# Windlass Documentation

## About

Welcome to the documentation for Windlass, a web framework built in JavaScript for Node.js based on the HTML5 specification.

## Getting Started

## Frequently Asked Questions

### Unique Identifier (UID)

The UID tag at the start of each section within this documentation specifies a unique tag for each section, which corresponds to an identical tag within the source code. This makes searching for the definition of a particular interface within the code base extremely quick and easy as the a UID only appears once within the code base.

Contents

[Windlass Documentation 1](#_Toc46939362)

[About 1](#_Toc46939363)

[Getting Started 1](#_Toc46939364)

[Frequently Asked Questions 1](#_Toc46939365)

[Unique Identifier (UID) 1](#_Toc46939366)

[Defaults 3](#_Toc46939367)

[ALIGN\_VALUES 3](#_Toc46939368)

[DIRECTION\_VALUES 3](#_Toc46939369)

[DISPLAY\_VALUES 3](#_Toc46939370)

[TRANSFORM\_VALUES 3](#_Toc46939371)

[DEFAULT\_PROPERTIES 4](#_Toc46939372)

[Color 6](#_Toc46939373)

[Neutrals 6](#_Toc46939374)

[Colors 7](#_Toc46939375)

[Typography 14](#_Toc46939376)

[Text Variants 14](#_Toc46939377)

[Text Properties 15](#_Toc46939378)

[Text 16](#_Toc46939379)

[Heading Variants 16](#_Toc46939380)

[Heading Properties 17](#_Toc46939381)

[Heading 17](#_Toc46939382)

## Defaults

### ALIGN\_VALUES

UID: [[wl\_IFUSr]](https://github.com/Niten001/windlass/blob/master/components/default/default.js#L11) [[?]](#_Unique_Identifier_(UID))

[ALIGN\_VALUES](https://github.com/Niten001/windlass/wiki/ALIGN_VALUES) provides an enumerable interface for the possible values alignment can take within component properties.

#### Values

|  |
| --- |
| JavaScript |
| const ALIGN\_VALUES = {  INHERIT: "inherit",  LEFT: "left",  CENTER: "center",  RIGHT: "right",  JUSTIFY: "justify",  }; |

### DIRECTION\_VALUES

UID: [[wl\_i9NPq]](https://github.com/Niten001/windlass/blob/master/components/default/default.js#L21) [[?]](#_Unique_Identifier_(UID))

[DIRECTION\_VALUES](#_DIRECTION_VALUES) provides an enumerable interface for the possible values direction can take within component properties.

#### Values

|  |
| --- |
| JavaScript |
| const DIRECTION\_VALUES = {  AUTO: "auto",  LEFT\_TO\_RIGHT: "ltr",  RIGHT\_TO\_LEFT: "rtl",  }; |

### DISPLAY\_VALUES

UID: [[wl\_GKg6F]](https://github.com/Niten001/windlass/blob/master/components/default/default.js#L29) [[?]](#_Unique_Identifier_(UID))

[DISPLAY\_VALUES](#_DISPLAY_VALUES) provides an enumerable interface for the possible values display can take within component properties.

#### Values

|  |
| --- |
| JavaScript |
| const DISPLAY\_VALUES = {  INITIAL: "initial",  BLOCK: "block",  INLINE: "inline",  }; |

### TRANSFORM\_VALUES

UID: [[wl\_bWc8v]](https://github.com/Niten001/windlass/blob/master/components/default/default.js#L37) [[?]](#_Unique_Identifier_(UID))

[TRANSFORM\_VALUES](#_TRANSFORM_VALUES) provides an enumerable interface for the possible values transform can take within component properties.

#### Values

|  |
| --- |
| JavaScript |
| const TRANSFORM\_VALUES = {  NONE: "none",  INITIAL: "initial",  INHERIT: "inherit",  CAPITALIZE: "capitalize",  UPPERCASE: "uppercase",  LOWERCASE: "lowercase",  }; |

### DEFAULT\_PROPERTIES

UID: [[wl\_EfQ6W]](https://github.com/Niten001/windlass/blob/master/components/default/default.js#L48) [[?]](#_Unique_Identifier_(UID))

[DEFAULT\_PROPERTIES](#_DEFAULT_PROPERTIES_1) provides an interface for the global properties used on all structures/components.

These properties can be accessed on the this.props variable for any structure/component whose properties are either the [DEFAULT\_PROPERTIES](#_DEFAULT_PROPERTIES_1) or an interface that extends the [DEFAULT\_PROPERTIES](#_DEFAULT_PROPERTIES_1).

#### Syntax

|  |
| --- |
| JavaScript |
| class DEFAULT\_PROPERTIES = {  String class (optional) (Default: "");  String content (optional) (Default: "");  DIRECTION\_VALUES direction (optional) (Default: "");  String id (optional) (Default: "");  String language (optional) (Default: "");  String style (optional) (Default: undefined);  String title (optional) (Default: "");  String[] styleList (optional) (Default: []);  }; |

#### Members

##### class

Type: **String**

##### Sets the *class* of the highest order element within the structure/component. (optional) (Default: "")

##### content

Type: **String**

Sets the *content* of the structure/component. (optional) (Default: "")

**[! Warning]**: The value of content is not sanitised by default, be careful what you allow to write to this property. You can sanitise the value of content by calling the sanitise() function around any values you write to this property.

##### direction

Type: [**DIRECTION\_VALUES**](#_DIRECTION_VALUES)

Sets the *dir* of the highest order element within the structure/component. (optional) (Default: "")

##### id

Type: **String**

Sets the *id* of the highest order element within the structure/component. (optional) (Default: "")

##### language

Type: **String**

Sets the *lang* of the highest order element within the structure/component. (optional) (Default: "")

##### style

Type: **String**

Sets the *style* of the structure/component. (optional) (Default: undefined)

##### title

Type: **String**

Sets the *title* of the highest order element within the structure/component. (optional) (Default: "")

##### styleList

Type: **String**

Sets the list of styles applied to the structure/component by the combineStyles() function, before being overridden by the style property. (optional) (Default: "")

## Color

### Neutrals

Provides a series of enumerable interfaces for neutral colors.

#### White

###### Values

|  |
| --- |
| JavaScript |
| const WHTIE = “#FFFFFF”; |

#### Black

###### Values

|  |
| --- |
| JavaScript |
| const BLACK = “#000000”; |

#### Grey



###### Values

|  |
| --- |
| JavaScript |
| const GREY = {  DEFAULT: “#747374”,  50: “#FAFAFA”,  100: “#DFDFDF”,  200: “#C4C4C4”,  300: “#AAA9AA”,  400: “#8F8E8F”,  500: “#747374”,  600: “#636263”,  700: “#525252”,  800: “#424142”,  900: “#202020”,  } |

#### Cool Grey



###### Values

|  |
| --- |
| JavaScript |
| const COOL\_GREY = {  DEFAULT: “#6D7188”,  50: “#F8FBFF”,  100: “#D8DDF3”,  200: “#BDC2D8”,  300: “#A3A7BE”,  400: “#888CA3”,  500: “#6D7188”,  600: “#5C6077”,  700: “#4B5066”,  800: “#3B3F56”,  900: “#191E34”,  } |

#### Warm Grey



###### Values

|  |
| --- |
| JavaScript |
| const WARM\_GREY = {  DEFAULT: “#7B7160”,  50: “#FFFAEC”,  100: “#E6DDCB”,  200: “#CBC2B0”,  300: “#B1A796”,  400: “#968C7B”,  500: “#7B7160”,  600: “#6A604F”,  700: “#59503E”,  800: “#493F2E”,  900: “#271E0C”,  } |

#### Neutrals

|  |
| --- |
| JavaScript |
| let NEUTRALS = GREY; |

### Colors

Provides a series of enumerable interfaces for colors.

#### Brown



###### Values

|  |
| --- |
| JavaScript |
| const BROWN = {  DEFAULT: “#583931”,  50: “#F4DFD7”,  100: “#DBC2BA”,  200: “#C4A69D”,  300: “#AD887E”,  400: “#8C665C”,  500: “#775147”,  600: “#6D483F”,  700: “#63423A”,  800: “#4C322D”,  900: “#331F1B”,  } |

#### Red



###### Values

|  |
| --- |
| JavaScript |
| const RED = {  DEFAULT: “#E80602”,  50: “#FCC5C2”,  100: “#F79D9B”,  200: “#F2716F”,  300: “#ED5553”,  400: “#D82D30”,  500: “#CE1616”,  600: “#BC0909”,  700: “#960404”,  800: “#6D0202”,  900: “#490001”,  } |

#### Crimson



###### Values

|  |
| --- |
| JavaScript |
| const CRIMSON = {  DEFAULT: “#D73800”,  50: “#FFD0BF”,  100: “#FCB69C”,  200: “#FE9873”,  300: “#FE7340”,  400: “#F24F21”,  500: “#DB3201”,  600: “#BF2701”,  700: “#A02101”,  800: “#7A1901”,  900: “#4F0B00”,  } |

#### Orange



###### Values

|  |
| --- |
| JavaScript |
| const ORANGE = {  DEFAULT: “#F97611”,  50: “#FCDFC2”,  100: “#FCD0A6”,  200: “#F7AD71”,  300: “#F28B41”,  400: “#ED7525”,  500: “#E26009”,  600: “#C64D01”,  700: “#A83E01”,  800: “#7C2B00”,  900: “#5E2100”,  } |

#### Amber



###### Values

|  |
| --- |
| JavaScript |
| const AMBER = {  DEFAULT: “#FCB30A”,  50: “#FCF0C4”,  100: “#FCE4A4”,  200: “#F9D172”,  300: “#F9C232”,  400: “#F9B516”,  500: “#EF9E07”,  600: “#D38208”,  700: “#A86201”,  800: “#874D01”,  900: “#633600”,  } |

#### Yellow



###### Values

|  |
| --- |
| JavaScript |
| const YELLOW = {  DEFAULT: “#F9DD22”,  50: “#FCF8D1”,  100: “#FDF4B7”,  200: “#FCEC85”,  300: “#FAE554”,  400: “#F4D229”,  500: “#EAB802”,  600: “#D19D02”,  700: “#B98201”,  800: “#A06701”,  900: “#874C00”,  } |

#### Lime



###### Values

|  |
| --- |
| JavaScript |
| const LIME = {  DEFAULT: “#AEF35A”,  50: “#F4FFD8”,  100: “#E9FCC2”,  200: “#DBF9A9”,  300: “#CBF98E”,  400: “#BAF76F”,  500: “#A6E94D”,  600: “#94DB36”,  700: “#7ECC18”,  800: “#62AA0A”,  900: “#437C01”,  } |

#### Light Green



###### Values

|  |
| --- |
| JavaScript |
| const LIGHT\_GREEN = {  DEFAULT: "#30E85B",  50: "#CEFFD8",  100: "#B4F7C2",  200: "#99EEAD",  300: "#7FE697",  400: "#64DD81",  500: "#4BCA6B",  600: "#33B855",  700: "#1AA53F",  800: "#0B842C",  900: "#00681D",  } |

#### Green



###### Values

|  |
| --- |
| JavaScript |
| const GREEN = {  DEFAULT: "#01862E",  50: "#C0E5C3",  100: "#9DD2A5",  200: "#7ABF87",  300: "#57AD69",  400: "#349A4B",  500: "#11842A",  600: "#0D7123",  700: "#095C19",  800: "#04460E",  900: "#003010",  } |

#### Teal



###### Values

|  |
| --- |
| JavaScript |
| const TEAL = {  DEFAULT: "#20C997",  50: "#C0F5E5",  100: "#A2F2DA",  200: "#7EEACA",  300: "#52E3B8",  400: "#30DBAA",  500: "#25C696",  600: "#1CA77D",  700: "#128260",  800: "#0A684C",  900: "#014933",  } |

#### Cyan



###### Values

|  |
| --- |
| JavaScript |
| const CYAN = {  DEFAULT: "#06E1E5",  50: "#CEFEFF",  100: "#B1F8F9",  200: "#97F3F4",  300: "#7CEBEF",  400: "#60E3E5",  500: "#46D8DB",  600: "#38BEC7",  700: "#26A0A9",  800: "#127F87",  900: "#005B66",  } |

#### Light Blue



###### Values

|  |
| --- |
| JavaScript |
| const LIGHT\_BLUE = {  DEFAULT: "#56CCF2",  50: "#CEF6FF",  100: "#B1EBF9",  200: "#97E0F4",  300: "#7CD4EF",  400: "#64CBEA",  500: "#4BBFE5",  600: "#34A7D1",  700: "#2391BA",  800: "#0E78A5",  900: "#016391",  } |

#### Blue



###### Values

|  |
| --- |
| JavaScript |
| const BLUE = {  DEFAULT: "#1A70E1",  50: "#C0DCF7",  100: "#A2C6EF",  200: "#85B1E8",  300: "#679BE0",  400: "#4D87DF",  500: "#2970CE",  600: "#1F5EB6",  700: "#164D9E",  800: "#0C3B85",  900: "#02296D",  } |

#### Indigo



###### Values

|  |
| --- |
| JavaScript |
| const INDIGO = {  DEFAULT: "#1D3BB7",  50: "#C7D6FC",  100: "#ADC0F4",  200: "#97ADED",  300: "#6E8AE5",  400: "#415ED3",  500: "#2848BA",  600: "#1D3CAD",  700: "#152F99",  800: "#091F7C",  900: "#010E47",  } |

#### Purple



###### Values

|  |
| --- |
| JavaScript |
| const PURPLE = {  DEFAULT: "#6512E1",  50: "#DDC2FC",  100: "#C8A7F2",  200: "#B58CEA",  300: "#9F6DE7",  400: "#7A3ED6",  500: "#6621CE",  600: "#5616BC",  700: "#3F1396",  800: "#2F097A",  900: "#19004F",  } |

#### Pink



###### Values

|  |
| --- |
| JavaScript |
| const PINK = {  DEFAULT: "#DA0299",  50: "#FCBDED",  100: "#F29FDD",  200: "#EA85CF",  300: "#E261C0",  400: "#E048BE",  500: "#CE21A0",  600: "#BA0982",  700: "#A30D73",  800: "#87055E",  900: "#630040",  } |

#### Colors

|  |
| --- |
| JavaScript |
| const COLORS = {  WHITE: WHITE,  BLACK: BLACK,  BROWN: BROWN,  RED: RED,  CRIMSON: CRIMSON,  ORANGE: ORANGE,  AMBER: AMBER,  YELLOW: YELLOW,  LIME: LIME,  LIGHT\_GREEN: LIGHT\_GREEN,  GREEN: GREEN,  TEAL: TEAL,  CYAN: CYAN,  LIGHT\_BLUE: LIGHT\_BLUE,  BLUE: BLUE,  INDIGO: INDIGO,  PURPLE: PURPLE,  PINK: PINK,  } |

#### Theme Colors

|  |
| --- |
| JavaScript |
| const THEME\_COLORS = {  PRIMARY: BLUE.DEFAULT,  SECONDARY: LIGHT\_BLUE.DEFAULT,  SUCCESS: GREEN[400],  INFO: PURPLE[400],  WARNING: AMBER[400],  DANGER: RED[400],  LIGHT: NEUTRALS[50],  DARK: NEUTRALS[900],  } |

## Typography

### Text Variants

Provides an enumerable interface for selecting which type of text structure to generate.

#### Values

|  |
| --- |
| JavaScript |
| const TEXT\_VARIANTS = {  DEFAULT: “span”,  BOLD: “bold”,  DELETE: “del”,  EMPHASIS: “em”,  HIGHLIGHT: “mark”,  INSERT: “ins”,  ITALIC: “i”,  SMALL: “small”,  STRIKETHROUGH: “s”,  STRONG: “strong”,  SUBSCRIPT: “sub”,  SUPERSCRIPT: “sup”,  UNDERLINE: “u”,  } |

### Text Properties

Extends [DEFAULT\_PROPERTIES](#_Default_Properties) to provide an interface for generating the text structures within a website.

#### Syntax

|  |
| --- |
| JavaScript |
| const TEXT\_PROPERTIES extends DEFAULT\_PROPERTIES {  ALIGN\_VALUES align (optional) = INHERIT;  COLORS color (optional) = INHERIT;  DISPLAY\_VALUES display (optional) = INITIAL;  String || Intl language (optional) = ‘en’;  Boolean paragraph (optional) = false;  TRANSFORM\_VALUES transform (optional) = INITIAL;  TEXT\_VARIANTS variant (optional) = DEFAULT;  Boolean wrap (optional) = true;  } |

#### Members

##### align

Type: ALIGN\_VALUES

Set the text-align on the component. (Optional) (Default: INHERIT)

##### color

Type: **COLORS**

The color of the component. It supports those theme colors that make sense for this component. (Optional) (Default: INITIAL)

##### display

Type: DISPLAY\_VALUES

Controls the display type. (Optional) (Default: INITIAL)

##### language

Type: String OR Intl

Sets the language of the text structure/component being generated. (Optional) (Default: “en”)

##### paragraph

Type: Boolean

If true, the text will be wrapped in a paragraph element.

##### transform

Type: TRANSFORM\_VALUES

Sets the text transform property of the text element being generated. (Optional) (Default:

##### variant

Type: TEXT\_VARIANTS

Sets what type of text structure is being generated. Can be any value from the [TEXT\_VARIANTS](#_Text_Variants) enumeration. (Optional) (Default: DARK)

##### wrap

Type: Boolean

If true, the text will not wrap, but instead will truncate with a text overflow ellipsis.

### Text

Contains the building blocks for text structures within a website

#### Syntax

|  |
| --- |
| JavaScript |
| String Text(  TEXT\_PROPERTIES props (Optional)  ); |

#### Members

##### props

Type: TEXT\_PROPERTIES

Sets the properties used to generate the text element. Must be an object of type [TEXT\_PROPERTIES](#_Text_Properties)

### Heading Variants

Provides an enumerable interface for selecting which type of heading structure to generate.

#### Values

|  |
| --- |
| JavaScript |
| const HEADING\_VARIANTS = {  DEFAULT: “”  HEADING\_1: “h1”,  HEADING\_2: “h2”,  HEADING\_3: “h3”,  HEADING\_4: “h4”,  HEADING\_5: “h5”,  HEADING\_6: “h6”,  TITLE: “title”,  SUBTITLE: “subtitle”,  } |

### Heading Properties

Extends [DEFAULT\_PROPERTIES](#_Default_Properties) to provide an interface for generating the heading structures within a website.

#### Syntax

|  |
| --- |
| JavaScript |
| const HEADING\_PROPERTIES extends DEFAULT\_PROPERTIES {  ALIGN\_VALUES align (optional) = INHERIT;  COLORS color (optional) = INITIAL;  DISPLAY\_VALUES display (optional) = INITIAL;  String || Intl language (optional) = ‘en’;  Boolean paragraph (optional) = false;  TRANSFORM\_VALUES transform (optional) = INITIAL;  HEADING\_VARIANTS variant (optional) = DEFAULT;  Boolean wrap (optional) = true;  } |

#### Members

##### align

Type: ALIGN\_VALUES

Set the text-align on the component. (Optional) (Default: INHERIT)

### Heading

Contains the building blocks for heading structures within a website

#### Syntax

|  |
| --- |
| JavaScript |
| String Heading(  HEADING\_PROPERTIES props (Optional)  ); |

#### Members

##### props

Type: HEADING\_PROPERTIES

Sets the properties used to generate the heading element. Must be an object of type HEADING\_PROPERTIES