Package: imola (via r-universe)

September 18, 2024

Type Package

Title CSS Layouts (Grid and Flexbox) Implementation for R/Shiny

Version 0.5.1

Description Allows easy creation of CSS layouts (grid and flexbox) directly from R without added CSS.

License MIT + file LICENSE

URL https://github.com/pedrocoutinhosilva/imola

Encoding UTF-8

LazyData true

VignetteBuilder knitr

Imports shiny, htmltools, magrittr, stringi, glue, yaml

Suggests testthat (>= 3.0.0), rvest, devtools, covr, rmarkdown, knitr

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.2

Config/testthat/edition 3

Language en-US

Repository https://pedrocoutinhosilva.r-universe.dev

RemoteUrl https://github.com/pedrocoutinhosilva/imola

RemoteRef HEAD

RemoteSha 5068fe7735c03784196f4ce28a6cc35584463f5f

Contents

`addBreakpoint` *Add a breakpoint to a breakpoint system*

Description

Adds a breakpoint to a breakpoint system object.

Usage

```
addBreakpoint(system, breakpoint)
```

Arguments

`system`	A breakpoint system object created with breakpointSystem.
`breakpoint`	A breakpoint created with breakpoint.

Value

A breakpoint system object.

`breakpoint` *Create a breakpoint*

Description

Creates a valid breakpoint object to use in a breakpoint system. While both the min and max arguments are optional, at least one of them must exist for the breakpoint to be considered valid.

Usage

```
breakpoint(name, min = NULL, max = NULL)
```

Arguments

`name` A string with the name that identifies the breakpoint.

`min` Optional numeric minimum value (in pixels) of the screen width where the breakpoint is active.

`max` Optional numeric maximum value (in pixels) of the screen width where the breakpoint is active.

Value

A breakpoint object.

`breakpointSystem` *Create a breakpoint system*

Description

Creates a breakpoint system object containing all the information about the system, including its name and set of available breakpoints.

Usage

```
breakpointSystem(name, ..., description = NULL)
```

Arguments

`name` A string with the name that identifies the breakpoint system.

`...` One or more breakpoint objects created with breakpoint.

`description` Optional description with information. about the breakpoint system. Can be used to pass on any information regarding the system (For example its origin or connected frameworks).

Value

A breakpoint system object.

`exportBreakpointSystem` *Export a breakpoint system*

Description

Exports a breakpoint system into a file for storage and later usage. Exported systems can be retrieved from their file form by using importBreakpointSystem.

Usage

```
exportBreakpointSystem(system, path)
```

Arguments

`system`	A string with the name of a registered breakpoint system, or a breakpoint system object generated with breakpointSystem.
`path`	The file path where to export the system to, including the file name and extension. The file name must end with a `.yaml` extension.

Value

No return value, called for side effects.

`exportTemplate` *Export a template*

Description

Exports a template into a file for storage and later usage. Exported template can be retrieved from their file form by using importTemplate.

Usage

```
exportTemplate(template, path)
```

Arguments

`template`	A template object generated with gridTemplate.
`path`	The file path where to export the system to, including the file name and extension. The file name must end with a `.yaml` extension.

Value

No return value, called for side effects.

`flexPage` *Create a css flexbox based page*

Description

Create a Shiny UI page using a flexPanel to wrap the page content. As other Shiny UI pages, it scaffolds the entire page and loads any required or registered dependencies.

Usage

```
flexPage(..., title = NULL, fill_page = TRUE, dependency = bootstrapLib())
```

Arguments

`...`	Arguments to be passed to flexPanel.
`title`	The browser window title (defaults to the host URL of the page).
`fill_page`	Boolean value if the page should automatically stretch to match the browser window height.
`dependency`	A set of web dependencies. This value can be a htmlDependency, for example the shiny bootstrap dependency (default value) or a tagList with different dependencies.

Value

A UI definition that can be passed to the shinyUI function.

Note

See `https://css-tricks.com/snippets/css/a-guide-to-flexbox/` for additional details on using css flexbox.

See Also

flexPanel

`flexPanel` *Create a css flexbox based HTML tag*

Description

Creates a HTML tag and automatically generates css style rules based on css flexbox, depending on the given arguments. Functionality acts as a way to generate css flexbox based HTML containers directly from R without the need to write any additional css rules directly.

Usage

```
flexPanel(
  ...,
  template = NULL,
  direction = "row",
  wrap = "nowrap",
  justify_content = "flex-start",
  align_items = "stretch",
  align_content = "flex-start",
  gap = 0,
  flex = c(1),
  grow = NULL,
  shrink = NULL,
  basis = NULL,
  breakpoint_system = getBreakpointSystem(),
  id = generateID()
)
```

Arguments

`...` Tag attributes (named arguments) and child elements (unnamed arguments).

Named arguments are treated as additional html attribute values to the parent tag.

Child elements may include any combination of other tags objects, HTML strings, and htmlDependencys.

`template` The name of a registered template to use as a base for the grid, or a template object from gridTemplate.

`direction` Direction of the flow of elements in the panel.

Accepts a valid css `flex-direction` value (`row` | `row-reverse` | `column` | `column-reverse`).

Default value of `row` value is used. Supports breakpoints.

`wrap` Should elements be allowed to wrap into multiple lines.

Accepts a valid css `flex-wrap` value (`nowrap` | `wrap` | `wrap-reverse`).

Supports breakpoints.

`justify_content`

The alignment along the main axis. Accepts a valid css `justify-content` value (`flex-start` | `flex-end` | `center` | `space-between` | `space-around` | `space-evenly` | `start` | `end` | `left` | `right`).

Supports breakpoints.

`align_items` Defines the default behavior for how flex items are laid out along the cross axis on the current line.

Accepts a valid css `align-items` value (`stretch` | `flex-start` | `flex-end` | `center` | `baseline` | `first baseline` | `last baseline` | `start` | `end` | `self-start` | `self-end`).

Supports breakpoints.

`align_content` Aligns a flex container's lines within when there is extra space in the cross-axis.

Accepts a valid css `align-content` value (flex-start | flex-end | center | space-between | space-around | space-evenly | stretch | start | end | baseline | first baseline | last baseline).

Supports breakpoints.

`gap` The space between elements in the panel. Controls both the space between rows and columns.

Accepts a css valid value, or 2 values separated by a space (if using different values for row and column spacing).

Supports breakpoints.

`flex` A vector of valid css 'flex' values. Defines how child elements in the panel can grow, shrink and their initial size.

Arguments that target child elements require a vector of values instead of a single value, with each entry of the vector affecting the nth child element.

If the vector has less entries that the total number of child elements, the values will be repeated until the pattern affects all elements in the panel. If the vector as more entries that the number of child elements, exceeding entries will be ignored. NA can also be used as a entry to skip adding a rule to a specific child element.

Accepts a valid css `flex` value vector of values.

By default c(1) is used, meaning all elements can grow and shrink as required, at the same rate. Supports breakpoints.

`grow` A vector of valid css 'flex-grow' values. Defines the rate of how elements can grow.

Entries will overwrite the 'flex' values, and can be used make more targeted rules.

Arguments that target child elements require a vector of values instead of a single value, with each entry of the vector affecting the nth child element.

If the vector has less entries that the total number of child elements, the values will be repeated until the pattern affects all elements in the panel. If the vector as more entries that the number of child elements, exceeding entries will be ignored. NA can also be used as a entry to skip adding a rule to a specific child element.

By default NULL is used, meaning values from the flex argument will be used instead. Supports breakpoints.

`shrink` A vector of valid css 'flex-shrink' values. Defines the rate of how elements can shrink. Entries will overwrite the nth 'flex' value, and can be used make more targeted rules.

Arguments that target child elements require a vector of values instead of a single value, with each entry of the vector affecting the nth child element.

If the vector has less entries that the total number of child elements, the values will be repeated until the pattern affects all elements in the panel. If the vector as more entries that the number of child elements, exceeding entries will be ignored. NA can also be used as a entry to skip adding a rule to a specific child element.

By default NULL is used, meaning values from the flex argument will be used instead. Supports breakpoints.

`basis` A vector of valid css ’flex-basis’ values. Defines the base size of elements. Entries will overwrite the nth ’flex’ value, and can be used make more targeted rules.

Arguments that target child elements require a vector of values instead of a single value, with each entry of the vector affecting the nth child element.

If the vector has less entries that the total number of child elements, the values will be repeated until the pattern affects all elements in the panel. If the vector as more entries that the number of child elements, exceeding entries will be ignored. NA can also be used as a entry to skip adding a rule to a specific child element.

By default NULL is used, meaning values from the flex argument will be used instead. Supports breakpoints.

`breakpoint_system`

Breakpoint system to use.

`id` The parent element id.

Details

Behaves similar to a normal HTML tag, but provides helping arguments that simplify the way flexbox css can be created from shiny.

Value

An HTML tagList.

Note

When creating responsive layouts based on css media rules, some arguments allow a named list can be passed instead of a single value.

The names in that list can be any of the breakpoints available in the `breakpoint_system` argument.

It is recommended to define the breakpoint system for the application globally before UI definitions, but the `breakpoint_system` in panel functions allows for more flexibility when reusing components from other projects.

See `https://css-tricks.com/snippets/css/a-guide-to-flexbox/` for additional details on using css flexbox.

For a full list of valid HTML attributes check visit `https://www.w3schools.com/tags/ref_attributes.asp`.

See Also

flexPage

`getBreakpointSystem` *Get a registered breakpoint system*

Description

Returns a breakpoint system object of a registered breakpoint system by its name or, the currently active breakpoint system if no system name is provided.

Usage

```
getBreakpointSystem(name = NULL)
```

Arguments

`name`	A string with the name of a registered breakpoint system, or `NULL` if looking for the currently active breakpoint system.

Value

A breakpoint system object.

`getTemplate` *Get a registered template*

Description

Returns a object form of a registered template by its name and type.

Usage

```
getTemplate(name, type)
```

Arguments

`name`	The name of a registered template.
`type`	The type of css grid for which the template can be used.

Value

A template object.

gridPage *Create a css grid based page*

Description

Create a Shiny UI page using a gridPanel to wrap the page content. As other Shiny UI pages, it scaffolds the entire page and loads any required or registered dependencies.

Usage

```
gridPage(..., title = NULL, fill_page = TRUE, dependency = bootstrapLib())
```

Arguments

`...`	Arguments to be passed to gridPanel.
`title`	The browser window title (defaults to the host URL of the page).
`fill_page`	Boolean value if the page should automatically stretch to match the browser window height.
`dependency`	A set of web dependencies. This value can be a htmlDependency, for example the shiny bootstrap dependency (default value) or a tagList with different dependencies.

Value

A UI definition that can be passed to the shinyUI function.

Note

See `https://css-tricks.com/snippets/css/complete-guide-grid/` for additional details on using css grids.

See Also

gridPanel

gridPanel *Create a css grid based HTML tag*

Description

Creates a HTML tag and automatically generates css style rules based on css grid, depending on the given arguments. Functionality acts as a way to generate css grid based HTML containers directly from R without the need to write any additional css rules directly.

Usage

```
gridPanel(
  ...,
  template = NULL,
  areas = NULL,
  rows = NULL,
  columns = NULL,
  gap = NULL,
  align_items = "stretch",
  justify_items = "stretch",
  auto_fill = TRUE,
  breakpoint_system = getBreakpointSystem(),
  id = generateID()
)
```

Arguments

`...` Tag attributes (named arguments) and child elements (unnamed arguments or with names used in areas).

Named arguments are treated as additional html attribute values to the parent tag, unless that name is used in the areas attribute as a grid area name.

Child elements may include any combination of other tags objects, HTML strings, and htmlDependencys.

`template` The name of a registered template to use as a base for the grid, or a template object from gridTemplate.

`areas` A list of vectors with area names, or a vector or strings representing each row of the grid. Each element should contain the names, per row, of each area of the grid.

Expected values follow the convention for the `grid-template-areas` css attribute.

for example `c("area-1 area-1", "area-2 area-3")` and `list(c("area-1", "area-1"), c(area-2"` are both valid representations of a 2x2 grid with 3 named areas.

Supports breakpoints.

`rows` A string of css valid sizes separated by a space. or a vector of sizes. For example both `"1fr 2fr"` or `c("1fr", "2fr")` are valid representations of the same 2 rows grid sizes.

Follows the convention for the `grid-template-rows` css attribute.

If not provided the existing space will be split equally accordingly to the areas defined in areas.

Supports breakpoints.

`columns` A string of css valid sizes separated by a space. or a vector of sizes. For example both `"1fr 2fr"` or `c("1fr", "2fr")` are valid representations of the same 2 columns grid sizes.

Follows the convention for the `grid-template-columns` css attribute.

If not provided the existing space will be split equally accordingly to the areas defined in areas.

Supports breakpoints.

`gap` The space between elements in the panel. Controls both the space between rows and columns.

Accepts a css valid value, or 2 values separated by a space (if using different values for row and column spacing).

Supports breakpoints.

`align_items` The cell behavior according to the `align-items` css property. Aligns grid items along the block (column) axis.

Accepts a valid css `align-items` value (`start` | `end` | `center` | `stretch`).

Supports breakpoints.

`justify_items` The cell behavior according to the `justify-items` css property. Aligns grid items along the inline (row) axis.

Accepts a valid css `justify-items` value (`start` | `end` | `center` | `stretch`).

Supports breakpoints.

`auto_fill` Should the panel stretch to fit its parent size (TRUE), or should its size be based on its children element sizes (FALSE).

Supports breakpoints.

`breakpoint_system`

Breakpoint system to use.

`id` The parent element id.

Details

Behaves similar to a normal HTML tag, but provides helping arguments that simplify the way grid css can be created from shiny.

Value

An HTML tagList.

Note

When creating responsive layouts based on css media rules, some arguments allow a named list can be passed instead of a single value.

The names in that list can be any of the breakpoints available in the `breakpoint_system` argument.

It is recommended to define the breakpoint system for the application globally before UI definitions, but the `breakpoint_system` in panel functions allows for more flexibility when reusing components from other projects.

See `https://css-tricks.com/snippets/css/complete-guide-grid/` for additional details on using css grids.

For a full list of valid HTML attributes check visit `https://www.w3schools.com/tags/ref_attributes.asp`.

See Also

gridPage

`gridTemplate` *Create a grid template*

Description

Creates a imola template as an object for future use. Depending on the given type, the template will then be available to be used as an argument to a panel or page function of that specific type. Templates are collections of arguments that can be grouped and stored for later usage via the "template" argument of panel and page functions.

Usage

```
gridTemplate(
  name,
  type = c("grid", "flex"),
  ...,
  breakpoint_system = getBreakpointSystem(),
  description = NULL
)
```

Arguments

`name` A string with the name that identifies the template.

`type` The type of css grid for which the template can be used. Value must be either "grid" or "flex".

`...` Collection of valid arguments that can be passed to a panel of the given type (see gridPanel and flexPanel for all options)

`breakpoint_system`
Breakpoint system to use.

`description` Optional description with information. about the template. Can be used to pass on any additional relevant information (For example its origin or connected frameworks).

Value

A template object.

`importBreakpointSystem` *Import a breakpoint system*

Description

Imports a breakpoint system from a file. Breakpoint systems can be exported into a file format using exportBreakpointSystem.

Usage

```
importBreakpointSystem(path)
```

Arguments

`path`	The file path of the file to import, including the file name and extension. The file name must end with a `.yaml` extension.

Value

A breakpoint system object.

`importTemplate` *Import a template*

Description

Imports a template from a file. Templates can be exported into a file format by using exportTemplate

Usage

```
importTemplate(path)
```

Arguments

`path`	The file path of the file to import, including the file name and extension. The file name must end with a `.yaml` extension.

Value

A template object.

`listBreakpointSystems` *List registered breakpoint systems*

Description

Lists all available breakpoint systems.

Usage

```
listBreakpointSystems()
```

Value

A named list of css templates and specific values.

`listTemplates` *List registered templates*

Description

Lists all available grid and flex templates. If type is given, returns only templates for the given grid type.

Usage

```
listTemplates(type = NULL)
```

Arguments

`type`	Optional argument for what type of css templates to return. value must be either "grid" or "flex". If no type is given, all templates of all types are returned.

Value

A named list of css templates and specific values.

print.imola.breakpoint

Custom print function for a breakpoint object.

Description

Custom print function for a breakpoint object.

Usage

```
## S3 method for class 'imola.breakpoint'
print(x, ...)
```

Arguments

x	the object to print.
...	Additional arguments.

Value

No return value, called for side effects.

print.imola.breakpoint.system

Custom print function for a breakpoint system object.

Description

Custom print function for a breakpoint system object.

Usage

```
## S3 method for class 'imola.breakpoint.system'
print(x, ...)
```

Arguments

x	the object to print.
...	Additional arguments.

Value

No return value, called for side effects.

print.imola.template *Custom print function for a template object.*

Description

Custom print function for a template object.

Usage

```
## S3 method for class 'imola.template'
print(x, ...)
```

Arguments

x	the object to print.
...	Additional arguments.

Value

No return value, called for side effects.

registerBreakpointSystem *Register a breakpoint system*

Description

Registers a breakpoint system object to make it available globally in `getOption("imola.breakpoint.systems")`. After registered it can be retrieved anywhere using getBreakpointSystem.

Usage

```
registerBreakpointSystem(system)
```

Arguments

system	A breakpoint system object created with breakpointSystem.

Value

No return value, called for side effects.

`registerTemplate` *Register a template*

Description

Registers a template object to make it available globally in getOption("imola.templates"). After registered it can be retrieved anywhere using getTemplate.

Usage

```
registerTemplate(template)
```

Arguments

`template`	A template object generated with gridTemplate.

Value

No return value, called for side effects.

`removeBreakpoint` *Remove a breakpoint from a breakpoint system*

Description

Removes a breakpoint from a breakpoint system object by name.

Usage

```
removeBreakpoint(system, name)
```

Arguments

`system`	A breakpoint system object created with breakpointSystem.
`name`	A string with the name of a breakpoint in the given system.

Value

A breakpoint system object.

`setActiveBreakpointSystem` *Set the active breakpoint system*

Description

Sets the current globally active breakpoint system. The active breakpoint system is used for grid function as the default system if no system is provided as an argument.

Usage

```
setActiveBreakpointSystem(system)
```

Arguments

`system`	A string with the name of a registered breakpoint system, or a breakpoint system object generated with breakpointSystem. If a breakpoint system object is used, it will be registered as well.

Value

A breakpoint system object.

`unregisterBreakpointSystem` *Unregister a breakpoint system*

Description

Removes a globally registered breakpoint system from `getOption("imola.breakpoint.systems")`.

Usage

```
unregisterBreakpointSystem(name)
```

Arguments

`name`	A string with the name of a registered breakpoint system. Registered systems are available in `getOption("imola.breakpoint.systems")`.

Value

No return value, called for side effects.

`unregisterTemplate` *Unregister a template*

Description

Removes a globally registered template from `getOption("imola.templates")`.

Usage

```
unregisterTemplate(name, type)
```

Arguments

`name`	A string with the name of a registered template. Registered templates are available in `getOption("imola.templates")`.
`type`	The type of css grid for which the template can be used.

Value

No return value, called for side effects.

Index