pug-mixins Documentation

Release 0.0.2

Álvaro Mondéjar Rubio

Aug 18, 2018

User's guide

CHAPTER 1

Installation

1.1 Manual

1. Download the project from https://github.com/mondeja/pug-mixins/archive/master.zip
2. Copy `src` folder in your project rename it as you want.

1.2 Command line

1.2.1 Linux

```
git clone https://github.com/mondeja/pug-mixins.git
cp -r pug-mixins/src mixins && rm -rf pug-mixins
```

You can see a directory named `mixins`, inside of which the library is located. You can rename it and copy inside your project and include the files.

1.2.2 Windows

```
git clone https://github.com/mondeja/pug-mixins.git
xcopy pug-mixins\src mixins
rmdir "pug-mixins" /S /Q
```

You can see a directory named `mixins`, inside of which the library is located. You can rename it and copy inside your project and include the files.

CHAPTER 2

Basic usage

Only include a file for use mixins located inside. You can do this specyfing pug includes at the beggining of your template.

For example if I have a file called `example.pug` inside `src/` folder and I need to use `+script` mixin located at `src/html/script.pug`:

```
include ./html/script.pug
```

See also:

Pug documentation and *Reference*.

CHAPTER 3

Custom contexts

Some mixins needs to pass specific locals at compilation script. For example, mixin *+table-json()* needs a context with `require` NodeJS function. The library includes some contexts in `src/contexts.js` which can be imported in your preprocessing script:

pug_mixins_context
: All context included in `contexts.js` file merged in one context.

require_context
: Context with only `require` NodeJS function: `{require: require}`.

CHAPTER 4

Reference

4.1 `audio/` — Podcasts and playlists

4.1.1 `ivoox.pug` – Ivoox podcasts players

Audios

```
mixin ivoox-audio(id)
  - src = `https://www.ivoox.com/player_ej_${id}_4_1.html?`
```

(continues on next page)

(continued from previous page)

```
  - if ("c1" in attributes)
    - c1 = attributes.c1;
    - delete attributes.c1;
    - src = src + `c1=${c1}`
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  iframe(frameborder="0"
         allowfullscreen="true"
         scrolling="no"
         src=`${src}`
         width="100%"
         height="200")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+ivoox-audio (*id)(c1=null*, *width="100%"*, *height="200"*)

Embed an audio from Ivoox.

Arguments

- **id** (*string, integer*) – Audio identificator. You can get it from the url of an audio page: `https://www.ivoox.com/...audios-mp3_rf_<THIS_NUMBER>_1.html`.
- **c1** (*string, optional*) – Main color of the embedded player in HEX format without # character. As default `null` (default Ivoox main color will be used: `"FF6600"`).

Usage

Input

```
+ivoox-audio(25288583)(c1="007300")
```

Output

```
<iframe frameborder="0" allowfullscreen="true" scrolling="no" src="https://www.ivoox.
↪com/player_ej_25288583_4_1.html?c1=007300" width="100%" height="200"></iframe>
```

Render

Podcasts

```
mixin ivoox-podcast(id)
  - if ("alt" in attributes)
    - alt = attributes.alt;
```

(continues on next page)

(continued from previous page)

```
    - delete attributes.alt;
  - else
    - alt = true;
  iframe(src=`https://www.ivoox.com/player_es_podcast_${id}_1.html`
         frameborder="0"
         allowfullscreen="0"
         scrolling="no"
         width="100%"
         height="440")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+ivoox-podcast (*id)(width="100%"*, *height="440"*)

Embed a podcast from Ivoox.

Arguments

- **id** (*string*) – Podcast identificator. You can get it from the url of a podcast page: `https://www.ivoox.com/..._sq_f1<THIS_NUMBER>_1.html`.

Usage

Input

```
+ivoox-podcast(70523)
```

Output

```
<iframe src="https://www.ivoox.com/player_es_podcast_70523_1.html" frameborder="0"␣
↪allowfullscreen="0" scrolling="no" width="100%", height="440"></iframe>
```

Render

Playlists

```
mixin ivoox-playlist(id)
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  iframe(src=`https://www.ivoox.com/player_es_channel_${id}_1.html`
         frameborder="0"
         allowfullscreen="0"
         scrolling="no"
         width="100%"
         height="440")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+ivoox-playlist (*id)(width="100%"*, *height="440"*)
Insert an Ivoox channel playlist.

Arguments

- **id** (*integer, string*) – Your channel identificator. You can obtain it from your channel page url: `https://www.ivoox.com/..._nq_<THIS_NUMBER>_1.html`.

Usage

Input

```
+ivoox-playlist(40562)
```

Output

```
<iframe src="https://www.ivoox.com/player_es_channel_40562_1.html" frameborder="0"␣
↪allowfullscreen="0" scrolling="no" width="100%" height="440"></iframe>
```

Render

Channel subscription widget

```
mixin ivoox-channel-subscription(id)
  - src = `https://www.ivoox.com/_ns_${id}_0_.html?`
  - if ("c1" in attributes)
    - c1 = attributes.c1;
    - delete attributes.c1;
    - src = src + `c1=${c1}&`
  - if ("c2" in attributes)
    - c2 = attributes.c2;
    - delete attributes.c2;
    - src = src + `c2=${c2}&`
  - if ("r" in attributes)
    - r = attributes.r;
    - delete attributes.r;
    - src = src + `r=${r}&`
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  iframe(src=`${src}`
         frameborder="0"
         scrolling="no"
         allowfullscreen="true"
         height="174")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+ivoox-channel-subscription (*id*)(*c1="555555"*, *c2="ffffff"*, *r=null*, *height="174"*)
Insert an Ivoox subscription widget.

Arguments

- **id** (*integer, string*) – Your channel identificator. You can obtain it from your channel page url: `https://www.ivoox.com/..._nq_<THIS_NUMBER>_1.html`.
- **c1** (*string*) – Widget background color with HEX format without # character. As default `"555555"`.
- **c1** – Widget text color with HEX format without # character. As default `"ffffff"`.

Usage

```
+ivoox-channel-subscription(40562)(c1="007300", c2="ffffff")
```

```
<iframe src="https://www.ivoox.com/_ns_40562_0_.html?c1=007300&c2=ffffff" frameborder=
↪"0" scrolling="no" allowfullscreen="true" height="174"></iframe>
```

4.1.2 `soundcloud.pug` – Soundcloud podcasts players

Embed Soundcloud user

```
mixin soundcloud-user(id)
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  - if ("color" in attributes)
    - color = attributes.color;
    - delete attributes.color;
  - else
    - color = "ff8800";
  - if ("auto_play" in attributes)
    - auto_play = attributes.auto_play;
    - delete attributes.auto_play;
  - else
    - auto_play = false;
  - if ("hide_related" in attributes)
    - hide_related = attributes.hide_related;
    - delete attributes.hide_related;
  - else
    - hide_related = false;
  - if ("show_comments" in attributes)
    - show_comments = attributes.show_comments;
    - delete attributes.show_comments;
  - else
    - show_comments = true;
  - if ("show_user" in attributes)
    - show_user = attributes.show_user;
```

(continues on next page)

(continued from previous page)

```
    - delete attributes.show_user;
  - else
    - show_user = true;
  - if ("show_reposts" in attributes)
    - show_reposts = attributes.show_reposts;
    - delete attributes.show_reposts;
  - else
    - show_reposts = false;
  - if ("show_teaser" in attributes)
    - show_teaser = attributes.show_teaser;
    - delete attributes.show_teaser;
  - else
    - show_teaser = true;
  iframe(src=`https://w.soundcloud.com/player/?url=https%3A//api.soundcloud.com/users/
↪${id}&color=%23${color}&auto_play=${auto_play}&hide_related=${hide_related}&show_
↪comments=${show_comments}&show_user=${show_user}&show_reposts=${show_reposts}&show_
↪teaser=${show_teaser}`
          width="100%"
          height="300"
          scrolling="no"
          frameborder="no"
          allow="autoplay")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+soundcloud-user(*id)(color="ff8800"*, *auto_play=false*, *hide_related=false*, *show_comments=true*, *show_user=true*, *show_reposts=false*, *show_teaser=true*, *width="100%"*, *height="300"*)

Arguments

- **id** (*string*) – Soundcloud user identificator. You can get it from an user Soundcloud page url.
- **color** (*string*) – Player color passed as HEX color withour # character.
- **autoplay** (*bool*) – Play tracklist when content is loaded. As default `false`.
- **hide_related** (*bool*) – Hide related content if `true`. As default `false`.
- **show_comments** (*bool*) – Show user comments on displayed tracks. As default `true`.
- **show_user** – Show uploader name. As default `true`.
- **show_reposts** (*bool*) – Show reposts. As default `false`.
- **show_teaser** (*bool*) – Show teaser. As default `true`.

Usage

Input

```
+soundcloud-user("alvaromondejar")(show_comments=false)
```

Output

```
<iframe src="https://w.soundcloud.com/player/?url=https%3A//api.soundcloud.com/users/
↪alvaromondejar&color=%23ff8800&auto_play=false&hide_related=false&show_
↪comments=false&show_user=true&show_reposts=false&show_teaser=true" width="100%"␣
↪height="300" scrolling="no" frameborder="no" allow="autoplay"></iframe>
```

Render

4.2 `blog/` — Blogging tools

4.2.1 `storify.pug` - Storify stories

Embed stories

```
mixin storify(user, slug)
  - if ("border" in attributes)
    - border = attributes.border;
    - delete attributes.border;
  - else
    - border = false;
  - if ("post_title" in attributes)
    - post_title = attributes.post_title;
    delete attributes.post_title;
  - else
    - post_title = slug.replace("-", " ")
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  .storify
    iframe(src=`https://storify.com/${user}/${slug}/embed?border=${border}`
           width="100%" height="750" frameborder="no" allowtransparency="true")
    script(src=`https://storify.com/${user}/${slug}.js?border=${border}`)
    - if (alt)
      noscript
```

(continues on next page)

(continued from previous page)

```
        | [
        a(href=`https://storify.com/${user}/${slug}` target="__blank") View the story
↪"#{post_title}" on Storify
        | ]
```

+storify (*user*, *slug)(border=false*, *post_title=null*, *width="100%"*, *height="750"*)
Embed a story from Storify.

Arguments

- **user** (*string*) – Story publisher username.
- **slug** (*string*) – Story identificator. You can obtain it from a story page url.
- **border** (*bool, optional*) – Specify `border` parameter value in `src` endpoint of included `iframe`. As default `false`.
- **post_title** (*string, optional*) – Post title used if `alt` is `true`. Note that if user browser supports `iframe` tags, this title will not be shown. If you don't pass this parameter, this will be calculated from `slug` using `slug.replace("-", " ")`. As default `null`.

Usage

```
+storify("journalstarnews", "2018-lincoln-marathon")(alt=false)
```

```
<div class="storify">
  <iframe src="https://storify.com/journalstarnews/2018-lincoln-marathon/embed?
↪border=false" width="100%" height="750" frameborder="no" allowtransparency="true"></
↪iframe>
  <script src="https://storify.com/journalstarnews/2018-lincoln-marathon.js?
↪border=false"></script>
</div>
```

4.3 board/ — Content-boards

4.3.1 livebinders.pug - Livebinders boards

Binders by iframes

```
mixin livebinder(id)
  - if ("present" in attributes)
    - present = attributes.present;
    - delete attributes.present;
  - else
    - present = true;
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
```

(continues on next page)

(continued from previous page)

```
    - alt = true;
  iframe(src=`https://www.livebinders.com/play/play?id=${id}&present=${present}`
         frameborder="0")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+livebinder (*id)(present=true*)

Insert a content-board from Livebinders.

Arguments

- **id** (*integer, string*) – Identificator of the binder. You can obtain it from a binder page url.

Usage

Input

```
+livebinder(2368302)(width="100%", height="400")
```

Output

```
<iframe src="https://www.livebinders.com/play/play?id=2368302&present=true"␣
↪frameborder="0" width="100%" height="400">Your browser does not support iframes.</
↪iframe>
```

Render

4.3.2 padlet.pug – Content boards from Padlet

Insert boards

```
mixin padlet(id)
  - if ("header" in attributes || "footer" in attributes)
    - iframe_only = false
  - else
    - iframe_only = true
  - if ("header" in attributes)
    - header = true;
    - delete attributes.header;
  - else
    - header = false;
  - if ("footer" in attributes)
    - footer = true;
    - delete attributes.footer;
  - else
```

(continues on next page)

(continued from previous page)

```
    - footer = false;
  - if (iframe_only)
    iframe(src=`https://padlet.com/embed/${id}`
           frameborder="0")&attributes(attributes)
  - else
    - if (header)
      .padlet-embed(style="border:1px solid rgba(0,0,0,0.1);border-radius:2px;box-
↪sizing:border-box;overflow:hidden;position:relative;width:100%;background:#F4F4F4")&
↪attributes(attributes)
    - else
      .padlet-embed(style="overflow:hidden;position:relative;")
        p(style="padding:0;margin:0")
          iframe(src=`https://padlet.com/embed/${id}`
             frameborder="0")&attributes(attributes)
        - if (footer)
          div(style="padding:8px;text-align:right;margin:0;")
            a(href="https://padlet.com?ref=embed"
              style="padding:0;margin:0;border:none;display:block;line-height:1;
↪height:16px"
              target="_blank")
              img(src="https://resources.padletcdn.com/assets/made_with_padlet.png"
                  width="86" height="16"
                  style="padding:0;margin:0;background:none;border:none;
↪display:inline;box-shadow:none"
                  alt="Made with Padlet")
```

+padlet (*id)(header=false*, *footer=false*)

Include a content board from Padlet.

Arguments

- **id** (*string*) – Board identificator.
- **header** (*bool, optional*) – If `true`, include the header. As default `false`.
- **footer** (*bool, optional*) – If `true`, include the footer. As default `false`.

Usage

Input

```
+padlet("v8nee7607dvc")(width="100%", height="600")
```

Output

```
<iframe src="https://padlet.com/embed/v8nee7607dvc" frameborder="0" width="100%"␣
↪height="600"></iframe>
```

Render

4.4 code/ — Programming tools

4.4.1 codepen.pug – Modern pens

Embed pens

```
mixin codepen(id, user)
  - if ("default_tab" in attributes)
    - default_tab = attributes.default_tab;
    - delete attributes.default_tab;
  - else
    - default_tab = "result";
  - if ("embed_version" in attributes)
    - embed_version = attributes.embed_version;
    - delete attributes.embed_version;
  - else
    - embed_version = 2;
  - src = `http://codepen.io/${user}/embed/${id}/?default-tab=${default_tab}&embed-
↪version=${embed_version}`
  - if ("theme_id" in attributes)
    - theme_id = attributes.theme_id;
    - delete attributes.theme_id;
    - src = src + `&theme-id=${theme_id}`
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  iframe(scrolling="no"
         frameborder="0"
         allowtransparency="true"
         allowfullscreen="true"
         src=`${src}`
         width="100%"
         height="300")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+codepen (*id*, *user)(default_tab="result"*, *embed_version=2*, *theme_id=null*, *width="100%"*, *height="400"*)

Embed a pen from Codepen passing a pen identificator and their user.

> **Note:** You can obtain identificator and user of a pen from the url of their page. For example: https://codepen.io/user/pen/id.

Arguments

- **id** (*string*) – Pen id.
- **user** (*string*) – Pen user.

- **default_tab** (*string, optional*) – Tab shown after the pen is loaded. As default `"result"`.
- **embed_version** (*integer, string, optional*) – Codepen embed API version used for current insertion. As default `2`.
- **theme_id** (*integer, string, optional*) – Identificator of your customized theme. You can create your own theme from Codepen Embed Theme Builder.

Usage

Input

```
+codepen("ZameGP", "mondeja")(theme_id=33136)
```

Output

```
<iframe src="http://codepen.io/mondeja/embed/ZameGP/?default-tab=result&embed-
↪version=2&theme-id=33136" scrolling="no" frameborder="0" allowtransparency="true"␣
↪allowfullscreen="true" width="100%" height="300"></iframe>
```

Render

4.4.2 `console.pug` – Insert consoles everywhere

Brython console

```
mixin brython-console()
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  iframe(src="http://brython.info/console.html"
         scrolling="no"
         width="75%"
         height="165")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+brython-console()*(width="75%" height="165")*

Embed a Python console iframe made with Brython for the browser.

Usage

Input

```
+brython-console()
```

Output

```
<iframe width="75%" height="165" src="http://brython.info/console.html"></iframe>
```

Render

Javascript console

```
mixin js-console()
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  iframe(src="https://jsconsole.com/"
         width="75%"
         height="165")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+js-console()*(width="75%" height="165")*

Embed a Javascript console iframe from http://jsconsole.com/.

Usage

Input

```
+js-console()
```

Output

```
<iframe width="75%" height="165" src="https://jsconsole.com/"></iframe>
```

Render

4.4.3 `gist.pug` – Github gists mixins

Embed gists

```
mixin gist(user, id)
  - if (id.length != 32)
    - throw new Error("Invalid gist id length. Must be 32 but is " + id.length)
  script(src=`https://gist.github.com/${user}/${id}.js`)&attributes(attributes)
```

+gist (*user*, *id*)

Embed a gist passing their user and identificator. You can obtain these parameters from the gist page url.

Arguments

- **user** (*string*) – Gist owner username.
- **id** (*string*) – Gist identificator

Usage

Input

```
gist("mondeja", "2bd82917552ff2589ddffc4a92744825")
```

Output

```
<script src="https://gist.github.com/mondeja/2bd82917552ff2589ddffc4a92744825.js"></
↪script>
```

Render

4.4.4 `jsfiddle.pug` – JsFiddle web editor

Fiddles

```
mixin jsfiddle(user, id)
  - src = `https://jsfiddle.net/${user}/${id}/embedded/`
  - if ("tabs" in attributes)
    - tabs = attributes.tabs;
    - delete attributes.tabs;
  - else
    - tabs = "js,html,css,result";
  - src = src + tabs + "/"
  - if ("theme" in attributes)
    - theme = attributes.theme;
    - delete attributes.theme;
  - else
    - theme = "light";
  - src = src + theme + "/?";
  - _first_color_param = false;
  - if ("font_color" in attributes)
```

(continues on next page)

(continued from previous page)

```
    - font_color = attributes.font_color;
    - delete attributes.font_color;
    - src = src + "fontColor=" + font_color;
    - _first_color_param = true;
  - if ("accent_color" in attributes)
    - accent_color = attributes.accent_color;
    - delete attributes.accent_color;
    - if (_first_color_param)
      - src = src + "&"
    - src = src + "accentColor=" + accent_color;
    - _first_color_param = true;
  - if ("code_background" in attributes)
    - code_background = attributes.code_background;
    - delete attributes.code_background;
    - if (_first_color_param)
      - src = src + "&"
    - src = src + "codeBackground=" + code_background;
    - _first_color_param = true;
  - if ("menu_background" in attributes)
    - menu_background = attributes.menu_background;
    - delete attributes.menu_background;
    - if (_first_color_param)
      - src = src + "&"
    - src = src + "menuBackground=" + menu_background;
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  iframe(src=`${src}`
         allowpaymentrequest="true"
         allowfullscreen="true"
         width="100%"
         height="400"
         frameborder="0")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+jsfiddle(*user*, *id)(tabs="js*, *html*, *css*, *result"*, *theme="light"*, *font_color=null*, *accent_color=null*, *code_background=null*, *menu_background=null*, *width="100%"*, *height="400"*)

Embed a fiddle from JsFiddle.

Arguments

- **user** (*string*) – Owner username of the fiddle.
- **id** (*string*) – Fiddle identificator. You can extract it from url of the fiddle page.
- **tabs** (*string*) – Tabs shown in the fiddle, separated by commas. By default, all: `"js,html,css,result"`.
- **theme** (*string*) – Fiddle theme. Valid are `"light"` and `"dark"`. As default `"light"`.

Note: Parameters `font_color`, `accent_color`, `code_background` and `menu_background` accept HEX colors string without `#` starting character.

Usage

Input

```
+jsfiddle("mondeja", "xf3tvvw0")(tabs="html,css,result", theme="dark")
```

Output

```
<iframe src="https://jsfiddle.net/mondeja/xf3tvvw0/embedded/html,css,result/dark/"␣
↪allowpaymentrequest="true" allowfullscreen="true" frameborder="0" width="100%"␣
↪height="400"></iframe>
```

Render

4.4.5 `pastebin.pug` – Clean Pastebin pastes

Embed pastes

```
mixin pastebin(id)
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  iframe(src=`https://pastebin.com/embed_iframe/${id}`
         width="100%"
         height="315")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+pastebin(*id*)

Insert a paste from Pastebin.

Arguments

- **id** (*string*) – Paste identificator, corresponds to some letters and numbers located at the end of a paste url. For example, if the url is https://pastebin.com/DHa6btr5, the identificator is `DHa6btr5`.

Usage

Input

```
+pastebin("DHa6btr5")(width="100%" height="315")
```

Output

```
<iframe width="100%" height="315" src="https://pastebin.com/embed_iframe/DHa6btr5"></
↪iframe>
```

Render

4.4.6 `pythontutor.pug` – Code executions step by step

Insert a PythonTutor `iframe`

```
mixin tutor(lang="python")
  - valid_langs = ["python", "c", "cpp", "java", "javascript", "ruby"];
  - if (valid_langs.indexOf(lang) > 0)
    - if (lang == "python")
      - endpoint = "visualize";
    - else
      - endpoint = lang;
  - else
    - throw new Error(lang + " is not a valid language. Valid languages: " + valid_
↪langs);
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  iframe(width="100%"
         height="500"
         frameborder="0"
         src=`http://pythontutor.com/${endpoint}.html#cumulative=false&
↪heapPrimitives=nevernest&mode=edit&origin=opt-frontend.js&py=3&rawInputLstJSON=%5B
↪%5D&textReferences=false`)
    - if (alt)
      | Your browser does not support iframes.
```

+tutor(*lang="python")(width="100%", height="500"*)

Insert a PythonTutor `iframe` passing a language name as first parameter.

Arguments

- **lang** (*string*) – Code language. Valid languages are: `"python"`, `"c"`, `"cpp"`, `"java"`, `"javascript"`, `"ruby"`.

Usage

Input

```
+tutor()
```

Output

```
<iframe width="100%" height="500" frameborder="0" src="http://pythontutor.com/
↪visualize.html#cumulative=false&heapPrimitives=nevernest&mode=edit&origin=opt-
↪frontend.js&py=3&rawInputLstJSON=%5B%5D&textReferences=false">Your browser does not␣
↪support iframes.</iframe>
```

Render

4.4.7 `shield.pug` – Developer shields

Shields by urls

```
mixin shield-urls(image_url, target_url)
  a(href=target_url target="__blank")&attributes(attributes)
    img(src=image_url)
```

+shield-urls (*image_url*, *target_url*)

Insert a shield passing the image url and link href.

Arguments

- **image_url** (*string*) – Image url of the shield.
- **target_url** (*string*) – Target url of the shield.

Usage

Input

```
+shield-urls("https://travis-ci.org/mondeja/pymarketcap.svg?branch=master", "https://
↪cnhv.co/1xgw5")
```

Output

```
<a href="https://cnhv.co/1xgw5" target="__blank">
  <img src="https://travis-ci.org/mondeja/pymarketcap.svg?branch=master">
</a>
```

Render

Shields by services

```
mixin shield(service, project_id)
  - if (attributes)
    - if ("branch" in attributes)
      - branch = attributes.branch;
      - delete attributes.branch;
    - else
      - branch = "master";
    - if ("version" in attributes)
      - version = attributes.version;
      - delete attributes.version;
    - else
      - version = "latest";
    - if ("url" in attributes)
      - url = attributes.url;
      - delete attributes.url;
    - else
      - url = null;
  - else
    - attributes = {};
  - if (service == "travis")
    - url = url || `https://travis-ci.org/${project_id}`;
    +shield-urls(
      `https://travis-ci.org/${project_id}.svg?branch=${branch}`,
      `${url}`
    )&attributes(attributes)
  - else if (service == "pypi-version")
    - url = url || `https://pypi.org/project/${project_id}`;
    +shield-urls(
      `https://img.shields.io/pypi/v/${project_id}.svg`,
      `${url}`
    )&attributes(attributes)
  - else if (service == "pypi-py-versions")
    - url = url || `https://pypi.org/project/${project_id}`;
    +shield-urls(
      `https://img.shields.io/pypi/pyversions/${project_id}.svg`,
      `${url}`
    )&attributes(attributes)
  - else if (service == "pypi-status")
    - url = url || `https://pypi.org/project/${project_id}`;
    +shield-urls(
      `https://img.shields.io/pypi/status/${project_id}.svg`,
      `${url}`
    )&attributes(attributes)
  - else if (service == "pypi-license")
    - url = url || `https://pypi.org/project/${project_id}`;
    +shield-urls(
      `https://img.shields.io/pypi/l/${project_id}.svg`,
      `${url}`
    )&attributes(attributes)
  - else if (service == "binder")
    - url = url || `https://mybinder.org/v2/gh/${project_id}/${branch}`;
    +shield-urls(
      `https://mybinder.org/badge.svg`,
      `${url}`
    )&attributes(attributes)
  - else if (service == "readthedocs" || service == "rtd")
    - url = url || `http://${project_id}.readthedocs.io/`;
```

(continues on next page)

(continued from previous page)

```
    +shield-urls(
      `https://readthedocs.org/projects/${project_id}/badge/?version=${version}`,
      `${url}`
    )&attributes(attributes)
  - else if (service == "ask-me-anything")
    - url = url || project_id;
    +shield-urls(
      `https://camo.githubusercontent.com/d52b9239d76d77ebff4fc954745ee8ba555338ee/
↪68747470733a2f2f696d672e736869656c64732e696f2f62616467652f41736b2532306d652d616e797468696e672d31616
↪
      `${url}`
    )&attributes(attributes)
  - else if (service == "github-issues")
    - url = url || `https://github.com/${project_id}/issues`;
    +shield-urls(
      `https://img.shields.io/github/issues/${project_id}.svg`,
      `${url}`
    )&attributes(attributes)
  - else if (service == "github-issues-closed")
    - url = url || `https://github.com/${project_id}/issues?q=is%3Aissue+is%3Aclosed`;
    +shield-urls(
      `https://img.shields.io/github/issues-closed/${project_id}.svg`,
      `${url}`
    )&attributes(attributes)
  - else if (service == "github-issues-closed-in")
    - url = url || `https://github.com/${project_id}/issues`;
    +shield-urls(
      `https://img.shields.io/issuestats/i/long/github/${project_id}.svg`,
      `${url}`
    )&attributes(attributes)
  - else if (service == "github-last-commit")
    - url = url || `https://github.com/${project_id}/commits`;
    +shield-urls(
      `https://img.shields.io/github/last-commit/${project_id}.svg`,
      `${url}`
    )&attributes(attributes)
  - else if (service == "github-contributors")
    - url = url || `https://github.com/${project_id}/graphs/contributors`;
    +shield-urls(
      `https://img.shields.io/github/contributors/${project_id}.svg`,
      `${url}`
    )&attributes(attributes)
  - else if (service == "isitmaintained-issues-open-perc")
    - url = url || `http://isitmaintained.com/project/${project_id}`;
    +shield-urls(
      `http://isitmaintained.com/badge/open/${project_id}.svg`,
      `${url}`
    )&attributes(attributes)
  - else if (service == "isitmaintained-issues-closed-in")
    - url = url || `http://isitmaintained.com/project/${project_id}`;
    +shield-urls(
      `http://isitmaintained.com/badge/resolution/${project_id}.svg`,
      `${url}`
    )&attributes(attributes)
```

+shield (*service*, *project_id*, *url=null*, *branch=null*, *version=null*)

Include shields passing, as minimum, their service and project id.

Arguments

- **service** (*string*) – Shield service. Valid services are:
  - `"travis"`: Insert a travis shield. Pass the parameter `branch` with a valid project branch or the branch used will be `"master"`.
  - `"pypi-version"`: Shield that shows version of a project hosted at Pypi defined by `project_id` parameter.
  - `"pypi-py-version"`: Shows versions defined by a python project setup's classifiers param hosted at Pypi.
  - `"pypi-status"`: Shows a shield with status information of a project hosted at Pypi.
  - `"pypi-license"`: Shows a shield with the license of a project hosted at Pypi.
  - `"binder"`: Inserts a Binderhub shield that links to an IPython console for a repository. Pass the parameter `branch` with a valid project branch or the branch used will be `"master"`. Only works for Github projects.
  - `"readthedocs"/"rtd"`: Insert a Readthedocs shield. You need to pass the optional argument `version`, by default `"latest"` will be used.
  - `"ask-me-anything"`: Insert an "Ask me anything" shield. You need to pass the url link target in the optional parameter `url`.
  - `"github-issues"`: Shows number of issues opened in a Github project.
  - `"github-issues-closed"`: Shows number of issues closed in a Github project.
  - `"github-issues-closed-in"`: Shows mean time spended closing a issue in a Github project.
  - `"github-last-commit"`: Shows a shield with how many time has elapsed since last commit in a project.
  - `"github-contributors"`: Shows number of contributors that have been participated in a project.
  - `"isitmaintained-issues-open-perc"`: Shows the percentage of issues open against total issues. This data is provided by Isitmaintained and is useful to know if a project are currently been maintained.
  - `"isitmaintained-issues-closed-in"`: Shows mean time spended closing a issue in a Github project provided by Isitmaintained.
- **project_id** (*string*) – Identificator of the project. Depends of service, some needs only a project name like the Pypi shields, but others like Github shields need a string including the repository author like `"mondeja/pug-mixins"`. Only ask yourself, this service works identificating their hosted projects by an author and project name or only needs the project name?
- **url** (*string, optional*) – Target url of a shield. Can be used to replace manually default shields link targets. As default `null`.
- **branch** (*string, optional*) – Branch of a project. As default `"master"`.
- **version** (*string, optional*) – Version of the project. Is used only by `"readthedocs"` service for now. As default `latest`.

Usage

Input

```
+shield("readthedocs", "pymarketcap")(version="latest")
```

Output

```
<a href="http://pymarketcap.readthedocs.io/" target="__blank">
  <img src="https://readthedocs.org/projects/pymarketcap/badge/?version=latest">
</a>
```

Render

4.5 `crypto/` — Blockchain utils

4.5.1 `coinbase.pug` – Cryptocurrencies payment processor

Coinbase allows customized easy payments processing with some cryptocurrencies like Bitcoin (BTC), Ethereum (ETH), Bitcoin Cash (BCH) and Litecoin (LTC).

In order to use all mixins explained in this page, you need to follow next steps:

1. Go to Coinbase Commerce page, sign up and/or sign in.
2. Go to Dashboard. At left-bottom corner of your screen must be a button with the text `Accept payments`, click it.
3. You can select between sell a product or accept donations. This option will change the process of payment.
4. Customize your product or organization for donations and Coinbase will show you some code for embed it in your page. Search in the code the line `href="https://coinbase.commerce.com...` and copy the identificator located after `checkout/`. This one must look like `05bc9951-75aa-4c8a-b0f4-aadc0f9ad3e7`. Use this identification to pass it at first parameter of `+coinbase` and `+coinbase-default` mixins.
5. Go to Settings section and add the index url of your documentation to whitelisted domains list.

Coinbase Commerce script

```
mixin coinbase-script()
  script(src="https://commerce.coinbase.com/v1/checkout.js")&attributes(attributes)
```

+coinbase-script()

All coinbase commerce mixins doesn't includes by default the necessary script that make works these payment implementations. You can include where you want with this mixin.

Usage

Input

```
+coinbase-script()
```

Output

```
<script src="https://commerce.coinbase.com/v1/checkout.js">
```

Custom links

```
mixin coinbase(id, span_text, include_script=false)
  a(href=`https://commerce.coinbase.com/checkout/${id}`)&attributes(attributes)
    span #{span_text}
  - if (include_script)
    script(src="https://commerce.coinbase.com/v1/checkout.js")
```

+coinbase(*id*, *span_text*, *include_script=false*)

Insert a link to your product/organization without any style.

Arguments

- **id** (*string*) – Your product indentificator, as explained above.
- **span_text** (*string*) – Text of the link.
- **include_script** (*bool, string*) – Interaction with coinbase commerce only is possible including a script. You can pass `include_script=true` to include it after the link, but maybe you prefer include inside in your `head` scripts section (see `+coinbase-script()` mixin above). As default, `"false"`.

Usage

Input

```
+coinbase("05bc9951-75aa-4c8a-b0f4-aadc0f9ad3e7", "Siglo25 donations")
```

Output

```
<a href="https://commerce.coinbase.com/checkout/05bc9951-75aa-4c8a-b0f4-aadc0f9ad3e7">
↪<span>Siglo25 donations</span></a>
```

Render

Without mixins

Input

```
a(href=`https://commerce.coinbase.com/checkout/05bc9951-75aa-4c8a-b0f4-aadc0f9ad3e7`)␣
↪My basic link
```

Output

```
<a href="https://commerce.coinbase.com/checkout/05bc9951-75aa-4c8a-b0f4-aadc0f9ad3e7">
↪My basic link</a>
```

Render

Default links

```
mixin coinbase-default(id, span_text, include_script=false)
  a.donate-with-crypto(href=`https://commerce.coinbase.com/checkout/${id}`)&
↪attributes(attributes)
    span #{span_text}
  - if (include_script)
    script(src="https://commerce.coinbase.com/v1/checkout.js")
```

+coinbase-default (*id*, *span_text*, *include_script=false*)

Insert a link that will be styled like a button with Coinbase colors palette.

Arguments

- **id** (*string*) – Your product indentificator, as explained above.
- **span_text** (*string*) – Text of the button.
- **include_script** (*bool, string*) – Interaction with coinbase commerce only is possible including a script. You can pass `include_script=true` to include it after the link, but maybe you prefer include inside in your `head` scripts section (see `+coinbase-script()` mixin above). As default, `"false"`.

Usage

Input

```
+coinbase-default("05bc9951-75aa-4c8a-b0f4-aadc0f9ad3e7", "Siglo25 donations",␣
↪include_script="true")
```

Output

```
<a class="donate-with-crypto" href="https://commerce.coinbase.com/checkout/05bc9951-
↪75aa-4c8a-b0f4-aadc0f9ad3e7">
  <span>Siglo25 donations</span>
</a>
<script src="https://commerce.coinbase.com/v1/checkout.js">
```

Render

4.6 `functional/` — Functional programming

4.6.1 `loop.pug` – Recursive HTML generation

Include blocks X times

```
mixin loop(count)
  - var n = 0
  while n < count
    block
    - n++
```

+loop(*count*)

Insert a block inherited from this mixin a defined number of times.

Arguments

- **count** (*int*) – Number of reptetitions of the inherited block.

Usage

Input

```
+loop(3)
  a(href="#") 3 links
```

Output

```
<a href="#">3 links</a>
<a href="#">3 links</a>
<a href="#">3 links</a>
```

Render

4.7 `html/` — HTML tags utilities

4.7.1 `ol.pug` – Ordered lists generation

Ordered list from string

```
mixin ol-string(value, separator=",")
  ol&attributes(attributes)
    each elem in value.split(separator)
      li #{elem}
```

+ol-string(*value*, *separator=","*)

Insert a ordered list from a string indicating a separator to split the string.

Arguments

- **value** (*string*) – String from which the list will be built.
- **separator** (*string*) – The string will be splitted into differents elements using a separator passed as value of this parameter. As default `","`.

Usage

Input

```
+ol-string("one,two,three,four,five")
```

Output

```
<ol>
  <li>one</li>
  <li>two</li>
  <li>three</li>
  <li>four</li>
  <li>five</li>
</ol>
```

Render

4.7.2 `script.pug` – Powerful `script` macros

Load external scripts

```
mixin script(src)
  script(src=`${src}`
         type="text/javascript"
         charset="utf-8")&attributes(attributes)
```

+script(*src*)

Insert an external script avoiding to specify `type` and `charset` HTML attributes.

Arguments

- **src** (*string*) – External script src.

Usage

Input

```
+script("path/to/file.js")(async defer)
```

Output

```
<script src="path/to/file.js" type="text/javascript" charset="UTF-8" async defer>
```

4.7.3 `table.pug` – Tables generation

Table from JSON matrix

```
mixin table-json(src, header=true)
  //- requires context --> {require: require}
  - matrix_table = require(src);
  - count = 0;
  table&attributes(attributes)
    each row in matrix_table
      tr
        each column in row
          - if (count == 0)
            - if (header == true)
              th #{column}
            - else
              td #{column}
          - else
```

(continues on next page)

(continued from previous page)

```
            td #{column}
        - count += 1;
```

+table-json (*src*, *header=true*)

> **Note:** This mixin requires *require_context* inyection.

Create a table indicating a JSON file path which has a two level matrix structure (arrays inside arrays) like:

```
[
    [1, 2, 3, 4, 5],
    [1, 2, 3, 4, 5]
]
```

Arguments

- **src** (*string*) – Path of the `.json` file to load.
- **header** (*bool, optional*) – Indicates if include `th` tags for the first row. As default `true`.

Usage

Inputs

```
[
    [1, 2, 3, 4, 5],
    [6, 7, 8, 9, 10],
    [11, 12, 13, 14, 15]
]
```

```
+table-json("data.json")(style="width:50%;")
```

Output

```
<table style="width:50%;">
  <tr>
    <th>1</th>
    <th>2</th>
    <th>3</th>
    <th>4</th>
    <th>5</th>
  </tr>
  <tr>
    <td>6</td>
    <td>7</td>
    <td>8</td>
    <td>9</td>
    <td>10</td>
```

(continues on next page)

(continued from previous page)

```
  </tr>
  <tr>
    <td>11</td>
    <td>12</td>
    <td>13</td>
    <td>14</td>
    <td>15</td>
  </tr>
</table>
```

Render

4.7.4 `ul.pug` – Unordered lists generation

Unordered list from string

```
mixin ul-string(value, separator=",")
  ul&attributes(attributes)
    each elem in value.split(separator)
      li #{elem}
```

+ul-string(*value*, *separator=','*)

Insert an unordered list from a string indicating a separator to split the string.

Arguments

- **value** (*string*) – String from which the list will be built.
- **separator** (*string*) – The string will be splitted into differents elements using a separator passed as value of this parameter. As default `","`.

Usage

Input

```
+ul-string("one,two,three,four,five")
```

Output

```
<ul>
  <li>one</li>
  <li>two</li>
  <li>three</li>
  <li>four</li>
  <li>five</li>
</ul>
```

Render

4.8 `map/` — Customized maps

4.8.1 `google.pug` – Embed Google Maps

In order to use the next mixins, you need to follow next steps:

1. Go to Maps Javascript API Google's documentation and get an API key. You can start the process for obtain it clicking on `GET STARTED` button.
2. In your API's google dashboard you must enable next APIs:
   - Google Maps Embed API
   - Google Maps Javascript API
   - Google Maps Static API

Note: Mixins explained in this page are covering Google Maps Embed API Documentation and there is the relation between mixins and maps types:

- `google-map-view()`: `view` mode.
- `google-map-place()`: `place` mode.

Common parameters

`+google-map-*` (*key*)(*maptype="roadmap"*)

All google maps mixins share some common parameters like:

Arguments

- **`key`** (*`string`*) – Your API key. Don't forget to enable Google Maps APIs as explained above because if aren't enabled, the iframe will not be displayed.

- **maptype** (*string, optional*) – As default, `"roadmap"`. You can selected between 2 styles of map:
  - `"roadmap"`: View of predetermined road map.
  - `"satellite"`: View of satellital images of Google Earth.

Embed maps by places

```
mixin google-map-place(query, key)
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  - if (query.length == 27 && query.indexOf(["+"]) < 1)
    - query = "place_id:" + query
  - else
    - query = query.replace(/ /g, "+");
  iframe(src=`https://google.com/maps/embed/v1/place?key=${key}&q=${query}`
         frameborder="0"
         width="100%"
         height="450")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+google-map-place (*query*, *key*)(*maptype="roadmap"*)

Insert a Google map given a place string like `"Eiffel Tower,Paris France"` or by place identificator.

Arguments

- **query** (*string*) – Place string or place identificator. You can pass it with spaces and the mixins will be responsible of replace all with `"+"` characters.
- **key** (*string*) – See *Common parameters*.
- **maptype** (*string, optional*) – See *Common parameters*.

Usage

Input

```
+google-map-place("Eiffel Tower,Paris France", "AIzaSyAYBpeIr5j02T63_
↪xtKTs3l5vC8eLsNqqI")
```

Output

```
<iframe src="https://google.com/maps/embed/v1/place?key=AIzaSyAYBpeIr5j02T63_
↪xtKTs3l5vC8eLsNqqI&q=Eiffel+Tower,Paris+France" frameborder="0" width="100%" height=
↪"450">Your browser does not support iframes.</iframe>
```

Render

Embed maps by coordinates

```
mixin google-map-view(center, zoom, key)
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  - if ("maptype" in attributes)
    - maptype = attributes.maptype;
    - delete attributes.maptype;
  - else
    - maptype = "roadmap";
  - src = `https://google.com/maps/embed/v1/view?key=${key}&center=${center}&zoom=$
↪{zoom}&maptype=${maptype}`
  - if ("language" in attributes)
    - src = src + `&language=${attributes.language}`;
    - delete attributes.language;
  - if ("region" in attributes)
    - src = src + `&region=${attributes.region}`;
    - delete attributes.region;
  iframe(frameborder="0"
         width="100%"
         height="450"
         src=`${src}`)&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+google-map-view(*center*, *zoom*, *key)(maptype="roadmap"*, *language=null*, *region=null*, *width="100%"*, *height="450"*)

Insert a Google map given location coordinates, a zoom distance and API key parameters. You can configure also map visualization type displayed when `iframe` content is loaded.

Arguments

- **center** (*string*) – Coordinates separatted by a comma. These can be obtained navigating through maps at maps.google.com. You can see in the url the coordinates at the right of a `@` symbol. The parameter in the url that contains a `z` caracter is the zoom (see parameter).
- **zoom** (*float, string*) – The higher will be this value, the distance to the earth will be less and viceversa.
- **key** (*string*) – See *Common parameters*.
- **maptype** (*string, optional*) – See *Common parameters*.
- **language** (*string*) – Defines the language to use for UI elements and for the display of labels on map tiles. By default, visitors will see a map in their own language.
- **region** (*string*) – defines the appropriate borders and labels to display, based on geopolitical sensitivities. Accepts a region code specified as a two-character ccTLD (top-level domain) value.

Usage

Input

```
+google-map-view("36.0135723,-5.6080321", 9.5, "AIzaSyAYBpeIr5j02T63_
↪xtKTs3l5vC8eLsNqqI")(maptype="satellite")
```

Output

```
<iframe frameborder="0" width="100%" height="450" src="https://www.google.com/maps/
↪embed/v1/view?key=AIzaSyAYBpeIr5j02T63_xtKTs3l5vC8eLsNqqI&center=36.0135723,-5.
↪6080321&zoom=9.5&maptype=satellite"></iframe>
```

Render

4.9 `nodemap/` — Nodegraph maps

4.9.1 `bubbl.pug` – Mind maps from Bubbl

Embed graph maps

```
mixin bubbl-map(id)
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  iframe(src=`https://bubbl.us/${id}?utm_source=page-embed&utm_medium=link&s=9040282`
         allowfullscreen="true"
         frameborder="0"
         width="100%"
         height="400")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+bubbl-map (*id)(width="100%", height="400"*)

Insert a mental map based on graphs nodes from https://bubbl.us

Arguments

- **id** (*string*) – Map identificator.

Usage

Input

```
+bubbl-map("NDcxMTQ1Ny85MDQwMjgyL2U5YTM3MWQyZTJlZDhkODdiOWNlMjRmYzg0MWFlYzIz-X")
```

Output

```
<iframe src="https://bubbl.us/
↪NDcxMTQ1Ny85MDQwMjgyL2U5YTM3MWQyZTJlZDhkODdiOWNlMjRmYzg0MWFlYzIz-X?utm_source=page-
↪embed&utm_medium=link&s=9040282 allowfullscreen="true" frameborder="0" width="100%"␣
↪height="400"></iframe>
```

Render

4.10 `social/` — Connect with people

4.10.1 `facebook.pug` – Facebook tools

Like buttons

```
mixin facebook-like(href)
  - if ("layout" in attributes)
    - layout = attributes.layout;
    - delete attributes.layout;
  - else
    - layout = "standard";
  - if ("action" in attributes)
    - action = attributes.action;
    - delete attributes.action;
  - else
    - action = "like"
  - if ("size" in attributes)
    - size = attributes.size;
    - delete attributes.size;
  - else
    - size = "small";
  - if ("show_faces" in attributes)
    - show_faces = attributes.show_faces;
    delete attributes.show_faces;
  - else
    - show_faces = true;
  - if ("share" in attributes)
    - share = attributes.share;
    - delete attributes.share;
  - else
    - share = true;
  - if ("colorscheme" in attributes)
    - colorscheme = attributes.colorscheme;
    - delete attributes.colorscheme;
  - else
    - colorscheme = "light";
  - src = `https://www.facebook.com/plugins/like.php?href=${href}&layout=${layout}&
↪action=${action}&size=${size}&show_faces=${show_faces}&share=${share}&colorscheme=$
↪{colorscheme}`
```

(continues on next page)

(continued from previous page)

```
  - if ("app_id" in attributes)
    - app_id = attributes.app_id;
    - delete attributes.app_id;
    - src = src + `&appId=${app_id}`
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  iframe(src=`${src}`
         scrolling="no" frameborder="0"
         allowtransparency="true"
         allow="encrypted-media"
         width="450"
         height="80")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+facebook-like (*href)(layout="standard"*, *action="like"*, *size="small"*, *show_faces=true*, *share=true*, *app_id=null*, *width="450"*, *height="80"*)

Creates a "like" facebook button pointing to content referenced by `href` parameter.

See also:

Facebook Like button documentation.

Arguments

- **href** (*string*) – Endpoint to your web content that you want to be liked when "Like" button will be clicked.
- **layout** (*string, optional*) – Select a design between next availables:
  - `standard` (default): Minimum width `225px`, default width `450px` and height `35px` (without photos) or `80px` (with photos).
  - `button_count`: Minimum width `55px`, default width `55px` and height `65px`.
  - `button`: Minimum width `90px`, default width `90px` and height `20px`.
  - `box_count`: Minimum width `47px`, default width `47px` and height `20px`.
- **action** (*string, optional*) – Verb that will be shown inside button. You can select between `"like"` and `"recommend"`. As default `"like"`.
- **size** (*string, optional*) – Button size, you can select it between `"small"` and `"big"`. As default `"small"`.
- **show_faces** (*bool, optional*) – Specify if must be shown profile photographies (only with `"standard"` design). As default `true`.
- **share** (*bool, optional*) – Specify if the `iframe` element must be include a "Share" button. As default `true`.
- **colorscheme** (*string, optional*) – Colors scheme that facebook plugin will use fot the button external text. Select it between `"light"` and `"dark"`. As default `"light"`.
- **app_id** (*integer, optional*) – App identificator used by recopile data from post interactions. As default `null`.

Usage

Input

```
+facebook-like("https://github.com/mondeja/pug-mixins")(share=false, height="50")
```

Output

```
<iframe src="https://www.facebook.com/plugins/like.php?href=https://github.com/
↪mondeja/pug-mixins&layout=standard&action=like&size=small&show_faces=true&
↪share=false&colorscheme=light" scrolling="no" frameborder="0" allowtransparency=
↪"true" allow="encrypted-media" width="450" height="50"></iframe>
```

Render

Embed posts

```
mixin facebook-post(user_id, post_number)
  - if ("show_text" in attributes)
    - show_text = attributes.show_text;
    - delete attributes.show_text;
  - else
    - show_text = true;
  - src = `https://www.facebook.com/plugins/post.php?href=https%3A%2F%2Fwww.facebook.
↪com%2F${user_id}%2Fposts%2F${post_number}&show_text=${show_text}`
  - if ("app_id" in attributes)
    - app_id = attributes.app_id;
    - delete attributes.app_id;
    - src = src + `&appId=${app_id}`
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  iframe(src=`${src}`
         scrolling="no" frameborder="0"
         allowtransparency="true"
         allow="encrypted-media"
         width="500"
         height="289")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+facebook-post (*user_id*, *post_number)(app_id=null)(width="450"*, *height="289"*)

Insert a post given a post number and user identificators.

See also:

Facebook embedded posts documentation.

Arguments

- **user_id** (*string*) – Post user identificator. You can obtain it accessing a post page and copying the string between `facebook.com/` and `/posts/`.
- **post_number** (*integer*) – Post number identificator. You can obtain it accessing a post page and copying the number after `/posts/` in the url.
- **app_id** (*integer, optional*) – App identificator used by recopile data from post interactions. As default `null`.

Usage

Input

```
+facebook-post("alvaro.mondejarrubio", 1880014045362442)(width="80%" height="300")
```

Output

```
<iframe src="https://www.facebook.com/plugins/post.php?href=https%3A%2F%2Fwww.
↪facebook.com%2Falvaro.mondejarrubio%2Fposts%2F1880014045362442&show_text=false"␣
↪scrolling="no" frameborder="0" allowtransparency="true" allow="encrypted-media"␣
↪width="80%" height="300"></iframe>
```

Render

4.11 `video/` — Videos and playlists

4.11.1 `dailymotion.pug` – Dailymotion video iframes

Embed videos

```
mixin dailymotion-video(id, autoplay=false)
  - if (autoplay == true)
    - _autoplay = 1
  - else
    - _autoplay = 0
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  iframe(frameborder="0"
         width="480"
         height="270"
         src=`https://www.dailymotion.com/embed/video/${id}?autoPlay=${_autoplay}`
         allowfullscreen="true"
         allow="autoplay")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+dailymotion-video (*id*, *autoplay=false)(width="480" height="270"*)
Embed a Dailymotion video.

Arguments

- **id** (*string*) – Identificator of the video. You can obtain it from their url.
- **autoplay** – Indicates if the video is played after loading. As default `false`.

Usage

Input

```
+dailymotion-video("x2c3xfh")(width="100%", height="360")
```

Output

```
<iframe frameborder="0" width="100%" height="360" src="https://www.dailymotion.com/
↪embed/video/x2c3xfh?autoPlay=0" allowfullscreen="true" allow="autoplay"></iframe>
```

Render

4.11.2 `vimeo.pug` – Vimeo video iframes

Embed videos

```
mixin vimeo-video(id)
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  iframe(src=`https://player.vimeo.com/video/${id}`
         frameborder="0"
         width="640"
         height="360"
         allowfullscreen)&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+vimeo-video (*id)(width="640"*, *height="360"*)
Insert a Vimeo video `iframe` given a identificator.

Arguments

- **id** (*string / integer*) – Video identificator, correspond to a number located at the end of a video url. For example, if a video has the url https://vimeo.com/96425312, the identificator is `96425312`.

Usage

Input

```
+vimeo-video("152881306")(width="100%" height="315")
```

Output

```
<iframe width="100%" height="315" src="https://player.vimeo.com/video/152881306"␣
↪frameborder="0" allowfullscreen></iframe>
```

Render

4.11.3 `youtube.pug` – Youtube videos and playlists

See also:

Youtube embedded players documentation

Embed videos

```
mixin youtube-video(id)
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
  - if ("autoplay" in attributes)
    - autoplay = attributes.autoplay;
    - delete attributes.autoplay;
```

(continues on next page)

(continued from previous page)

```
  - else
    - autoplay = false;
  - if (autoplay == true)
    - autoplay = 1;
  - else
    - autoplay = 0;
  - src = `https://www.youtube.com/embed/${id}?autoplay=${autoplay}`
  iframe(src=`${src}`
         frameborder="0"
         allow="autoplay;encrypted-media"
         allowfullscreen="true"
         width="560"
         height="315")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+youtube-video (*id)(autoplay=false*, *width="560"*, *height="315"*)

Insert a Youtube video `iframe` given a identificator.

Arguments

- **id** (*string*) – Video identificator, corresponds to some letters and numbers located at the end of a video url. For example, if a video has the url https://www.youtube.com/watch?v=xK1bZBzowF0, the identificator is `xK1bZBzowF0`.
- **autoplay** (*bool, optional*) – If `true`, video will be inmediatly played after loaded.

Usage

Input

```
+youtube-video("xK1bZBzowF0")(width="100%")
```

Output

```
<iframe width="100%" height="315" src="https://www.youtube.com/embed/xK1bZBzowF0"␣
↪frameborder="0" allow="autoplay; encrypted-media" allowfullscreen></iframe>
```

Render

Embed playlists

```
mixin youtube-playlist(id)
  - if ("alt" in attributes)
    - alt = attributes.alt;
    - delete attributes.alt;
  - else
    - alt = true;
```

(continues on next page)

(continued from previous page)

```
  - if ("autoplay" in attributes)
    - autoplay = attributes.autoplay;
    - delete attributes.autoplay;
  - else
    - autoplay = false;
  - if (autoplay == true)
    - autoplay = 1;
  - else
    - autoplay = 0;
  iframe(src=`https://www.youtube.com/embed/videoseries?list=${id}&autoplay=$
↪{autoplay}`
          frameborder="0"
          allow="autoplay;encrypted-media"
          allowfullscreen="true"
          width="560"
          height="315")&attributes(attributes)
    - if (alt)
      | Your browser does not support iframes.
```

+youtube-playlist (*id)(width="560"*, *height="315"*)

Insert a Youtube playlist `iframe` given an identificator.

Arguments

- **id** (*string*) – Playlist identificator, corresponds to some letters and numbers located at the end of a playlist url. For example, if a playlist has the url https://www.youtube.com/playlist?list=PL9fkilWKH-6BwYucD1jZ8RZS0wHHCik7d, the identificator is `PL9fkilWKH-6BwYucD1jZ8RZS0wHHCik7d`.

Usage

Input

```
+youtube-playlist("PL9fkilWKH-6BwYucD1jZ8RZS0wHHCik7d")(width="100%")
```

Output

```
<iframe width="100%" height="315" src="https://www.youtube.com/embed/videoseries?
↪list=PL9fkilWKH-6BwYucD1jZ8RZS0wHHCik7d" frameborder="0" allow="autoplay; encrypted-
↪media" allowfullscreen></iframe>
```

Render

CHAPTER 5

Contributing

5.1 Basic guidelines

1. Fork the project from https://github.com/mondeja/pug-mixins
2. Download the source code from github with `git clone https://github.com/<your-user>/pug-mixins.git` or manually from `https://github.com/<your-user>/pug-mixins/archive/master.zip`.
3. Install python dependencies with `pip install -r requirements.txt`.
4. Fix a bug or create a new feature (and write tests if this is your case).
5. Run tests with `make tests` or `pytest test`. If something is broken, fix it.
6. Write documentation for this new features if necessary and build it. See *doc/ – Writing documentation*.
7. Commit your changes and pull `git pull origin master`.
8. Make a pull request from the github interface. See CI testing for your pull request. If something is broken, fix it and restart the pull request.

See also:

Development status

5.2 Project directories tree

- `src`: Source code of the project. All mixins are catalogued by folders according to their use, not by html tags.
- `doc`: Project documentation. See *doc/ – Writing documentation*.
- `test`: Project tests wrote with `pytest`. See *test/ – Testing*.
- `scripts`: Useful scripts. You can see here a python script called `doc_prebuild.py` that loads source code mixins and insert on documentation through jinja2 renderization.

5.3 doc/ – Writing documentation

Documentation for each new module must be located at `doc/source/_templates/`. This folder has the same directories structure of `src/`. If you see the existing templates you will observe that mixins source code are included by their name, replacing - caracters with _ and enclosed by double `{` and `}` caracters.

5.3.1 Build steps

The script located at `scripts/prebuild_doc.py` render all source code mixins in their respective files. Execute it normally or use `make doc-prebuild` (Linux/Mac users).

Generated output files will be located at `doc/source/user_guide/modules/`. This files need to be referenced from `doc/source/user_guide/reference.rst` toctrees.

So the build documentation process involves 2 steps, first prebuild with previously explained script and the build process with `sphinx`. You can execute this 2 steps secuentially running `make doc-build`.

5.3.2 Mixins structure

Please, respect documentation structure for each mixin. These must include, at least, the next information in order:

- Source code of the mixin.
- Mixin documentation (description, attributes...)
- **Usage example with:**
  - Input example.
  - HTML output of that example.
  - Renderization output.

CHAPTER 6

test/ – Testing

Note: To run tests you need to install python dependencies with `pip install -r requirements.txt` and pug client with `npm install pug`.

- **run tests**: `pytest tests`

6.1 Writing tests

Every new feature need to be tested. All tests are located in `test/`, and this have the same project structure than `src/`, but all names are preceded by `test_`.

At the beggining of every test, set in global variables: filepath of `.pug` source file to test and directory located in.

The basic process of each test involves next steps:

- Create a `.pug` page inside project source directory that will be tested. This page must contains a mixin execution string and an include pointing to the mixins source code that will be tested (in the same folder). See *pug_utils.create_pug_page()*.
- Renderize created pug page and test HTML output.
- If possible, test web endpoints (see `http_utils` below).

6.2 Fixtures

Pytest fixtures are inyected from `test/conftest.py` and located at `test/_fixtures/`.

6.2.1 pug_utils.py

pug_utils.**create_pug_page**(*src*, *content*, *includes=[]*)
Creates a pug page with the content provided inserting includes to files in the same folder.

Parameters

- **src** (*str*) – Filepath of new page.
- **content** (*str*) – Content of the file, without includes.
- **includes** (*list*) – Files to include with pug's include syntax (`include ./<...>.pug`) at the beggining of the file. As default `[]`.

Returns `src` parameter value.

Return type str

pug_utils.**extract_pug_mixin**(*src*, *mixin_name*)
Extract a mixin code from a `.pug` file

Parameters

- **src** (*str*) – Source filepath.
- **mixin_name** (*str*) – Mixin name to extract.

Returns The whole mixin, including definition and body.

Return type str

6.2.2 http_utils.py

class http_utils.**Response**(*text*, *status_code*, *url*)
Represents HTTP response objects.

Parameters

- **text** (*str*) – Data retrieved by the request.
- **status_code** (*int*) – Status code of the response.
- **url** (*str*) – Request url.

http_utils.**get**(*url*, *timeout=10*)
Performs a GET request giving an url as first parameter.

Parameters

- **url** (*str*) – Request url.
- **timeout** (*int, float, optional*) – Request expiration time in seconds.

http_utils.**assert_200**(*url*)
Assert if requesting with GET an url this returns 200 status code.

Parameters **url** (*str*) – Request url.

Returns Response object if is asserted `True`.

Return type *http_utils.Response*

http_utils.**assert_403**(*url*)
Assert if requesting with GET an url this returns 403 HTTP error (Forbidden request).

Parameters **url** (*str*) – Request url.

Returns Response object if is asserted `True`.

Return type *http_utils.Response*

6.3 Global variables

Global tests variables are located at `_fixtures/constests.py` module.

6.3.1 constests.py

PUG_CLI_PATH
Filepath of pug client.

***_DIR**
You can get every directory path inside `src/` folder importing from `conftest.py` the uppercase name of a src folder following by `_DIR`. For example, filepath of `src/code` is the value of `CODE_DIR`.

CHAPTER 7

Development status

7.1 audio/ — Podcasts and playlists

7.1.1 ivoox.pug – Ivoox podcasts players 

Tracking

Basic

Mixin	Implemented	Tested	Documented
+ivoox-audio()			
+ivoox-podcast()			
+ivoox-playlist()			
+ivoox-channel-subscription()			
+ivoox-podcast-subscription()			

Advanced

- ***+ivoox-audio()***
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested
- ***+ivoox-podcast()***
  - Default `width` and `height` parameters
  - `alt` parameter support

  - HTTP endpoint tested

- *+ivoox-playlist()*
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested
- *+ivoox-channel-subscription()*
  - Default `height` parameter
  - `alt` parameter support
  - HTTP endpoint tested
- **+ivoox-podcast-subscription()**
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested

TODO

- Understand and explain at documentation what really does `r` optional parameter of *+ivoox-channel-subscription()* mixin.
- Develop `+ivoox-podcast-subscription()` mixin.

7.1.2 `soundcloud.pug` – Soundcloud podcasts players

Tracking

Basic

Mixin	Implemented	Tested	Documented
+soundcloud-user()			

Advanced

- *+soundcloud-user()*
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested

7.2 `blog/` — Blogging tools

7.2.1 `storify.pug` - Storify stories

Tracking

Basic

Mixin	Implemented	Tested	Documented
+storify()			

Advanced

- ***+storify()***
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested

TODO

- Explain better what does `border` parameter of *+storify()* mixin.

7.3 `code/` — Programming tools

7.3.1 `codepen.pug` – Modern pens

Tracking

Basic

Mixin	Implemented	Tested	Documented
+codepen()			

Advanced

- ***+codepen()***
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested

TODO

- Add `test_codepen_invalid()` test.
- Add `test_codepen_default_tab` test.
- Add `test_codepen_invalid_default_tab` test.
- Add `test_codepen_embed_version` test.
- Add `test_codepen_invalid_embed_version` test.

7.3.2 `console.pug` – Insert consoles everywhere

Tracking

Basic

Mixin	Implemented	Tested	Documented
+brython-console()			
+js-console()			

Advanced

- ***+brython-console()***
  - Default `width` and `height` parameters
  - `alt` parameter support
  - Test HTTP endpoint
- ***+js-console()***
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested

TODO

- Understand and explain at documentation what really does `r` optional parameter of *+ivoox-channel-subscription()* mixin.
- Develop `+ivoox-podcast-subscription()` mixin.

7.3.3 `gist.pug` – Github gists mixins

Tracking

Basic

Mixin	Implemented	Tested	Documented
+gist()			

Advanced

- ***+gist()***
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested

7.3.4 `jsfiddle.pug` – JsFiddle web editor

Tracking

Basic

Mixin	Implemented	Tested	Documented
+jsfiddle()			

Advanced

- ***+jsfiddle()***
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested

TODO

- Test `tabs` parameter.
- Test `theme` parameter.
- Test `font_color` parameter.
- Test `accent_color` parameter.
- Test `code_background` parameter.
- Test `menu_background` parameter.

7.3.5 `pastebin.pug` – Clean Pastebin pastes

Tracking

Basic

Mixin	Implemented	Tested	Documented
+pastebin()			

Advanced

- ***+pastebin()***
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested

7.3.6 `pythontutor.pug` – Code executions step by step

Tracking

Basic

Mixin	Implemented	Tested	Documented
+tutor()			

Advanced

- ***+tutor()***
  - Default `width` and `height` optional attributes
  - `alt` parameter support
  - HTTP endpoint tested

TODO

- Implement paramters for Pythontutor `iframe` url.

7.3.7 `shield.pug` – Developer shields

Tracking

Basic

Mixin	Implemented	Tested	Documented
+shield-urls()			
+shield()			

Advanced

- ***+shield()***
  - **HTTP endpoints tested by services:**
    * "travis"
    * "pypi-version"
    * "pypi-py-versions"
    * "pypi-status"
    * "pypi-license"
    * "binder"
    * "readthedocs"
    * "ask-me-anything"
    * "github-issues"
    * "github-issues-closed"
    * "github-issues-closed-in"
    * "github-last-commit"
    * "github-contributors"
    * "isitmaintained-issues-open-perc"
    * "isitmaintained-issues-closed-in"

7.4 `crypto/` — Blockchain utils

7.4.1 `coinbase.pug` – Cryptocurrencies payment processor

Tracking

Basic

Mixin	Implemented	Tested	Documented
+coinbase-script()			
+coinbase()			
+coinbase-default()			

Advanced

- ***+coinbase-script()***
  - HTTP endpoint tested

TODO

- Test `include_script` parameter.

7.5 `functional/` — Functional programming

7.5.1 `loop.pug` – Recursive HTML generation

Tracking

Basic

Mixin	Implemented	Tested	Documented
+loop()			

7.6 `html/` — HTML tags utilities

7.6.1 `ol.pug` – Ordered lists generation

Tracking

Basic

Mixin	Implemented	Tested	Documented
+ol-string()			

Advanced

- *+ol-string()*
  - `separator` parameter tested

7.6.2 `table.pug` – Tables generation

Tracking

Basic

Mixin	Implemented	Tested	Documented
+table-json()			

Advanced

- **table-json()**
  - `header` optional parameter tested

7.6.3 `script.pug` – Powerful `script` macros

Tracking

Basic

Mixin	Implemented	Tested	Documented
+script()			

7.6.4 `ul.pug` – Unordered lists generation

Tracking

Basic

Mixin	Implemented	Tested	Documented
+ul-string()			

Advanced

- *+ul-string()*
  - `separator` parameter tested

7.7 map/ — Customized maps

7.7.1 google.pug – Embed Google Maps

Tracking

Basic

Mixin	Implemented	Tested	Documented
+google-map-view()			
+google-map-place()			

Advanced

- ***+google-map-view()*:**
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested
- ***+google-map-place()*:**
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested
  - Optional parameters for attributes saving: `attribution_source`, `attribution_web_url` and `attribution_ios_deep_link_id`.

TODO

- Add `google-map-place()` mixin covering place mode of Google Maps Embed API.
- Add `google-map-directions()` mixin covering directions mode of Google Maps Embed API.
- Add `google-map-search()` mixin covering search mode of Google Maps Embed API.
- Add `google-map-streetview()` mixin covering street view mode of Google Maps Embed API.

7.8 nodemap/ — Nodegraph maps

7.8.1 bubbl.pug – Mind maps from Bubbl

Tracking

Basic

Mixin	Implemented	Tested	Documented
+bubbl-map()			

Advanced

- ***+bubbl-map()***
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested

7.9 social/ — Connect with people

7.9.1 facebook.pug – Facebook tools

Tracking

Basic

Mixin	Implemented	Tested	Documented
+facebook-like()			
+facebook-post()			

Advanced

- ***+facebook-like()***
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested
- ***+facebook-post()***
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested

7.10 `video/` — Videos and playlists

7.10.1 `dailymotion.pug` – Dailymotion video iframes

Tracking

Basic

Mixin	Implemented	Tested	Documented
+dailymotion-video()			

Advanced

- ***+dailymotion-video()***
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested

7.10.2 `vimeo.pug` – Vimeo video iframes

Tracking

Basic

Mixin	Implemented	Tested	Documented
+vimeo-video()			

Advanced

- ***+vimeo-video()***
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested

7.10.3 `youtube.pug` – Youtube videos and playlists

Tracking

Basic

Mixin	Implemented	Tested	Documented
+youtube-video()			
+youtube-playlist()			

Advanced

- **youtube-*()**
  - **Optional parameters support:**
    - autoplay
    - cc_load_policy
    - color
    - controls
    - disablekb
    - enablejsapi
    - end
    - fs
    - hl
    - iv_load_policy
    - loop
    - modestbranding
    - origin
    - playlist
    - playsinline
    - rel
    - showinfo
    - start
    - widget_referrer
- ***+youtube-video()***
  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested
- ***+youtube-playlist()***

  - Default `width` and `height` parameters
  - `alt` parameter support
  - HTTP endpoint tested

TODO

- Add `youtube-user-uploads()` mixin covering `Loading a user's uploaded videos` section of Youtube embedded players documentation.
- Add `youtube-search()` mixin covering `Loading search results for a specified query` section of Youtube embedded players documentation.

CHAPTER 8

Changelog

8.1 0.0.2

- :js:func‘+google-map‘ mixin renamed as *+google-map-view()*.
- **New folders, files and mixins:**
  - **code/**
    - **pythontutor.pug**
      - *+tutor()*
  - **html/**
    - **ol.pug**
      - *+ol-string()*
    - **table.pug**
      - *+table-json()*
    - **ul.pug**
      - *+ul-string()*
  - **map/**
    - **google.pug**
      - *+google-map-place()*

8.2 0.0.1

- New folders, files and mixins:
  - **audio/**

        * **ivoox.pug**
            - *+ivoox-audio()*
            - *+ivoox-podcast()*
            - *+ivoox-playlist()*
            - *+ivoox-channel-subscription()*

- **blog/**
        * **storify.pug**
            - *+storify()*

- **board/**
        * livebinders.pug
        * **padlet.pug**
            - *+padlet()*

- **code/**
        * **codepen.pug**
            - *+codepen()*
        * **console.pug**
            - *+brython-console()*
            - *+js-console()*
        * **gist.pug**
            - *+gist()*
        * **jsfiddle.pug**
            - *+jsfiddle()*
        * **pastebin.pug**
            - *+pastebin()*
        * **shield.pug**
            - *+shield-urls()*
            - *+shield()*

- **crypto/**
        * **coinbase.pug**
            - *+coinbase-script()*
            - *+coinbase()*
            - *+coinbase-default()*

- **functional/**
        * **loop.pug**
            - *+loop()*

- **html/**

        * **script.pug**
            - *+script()*

- **map/**
        * **google.pug**
            - +google-map()

- **nodemap/**
        * **bubbl.pug**
            - *+bubbl-map()*

- **social/**
        * **facebook.pug**
            - *+facebook-like()*
            - *+facebook-post()*

- **video/**
        * **dailymotion.pug**
            - *+dailymotion-video()*
        * **vimeo.pug**
            - *+vimeo-video()*
        * **youtube.pug**
            - *+youtube-video()*
            - *+youtube-playlist()*

CHAPTER 9

Indices and tables

- genindex
- modindex
- search

Index

Symbols

A

C

E

G

P

R