```csharp
public void Clear();
```

See Also

[HTMLchunk Class](#) | [Majestic12 Namespace](#) | [Send comments on this topic.](#) | [Copyright (c) Alex Chudnovsky, Majestic-12 (UK)](#)

---
High-performance .NET C# HTML parser Library
This function will convert parameters stored in sParams/sValues arrays into oParams hash. Useful if generally parsing is done when bHashMode is FALSE. Hash operations are not the fastest, so it's best not to use this function.

```csharp
public void ConvertParamsToHash();
```

See Also

HTMLchunk Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky. Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLchunk.Dispose Method

```csharp
public void Dispose();
```

Implements

`IDisposable.Dispose`

See Also

[HTMLchunk Class] | [Majestic12 Namespace]

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
Generates HTML based on current chunk's data Note: this is not a high performance method and if you want ORIGINAL HTML that was parsed to create this chunk then use relevant HTMLparser method to obtain such HTML

```csharp
public string GenerateHTML();
```

Return Value

HTML equivalent of this chunk

See Also

[HTMLchunk Class](#) | [Majestic12 Namespace](#)

---

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTML parser Library
Generates HTML for param/value pair

```csharp
public string GenerateParamHTML(
    string sParam,
    string sValue,
    char cParamChar
);
```

**Parameters**

- **sParam**
  - Param

- **sValue**
  - Value (empty if not specified)

- **cParamChar**

**Return Value**

- String with HTML

**See Also**

- [HTMLchunk Class](#) | [Majestic12 Namespace](#)

---

**Send comments on this topic.**

**Copyright (c) Alex Chudnovsky, Majestic-12 (UK)**
High-performance .NET C# HTMLparser Library
HTMLchunk.GenerateParamsHTML Method

Generates HTML for params in this chunk

```csharp
public string GenerateParamsHTML();
```

Return Value
String with HTML corresponding to params

See Also
HTMLchunk Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLchunk.GetParamValue Method

Returns value of a parameter

```csharp
public string GetParamValue(string sParam);
```

Parameters

- `sParam` Parameter

Return Value

Parameter value or empty string

See Also

- [HTMLchunk Class](#) | [Majestic12 Namespace](#)

---

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLchunk.MakeSafeParamValue Method

Makes parameter value safe to be used in param - this will check for any conflicting quote chars, but not full entity-encoding

```csharp
public static string MakeSafeParamValue(
    string sLine,
    char cQuoteChar
);
```

Parameters

`sLine`
Line of text

`cQuoteChar`
Quote char used in param - any such chars in text will be entity-encoded

Return Value
Safe text to be used as param's value

See Also

HTMLchunk Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
Sets encoding to be used for conversion of binary data into string

```csharp
public void SetEncoding(
    Encoding p_oEnc
);
```

Parameters

- `p_oEnc` Encoding object

See Also

- HTMLchunk Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
### HTMLchunkType Enumeration

Type of parsed HTML chunk (token), each non-null returned chunk from HTMLparser will have oType set to one of these values.

```java
public enum HTMLchunkType {
    Text, OpenTag, CloseTag, Comment, Script
}
```

#### Members

<table>
<thead>
<tr>
<th>Member Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text</td>
<td>Text data from HTML</td>
</tr>
<tr>
<td>OpenTag</td>
<td>Open tag, possibly with attributes</td>
</tr>
<tr>
<td>CloseTag</td>
<td>Closed tag (it may still have attributes)</td>
</tr>
<tr>
<td>Comment</td>
<td>Comment tag () depending on HTMLparser boolean flags you may have: a) nothing to oHTML variable - for faster performance, call SetRawHTML function in parser b) data BETWEEN tags (but not including comment tags themselves) - DEFAULT c) complete RAW HTML representing data between tags and tags themselves (same as you get in a) when you call SetRawHTML function)</td>
</tr>
<tr>
<td>Script</td>
<td>Script tag () depending on HTMLparser boolean flags a) nothing to oHTML variable - for faster performance, call SetRawHTML function in parser b) data BETWEEN tags (but not including comment tags</td>
</tr>
</tbody>
</table>
Requirements

Namespace: Majestic12
Assembly: HTMLparserDotNet11 (in HTMLparserDotNet11.exe)

See Also

Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLparser Class

Allows to parse HTML by splitting it into small token (HTMLchunks) such as tags, text, comments etc. Do NOT create multiple instances of this class - REUSE single instance Do NOT call same instance from multiple threads - it is NOT thread safe

For a list of all members of this type, see HTMLparser Members.

System.Object Majestic12.HTMLparser

| public class HTMLparser : IDisposable |

Thread Safety

Public static (Shared in Visual Basic) members of this type are safe for multithreaded operations. Instance members are not guaranteed to be thread-safe.

Requirements

Namespace: Majestic12
Assembly: HTMLparserDotNet11 (in HTMLparserDotNet11.exe)

See Also

HTMLparser Members | Majestic12 Namespace

Send comments on this topic.
Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
# HTMLparser Members

## HTMLparser overview

### Public Static Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✧ <strong>CalculateWidth</strong></td>
<td>Parses WIDTH param and calculates width</td>
</tr>
<tr>
<td>✧ <strong>DecodeEntities</strong></td>
<td>This function will decode any entities found in a string - not fast!</td>
</tr>
<tr>
<td>✧ <strong>IsBiggerFont</strong></td>
<td>Checks if first font is bigger than the second</td>
</tr>
<tr>
<td>✧ <strong>IsEqualOrBiggerFont</strong></td>
<td>Checks if first font is equal or bigger than the second</td>
</tr>
<tr>
<td>✧ <strong>ParseFontSize</strong></td>
<td>Parses font's tag size param</td>
</tr>
</tbody>
</table>

### Public Instance Constructors

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✧ HTMLparser</td>
<td>Overloaded. Initializes a new instance of the HTMLparser class.</td>
</tr>
</tbody>
</table>

### Public Instance Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✧ bAutoExtractBetweenTagsOnly</td>
<td>If true (and either bAutoKeepComments or bAutoKeepScripts is true), then oHTML will be set to data BETWEEN tags excluding those tags themselves, as otherwise FULL HTML will be set, ie: &quot; but if this is set to true then only ' comments ' will be returned</td>
</tr>
<tr>
<td><strong>bAutoKeepComments</strong></td>
<td>If true (default) then HTML for comments tags themselves AND between them will be set to oHTML variable, otherwise it will be empty but you can always set it later</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>bAutoKeepScripts</strong></td>
<td>If true (default: false) then HTML for script tags themselves AND between them will be set to oHTML variable, otherwise it will be empty but you can always set it later</td>
</tr>
<tr>
<td><strong>bAutoMarkClosedTagsWithParamsAsOpen</strong></td>
<td>Long winded name... by default if tag is closed BUT it has got parameters then we will consider it open tag, this is not right for proper XML parsing</td>
</tr>
<tr>
<td><strong>bCompressWhiteSpaceBeforeTag</strong></td>
<td>If true (default), then all whitespace before TAG starts will be compressed to single space char (32 or 0x20) this makes parser run a bit faster, if you need exact whitespace before tags then change this flag to FALSE</td>
</tr>
<tr>
<td><strong>oHE</strong></td>
<td>Heuristics engine used by Tag Parser to</td>
</tr>
</tbody>
</table>
quickly match known tags and attribute names, can be disabled or you can add more tags to it to fit your most likely cases, it is currently tuned for HTML

### Public Instance Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bDecodeEntities</td>
<td>If true (default) then heuristics engine will be used to match tags and attributes quicker, it is possible to add new tags to it, oHE</td>
</tr>
<tr>
<td>bDecodeMiniEntities</td>
<td></td>
</tr>
<tr>
<td>bEnableHeuristics</td>
<td></td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ChangeToEntities</td>
<td>Parses line and changes known entity characters into proper HTML entries</td>
</tr>
<tr>
<td>CleanUp</td>
<td>Cleans up parser in preparation for next parsing</td>
</tr>
<tr>
<td>Close</td>
<td>Closes object and releases all allocated resources</td>
</tr>
<tr>
<td>Dispose</td>
<td></td>
</tr>
<tr>
<td>Equals (inherited from Object)</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td>GetHashCode (inherited from Object)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>GetType</strong> (inherited from <strong>Object</strong>)</td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>Init</strong></td>
<td>Overloaded. Initialises parses with HTML to be parsed from provided string</td>
</tr>
<tr>
<td><strong>InitMiniEntities</strong></td>
<td>Inits mini-entities mode: only &quot;nbsp&quot; will be converted into space, all other entities will be left as is</td>
</tr>
<tr>
<td><strong>LoadFromFile</strong></td>
<td>Loads HTML from file</td>
</tr>
<tr>
<td><strong>ParseNext</strong></td>
<td>Parses next chunk and returns it with</td>
</tr>
<tr>
<td><strong>ParseNextTag</strong></td>
<td>Returns next tag or null if end of document, text will be ignored completely</td>
</tr>
<tr>
<td><strong>Reset</strong></td>
<td>Resets current parsed data to start</td>
</tr>
<tr>
<td><strong>SetChunkHashMode</strong></td>
<td>Sets chunk param hash mode</td>
</tr>
<tr>
<td><strong>SetEncoding</strong></td>
<td>Overloaded. Sets encoding</td>
</tr>
<tr>
<td><strong>SetRawHTML</strong></td>
<td>Sets oHTML variable in a chunk to the raw HTML that was parsed for that chunk.</td>
</tr>
<tr>
<td><strong>ToString</strong> (inherited from <strong>Object</strong>)</td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> (inherited from <strong>Object</strong>)</td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong> (inherited from <strong>Object</strong>)</td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>
See Also

HTMLparser Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTML parser Library
HTMLparser Constructor

Initializes a new instance of the HTMLparser class.

Overload List

Initializes a new instance of the HTMLparser class.

```java
public HTMLparser();
```
Constructs parser object using provided HTML as source for parsing

```java
public HTMLparser(string);
```

See Also

HTMLparser Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLparser Constructor ()

Initializes a new instance of the HTMLparser class.

```java
public HTMLparser();
```

See Also

- HTMLparser Class
- Majestic12 Namespace
- HTMLparser Constructor Overload List
- Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTML parser Library
Constructs parser object using provided HTML as source for parsing

```java
public HTMLparser(
    string p_oHTML
);
```

Parameters

*p_oHTML*

See Also

- HTMLparser Class
- Majestic12 Namespace
- HTMLparser Constructor Overload List

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
### HTMLparser Fields

The fields of the **HTMLparser** class are listed below. For a complete list of **HTMLparser** class members, see the [HTMLparser Members](#) topic.

#### Public Instance Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bAutoExtractBetweenTagsOnly</code></td>
<td>If true (and either <code>bAutoKeepComments</code> or <code>bAutoKeepScripts</code> is true), then <code>oHTML</code> will be set to data BETWEEN tags excluding those tags themselves, as otherwise FULL HTML will be set, ie: &quot; but if this is set to true then only ' comments ' will be returned</td>
</tr>
<tr>
<td><code>bAutoKeepComments</code></td>
<td>If true (default) then HTML for comments tags themselves AND between them will be set to <code>oHTML</code> variable, otherwise it will be empty but you can always set it later</td>
</tr>
<tr>
<td><code>bAutoKeepScripts</code></td>
<td>If true (default: false) then HTML for script tags themselves AND between them will be set to <code>oHTML</code> variable, otherwise it will be empty but you can always set it later</td>
</tr>
<tr>
<td>Long winded name...</td>
<td></td>
</tr>
<tr>
<td><strong>bAutoMarkClosedTagsWithParamsAsOpen</strong></td>
<td>by default if tag is closed BUT it has got parameters then we will consider it open tag, this is not right for proper XML parsing</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>♭ bCompressWhiteSpaceBeforeTag</strong></td>
<td>If true (default), then all whitespace before TAG starts will be compressed to single space char (32 or 0x20) this makes parser run a bit faster, if you need exact whitespace before tags then change this flag to FALSE</td>
</tr>
<tr>
<td><strong>♭ oHE</strong></td>
<td>Heuristics engine used by Tag Parser to quickly match known tags and attribute names, can be disabled or you can add more tags to it to fit your most likely cases, it is currently tuned for HTML</td>
</tr>
</tbody>
</table>

**See Also**

HTMLparser Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
If true (and either bAutoKeepComments or bAutoKeepScripts is true), then oHTML will be set to data BETWEEN tags excluding those tags themselves, as otherwise FULL HTML will be set, ie: " but if this is set to true then only ' comments ' will be returned

```csharp
public bool bAutoExtractBetweenTagsOnly;
```

See Also

- HTMLparser Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTML parser Library
**HTMLparser.bAutoKeepComments Field**

If true (default) then HTML for comments tags themselves AND between them will be set to oHTML variable, otherwise it will be empty but you can always set it later

```csharp
public bool bAutoKeepComments;
```

See Also

[HTMLparser Class](#) | [Majestic12 Namespace](#)
---

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTML parser Library
**HTMLparser.bAutoKeepScripts Field**

If true (default: false) then HTML for script tags themselves AND between them will be set to oHTML variable, otherwise it will be empty but you can always set it later

```csharp
public bool bAutoKeepScripts;
```

See Also

[HTMLparser Class](#) | [Majestic12 Namespace](#)

[Send comments on this topic.](#)

[Copyright (c) Alex Chudnovsky, Majestic-12 (UK)](#)
High-performance .NET C# HTML parser Library
HTMLparser.bAutoMarkClosedTagsWithParamsAsOpen Field

Long winded name... by default if tag is closed BUT it has got parameters then we will consider it open tag, this is not right for proper XML parsing

```
public bool bAutoMarkClosedTagsWithParamsAsOpen;
```

See Also

- HTMLparser Class
- Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
If true (default), then all whitespace before TAG starts will be compressed to single space char (32 or 0x20) this makes parser run a bit faster, if you need exact whitespace before tags then change this flag to FALSE.

```csharp
public bool bCompressWhiteSpaceBeforeTag;
```

See Also

[HTMLparser Class](#) | [Majestic12 Namespace](#)

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLparser.oHE Field

Heuristics engine used by Tag Parser to quickly match known tags and attribute names, can be disabled or you can add more tags to it to fit your most likely cases, it is currently tuned for HTML

```java
public HTMLheuristics oHE;
```

See Also

HTMLparser Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
The properties of the HTMLparser class are listed below. For a complete list of HTMLparser class members, see the HTMLparser Members topic.

**Public Instance Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bDecodeEntities</code></td>
<td>If true (default) then heuristics engine will be used to match tags and attributes quicker, it is possible to add new tags to it.</td>
</tr>
<tr>
<td><code>bDecodeMiniEntities</code></td>
<td></td>
</tr>
<tr>
<td><code>bEnableHeuristics</code></td>
<td></td>
</tr>
</tbody>
</table>

**See Also**

[HTMLparser Class] | [Majestic12 Namespace]

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
public bool bDecodeEntities {get; set;}

See Also

HTMLparser Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
public bool bDecodeMiniEntities {get; set;}

See Also

HTMLparser Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLparser.bEnableHeuristics Property

If true (default) then heuristics engine will be used to match tags and attributes quicker, it is possible to add new tags to it, oHE

```csharp
public bool bEnableHeuristics {get; set;}
```

See Also

- [HTMLparser Class](#) | [Majestic12 Namespace](#)

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
**HTMLparser Methods**

The methods of the **HTMLparser** class are listed below. For a complete list of **HTMLparser** class members, see the **HTMLparser Members** topic.

### Public Static Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$CalculateWidth</td>
<td>Parses WIDTH param and calculates width</td>
</tr>
<tr>
<td>$DecodeEntities</td>
<td>This function will decode any entities found in a string - not fast!</td>
</tr>
<tr>
<td>$IsBiggerFont</td>
<td>Checks if first font is bigger than the second</td>
</tr>
<tr>
<td>$IsEqualOrBiggerFont</td>
<td>Checks if first font is equal or bigger than the second</td>
</tr>
<tr>
<td>$ParseFontSize</td>
<td>Parses font's tag size param</td>
</tr>
</tbody>
</table>

### Public Instance Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ChangeToEntities</td>
<td>Parses line and changes known entity characters into proper HTML entries</td>
</tr>
<tr>
<td>$CleanUp</td>
<td>Cleans up parser in preparation for next parsing</td>
</tr>
<tr>
<td>$Close</td>
<td>Closes object and releases all allocated resources</td>
</tr>
<tr>
<td>$Dispose</td>
<td></td>
</tr>
<tr>
<td>$Equals (inherited from <strong>Object</strong>)</td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td>$GetHashCode (inherited from <strong>Object</strong>)</td>
<td>Serves as a hash function for a particular type, suitable for use in hashing algorithms and data</td>
</tr>
</tbody>
</table>
structures like a hash table.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GetType</strong> <em>(inherited from Object)</em></td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td><strong>Init</strong></td>
<td>Overloaded. Initialises parses with HTML to be parsed from provided string</td>
</tr>
<tr>
<td><strong>InitMiniEntities</strong></td>
<td>Inits mini-entities mode: only &quot;nbsp&quot; will be converted into space, all other entities will be left as is</td>
</tr>
<tr>
<td><strong>LoadFromFile</strong></td>
<td>Loads HTML from file</td>
</tr>
<tr>
<td><strong>ParseNext</strong></td>
<td>Parses next chunk and returns it with</td>
</tr>
<tr>
<td><strong>ParseNextTag</strong></td>
<td>Returns next tag or null if end of document, text will be ignored completely</td>
</tr>
<tr>
<td><strong>Reset</strong></td>
<td>Resets current parsed data to start</td>
</tr>
<tr>
<td><strong>SetChunkHashMode</strong></td>
<td>Sets chunk param hash mode</td>
</tr>
<tr>
<td><strong>SetEncoding</strong></td>
<td>Overloaded. Sets encoding</td>
</tr>
<tr>
<td><strong>SetRawHTML</strong></td>
<td>Sets oHTML variable in a chunk to the raw HTML that was parsed for that chunk.</td>
</tr>
<tr>
<td><strong>ToString</strong> <em>(inherited from Object)</em></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>

**Protected Instance Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong> <em>(inherited from Object)</em></td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
</tbody>
</table>
MemberwiseClone (inherited from Object)

Creates a shallow copy of the current Object.

See Also

HTMLparser Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLparser.CalculateWidth Method

Parses WIDTH param and calculates width

```csharp
public static int CalculateWidth(
    string sWidth,
    int iAvailWidth,
    ref bool bRelative)
```

Parameters

- `sWidth`  
  WIDTH param from tag

- `iAvailWidth`  
  Currently available width for relative calculations, if negative width will be returned as is

- `bRelative`  
  Flag that will be set to true if width was relative

Return Value

Width in pixels

See Also

- [HTMLparser Class](#) | [Majestic12 Namespace](#)

---

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
Parses line and changes known entity characters into proper HTML entities

```csharp
public string ChangeToEntities(
    string sLine
);
```

Parameters

*sLine*

Line of text

Return Value

Line of text with proper HTML entities

See Also

HTMLparser Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLparser.CleanUp Method

Cleans up parser in preparation for next parsing

public void CleanUp();

See Also

HTMLparser Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
**HTMLparser.Close Method**

Closes object and releases all allocated resources

```java
public void Close();
```

**See Also**

[HTMLparser Class](#) | [Majestic12 Namespace](#)

[Send comments on this topic.](#)

[Copyright (c) Alex Chudnovsky, Majestic-12 (UK)](#)
High-performance .NET C# HTMLparser Library
This function will decode any entities found in a string - not fast!

```
public static string DecodeEntities(
    string sData
);
```

Return Value

Possibly decoded string

See Also

- [HTMLparser Class](#) | [Majestic12 Namespace](#)

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTML parser Library
public void Dispose();

Implements
IDisposable.Dispose

See Also
HTMLparser Class | Majestic12 Namespace

Send comments on this topic.
Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
HTMLparser.Init Method

Initialises parses with HTML to be parsed from provided data buffer: this is best in terms of correctness of parsing of various encodings that can be used in HTML

Overload List

Initialises parses with HTML to be parsed from provided data buffer: this is best in terms of correctness of parsing of various encodings that can be used in HTML

public void Init(byte[]);

Initialises parses with HTML to be parsed from provided string

public void Init(string);

See Also

HTMLparser Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTML parser Library
Initialises parses with HTML to be parsed from provided data buffer: this is best in terms of correctness of parsing of various encodings that can be used in HTML

```csharp
public void Init(
    byte[] p_bHTML
);
```

**Parameters**

- `p_bHTML` Data buffer with HTML in it

**See Also**

- [HTMLparser Class](#) | [Majestic12 Namespace](#) | [HTMLparser.Init Overload List](#)

[Send comments on this topic.](#)

[Copyright (c) Alex Chudnovsky, Majestic-12 (UK)](#)
High-performance .NET C# HTMLparser Library
HTMLparser.Init Method (String)

Initialises parses with HTML to be parsed from provided string

```java
public void Init(
    string p_oHTML
);
```

Parameters

- `p_oHTML` 
  String with HTML in it

See Also

- HTMLparser Class | Majestic12 Namespace | HTMLparser.Init Overload List
- Send comments on this topic.
- Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLparser.InitMiniEntities Method

Inits mini-entities mode: only "nbsp" will be converted into space, all other entities will be left as is

public void InitMiniEntities();

See Also

HTMLparser Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
**HTMLParser.IsBiggerFont Method**

Checks if first font is bigger than the second

```csharp
public static bool IsBiggerFont(
    FontSize oFont1,
    FontSize oFont2
);
```

**Parameters**

- **oFont1**
  - Font #1

- **oFont2**
  - Font #2

**Return Value**

True if Font #1 bigger than the second, false otherwise

**See Also**

- [HTMLParser Class](#) | [Majestic12 Namespace](#)
High-performance .NET C# HTML parser Library
HTMLparser.IsEqualOrBiggerFont Method

Checks if first font is equal or bigger than the second

```csharp
public static bool IsEqualOrBiggerFont(
    FontSize oFont1,
    FontSize oFont2
);
```

Parameters

- `oFont1`  
  Font #1

- `oFont2`  
  Font #2

Return Value

True if Font #1 equal or bigger than the second, false otherwise

See Also

- [HTMLparser Class](#) | [Majestic12 Namespace](#)

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
| High-performance .NET C# HTML parser Library |
**HTMLparser.LoadFromFile Method**

Loads HTML from file

```csharp
public void LoadFromFile(
    string sFileName
);
```

**Parameters**

- **sFileName**
  - Full filename

**See Also**

- [HTMLparser Class](#) | [Majestic12 Namespace](#)

---

*Send comments on this topic.*

*Copyright (c) Alex Chudnovsky, Majestic-12 (UK)*
High-performance .NET C# HTMLparser Library
HTMLparser.ParseFontSize Method

Parses font's tag size param

```java
public static FontSize ParseFontSize(
    string sSize,
    FontSize oCurSize
);
```

Parameters

- **sSize**
  - String value of the size param
- **oCurSize**

Return Value

Relative size of the font size or Unknown if it was not determined

See Also

- [HTMLparser Class](#) | [Majestic12 Namespace](#)

---

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
HTMLparser.ParseNext Method

Parses next chunk and returns it with

```csharp
public HTMLchunk ParseNext();
```

Return Value

HTMLchunk or null if end of data reached

See Also

HTMLparser Class | Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTML parser Library
**HTMLparser.ParseNextTag Method**

Returns next tag or null if end of document, text will be ignored completely

```java
public HTMLchunk ParseNextTag();
```

**Return Value**

Tag chunk or null

**See Also**

[HTMLparser Class](#) | [Majestic12 Namespace](#)

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTML parser Library
HTMLparser.Reset Method

Resets current parsed data to start

```java
public void Reset();
```

See Also

[HTMLparser Class] [Majestic12 Namespace]

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
Sets chunk param hash mode

```csharp
public void SetChunkHashMode(bool bHashMode);
```

**Parameters**

*bHashMode*
- If true then tag's params will be kept in Chunk's hashtable (slower), otherwise kept in arrays (sParams/sValues)

**See Also**

[HTMLparser Class](#) | [Majestic12 Namespace](#)

---

*Send comments on this topic.*

[Copyright (c) Alex Chudnovsky, Majestic-12 (UK)](#)
High-performance .NET C# HTMLparser Library
**HTMLparser.SetEncoding Method**

Sets current encoding in format used in HTTP headers and HTML META tags

**Overload List**

Sets current encoding in format used in HTTP headers and HTML META tags

- `public bool SetEncoding(string);`

Sets encoding

- `public void SetEncoding(Encoding);`

**See Also**

- [HTMLparser Class](#) | [Majestic12 Namespace](#)

---

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
**HTMLparser.SetEncoding Method (String)**

Sets current encoding in format used in HTTP headers and HTML META tags

```csharp
public bool SetEncoding(
    string sCharSet
);
```

**Parameters**

- `sCharSet`_charset as

**Return Value**

True if encoding was set, false otherwise (in which case Default encoding will be used)

**See Also**

- [HTMLparser Class](#) | [Majestic12 Namespace](#) | [HTMLparser.SetEncoding Overload List](#)

---

[Send comments on this topic.](#)

[Copyright (c) Alex Chudnovsky, Majestic-12 (UK)](#)
High-performance .NET C# HTMLparser Library
Sets encoding

```csharp
public void SetEncoding(Encoding p_oEnc);
```

Parameters

`p_oEnc`  
Encoding object

See Also

- [HTMLparser Class](#) | [Majestic12 Namespace](#) | [HTMLparser.SetEncoding Overload List](#)

---

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
Sets oHTML variable in a chunk to the raw HTML that was parsed for that chunk.

```csharp
public void SetRawHTML(
    HTMLchunk oChunk
);
```

**Parameters**

*oChunk*

Chunk returned by ParseNext function, it must belong to the same HTMLparser that was initiated with the same HTML data that this chunk belongs to

**See Also**

[HTMLparser Class](#) | [Majestic12 Namespace](#)

---

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)
High-performance .NET C# HTMLparser Library
Majestic12 Hierarchy

System.Object

Majestic12.HTMLchunk ---- System.IDisposable
Majestic12.HTMLparser ---- System.IDisposable

System.ValueType

System.Enum ---- System.IComparable, System.IConvertible, System.IFormattable

Majestic12.HTMLchunkType

See Also

Majestic12 Namespace

Send comments on this topic.

Copyright (c) Alex Chudnovsky, Majestic-12 (UK)