HTML5Tagger Python module

Import html5tagger as **h5t**, or

In [1]:
```
from html5tagger import Document, E
```

You can create HTML snippets by starting with **E** (for an empty builder) and adding elements with dot notation. Content goes within parenthesis.

In [2]:
```
snippet = E.table(E.tr.th("First").th("Second").th("Third").tr.td(1).td(2).td(3))

print(snippet)  # Use print or str(snippet) to get code as text
snippet  # IPython notebooks use snippet._repr_html_() and render the output
```

```
<table><tr><th>First<th>Second<th>Third<tr><td>1<td>2<td>3</table>
```

Out[2]:

First	Second	Third
1	2	3

In contrast to E which creates snippets, calling **Document** creates a new document (i.e. it begins with a DOCTYPE declaration). A minimal head structure is created using provided title and/or urls. HTML attributes may be defined by keyword arguments.

In [3]:
```
doc = Document("Test page", lang="en")
doc.div(id="content")
with doc.ul:  # Nest using the with statement
    doc.li("Write documents in Python").li("Pros")
    with doc.ul:
        doc.li("No brackets or closing tags").li("Integrates with other code")
        doc.ul(E.li("Easy").li("Efficient"))  # Nest using (...)
    doc.li.a(href='javascript:alert("Link clicked")')("A link")
```

In [4]:
```
print(doc)  # Print doc's code
E.iframe(width="100%", height=200, srcdoc=doc)  # Make an iframe with doc's code in its srcdoc
```

```
<!DOCTYPE html><html lang=en><meta charset=utf8><title>Test page</title><div id=content></d
iv><ul><li>Write documents in Python<li>Pros<ul><li>No brackets or closing tags<li>Integrat
es with other code<ul><li>Easy<li>Efficient</ul></ul><li><a href="javascript:alert(&quot;Li
nk clicked&quot;)">A link</a></ul>
```

Out[4]:

- Write documents in Python
- Pros
  - No brackets or closing tags
  - Integrates with other code
    - Easy
    - Efficient
- A link

In [5]:
```
print(_)  # Print the iframe source
```

```
<iframe width="100%" height=200 srcdoc="<!DOCTYPE html><html lang=en><meta charset=utf8><ti
tle>Test page</title><div id=content></div><ul><li>Write documents in Python<li>Pros<ul><li
>No brackets or closing tags<li>Integrates with other code<ul><li>Easy<li>Efficient</ul></u
l><li><a href=&amp;quot;javascript:alert(&amp;amp;quot;Link clicked&amp;amp;quot;)&amp;quot;>A link</a></ul
>"></iframe>
```

Escaping and quotes are quite minimal but sufficient.

In [6]:
```
doc = Document("Table test", lang="en")
with doc.table:
    for r in range(3):
        doc.tr.th("R", r)
        for c in range(4):
            doc.td(r, c)

print(doc)
doc
```

```
<!DOCTYPE html><html lang=en><meta charset=utf8><title>Table test</title><table><tr><th>R0<
td>00<td>01<td>02<td>03<tr><th>R1<td>10<td>11<td>12<td>13<tr><th>R2<td>20<td>21<td>22<td>23
</table>
```

Out[6]:

R0	00	01	02	03
R1	10	11	12	13
R2	20	21	22	23

Boolean values convert into short attributes. Underscore at the end of name is ignored so that Python's reserved names such as **for** can be specified. Other underscores convert into hyphens.

In [7]:
```
E.input(type="checkbox", id="somebox", checked=True).label(for_="somebox")("Yes, please!")
```

Out[7]: ☑ Yes, please!

In [8]:
```
print(_)
```

```
<input type=checkbox id=somebox checked><label for=somebox>Yes, please!</label>
```

In [9]:
```
doc = E.h1("Introduction")
doc.p("This module is intended to be used for HTML formatting using Python code and control str
doc.p("We hope that you find this useful too")
doc.p("Sincerely,").br("Developers!")._comment("This is a comment")

print(doc)
doc
```

```
<h1>Introduction</h1><p>This module is intended to be used for HTML formatting using Python
code and control structures.<p>We hope that you find this useful too<p>Sincerely,<br>Develo
pers!<!--This is a comment-->
```

Out[9]:

Introduction

This module is intended to be used for HTML formatting using Python code and control structures.

We hope that you find this useful too

Sincerely,
Developers!

In [10]:
```
E.strong("Strong text")(", normal text").em(", an emphasis and ").mark(style="background: #ff0'
```

Out[10]: **Strong text**, normal text, *an emphasis and* marked text

In [11]:
```
print(_)
```

```
<strong>Strong text</strong>, normal text<em>, an emphasis and </em><mark style="backgroun
d: #ff0">marked text</mark>
```

In [12]:
```
print(Document(_urls=("style.css", "favicon.ico", "jquery.js")))
```

```
<!DOCTYPE html><link rel=stylesheet href="style.css"><link rel=icon href="favicon.ico"><scr
ipt src="jquery.js"></script>
```

In [13]:
```
%timeit str(Document("benchmarking", lang="en", _urls=("foo.js", "bar.js")))
```

```
73.5 µs ± 11 µs per loop (mean ± std. dev. of 7 runs, 10000 loops each)
```

In [14]:
```
with E.svg as svg:
    svg.circle(id="circ", r=50, cx=50, cy=50, fill="red")
    svg.rect(x=120, y=5, width=90, height=90, stroke="blue", fill="none")

print(svg)
svg
```

```
<svg><circle id=circ r=50 cx=50 cy=50 fill=red></circle><rect x=120 y=5 width=90 height=90
stroke=blue fill=none></rect></svg>
```

Out[14]: