Stability

The following stability guarantees are made, this package tries to adhere to semantic versioning.

`unstable`	API may change with any version bump.
`stable`	API will not change without a major version bump or a minor version bump before `1.0.0`, if such a change occures it is a bug and unintended.

Custom Types

The following custom types are used to pass around information easily:

`sanitized-selector` `stable`

Defines a selector for an ancestor or primary element.

```
(
  target: queryable,
  filter: ((context, candidates) => bool) | none,
)
```

`hydra-selector` `stable`

Defines a pair of primary and ancestor element selectors.

```
(
  primary: sanitized-selector,
  ancestors: sanitized-selector | none,
)
```

`candidates` `stable`

Defines the candidates that have been found in a specific context.

```
(
  primary: (prev: content | none, next: content | none, last: content | none),
  ancestor: (prev: content | none, next: content | none),
)
```

`context` `unstable`

Defines the options passed to hydra nad resolved contextual information needed for querying and displaying.

```
(
  prev-filter: (context, candidates) => bool,
  next-filter: (context, candidates) => bool,
  display: (context, content) => content,
  skip-starting: bool,
  use-last: bool,
  book: bool,
  anchor: label | none,
  anchor-loc: location,
  primary: sanitized-selector,
  ancestors: sanitized-selector,
)
```

`hydra` `stable`

The package entry point. All functions validate their inputs and panic using error messages directed at the end user.

- anchor()
- hydra()

```
anchor()
```

An anchor used to search from. When using `hydra` ouside of the page header, this should be placed inside the pge header to find the correct searching context. `hydra` always searches from the last anchor it finds, if and only if it detects that it is outside of the top-margin.

```
hydra(
    prev-filter,
    next-filter,
    display,
    skip-starting,
    use-last,
    dir: direction auto,
    binding: alignement auto,
    book: bool,
    anchor: label none,
    ..sel
  ) -> content
```

Parameters:

`dir` (`direction` or `auto` = `auto`) – The reading direction of the document. If this is `auto`, the text direction is used. Be cautious about leaving this option on `auto` if you switch text direction mid-page and use hydra outside of footers or headers.

`binding` (`alignement` or `auto` = `auto`) – The binding of the document. If this is `auto`, the binding is inferred from `dir`, similar to how it is done in page. Be cautious about leaving this on option on `auto` if you switch text direction mid-page and use hydra outside of footers or headers.

`book` (`bool` = `false`) – The binding direction if it should be considered, `none` if not. If the binding direction is set it'll be used to check for redundancy when an element is visible on the last page. Make sure to set `binding` and `dir` if the document is not using left-to-right reading direction.

`anchor` (`label` or `none` = `<hydra-anchor>`) – The label to use for the anchor if `hydra` is used outside the header. If this is `none`, the anchor is not searched.

`core` `unstable`

The core logic module. Some functions may return results with error messages that can be used to panic or recover from instead of panicking themselves.

- display()
- execute()
- get-candidates()
- get-page-binding()
- get-text-dir()
- get-top-margin()
- is-active-redundant()
- is-active-visible()
- is-on-starting-page()
- locate-last-anchor()

display(ctx: `context`, candidate: `content`) -> `content`

Display a heading's numbering and body.

Parameters:

`ctx` (`context`) – The context in which the element was found.

`candidate` (`content`) – The heading to display, panics if this is not a heading.

execute(ctx: `context`) -> `content`

Execute the core logic to find and display elements for the current context.

This function is contextual.

Parameters:

`ctx` (`context`) – The context for which to find and display the element.

get-candidates(ctx: `context`, scope-prev: `bool`, scope-next: `bool`) -> `candidates`

Get the element candidates for the given context.

This function is contextual.

Parameters:

`ctx` (`context`) – The context for which to get the candidates.

`scope-prev` (`bool` = `true`) – Whether the search should be scoped by the first ancestor element in this direction.

`scope-next` (`bool` = `true`) – Whether the search should be scoped by the first ancestor element in this direction.

get-page-binding() -> `alignment`

Returns the current page binding.

This function is contextual.

get-text-dir() -> `direction`

Returns the current text direction.

This function is contextual.

get-top-margin() -> `length`

Returns the current top margin.

This function is contextual.

is-active-redundant(ctx: `context`, candidates: `candidates`) -> `bool`

Check if showing the active element would be redudnant in the current context.

This function is contextual.

Parameters:

`ctx` (`context`) – The context in which the redundancy of the previous primary candidate should be checked.

`candidates` (`candidates`) – The candidates for this context.

is-active-visible(ctx: `context`, candidates: `candidates`) -> `bool`

Checks if the previous primary candidate is still visible.

This function is contextual.

Parameters:

`ctx` (`context`) – The context in which the visibility of the previous primary candidate should be checked.

`candidates` (`candidates`) – The candidates for this context.

is-on-starting-page(ctx: `context`, candidates: `candidates`) -> `bool`

Checks if the current context is on a starting page, i.e. if the next candidates are on top of this context's page.

This function is contextual.

Parameters:

`ctx` (`context`) – The context in which the visibility of the next candidates should be checked.

`candidates` (`candidates`) – The candidates for this context.

locate-last-anchor(ctx: `context`) -> `location`

Get the last anchor location. Panics if the last anchor was not on the page of this context.

This function is contextual.

Parameters:

`ctx` (`context`) – The context from which to start.

`selectors` stable

Contains functions used for creating custom selectors.

- by-level()
- custom()
- sanitize()

by-level(min: `int` `none`, max: `int` `none`, ..exact: `int` `none`) -> `hydra-selector`

Create a heading selector for a given range of levels.

Parameters:

`min` (`int` or `none` = `none`) – The inclusive minimum level to consider as the primary heading

`max` (`int` or `none` = `none`) – The inclusive maximum level to consider as the primary heading

`..exact` (`int` or `none`) – The exact level to consider as the primary element

custom(element: `function` `selector`, filter: `function`, ancestors: `function` `selector`, ancestors-filter: `function`) -> `hydra-selector`

Create a custom selector for `hydra`.

Parameters:

`element` (`function` or `selector`) – The primary element to search for.

`filter` (`function` = `none`) – The filter to apply to the element.

`ancestors` (`function` or `selector` = `none`) – The ancestor elements, this should match all of its ancestors.

`ancestors-filter` (`function` = `none`) – The filter applied to the ancestors.

sanitize(name: `str`, sel: `any`, message: `str` `auto`) -> `hydra-selector`

Turn a selector or function into a hydra selector.

This function is considered unstable.

Parameters:

`name` (`str`) – The name to use in the assertion message.

`sel` (`any`) – The selector to sanitize.

`message` (`str` or `auto` = `auto`) – The assertion message to use.

`util` `unstable`

Utlity functions and values.

`util/core` `unstable`

Utlity functions.

- or-default()
- page-binding()
- text-direction()

or-default(value: `any`, default: `function`, check: `any`) -> `any`

Substitute `value` for the return value of `default()` if it is a sentinel value.

Parameters:

`value` (`any`) – The value to check.

`default` (`function`) – The function to produce the default value with.

`check` (`any` = `none`) – The sentinel value to check for.

page-binding(dir) -> `alignement`

Returns the page binding for a text direction.

Source: `page.rs#L368-L373`

dir (direction): The direction to get the page binding for.

text-direction(lang) -> `direction`

Returns the text direction for a given language, defaults to `ltr` for unknown languages.

Source: `lang.rs#L50-L57`

lang (str): The languge to get the text direction for.

auto-or

An alias for `or-default` with `check: auto`.

none-or

An alias for `or-default` with `check: none`.

queryable-functions

A list of queryable element functions.

`util/assert` `unstable`

Assertions used for input and state validation.

- element()
- enum()
- queryable()
- types()

element(name: `str`, element: `any`, ..expected-funcs: `type`, message: `str` `auto`)

Assert that `element` is an element creatd by one of the given `expected-funcs`.

Parameters:

`name` (`str`) – The name use for the value in the assertion message.

`element` (`any`) – The value to check for.

`..expected-funcs` (`type`) – The expected element functions of `element`.

`message` (`str` or `auto` = `auto`) – The assertion message to use.

enum(name: `str`, value: `any`, ..expected-values: `type`, message: `str` `auto`)

Assert that `value` is any of the given `expected-values`.

Parameters:

`name` (`str`) – The name use for the value in the assertion message.

`value` (`any`) – The value to check for.

`..expected-values` (`type`) – The expected variants of `value`.

`message` (`str` or `auto` = `auto`) – The assertion message to use.

queryable(name: `str`, value: `any`, message: `str` `auto`)

Assert that `value` can be used in `query`.

Parameters:

`name` (`str`) – The name use for the value in the assertion message.

`value` (`any`) – The value to check for.

`message` (`str` or `auto` = `auto`) – The assertion message to use.

types(name: `str`, value: `any`, ..expected-types: `type`, message: `str` `auto`)

Assert that `value` is of any of the given `expected-types`.

Parameters:

`name` (`str`) – The name use for the value in the assertion message.

`value` (`any`) – The value to check for.

`..expected-types` (`type`) – The expected types of `value`.

`message` (`str` or `auto` = `auto`) – The assertion message to use.