Tree Builder Examples

Contents

Import

All examples require:

```
from hte import Html5TreeBuilder, Raw
tb = Html5TreeBuilder()
```

Render

Assuming the tree starts at `doc`, render the tree by:

```
print(doc.render())
```

Attributes

Most elements take attributes though not all attributes are valid for all elements.

Elements should have only one instance on any attribute. They are passed in a dictionary.

```
div = tb.div(_name="mydiv", _title="mainsection")
```

produces:

```
<div name="mydiv" title="mainsection"></div>
```

For some popular HTML5 tags:

```
tb.a("x", _href="x.html")
tb.pre("this is text")
tb.table(tb.tr(tb.th("Name"), tb.th("Age")), tb.tr(tb.td("Bill"), tb.td("23")))
```

produces:

```
<a href="x.html">x</a>
<pre>this is text</pre>
<table><tr><th>Name</th><th>Age</th></tr><tr><td>Bill</td><td>23</td></tr></table>
```

Almost all elements can take the `id` attribute. The value of an id attribute should only exist once in a document.

```
div = tb.div(_id="myid")
```

produces:

```
<div id="myid"></div>
```

Almost all elements can take the class attribute. The value of a class attribute may exist more than once in a document. A single element may have multiple class values.

```
div = tb.div(_class="light")
```

produces:

```
<div class="light"></div>
```

For multiple classes:

```
div = tb.div(_class="light bold")
```

produces:

```
<div class="light bold"></div>
```

Empty Elements

Some elements (e.g., `<br>`) do not have children and only an opening tag:

```
br = tb.br()
```

produces:

```
<br>
```

```
hr = tb.hr()
```

produces:

```
<hr>
```

Non-Element Content

Text

Textual content is added to the tree using strings. This is done simply as:

```
tb = Html5TreeBuilder()
doc = tb.html()
doc.add("My name is Bob!")
```

which produces:

```
<html>My name is Bob!</html>
```

It is necessary for textual content to be escaped properly tp render correctly in the browser. This is done automatically.

Raw

However, it is sometimes necessary to dump raw, prepared content into the tree that will be rendered as is. This is done using `Raw`.

```
tb = Html5TreeBuilder()
doc = tb.html()
doc.add(Raw("<data>raw</data>"), "<data>processed</data>")
```

produces:

```
<html><data>raw</data>&lt;data&gt;processed&lt;/data&gt;</html>
```

where the raw content is left untouched, but the simple text content is escaped.

Table

```
data = [("Mark", 25), ("Lucy", 23), ("Joe", 32), ("Jane", 41)]
table = tb.table()
table.add(tb.tr(tb.th("Name"), tb.th("Age")))
for name, age in data.items():
        table.add(tb.tr(tb.td(name), tb.td(age)))
```

or:

```
data = [("Mark", 25), ("Lucy", 23), ("Joe", 32), ("Jane", 41)]
table = tb.table()
table.add(tb.tr(tb.th("Name"), tb.th("Age")))
table.add([tb.tr[tb.td(name), tb.td(age)) for name, age in data.items()])
```

produces:

```
<table>
        <tr><th>Name</th><th>Age</th></tr>
        <tr><td>Mark</td><td>25</td></tr>
        <tr><td>Lucy</td><td>23</td></tr>
        <tr><td>Joe</td><td>32</td></tr>
        <tr><td>Jane</td><td>41</td></tr>
</table>
```

Select Menus

Drop Down

```
names = ["john", "bill", "louise"]
selectnames = tb.select([tb.option(name, _value=name) for name in names], _name="name")
```

produces:

```
<select name="name">
        <option value="john">john</option>
        <option value="bill">bill</option>
        <option value="louise">louise</option>
</select>
```

Multiple Selection

```
cities = ["Montreal", "Toronto", "Yellowknife"]
selectnames = tb.select(_name="city", _multiple=True)
selectnames.add([tb.option(city, _value=city) for city in cities])
```

produces:

```
<select name="city" multiple>
        <option value="Montreal">Montreal</option>
        <option value="Toronto">Toronto</option>
        <option value="Yellowknife">Yellowknife</option>
</select>
```

Form

```
form = tb.form(_method="get", _action="showuser.html")
form.add(tb.label("First name"), tb.input(_type="text", _size="40", _name="firstname"))
form.add(tb.label("Last name"), tb.input(_type="text", _size="40", _name="lastname"))
form.add(tb.input(_type="submit", _value="Submit"))
```

produces:

```
<form action="showuser.html" method="get">
        <label>First name</label><input name="firstname" size="40" type="text">
        <label>Last name</label><input name="lastname" size="40" type="text">
    <input type="submit" value="Submit">
</form>
```