korona Documentation

Release 0.4.3.-dev

Bharadwaj Yarlagadda

September 29, 2016

Contents

Korona helps you to generate HTML pages from the given inputs(HTML Tags and Attributes) without writing the HTML code.

CHAPTER 1

Links

- Project: https://github.com/bharadwajyarlagadda/korona
- Documentation: http://korona.readthedocs.io
- Pypi: https://pypi.python.org/pypi/korona
- TravisCI: https://travis-ci.org/bharadwajyarlagadda/korona

CHAPTER 2

Features

- Supported on Python 2.7 and Python 3.3+
- With this package, you can avoid writing direct HTML code.

CHAPTER 3

Quickstart

Install using pip:

```
pip install korona
```

CHAPTER 4

Guide

4.1 Installation

korona requires Python 2.7 or 3.3+.

To install from PyPI:

```
pip install korona
```

You can also install korona with all the latest changes:

```
$ git clone git@github.com:bharadwajyarlagadda/korona.git
$ cd korona
$ python setup.py install
```

4.2 User's Guide

Korona helps you to build html pages.

Korona also helps you to build individual html tags.

4.2.1 <a>

Korona supports some of the anchor tag attributes like:

- `charset`
- `coords`
- `download`
- `href`
- `hreflang`
- `name`
- `rel`
- `rev`
- `shape`
- `target`

- `type`
- `text` (text as in <a>{text})

Korona can build an `<anchor>` tag.

```
from korona.html.tags import A

attributes = {'charset': 'UTF-8', 'href': 'www.google.com', 'hreflang': 'en', 'text': 'google'}

# You can pass in the attributes in the form of a dictionary.
anchor1 = A(**attributes)
# You can also pass in the attributes as args.
anchor2 = A(charset='UTF-8', href='www.google.com', hreflang='en', text='google')
anchor_tag1 = anchor1.construct()
anchor_tag2 = anchor2.construct()

assert anchor_tag1 == '<a charset="UTF-8" href="www.google.com" hreflang="en" >google</a>'
assert anchor_tag1 == anchor_tag2
```

4.2.2 <abbr>

Korona can build an `<abbr>` tag.

```
from korona.html.tags import Abbr

attributes = {'text': 'WHO'}

# You can pass in the attributes in the form of a dictionary.
abbreviate1 = Abbr(**attributes)
# You can also pass in the attributes as args.
abbreviate2 = Abbr(text='WHO')
abbreviate_tag1 = abbreviate1.construct()
abbreviate_tag2 = abbreviate2.construct()

assert abbreviate_tag1 == '<abbr>WHO </abbr>'
assert abbreviate_tag1 == abbreviate_tag2
```

Note: korona only supports `text` attribute for `<abbr>` tag for now.

4.2.3 <acronym>

Korona can build an `<acronym>` tag.

```
from korona.html.tags import Acronym

attributes = {'text': 'ASAP'}

# You can pass in the attributes in the form of a dictionary.
acronym1 = Acronym(**attributes)
# You can also pass in the attributes as args.
acronym2 = Acronym(text='ASAP')
acronym_tag1 = acronym1.construct()
acronym_tag2 = acronym2.construct()
```

```
assert acronym_tag1 == '<acronym>ASAP </acronym>'
assert acronym_tag1 == acronym_tag2
```

Note: korona only supports `text` attribute for `<acronym>` tag for now.

4.2.4 <address>

Korona can build an `<address>` tag.

```
from korona.html.tags import Address

attributes = {'text': 'abcd@yahoo.com'}

# You can pass in the attributes in the form of a dictionary.
address1 = Address(**attributes)
# You can also pass in the attributes as args.
address2 = Address(text='abcd@yahoo.com')
address_tag1 = address1.construct()
address_tag2 = address2.construct()

assert address_tag1 == '<address>abcd@yahoo.com </address>'
assert address_tag1 == address_tag2
```

4.2.5 <area>

Korona supports some of the area tag attributes like:

- `alt`
- `coords`
- `download`
- `href`
- `hreflang`
- `media`
- `nohref`
- `rel`
- `shape`
- `target`
- `type`

Korona can build an `<area>` tag.

```
from korona.html.tags import Area

attributes = {'href': 'www.example.com', 'hreflang': 'en', 'alt': 'example'}

# You can pass in the attributes in the form of a dictionary.
area1 = Area(**attributes)
# You can also pass in the attributes as args.
```

```
area2 = Area(href='www.example.com', hreflang='en', alt='example')

area_tag1 = area1.construct()
area_tag2 = area2.construct()

assert area_tag1 == '<area href="www.example.com" hreflang="en" alt="example" >
assert area_tag1 == area_tag2
```

4.2.6 <article>

Korona can build an `<article>` tag.

```
from korona.html.tags import Article

attributes = {'text': '<p>Hi there</p>'}

# You can pass in the attributes in the form of a dictionary.
article1 = Article(**attributes)
# You can also pass in the attributes as args.
article2 = Article(text='<p>Hi there</p>')
article_tag1 = article1.construct()
article_tag2 = article2.construct()

assert article_tag1 == '<article><p>Hi there</p> </article>'
assert article_tag1 == article_tag2
```

4.2.7 <b>

Korona can build `<b>` tag.

```
from korona.html.tags import B

attributes = {'text': 'example'}

# You can pass in the attributes in the form of a dictionary.
bold1 = B(**attributes)
# You can also pass in the attributes as args.
bold2 = B(text='example')

bold_tag1 = bold1.construct()
bold_tag2 = bold2.construct()

assert bold_tag1 == '<b>example </b>'
assert bold_tag1 == bold_tag2
```

4.2.8 <base>

Korona can build `<base>` tag.

```
from korona.html.tags import Base

attributes = {'href': 'www.google.com', 'target': 'example'}

# You can pass in the attributes in the form of a dictionary.
```

```
base1 = Base(**attributes)
# You can also pass in the attributes as args.
base2 = Base(href='www.google.com', target='example')

base_tag1 = base1.construct()
base_tag2 = base2.construct()

assert base_tag1 == '<base href="www.google.com" target="example" >'
assert base_tag1 == base_tag2
```

4.2.9 <button>

Korona supports some of the button tag attributes like:

- `autofocus`
- `disabled`
- `form`
- `formaction`
- `formenctype`
- `formmethod`
- `formnovalidate`
- `formtarget`
- `name`
- `type`
- `value`
- `text` (text as in <button>{text}</button>)

Korona can build `<button>` tag.

```
from korona.html.tags import Button

attributes = {'name': 'HTML1',
              'type': 'submit',
              'value': 'HTML1',
              'text': 'HTML1'}

# You can pass in the attributes in the form of a dictionary.
button1 = Button(**attributes)
# You can also pass in the attributes as args.
button2 = Button(name='HTML1', type='submit', value='HTML1', text='HTML1')

button_tag1 = button1.construct()
button_tag2 = button2.construct()

assert button_tag1 == '<button name="HTML1" type="submit" value="HTML1" >HTML1</button>'
assert button_tag1 == button_tag2
```

4.2.10 <canvas>

Korona can build <canvas> tag.

```
from korona.html.tags import Canvas

attributes = {'height': '100', 'width': '200'}

# You can pass in the attributes in the form of a dictionary.
canvas1 = Canvas(**attributes)
# You can also pass in the attributes as args.
canvas2 = Canvas(height='100', width='200')

canvas_tag1 = canvas1.construct()
canvas_tag2 = canvas2.construct()

assert canvas_tag1 == '<canvas height="100" width="200" ></canvas>'
assert canvas_tag1 == canvas_tag2
```

Note: korona doesn't support canvas `text` for now.

4.2.11 <caption>

Korona can build <caption> tag.

```
from korona.html.tags import Caption

attributes = {'align': 'top', 'text': 'abcd'}

# You can pass in the attributes in the form of a dictionary.
caption1 = Caption(**attributes)
# You can also pass in the attributes as args.
caption2 = Caption(align='top', text='abcd')

caption_tag1 = caption1.construct()
caption_tag2 = caption2.construct()

assert caption_tag1 == '<caption align="top" >abcd</caption>'
assert caption_tag1 == caption_tag2
```

4.2.12 <cite>

Korona can build <cite> tag.

```
from korona.html.tags import Cite

attributes = {'text': 'abcd'}

# You can pass in the attributes in the form of a dictionary.
cite1 = Cite(**attributes)
# You can also pass in the attributes as args.
cite2 = Cite(text='abcd')

cite_tag1 = cite1.construct()
```

```
cite_tag2 = cite2.construct()

assert cite_tag1 == '<cite>abcd </cite>'
assert cite_tag1 == cite_tag2
```

4.2.13 <col>

Korona supports some of the col tag attributes like:

- `align`
- `char`
- `charoff`
- `span`
- `valign`
- `width`

Korona can build `<col>` tag.

```
from korona.html.tags import Col

attributes = {'align': 'char', 'char': '.', 'charoff': '2'}

# You can pass in the attributes in the form of a dictionary.
col1 = Col(**attributes)
# You can also pass in the attributes as args.
col2 = Col(align='char', char='.', charoff='2')

col_tag1 = col1.construct()
col_tag2 = col2.construct()

assert col_tag1 == '<col align="char" char="." charoff="2" >'
assert col_tag1 == col_tag2
```

4.2.14 <colgroup>

Korona supports some of the colgroup tag attributes like:

- `align`
- `char`
- `charoff`
- `span`
- `valign`
- `width`

Korona can build `<colgroup>` tag.

```
from korona.html.tags import ColGroup

attributes = {'align': 'char', 'char': '.', 'charoff': '2'}
```

```
# You can pass in the attributes in the form of a dictionary.
colgroup1 = ColGroup(**attributes)
# You can also pass in the attributes as args.
colgroup2 = ColGroup(align='char', char='.', charoff='2')

colgroup_tag1 = colgroup1.construct()
colgroup_tag2 = colgroup2.construct()

assert colgroup_tag1 == '<colgroup align="char" char="." charoff="2" ></colgroup>'
assert colgroup_tag1 == colgroup_tag2
```

4.2.15 <dd>

Korona can build `<dd>` tag.

```
from korona.html.tags import DD

attributes = {'text': 'abc'}

# You can pass in the attributes in the form of a dictionary.
dd1 = DD(**attributes)
# You can also pass in the attributes as args.
dd2 = DD(text='abc')

dd_tag1 = dd1.construct()
dd_tag2 = dd2.construct()

assert dd_tag1 == '<dd>abc </dd>'
assert dd_tag1 == dd_tag2
```

4.2.16 <del>

Korona supports some of the del tag attributes like:

- `cite`
- `datetime`

Korona can build `<del>` tag.

```
from korona.html.tags import Del

attributes = {'cite': 'www.abcd.com', 'text': 'abcd'}

# You can pass in the attributes in the form of a dictionary.
del1 = Del(**attributes)
# You can also pass in the attributes as args.
del2 = Del(cite='www.abcd.com', text='abcd')

del_tag1 = del1.construct()
del_tag2 = del2.construct()

assert del_tag1 == '<del cite="www.abcd.com" >abcd</del>'
assert del_tag1 == del_tag2
```

4.2.17 <details>

Korona supports `open` attribute for `<details>` tag. Korona can help you build `<details>` tag.

```
from korona.html.tags import Details

attributes = {'open': True, 'text': 'abcd'}

# You can pass in the attributes in the form of a dictionary.
details1 = Details(**attributes)
# You can also pass in the attributes as args.
details2 = Details(open=True, text='abcd')

details_tag1 = details1.construct()
details_tag2 = details2.construct()

assert details_tag1 == '<details open >abcd</details>
assert details_tag1 == details_tag2
```

4.2.18 <dialog>

Korona supports `open` attribute for `<dialog>` tag. Korona can help you build `<dialog>` tag.

```
from korona.html.tags import Dialog

attributes = {'open': True, 'text': 'abcd'}

# You can pass in the attributes in the form of a dictionary.
dialog1 = Dialog(**attributes)
# You can also pass in the attributes as args.
dialog2 = Dialog(open=True, text='abcd')

dialog_tag1 = dialog1.construct()
dialog_tag2 = dialog2.construct()

assert dialog_tag1 == '<dialog open >abcd</dialog>
assert dialog_tag1 == dialog_tag2
```

4.2.19

Korona supports `align` attribute for `<div>` tag. Korona can help you build `<div>` tag.

```
from korona.html.tags import Div

attributes = {'align': 'left', 'text': 'abcd'}

# You can pass in the attributes in the form of a dictionary.
div1 = Div(**attributes)
# You can also pass in the attributes as args.
div2 = Div(align='left', text='abcd')

div_tag1 = div1.construct()
div_tag2 = div2.construct()

assert div_tag1 == '<div align="left" >abcd</div>
assert div_tag1 == dialog_tag2
```

4.2.20 <dl>

Korona can build <dl> tag.

```
from korona.html.tags import DL

attributes = {'text': 'abc'}

# You can pass in the attributes in the form of a dictionary.
dl1 = DL(**attributes)
# You can also pass in the attributes as args.
dl2 = DL(text='abc')

dl_tag1 = dl1.construct()
dl_tag2 = dl2.construct()

assert dl_tag1 == '<dl>abc</dl>'
assert dl_tag1 == dl_tag2
```

4.2.21 <dt>

Korona can build <dt> tag.

```
from korona.html.tags import DT

attributes = {'text': 'abc'}

# You can pass in the attributes in the form of a dictionary.
dt1 = DT(**attributes)
# You can also pass in the attributes as args.
dt2 = DT(text='abc')

dt_tag1 = dt1.construct()
dt_tag2 = dt2.construct()

assert dt_tag1 == '<dt>abc</dt>'
assert dt_tag1 == dt_tag2
```

4.2.22 <embed>

Korona supports some of the embed tag attributes like:

- height
- src
- type
- width

Korona can build <embed> tag.

```
from korona.html.tags import Embed

attributes = {'src': 'helloworld.swf', 'height': '200', 'width': '100'}

# You can pass in the attributes in the form of a dictionary.
embed1 = Embed(**attributes)
```

```
# You can also pass in the attributes as args.
embed2 = Embed(src='helloworld.swf', height='200', width='100')

embed_tag1 = embed1.construct()
embed_tag2 = embed2.construct()

assert embed_tag1 == '<embed src="helloworld.swf" width="100" height="200" >'
assert embed_tag1 == embed_tag2
```

4.2.23 <fieldset>

Korona supports some of the fieldset tag attributes like:

- `disabled`
- `form`
- `name`

Korona can build `<fieldset>` tag.

```
from korona.html.tags import FieldSet

attributes = {'disabled': True, 'form': 'form1', 'name': 'name1'}

# You can pass in the attributes in the form of a dictionary.
fieldset1 = FieldSet(**attributes)
# You can also pass in the attributes as args.
fieldset2 = FieldSet(disabled=True, form='form1', name='name1')

fieldset_tag1 = fieldset1.construct()
fieldset_tag2 = fieldset2.construct()

assert fieldset_tag1 == '<fieldset form="form1" name="name1" disabled ></fieldset>'
assert fieldset_tag1 == fieldset_tag2
```

4.2.24 <figure>

Korona can build `<figure>` tag.

```
from korona.html.tags import Figure

attributes = {'text': 'abc'}

# You can pass in the attributes in the form of a dictionary.
figure1 = Figure(**attributes)
# You can also pass in the attributes as args.
figure2 = Figure(text='abc')

figure_tag1 = figure1.construct()
figure_tag2 = figure2.construct()

assert figure_tag1 == '<figure>abc</figure>'
assert figure_tag1 == figure_tag2
```

Note: Korona for now does not support any inner tags in <figure> tag.

4.2.25 <footer>

Korona can build `<footer>` tag.

```
from korona.html.tags import Footer

attributes = {'text': 'abc'}

# You can pass in the attributes in the form of a dictionary.
footer1 = Footer(**attributes)
# You can also pass in the attributes as args.
footer2 = Footer(text='abc')

footer_tag1 = figure1.construct()
footer_tag2 = figure2.construct()

assert footer_tag1 == '<footer>abc</footer>'
assert footer_tag1 == footer_tag2
```

Note: Korona for now does not support any inner tags in <footer> tag.

4.2.26 <form>

Korona supports some of the form tag attributes like:

- `accept`
- `action`
- `autocomplete`
- `enctype`
- `method`
- `name`
- `novalidate`
- `target`
- `text` (text as in <form>{text}</form>)

Korona can build `<form>` tag.

```
from korona.html.tags import Form

attributes = {'action': 'demo.asp', 'method': 'get', 'name': 'name1', 'target': '_top'}

# You can pass in the attributes in the form of a dictionary.
form1 = Form(**attributes)
# You can also pass in the attributes as args.
form2 = Form(action='demo.asp', method='get', name='name1', target='_top')
```

```
form_tag1 = form1.construct()
form_tag2 = form2.construct()

assert form_tag1 == '<form action="demo.asp" method="get" name="name1" target="_top" ></form>'
assert form_tag1 == form_tag2
```

4.2.27 <frame>

Korona supports some of the frame tag attributes like:

- `frameborder`
- `longdesc`
- `marginheight`
- `marginwidth`
- `name`
- `noresize`
- `scrolling`
- `src`

Koron can build `<frame>` tag.

```
from korona.html.tags import Frame

attributes = {'src': 'frame_a.htm', 'scrolling': 'auto', 'marginheight': '250', 'marginwidth': '100',

# You can pass in the attributes in the form of a dictionary.
frame1 = Frame(**attributes)
# You can also pass in the attributes as args.
frame2 = Frame(src='frame_a.htm', scrolling='auto', marginheight='250', marginwidth='100', name='name

frame_tag1 = frame1.construct()
frame_tag2 = frame2.construct()

assert frame_tag1 == '<frame src="frame_a.htm" longdesc="a.txt" marginheight="250" marginwidth="100"
assert frame_tag1 == frame_tag2
```

4.2.28 <frameset>

Korona supports some of the frameset tag attributes like:

- `cols`
- `rows`

Korona can build `<frameset>` tag.

```
from korona.html.tags import FrameSet

attributes = {'cols': '25%'}

# You can pass in the attributes in the form of a dictionary.
frameset1 = FrameSet(**attributes)
# You can also pass in the attributes as args.
```

```
frameset2 = FrameSet(cols='25%')

frameset_tag1 = frameset1.construct()
frameset_tag2 = frameset2.construct()

assert frameset_tag1 == '<frameset cols="25%" ></frameset>'
assert frameset_tag1 == frameset2_tag
```

4.2.29 <h1>

Korona supports `align` attribute for <h1> tag. Korona can help you build <h1> tag.

```
from korona.html.tags import H1

attributes = {'align': 'left', 'text': 'abcd'}

# You can pass in the attributes in the form of a dictionary.
h1_one = H1(**attributes)
# You can also pass in the attributes as args.
h1_two = H1(align='left', text='abcd')

h1_tag1 = h1_one.construct()
h1_tag2 = h1_two.construct()

assert h1_tag1 == '<h1 align="left" >abcd</h1>
assert h1_tag1 == h1_tag2
```

4.2.30 <h2>

Korona supports `align` attribute for <h2> tag. Korona can help you build <h2> tag.

```
from korona.html.tags import H2

attributes = {'align': 'left', 'text': 'abcd'}

# You can pass in the attributes in the form of a dictionary.
h2_one = H2(**attributes)
# You can also pass in the attributes as args.
h2_two = H2(align='left', text='abcd')

h2_tag1 = h2_one.construct()
h2_tag2 = h2_two.construct()

assert h2_tag1 == '<h2 align="left" >abcd</h2>
assert h2_tag1 == h2_tag2
```

4.2.31 <h3>

Korona supports `align` attribute for <h3> tag. Korona can help you build <h3> tag.

```
from korona.html.tags import H3

attributes = {'align': 'left', 'text': 'abcd'}
```

```
# You can pass in the attributes in the form of a dictionary.
h3_one = H3(**attributes)
# You can also pass in the attributes as args.
h3_two = H3(align='left', text='abcd')

h3_tag1 = h3_one.construct()
h3_tag2 = h3_two.construct()

assert h3_tag1 == '<h3 align="left" >abcd</h3>
assert h3_tag1 == h3_tag2
```

4.2.32 <h4>

Korona supports `align` attribute for `<h4>` tag. Korona can help you build `<h4>` tag.

```
from korona.html.tags import H4

attributes = {'align': 'left', 'text': 'abcd'}

# You can pass in the attributes in the form of a dictionary.
h4_one = H4(**attributes)
# You can also pass in the attributes as args.
h4_two = H4(align='left', text='abcd')

h4_tag1 = h4_one.construct()
h4_tag2 = h4_two.construct()

assert h4_tag1 == '<h4 align="left" >abcd</h4>
assert h4_tag1 == h4_tag2
```

4.2.33 <h5>

Korona supports `align` attribute for `<h5>` tag. Korona can help you build `<h5>` tag.

```
from korona.html.tags import H5

attributes = {'align': 'left', 'text': 'abcd'}

# You can pass in the attributes in the form of a dictionary.
h5_one = H5(**attributes)
# You can also pass in the attributes as args.
h5_two = H5(align='left', text='abcd')

h5_tag1 = h5_one.construct()
h5_tag2 = h5_two.construct()

assert h5_tag1 == '<h5 align="left" >abcd</h5>
assert h5_tag1 == h5_tag2
```

4.2.34 <h6>

Korona supports `align` attribute for `<h6>` tag. Korona can help you build `<h6>` tag.

```
from korona.html.tags import H6

attributes = {'align': 'left', 'text': 'abcd'}

# You can pass in the attributes in the form of a dictionary.
h6_one = H6(**attributes)
# You can also pass in the attributes as args.
h6_two = H6(align='left', text='abcd')

h6_tag1 = h6_one.construct()
h6_tag2 = h6_two.construct()

assert h6_tag1 == '<h6 align="left" >abcd</h6>
assert h6_tag1 == h6_tag2
```

4.2.35 <head>

Korona can help you build `<head>` tag.

```
from korona.html.tags import Head

attributes = {'text': 'abcd'}

# You can pass in the attributes in the form of a dictionary.
head1 = Head(**attributes)
# You can also pass in the attributes as args.
head2 = Head(text='abcd')

head_tag1 = head1.construct()
head_tag2 = head2.construct()

assert head_tag1 == '<head>abcd</head>'
assert head_tag1 == head_tag2
```

Note: <head> tag for now only supports `text`. It doesn't has the capability for creating inner tags such as `<title>`, `<style>`, etc.

4.2.36 <header>

Korona can help you build `<header>` tag.

```
from korona.html.tags import Header

attributes = {'text': 'abc'}

# You can pass in the attributes in the form of a dictionary.
header1 = Header(**attributes)
# You can also pass in the attributes as args.
header2 = Header(text='abc')

header_tag1 = header1.construct()
header_tag2 = header2.construct()
```

```
assert header_tag1 == '<header>abc</header>'
assert header_tag1 == header_tag2
```

Note: Korona for now does not support any inner tags in <header> tag.

4.2.37 <hr>

Korona can help you build <hr> tag.

```
from korona.html.tags import HR

attributes = {'align': 'center', 'size': '100'}

# You can pass in the attributes in the form of a dictionary.
hr1 = HR(**attributes)
# You can also pass in the attributes as args.
hr2 = HR(align='center', size='100')

hr_tag1 = hr1.construct()
hr_tag2 = hr2.construct()

assert hr_tag1 == '<hr align="center" size="100" >'
assert hr_tag1 == hr_tag2
```

4.2.38 <html>

Korona supports some of the html tag attributes like:

- manifest
- xmlns

Korona can help you build <html> tag.

```
from korona.html.tags import Html

attributes = {'manifest': 'demo.appcache', 'text': 'abcd'}

# You can pass in the attributes in the form of a dictionary.
html1 = Html(**attributes)
# You can also pass in the attributes as args.
html2 = Html(manifest='demo.appcache', text='abcd')

html_tag1 = html1.construct()
html_tag2 = html2.construct()

assert html_tag1 == '<html manifest="demo.appcache" >abcd</html>'
assert html_tag1 == html_tag2
```

4.2.39 <i>

Korona can help you build <i> tag.

```
from korona.html.tags import I

attributes = {'text': 'abcd'}

# You can pass in the attributes in the form of a dictionary.
italics1 = I(**attributes)
# You can also pass in the attributes as args.
italics2 = I(text='abcd')

italics_tag1 = italics1.construct()
italics_tag2 = italics2.construct()

assert italics_tag1 == '<i>abcd</i>
assert italics_tag1 == italics_tag2
```

4.2.40 <iframe>

Korona supports `iframe` tag attributes like:

- `align`
- `frameborder`
- `height`
- `longdesc`
- `marginheight`
- `marginwidth`
- `name`
- `sandbox`
- `scrolling`
- `src`
- `srcdoc`
- `width`

Korona can help you build `<iframe>` tag.

```
from korona.html.tags import IFrame

attributes = {'src': '/demo.asp', 'height': '100', 'width': '200'}

# You can pass in the attributes in the form of a dictionary.
iframe1 = IFrame(**attributes)
# You can also pass in the attributes as args.
iframe2 = IFrame(src='/demo.asp', height='100', width='200')

iframe_tag1 = iframe1.construct()
iframe_tag2 = iframe2.construct()

assert iframe_tag1 == '<iframe src="/demo.asp" width="200" height="100" ></iframe>'
assert iframe_tag1 == iframe_tag2
```

4.2.41 <img>

Korona supports `img` tag attributes like:

- `align`
- `alt`
- `border`
- `crossorigin`
- `height`
- `hspace`
- `ismap`
- `longdesc`
- `src`
- `usemap`
- `vspace`
- `width`

Korona can help you build `<img>` tag.

```
from korona.html.tags import Img

attributes = {'height': '30', 'width': '30', 'hspace': '20', 'vspace': '20'}

# You can pass in the attributes in the form of a dictionary.
img1 = Img(**attributes)
# You can also pass in the attributes as args.
img2 = Img(height='30', width='30', hspace='20', vspace='20')

img_tag1 = img1.construct()
img_tag2 = img2.construct()

assert img_tag1 == '<img height="30" hspace="20" vspace="20" width="30" >'
assert img_tag1 == img_tag2
```

4.2.42 <input>

Korona supports input tag attributes like:

- `accept` (str): Specifies the types of files that the server accepts. The accept attribute can only be used with <input type="file">.

```
from korona.html.tags import Input

input1 = Input(**{'accept': 'audio/*', 'type': 'file'})
input1.construct()
# <input type="file" accept="audio/*" >
```

- `align` (str): Specifies the alignment of an image input. The align attribute is only used with <input type="image">.

```
from korona.html.tags import Input

input1 = Input(**{'align': 'left', 'type': 'image'})
input1.construct()
# <input type="image" align="left" >
```

- `alt` (str): Specifies an alternate text for images. The alt attribute can only be used with <input type="image">.

```
from korona.html.tags import Input

input1 = Input(**{'alt': 'temp', 'type': 'image'})
input1.construct()
# <input type="image" alt="temp" >
```

- `autocomplete` (str): Specifies whether an <input> element should have autocomplete enabled. The autocomplete attribute works with the following <input> types: text, search, url, tel, email, password, datepickers, range, and color.

```
from korona.html.tags import Input

input1 = Input(**{'autocomplete': 'on', 'type': 'email'})
input1.construct()
# <input type="email" autocomplete="on" >
```

- `autofocus` (bool): Specifies that an <input> element should automatically get focus when the page loads.

```
from korona.html.tags import Input

input1 = Input(**{'autofocus': True})
input1.construct()
# <input autofocus >
```

- `checked` (bool): Specifies that an <input> element should be pre-selected when the page loads. The checked attribute can be used with <input type="checkbox"> and <input type="radio">.

```
from korona.html.tags import Input

input1 = Input(**{'checked': True, 'type': 'radio'})
input1.construct()
# <input type="radio" checked >
```

- `dirname` (str): Specifies that the text direction will be submitted. The dirname attribute's value is always the name of the input field, followed by ".dir".

```
from korona.html.tags import Input

input1 = Input(**{'name': 'fname', 'dirname': 'fname.dir'})
input1.construct()
# <input name="fname" dirname="fname.dir" >
```

- `disabled` (bool): Specifies that an <input> element should be disabled.

```
from korona.html.tags import Input

input1 = Input(**{'disabled': True})
input1.construct()
# <input disabled >
```

- `form` (str): Specifies one or more forms the <input> element belongs to.

```
from korona.html.tags import Input

input1 = Input(**{'form': 'form1'})
input1.construct()
# <input form="form1" >
```

- `formaction` (str): Specifies the URL of the file that will process the input control when the form is submitted. The formaction attribute is used with type="submit" and type="image".

```
from korona.html.tags import Input

input1 = Input(**{'formaction': 'demo.asp', 'type': 'submit'})
input1.construct()
# <input type="submit" formaction="demo.asp" >
```

- `formenctype` (str): Specifies how the form-data should be encoded when submitting it to the server. The formenctype attribute is used with type="submit" and type="image".

```
from korona.html.tags import Input

input1 = Input(**{'formenctype': 'application/x-www-form-urlencoded', 'type': 'submit'})
input1.construct()
# <input type="submit" formenctype="application/x-www-form-urlencoded" >
```

- `formmethod` (str): Defines the HTTP method for sending data to the action URL. The formmethod attribute can be used with type="submit" and type="image".

```
from korona.html.tags import Input

input1 = Input(**{'formmethod': 'get', 'type': 'submit'})
input1.construct()
# <input type="submit" formmethod="get" >
```

- `formnovalidate` (bool): Defines that form elements should not be validated when submitted. The formnovalidate attribute can be used with type="submit".

```
from korona.html.tags import Input

input1 = Input(**{'formnovalidate': True, 'type': 'submit'})
input1.construct()
# <input type="submit" formnovalidate >
```

- `formtarget` (str): Specifies where to display the response that is received after submitting the form. The formtarget attribute can be used with type="submit" and type="image".

```
from korona.html.tags import Input

input1 = Input(**{'formtarget': '_parent', 'type': 'submit'})
input1.construct()
# <input type="submit" formtarget="_parent" >
```

- `height` (int/float): Specifies the height of an <input> element. The height attribute is used only with <input type="image">.

```
from korona.html.tags import Input

input1 = Input(**{'height': '200', 'type': 'image'})
input1.construct()
# <input type="image" height="200" >
```

- `list` (str): Refers to a <datalist> element that contains pre-defined options for an <input> element.

```
from korona.html.tags import Input

input1 = Input(**{'list': 'list1'})
input1.construct()
# <input list="list1" >
```

- `max` (int/date/time): Specifies the maximum value for an <input> element. The max attribute works with the following input types: number, range, date, datetime, datetime-local, month, time and week.

```
from korona.html.tags import Input

input1 = Input(**{'max': date.today(), 'type': 'date'})
input1.construct()
# <input type="date" max="2016-09-24" >
```

- `maxlength` (int): Specifies the maximum number of characters allowed in an <input> element.

```
from korona.html.tags import Input

input1 = Input(**{'maxlength': '5'})
input1.construct()
# <input maxlength="5" >
```

- `min` (int/date/time): Specifies a minimum value for an <input> element. The min attribute works with the following input types: number, range, date, datetime, datetime-local, month, time and week.

```
from korona.html.tags import Input

input1 = Input(**{'min': date.today(), 'type': 'date'})
input1.construct()
# <input type="date" min="2016-09-24" >
```

- `multiple` (bool): Specifies that a user can enter more than one value in an <input> element. The multiple attribute works with the following input types: email, and file.

```
from korona.html.tags import Input

input1 = Input(**{'multiple': True, 'type': 'file'})
input1.construct()
# <input type="file" multiple >
```

- `name` (str): Specifies the name of an <input> element.

```
from korona.html.tags import Input

input1 = Input(**{'name': 'name1'})
input1.construct()
# <input name="name1" >
```

- `pattern` (str): Specifies a regular expression that an <input> element's value is checked against. The pattern attribute works with the following input types: text, date, search, url, tel, email, and password.

```
from korona.html.tags import Input

input1 = Input(**{'pattern': '[A-Za-z]{3}', 'type': 'email'})
input1.construct()
# <input type="email" pattern="[A-Za-z]{3}" >
```

- `placeholder` (str): Specifies a short hint that describes the expected value of an <input> element. The placeholder attribute works with the following input types: text, search, url, tel, email, and password.

```
from korona.html.tags import Input

input1 = Input(**{'placeholder': 'Full name', 'type': 'text'})
input1.construct()
# <input type="text" placeholder="Full name" >
```

- `readonly` (bool): Specifies that an input field is read-only.

```
from korona.html.tags import Input

input1 = Input(**{'readonly': True})
input1.construct()
# <input readonly >
```

- `required` (bool): Specifies that an input field must be filled out before submitting the form. The required attribute works with the following input types: text, search, url, tel, email, password, date pickers, number, checkbox, radio, and file.

```
from korona.html.tags import Input

input1 = Input(**{'required': True, 'type': 'search'})
input1.construct()
# <input type="search" required >
```

- `size` (int): Specifies the width, in characters, of an <input> element. The size attribute works with the following input types: text, search, tel, url, email, and password.

```
from korona.html.tags import Input

input1 = Input(**{'size': '10', 'type': 'password'})
input1.construct()
# <input type="password" size="10" >
```

- `src` (str): Specifies the URL of the image to use as a submit button. The src attribute is required for <input type="image">, and can only be used with <input type="image">.

```
from korona.html.tags import Input

input1 = Input(**{'src': 'demo.asp', 'type': 'image'})
input1.construct()
# <input type="image" src="demo.asp" >
```

- `step` (int): Specifies the legal number intervals for an input field. The step attribute works with the following input types: number, range, date, datetime, datetime-local, month, time and week.

```
from korona.html.tags import Input

input1 = Input(**{'step': 1, 'type': 'range'})
input1.construct()
# <input type="range" step="1" >
```

- `type` (str): Specifies the type <input> element to display.

```
from korona.html.tags import Input

input1 = Input(**{'type': 'submit'})
```

```
input1.construct()
# <input type="submit" >
```

- `value` (str): Specifies the value of an <input> element. The value attribute cannot be used with <input type="file">.

```
from korona.html.tags import Input

input1 = Input(**{'value': 'value1'})
input1.construct()
# <input value="value1" >
```

- `width` (int/float): Specifies the width of an <input> element. The width attribute is used only with <input type="image">.

```
from korona.html.tags import Input

input1 = Input(**{'width': '100', 'type': 'image'})
input1.construct()
# <input type="image" width="100" >
```

4.3 API Reference

class `korona.html.tags.`**A**(*charset=None*, *coords=None*, *download=None*, *href=None*, *hreflang=None*, *name=None*, *rel=None*, *rev=None*, *shape=None*, *target=None*, *type=None*, *text=None*)

Class for constructing anchor tag.

Parameters

- **charset** (*str*) – Specifies the character-set of a linked document.
- **coords** (*mixed*) – A list/tuple/string of coordinate values.
- **download** (*str*) – Specifies that the target will be downloaded when a user clicks on the hyper link.
- **href** (*str*) – Specifies the URL of the page the link goes to.
- **hreflang** (*str*) – Specifies the language of the linked document.
- **name** (*str*) – Specifies the name of an anchor.
- **rel** (*str*) – Specifies the relationship between the current document and the linked document.
- **rev** (*str*) – Specifies the relationship between the linked document and the current document.
- **shape** (*str*) – Specifies the shape of a link.
- **target** (*str*) – Specifies where to open the linked document.
- **type** (*str*) – Specifies the media type of the linked document.
- **text** (*str*) – Anchor tag text. (Ex. <a>text)

New in version 0.1.0.

Changed in version 0.2.0: Renamed the method construct_tag to construct.

Changed in version 0.3.1: Added URL validation for href attribute.

Changed in version 0.4.3-dev: Now the user can be displayed with warnings for charset attribute. Used custom exceptions for all the attributes. Removed the method `validate_values()`.

construct()
Returns the constructed anchor tag <a>.

get_coords(*shape*, *coords*)
Returns coordinates after a series of validations.

Parameters

- **shape** (*str*) – Shape of a link (Ex. rect/circle/poly/etc.)
- **coords** (*mixed*) – A list/tuple/str of coordinate values.

Returns A string of coordinate values.

Return type str

pre_validate(*href*, *attribute_name*, *value*)
Validates whether an attribute is dependant on another attribute or not. Some of the attributes requires href attribute to be set.

validate_charset(*charset*)
Validates charset attribute. Warn the user showing the common character sets used.

class korona.html.tags.**Abbr**(*text=None*)
Class for constructing abbr tag.

Parameters **text** (*str*) – Abbr tag text. (Ex. <abbr>text</abbr>)

New in version 0.1.0.

Changed in version 0.2.0: Renamed `construct_tag()` to `construct()`.

construct()
Returns the constructed abbr tag <abbr></abbr>.

class korona.html.tags.**Acronym**(*text=None*)
Class for constructing acronym tag.

Parameters **text** (*str*) – Acronym tag text. (Ex. <acronym>text</acronym>)

New in version 0.1.0.

Changed in version 0.2.0: Renamed `construct_tag()` to `construct()`.

construct()
Returns the constructed acronym tag <acronym></acronym>.

class korona.html.tags.**Address**(*text=None*)
Class for constructing address tag.

Parameters **text** (*str*) – Address tag text. (Ex. <address>text</address>)

New in version 0.1.0.

Changed in version 0.2.0: Renamed the method construct_tag to construct.

construct()
Returns the constructed address tag <address></address>.

class korona.html.tags.**Area**(*alt=None*, *coords=None*, *download=None*, *href=None*, *hreflang=None*, *media=None*, *nohref=False*, *rel=None*, *shape=None*, *target=None*, *type=None*)
Class for constructing area tag.

Parameters

- **alt** (*str*) – Specifies an alternate text for the area. Required if the href attribute is present.
- **coords** (*mixed*) – Specifies the coordinates of the area.
- **download** (*str*) – Specifies that the target will be downloaded when a user clicks on the hyper link.
- **href** (*str*) – Specifies the hyperlink target for the area.
- **hreflang** (*str*) – Specifies the language of the target URL.
- **media** (*str*) – Specifies what media/device the target URL is optimized for.
- **nohref** (*bool*) – Specifies that an area has no associated link.
- **rel** (*str*) – Specifies the relationship between the current document and the target URL.
- **shape** (*str*) – Specifies the shape of the area.
- **target** (*str*) – Specifies where to open the target URL.
- **type** (*str*) – Specifies the media type of the target URL.

New in version 0.1.0.

Changed in version 0.2.0: Renamed the method construct_tag to construct.

Changed in version 0.3.1: Added URL validation for href attribute.

construct()
Returns the constructed area tag <area></area>.

get_coords(*shape*, *coords*)
Returns coordinates after a series of validations.

Parameters

- **shape** (*str*) – Shape of a link (Ex. rect/circle/poly/etc.)
- **coords** (*mixed*) – A list/tuple/str of coordinate values.

Returns A string of coordinate values.

Return type str

pre_validate(*href*, *attribute_name*, *value*)
Validates whether an attribute is dependant on another attribute or not. Some of the attributes requires href attribute to be set.

validate_alt(*href*, *attribute_name*, *value*)
Validates area's alt attribute.

validate_values(*href*, *attribute_name*, *value*)
Validates whether the given attribute value is a valid value or not. Some of the attributes have confined values. Even if we give some other value, the html output would not be correct.

class korona.html.tags.**Article**(*text=None*)
Class for constructing article tag.

Parameters **text** (*str*) – Article tag text. (Ex. <article>text</article>)

New in version 0.1.0.

Changed in version 0.2.0: Renamed the method construct_tag to construct.

construct()
Returns the constructed article tag <article></article>.

class korona.html.tags.**B**(*text=None*)
Class for constructing bold tag.

Parameters **text** (*str*) – Bold tag text. (Ex. text)

New in version 0.1.0.

Changed in version 0.2.0: Renamed the method construct_tag to construct.

construct()
Returns the constructed bold tag .

class korona.html.tags.**Base**(*href=None*, *target=None*)
Class for constructing base tag.

Parameters

- **href** (*str*) – Specifies the base URL for all relative URLs in the page.
- **target** (*str*) – Specifies the default target for all hyperlinks and forms in the page.

New in version 0.1.0.

Changed in version 0.2.0: Renamed the method construct_tag to construct.

Changed in version 0.3.1: Added URL validation for href attribute.

construct()
Returns the constructed base tag <base>.

validate_values(*href*, *target*)
Validates the following: - Either of href or target attribute value is given. - Check whether both href and target attribute values are strings or not.

class korona.html.tags.**Button**(*autofocus=False*, *disabled=False*, *form=None*, *formaction=None*, *formenctype=None*, *formmethod=None*, *formnovalidate=False*, *formtarget=None*, *name=None*, *type=None*, *value=None*, *text=None*)
Class for constructing button tag.

Parameters

- **autofocus** (*bool*) – Specifies that a button should automatically get focus when the page loads.
- **disabled** (*bool*) – Specifies that a button should be disabled.
- **form** (*str*) – Specifies one or more forms the button belongs to.
- **formaction** (*str*) – Specifies where to send the form-data when a form is submitted. Only for type="submit".
- **formenctype** (*str*) – Specifies how form-data should be encoded before sending it to a server. Only for type="submit".
- **formmethod** (*str*) – Specifies how to send the form-data (which HTTP method to use). Only for type="submit".
- **formnovalidate** (*bool*) – Specifies that the form-data should not be validated on submission. Only for type="submit".
- **formtarget** (*str*) – Specifies where to display the response after submitting the form. Only for type="submit".

- **name** (*str*) – Specifies a name for the button.
- **type** (*str*) – Specifies the type of button.
- **value** (*str*) – Specifies an initial value for the button.

New in version 0.1.0.

Changed in version 0.2.0: Renamed the method construct_tag to construct.

Changed in version 0.4.3-dev: Removed `validate_values()` method.

construct()
Returns the constructed base tag <button>.

pre_validate(*type*, *attribute_name*, *value*)
Validates whether an attribute is dependant on another attribute or not. Some of the attributes requires type attribute to be set to 'submit'.

validate_type(*value*)
Validate the type attribute for a <button> element. Different browsers use different default types for the <button> element.

class korona.html.tags.**Canvas**(*height=None*, *width=None*)
Class for constructing canvas tag.

Parameters

- **height** (*str*) – Specifies the height of the canvas.
- **width** (*str*) – Specifies the width of the canvas.

New in version 0.1.0.

Changed in version 0.2.0: Renamed the method construct_tag to construct.

construct()
Returns the constructed canvas tag <canvas>.

class korona.html.tags.**Caption**(*align=None*, *text=None*)
Class for constructing caption tag.

Parameters

- **align** (*str*) – Defines the alignment of the caption.
- **text** (*str*) – Specifies the caption text.

New in version 0.1.0.

Changed in version 0.2.0: Renamed the method construct_tag to construct.

construct()
Returns the constructed caption tag <caption>.

class korona.html.tags.**Cite**(*text*)
Class for constructing cite tag.

Parameters **text** (*str*) – Specifies the citation text.

New in version 0.1.0.

Changed in version 0.2.0: Renamed the method construct_tag to construct.

construct()
Returns the constructed cite tag <cite>.

class korona.html.tags.**Col**(*align=None*, *char=None*, *charoff=None*, *span=None*, *valign=None*, *width=None*)

Class for constructing col tag.

Parameters

- **align** (*str*) – Specifies the alignment of the content related to a <col> element.
- **char** (*str*) – Specifies the alignment of the content related to a <col> element to a character.
- **charoff** (*int*) – Specifies the number of characters the content will be aligned from the character specified by the char attribute.
- **span** (*int*) – Specifies the number of columns a <col> element should span.
- **valign** (*str*) – Specifies the vertical alignment of the content related to a <col> element.
- **width** (*str*) – Specifies the width of a <col> element.

New in version 0.2.0.

construct()

Returns the constructed col tag <col>.

validate_char_attribute(*align*, *value*)

Validates char attribute. The char attribute can only be used if the align attribute is set to "char".

validate_charoff_attribute(*align*, *char*, *value*)

Validates charoff attribute. The charoff attribute can only be used if the char attribute is specified and the align attribute is set to "char".

class korona.html.tags.**ColGroup**(*align=None*, *char=None*, *charoff=None*, *span=None*, *valign=None*, *width=None*)

Class for constructing colgroup tag.

Parameters

- **align** (*str*) – Aligns the content in a column group.
- **char** (*str*) – Aligns the content in a column group to a character.
- **charoff** (*int*) – Sets the number of characters the content will be aligned from the character specified by the char attribute.
- **span** (*int*) – Specifies the number of columns a column group should span.
- **valign** (*str*) – Vertical aligns the content in a column group.
- **width** (*str*) – Specifies the width of a column group.

New in version 0.2.0.

construct()

Returns the constructed colgroup tag <colgroup>.

validate_char_attribute(*align*, *value*)

Validates char attribute. The char attribute can only be used if the align attribute is set to "char".

validate_charoff_attribute(*align*, *char*, *value*)

Validates charoff attribute. The charoff attribute can only be used if the char attribute is specified and the align attribute is set to "char".

class korona.html.tags.**DD**(*text=None*)

Class for constructing dd tag.

Parameters **text** (*str*) – Specifies the dd text. (As in <dd>{text}</dd>)

New in version 0.2.0.

construct()
Returns the constructed dd tag <dd></dd>.

class korona.html.tags.**Del**(*cite=None*, *datetime=None*, *text=None*)
Class for constructing del tag.

Parameters

- **cite** (*str*) – Specifies a URL to a document that explains the reason why the text was deleted.
- **datetime** (*datetime*) – Specifies the date and time of when the text was deleted.

New in version 0.2.0.

construct()
Returns the constructed del tag <del>.

class korona.html.tags.**Details**(*open=False*, *text=None*)
Class for constructing details tag.

Parameters

- **open** (*bool*) – Specifies that the details should be visible (open) to the user.
- **text** (*str*) – Specifies the details text. (As in <details>{text}</details>)

New in version 0.2.0.

construct()
Returns the constructed details tag <details></details>.

class korona.html.tags.**Dialog**(*open=False*, *text=None*)
Class for constructing dialog tag.

Parameters

- **open** (*bool*) – Specifies that the dialog element is active and that the user can interact with it.
- **text** (*str*) – Specifies the dialog text. (As in <dialog>{text}</dialog>)

New in version 0.2.0.

construct()
Returns the constructed dialog tag <dialog></dialog>.

class korona.html.tags.**Div**(*align=None*, *text=None*)
Class for constructing div tag.

Parameters

- **align** (*str*) – Specifies the alignment of the content inside a element.
- **text** (*str*) – Specifies the div text. (As in {text})

New in version 0.2.0.

construct()
Returns the constructed div tag .

class korona.html.tags.**DL**(*text=None*)
Class for constructing dl tag.

Parameters **text** (*str*) – Specifies the dl text. (As in <dl>{text}</dl>)

New in version 0.2.0.

construct()
Returns the constructed dl tag <dl></dl>.

class korona.html.tags.**DT**(*text=None*)
Class for constructing dt tag.

Parameters **text** (*str*) – Specifies the dt text. (As in <dt>{text}</dt>)

New in version 0.2.0.

construct()
Returns the constructed dt tag <dt></dt>.

class korona.html.tags.**Embed**(*height=None*, *width=None*, *src=None*, *type=None*)
Class for constructing embed tag.

Parameters

- **height** (*str*) – Specifies the height of the embedded content (in pixels).
- **width** (*str*) – Specifies the width of the embedded content (in pixels).
- **src** (*str*) – Specifies the address of the external file to embed.
- **type** (*str*) – Specifies the media type of the embedded content.

New in version 0.2.0.

construct()
Returns the constructed embed tag <embed>.

class korona.html.tags.**FieldSet**(*disabled=False*, *form=None*, *name=None*)
Class for constructing fieldset tag.

Parameters

- **disabled** (*bool*) – Specifies that a group of related form elements should be disabled.
- **form** (*str*) – Specifies one or more forms the fieldset belongs to.
- **name** (*str*) – Specifies a name for the fieldset.

New in version 0.2.0.

construct()
Returns the constructed fieldset tag <fieldset></fieldset>.

class korona.html.tags.**Figure**(*text=None*)
Class for constructing figure tag.

Parameters **text** (*str*) – Specifies the figure text. (As in <figure>{text}</figure>)

New in version 0.2.0.

construct()
Returns the constructed figure tag <figure></figure>.

class korona.html.tags.**Footer**(*text=None*)
Class for constructing the footer tag.

Parameters **text** (*str*) – Specifies the footer text. (As in <footer>{text}</footer>)

New in version 0.2.0.

construct()
Returns the constructed footer tag <footer></footer>.

class korona.html.tags.**Form**(*accept=None*, *action=None*, *autocomplete=None*, *enctype=None*, *method=None*, *name=None*, *novalidate=False*, *target=None*, *text=None*)

Class for constructing <form> tag.

Parameters

- **accept** (*str*) – Specifies a comma-separated list of file types that the server accepts (that can be submitted through the file upload).
- **action** (*str*) – Specifies where to send the form-data when a form is submitted.
- **autocomplete** (*str*) – Specifies whether a form should have autocomplete on or off.
- **enctype** (*str*) – Specifies how the form-data should be encoded when submitting it to the server (only for method="post").
- **method** (*str*) – Specifies the HTTP method to use when sending form-data.
- **name** (*str*) – Specifies the name of a form.
- **novalidate** (*bool*) – Specifies that the form should not be validated when submitted.
- **target** (*str*) – Specifies where to display the response that is received after submitting the form.
- **text** (*str*) – Specifies the form text. (As in <form>{text}</form>)

New in version 0.2.0.

construct()

Returns the constructed form tag <form></form>.

validate_enctype_attribute(*method*, *enctype*)

Validates enctype attribute. The enctype attribute can be used only if method is post.

class korona.html.tags.**Frame**(*frameborder=None*, *longdesc=None*, *marginheight=None*, *marginwidth=None*, *name=None*, *noresize=None*, *scrolling=None*, *src=None*)

Class for constructing <frame> tag.

Parameters

- **frameborder** (*str*) – Specifies whether or not to display a border around a frame.
- **longdesc** (*str*) – Specifies a page that contains a long description of the content of a frame.
- **marginheight** (*str*) – Specifies the top and bottom margins of a frame.
- **marginwidth** (*str*) – Specifies the left and right margins of a frame.
- **name** (*str*) – Specifies the name of a frame.
- **noresize** (*str*) – Specifies that a frame is not resizable.
- **scrolling** (*str*) – Specifies whether or not to display scrollbars in a frame.
- **src** (*str*) – Specifies the URL of the document to show in a frame.

New in version 0.2.0.

Changed in version 0.3.1: Added URL validation for src attribute.

construct()

Returns the constructed tag <frame>.

class korona.html.tags.**FrameSet**(*cols=None*, *rows=None*)
Class for constructing <frameset> tag.

Parameters

- **cols** (*str*) – Specifies the number and size of columns in a frameset.
- **rows** (*str*) – Specifies the number and size of rows in a frameset.

New in version 0.2.0.

construct()
Returns the constructed tag <frameset>.

class korona.html.tags.**H1**(*align=None*, *text=None*)
Class for constructing <h1> tag.

Parameters

- **align** (*str*) – Specifies the alignment of a heading.
- **text** (*str*) – Specifies the <h1> text. (As in <h1>{text}</h1>)

New in version 0.3.0.

construct()
Returns the constructed tag <h1>.

class korona.html.tags.**H2**(*align=None*, *text=None*)
Class for constructing <h2> tag.

Parameters

- **align** (*str*) – Specifies the alignment of a heading.
- **text** (*str*) – Specifies the <h2> text. (As in <h2>{text}</h2>)

New in version 0.3.0.

construct()
Returns the constructed tag <h2>.

class korona.html.tags.**H3**(*align=None*, *text=None*)
Class for constructing <h3> tag.

Parameters

- **align** (*str*) – Specifies the alignment of a heading.
- **text** (*str*) – Specifies the <h3> text. (As in <h3>{text}</h3>)

New in version 0.3.0.

construct()
Returns the constructed tag <h3>.

class korona.html.tags.**H4**(*align=None*, *text=None*)
Class for constructing <h4> tag.

Parameters

- **align** (*str*) – Specifies the alignment of a heading.
- **text** (*str*) – Specifies the <h4> text. (As in <h4>{text}</h4>)

New in version 0.3.0.

construct()
Returns the constructed tag <h4>.

class korona.html.tags.**H5**(*align=None*, *text=None*)
Class for constructing <h5> tag.

Parameters

- **align** (*str*) – Specifies the alignment of a heading.
- **text** (*str*) – Specifies the <h5> text. (As in <h5>{text}</h5>)

New in version 0.3.0.

construct()
Returns the constructed tag <h5>.

class korona.html.tags.**H6**(*align=None*, *text=None*)
Class for constructing <h6> tag.

Parameters

- **align** (*str*) – Specifies the alignment of a heading.
- **text** (*str*) – Specifies the <h6> text. (As in <h6>{text}</h6>)

New in version 0.3.0.

construct()
Returns the constructed tag <h6>.

class korona.html.tags.**Head**(*text=None*)
Class for constructing <head> tag.

Parameters **text** (*str*) – Specifies the head text. (As in <head>{text}</head>)

New in version 0.3.0.

construct()
Returns the constructed tag <head>.

class korona.html.tags.**Header**(*text=None*)
Class for constructing the header tag.

Parameters **text** (*str*) – Specifies the header text. (As in <header>{text}</header>)

New in version 0.3.0.

construct()
Returns the constructed header tag <header></header>.

class korona.html.tags.**HR**(*align=None*, *noshade=False*, *size=None*, *width=None*)
Class for constructing hr tag.

Parameters

- **align** (*str*) – Specifies the alignment of a <hr> element.
- **noshade** (*bool*) – Specifies that a <hr> element should render in one solid color (noshaded), instead of a shaded color.
- **size** (*str*) – Specifies the height of a <hr> element.
- **width** (*str*) – Specifies the width of a <hr> element

New in version 0.3.0.

construct()
Returns the constructed hr tag <hr>.

class korona.html.tags.**Html**(*manifest=None*, *xmlns=None*, *text=None*)
Class for constructing html tag.

Parameters

- **manifest** (*str*) – Specifies the address of the document's cache manifest (for offline browsing)
- **xmlns** (*str*) – Specifies the XML namespace attribute (If you need your content to conform to XHTML)
- **text** (*str*) – Specifies the html text. (As in <html>{text}</html>)

New in version 0.4.0.

construct()
Returns the constructed html tag <html>.

class korona.html.tags.**I**(*text=None*)
Class for constructing tag.

Parameters **text** (*str*) – Specifies the italics text. (As in {text})

New in version 0.4.0.

construct()
Returns the constructed italics tag .

class korona.html.tags.**IFrame**(*align=None*, *frameborder=None*, *height=None*, *longdesc=None*, *marginheight=None*, *marginwidth=None*, *name=None*, *sandbox=None*, *scrolling=None*, *src=None*, *srcdoc=None*, *width=None*)
Class for constructing <iframe> tag.

Parameters

- **align** (*str*) – Specifies the alignment of an <iframe> according to surrounding elements.
- **frameborder** (*str*) – Specifies whether or not to display a border around an <iframe>.
- **height** (*str*) – Specifies the height of an <iframe>.
- **longdesc** (*str*) – Specifies a page that contains a long description of the content of an <iframe>.
- **marginheight** (*str*) – Specifies the top and bottom margins of the content of an <iframe>.
- **marginwidth** (*str*) – Specifies the left and right margins of the content of an <iframe>.
- **name** (*str*) – Specifies the name of an <iframe>.
- **sandbox** (*str*) – Enables an extra set of restrictions for the content in an <iframe>.
- **scrolling** (*str*) – Specifies whether or not to display scrollbars in an <iframe>.
- **src** (*str*) – Specifies the address of the document to embed in the <iframe>.
- **srcdoc** (*str*) – Specifies the HTML content of the page to show in the <iframe>.
- **width** (*str*) – Specifies the width of an <iframe>.

New in version 0.4.0.

construct()
Returns the constructed iframe tag <iframe>.

validate_sandbox(*sandbox*)
Validates sandbox attribute. The value of the sandbox attribute can either be just sandbox (then all restrictions are applied), or a space-separated list of pre-defined values that will REMOVE the particular restrictions.

class korona.html.tags.**Img**(*align=None*, *alt=None*, *border=None*, *crossorigin=None*, *height=None*, *hspace=None*, *ismap=False*, *longdesc=None*, *src=None*, *usemap=None*, *vspace=None*, *width=None*)
Class for constructing <img> tag.

Parameters

- **align** (*str*) – Specifies the alignment of an image according to surrounding elements.
- **alt** (*str*) – Specifies an alternate text for an image.
- **border** (*str*) – Specifies the width of the border around an image.
- **crossorigin** (*str*) – Allow images from third-party sites that allow cross-origin access to be used with canvas.
- **height** (*str*) – Specifies the height of an image.
- **hspace** (*str*) – Specifies the whitespace on left and right side of an image.
- **ismap** (*bool*) – Specifies an image as a server-side image-map.
- **longdesc** (*str*) – Specifies a URL to a detailed description of an image.
- **src** (*str*) – Specifies the URL of an image.
- **usemap** (*str*) – Specifies an image as a client-side image-map.
- **vspace** (*str*) – Specifies the whitespace on top and bottom of an image.
- **width** (*str*) – Specifies the width of an image.

New in version 0.4.0.

construct()
Returns the constructed image tag <img>.

class korona.html.tags.**Input**(*accept=None*, *align=None*, *alt=None*, *autocomplete=None*, *autofocus=False*, *checked=False*, *name=None*, *dirname=None*, *disabled=False*, *form=None*, *formaction=None*, *formenctype=None*, *formmethod=None*, *formnovalidate=False*, *formtarget=None*, *height=None*, *list=None*, *max=None*, *min=None*, *maxlength=None*, *multiple=False*, *pattern=None*, *placeholder=None*, *readonly=False*, *required=False*, *size=None*, *src=None*, *step=None*, *type=None*, *value=None*, *width=None*)
Class for constructing <input> tag.

Parameters

- **accept** (*str*) – Specifies the types of files that the server accepts (only for type="file").
- **align** (*str*) – Specifies the alignment of an image input (only for type="image").
- **alt** (*str*) – Specifies an alternate text for images (only for type= "image").
- **autocomplete** (*str*) – Specifies whether an <input> element should have autocomplete enabled.
- **autofocus** (*bool*) – Specifies that an <input> element should automatically get focus when the page loads.

- **checked** (*bool*) – Specifies that an <input> element should be pre-selected when the page loads (for type="checkbox" or type="radio").
- **dirname** (*str*) – Specifies that the text direction will be submitted.
- **disabled** (*bool*) – Specifies that an <input> element should be disabled.
- **form** (*str*) – Specifies one or more forms the <input> element belongs to.
- **formaction** (*str*) – Specifies the URL of the file that will process the input control when the form is submitted (for type="submit" and type="image").
- **formenctype** (*str*) – Specifies how the form-data should be encoded when submitting it to the server (for type="submit" and type="image").
- **formmethod** (*str*) – Defines the HTTP method for sending data to the action URL (for type="submit" and type="image").
- **formnovalidate** (*bool*) – Defines that form elements should not be validated when submitted.
- **formtarget** (*str*) – Specifies where to display the response that is received after submitting the form (for type="submit" and type="image").
- **height** (*int/float*) – Specifies the height of an <input> element (only for type="image").
- **list** (*str*) – Refers to a <datalist> element that contains pre-defined options for an <input> element.
- **max** (*int/date/time*) – Specifies the maximum value for an <input> element.
- **maxlength** (*int*) – Specifies the maximum number of characters allowed in an <input> element.
- **min** (*int/date/time*) – Specifies a minimum value for an <input> element.
- **multiple** (*bool*) – Specifies that a user can enter more than one value in an <input> element.
- **name** (*str*) – Specifies the name of an <input> element.
- **pattern** (*str*) – Specifies a regular expression that an <input> element's value is checked against.
- **placeholder** (*str*) – Specifies a short hint that describes the expected value of an <input> element.
- **readonly** (*bool*) – Specifies that an input field is read-only.
- **required** (*bool*) – Specifies that an input field must be filled out before submitting the form.
- **size** (*int*) – Specifies the width, in characters, of an <input> element.
- **src** (*str*) – Specifies the URL of the image to use as a submit button (only for type="image").
- **step** (*int*) – Specifies the legal number intervals for an input field.
- **type** (*str*) – Specifies the type <input> element to display.
- **value** (*str*) – Specifies the value of an <input> element.
- **width** (*int/float*) – Specifies the width of an <input> element (only for type="image").

New in version 0.4.2.

Changed in version 0.4.3-dev: Renamed the method `validate_value()` to `validate_value_attribute()`.

construct()
Constructs the <input> tag.

validate_dirname(*name*, *dirname*)
Validates the dirname attribute for <input> tag. The dirname attribute's value should always have the name of the input field, followed by ".dir".

validate_input_attribute(*type*, *attribute_name*, *attribute_value*)
Validates input tag attribute. Some of the input attributes depend on the input type given by the consumer.

For example, max and min attributes works with the following input types: number, range, date, datetime, datetime-local, month, time and week.

validate_max_min_attributes(*type*, *attribute_name*, *attribute_value*)
Validates the max attribute whether it is an integer/float/datetime value.

validate_value_attribute(*type*, *value*)
Validates the value attribute for <input> tag. The value attribute cannot be used with <input type="file">.

CHAPTER 5

Project Info

5.1 License

5.2 Versioning

This project follows Semantic Versioning.

It is recommended to only use or import objects from the main module, korona.

5.3 Changelog

5.3.1 v0.4.3-dev

Added

- Added custom exceptions for korona package.
- Added warnings for korona package.
- Added class `GlobalAttributes()` for constructing HTML global attributes.

Changed

- Moved all the HTML tag attributes from `html/root/attributes.py` to their respective tag files.
- Moved all the global attributes from `html/root/attributes.py` to `html/root/global_attributes.py`.
- In the class for constructing `anchor` tag:
  - The user can be displayed with warnings for charset attribute.
  - Used custom exceptions for all the attributes.
  - Changed the name of the method from `validate_values()` to `validate_attribute_values()`.

Removed

- Removed `html/root/attributes.py`.
- Removed `validate_values()` method from both `html.tags.anchor.A()` and `html.tags.button.Button()` classes.

5.3.2 v0.4.2 (2016-09-24)

Added

- Added classes for building some of the tags:
  - `<input>`

5.3.3 v0.4.1 (2016-09-16)

Changed

- Moved all the templates for html tags from `templates/html/tags.py` to `templates/html/tags/`

5.3.4 v0.4.0 (2016-09-15)

Added

- Added classes for building some of the tags:
  - `<html></html>`
  - `<i></i>`
  - `<iframe></iframe>`
  - `<img>`

Changed

- Moved all the classes for constructing tags to separate files for easy accessibility.

5.3.5 v0.3.1 (2016-09-05)

Added

- Added validation for URL strings in constructing the tags.

5.3.6 v0.3.0 (2016-09-04)

Added

- Added classes for building some of the tags:
  - `<h1></h1>`
  - `<h2></h2>`
  - `<h3></h3>`
  - `<h4></h4>`
  - `<h5></h5>`
  - `<h6></h6>`
  - `<head></head>`
  - `<header></header>`
  - `<hr>`

5.3.7 v0.2.0 (2016-09-03)

Added

- Added classes for building some of the tags:
  - `<colgroup></colgroup>`
  - `<dd></dd>`
  - `<del></del>`
  - `<details></details>`
  - `<dialog></dialog>`
  - `<div></div>`
  - `<dl></dl>`
  - `<dt></dt>`
  - `<embed>`
  - `<fieldset></fieldset>`
  - `<figure></figure>`
  - `<footer></footer>`
  - `<form></form>`
  - `<frame>`

- <frameset></frameset>

5.3.8 v0.1.0 (2016-08-28)

Added

- First release.
- Added classes for building some of the tags:
  - <a>
  - <abbr></abbr>
  - <acronym></acronym>
  - <address></address>
  - <area>
  - <article></article>
  -
  - <base>
  - <button></button>
  - <canvas></canvas>
  - <caption></caption>
  - <cite></cite>

5.4 Authors

5.4.1 Lead

- Bharadwaj Yarlagadda, yarlagaddabharadwaj@gmail.com, bharadwajyarlagadda@github

5.5 Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

5.5.1 Types of Contributions

Report Bugs

Report bugs at https://github.com/bharadwajyarlagadda/korona.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.

- Detailed steps to reproduce the bug.

Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" is open to whoever wants to implement it.

Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" or "help wanted" is open to whoever wants to implement it.

Write Documentation

korona could always use more documentation, whether as part of the official korona docs, in docstrings, or even on the web in blog posts, articles, and such.

Submit Feedback

The best way to send feedback is to file an issue at https://github.com/bharadwajyarlagadda/korona.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

5.5.2 Get Started!

Ready to contribute? Here's how to set up `korona` for local development.

1. Fork the `korona` repo on GitHub.
2. Pull your fork locally:

```
$ git clone git@github.com:<username>/korona.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenv installed, this is how you set up your fork for local development:

```
$ cd korona
$ pip install -r requirements-dev.txt
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass linting and all unit tests by testing with tox across all supported Python versions:

```
$ invoke tox
```

6. Add yourself to `AUTHORS.rst`.

7. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

8. Submit a pull request through the GitHub website.

5.5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the README.rst.
3. The pull request should work for Python 2.7, 3.4, and 3.5. Check https://travis-ci.org/bharadwajyarlagadda/korona/pull_requests and make sure that the tests pass for all supported Python versions.

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

G

H

I

P

V