

# Package ‘InteractiveComplexHeatmap’

May 1, 2024

**Type** Package

**Title** Make Interactive Complex Heatmaps

**Version** 1.13.0

**Date** 2023-08-10

**Depends** R (>= 4.0.0), ComplexHeatmap (>= 2.11.0)

**Imports** grDevices, stats, shiny, grid, GetoptLong, S4Vectors (>= 0.26.1), digest, IRanges, kableExtra (>= 1.3.1), utils, svglite, htmltools, clisymbols, jsonlite, RColorBrewer, fontawesome

**Suggests** knitr, rmarkdown, testthat, EnrichedHeatmap, GenomicRanges, data.table, circlize, GenomicFeatures, tidyverse, tidyHeatmap, cluster, org.Hs.eg.db, simplifyEnrichment, GO.db, SC3, GOexpress, SingleCellExperiment, scater, gplots, pheatmap, airway, DESeq2, DT, cola, BiocManager, gridtext, HilbertCurve (>= 1.21.1), shinydashboard, SummarizedExperiment, pkgndep, ks

**VignetteBuilder** knitr

**Description** This package can easily make heatmaps which are produced by the ComplexHeatmap package into interactive applications. It provides two types of interactivities:

1. on the interactive graphics device, and 2. on a Shiny app. It also provides functions for integrating the interactive heatmap widgets for more complex Shiny app development.

**biocViews** Software, Visualization, Sequencing

**URL** <https://github.com/jokergoo/InteractiveComplexHeatmap>

**BugReports** <https://github.com/jokergoo/InteractiveComplexHeatmap/issues>

**License** MIT + file LICENSE

**git\_url** <https://git.bioconductor.org/packages/InteractiveComplexHeatmap>

**git\_branch** devel

**git\_last\_commit** bd8471f

**git\_last\_commit\_date** 2024-04-30

**Repository** Bioconductor 3.20

**Date/Publication** 2024-05-01

**Author** Zuguang Gu [aut, cre] (<<https://orcid.org/0000-0002-7395-8709>>)

**Maintainer** Zuguang Gu <z.gu@dkfz.de>

## Contents

all_column_indices . . . . .	2
all_row_indices . . . . .	3
getPositionFromBrush . . . . .	3
getPositionFromClick . . . . .	4
getPositionFromDbclick . . . . .	5
getPositionFromHover . . . . .	5
HeatmapInfoOutput . . . . .	6
htPositionsOnDevice . . . . .	7
htShiny . . . . .	8
htShinyExample . . . . .	11
ht_shiny . . . . .	11
interactivate . . . . .	12
interactivate.DESeqDataSet . . . . .	13
interactivate.kde . . . . .	13
interactivateDensity2D . . . . .	14
InteractiveComplexHeatmapModal . . . . .	15
InteractiveComplexHeatmapOutput . . . . .	17
InteractiveComplexHeatmapWidget . . . . .	19
is_in_sub_heatmap . . . . .	21
makeInteractiveComplexHeatmap . . . . .	21
originalHeatmapOutput . . . . .	23
rand_mat . . . . .	24
record_observation . . . . .	25
selectArea . . . . .	26
selectPosition . . . . .	27
subHeatmapOutput . . . . .	28
<b>Index</b>	<b>29</b>

---

all_column_indices	<i>Get all column indices from the selected data frame</i>
--------------------	--

---

### Description

Get all column indices from the selected data frame

### Usage

```
all_column_indices(df)
```

**Arguments**

df                    The selected data frame.

**Examples**

```
# There is no example
NULL
```

---

all\_row\_indices            *Get all row indices from the selected data frame*

---

**Description**

Get all row indices from the selected data frame

**Usage**

```
all_row_indices(df)
```

**Arguments**

df                    The selected data frame.

**Examples**

```
# There is no example
NULL
```

---

getPositionFromBrush    *Get the position of the brushed area on the heatmap image*

---

**Description**

Get the position of the brushed area on the heatmap image

**Usage**

```
getPositionFromBrush(brush, ratio = 1)
```

**Arguments**

brush                    The input brush object. Assume heatmap\_brush is the ID set to argument brush in [plotOutput](#), then the value here is input\$heatmap\_brush.

ratio                    The relative resolution. The value should be the ratio between res set in [makeInteractiveComplexHeatmap](#) and 72 (res/72).

**Value**

A list of length two. The two elements corresponds to the coordinates of the two diagonal points.

**See Also**

[getPositionFromClick](#), [getPositionFromHover](#), [getPositionFromDbclick](#).

**Examples**

```
# There is no example  
NULL
```

---

getPositionFromClick *Get the position of clicked point on the heatmap image*

---

**Description**

Get the position of clicked point on the heatmap image

**Usage**

```
getPositionFromClick(click, ratio = 1)
```

**Arguments**

click	The input click object. Assume heatmap_click is the ID set to argument click in <a href="#">plotOutput</a> , then the value here is input\$heatmap_click.
ratio	The relative resolution. The value should be the ratio between res set in <a href="#">makeInteractiveComplexHeatmap</a> and 72 (res/72).

**Value**

A [unit](#) object of length two which are the coordinates of the clicked points.

**See Also**

[getPositionFromBrush](#), [getPositionFromHover](#), [getPositionFromDbclick](#).

**Examples**

```
# There is no example  
NULL
```

---

`getPositionFromDblclick`*Get the position of double clicked point on the heatmap image*

---

**Description**

Get the position of double clicked point on the heatmap image

**Usage**

```
getPositionFromDblclick(dblclick, ratio = 1)
```

**Arguments**

`dblclick` The input `dblclick` object. Assume `heatmap_dblclick` is the ID set to argument `dblclick` in `plotOutput`, then the value here is `input$heatmap_dblclick`.

`ratio` The relative resolution. The value should be the ratio between `res` set in `makeInteractiveComplexHeatmap` and 72 (`res/72`).

**Value**

A `unit` object of length two which are the coordinates of the double clicked points.

**Examples**

```
# There is no example  
NULL
```

---

`getPositionFromHover` *Get the position of hovered point on the heatmap image*

---

**Description**

Get the position of hovered point on the heatmap image

**Usage**

```
getPositionFromHover(hover, ratio = 1)
```

**Arguments**

`hover` The input `hover` object. Assume `heatmap_hover` is the ID set to argument `hover` in `plotOutput`, then the value here is `input$heatmap_hover`.

`ratio` The relative resolution. The value should be the ratio between `res` set in `makeInteractiveComplexHeatmap` and 72 (`res/72`).

**Value**

A `unit` object of length two which are the coordinates of the hover points.

**Examples**

```
# There is no example
NULL
```

---

HeatmapInfoOutput	<i>UI for the output</i>
-------------------	--------------------------

---

**Description**

UI for the output

**Usage**

```
HeatmapInfoOutput(heatmap_id, title = NULL, width = 400,
  output_ui = default_output_ui(heatmap_id),
  output_ui_float = FALSE, action = NULL, response = NULL, internal = FALSE)
```

**Arguments**

<code>heatmap_id</code>	ID of the plot.
<code>title</code>	Title of the output.
<code>width</code>	Width of the output div.
<code>output_ui</code>	A <a href="#">htmlOutput</a> or other <code>*Output</code> object (defined in shiny or other related packages).
<code>output_ui_float</code>	Whether the UI defined by <code>output_ui</code> floats at the mouse positions.
<code>action</code>	It is only used when <code>output_ui_float = TRUE</code> to properly bind the floating frame to the event on heatmap (i.e. click, hover or dblclick). If <a href="#">HeatmapInfoOutput</a> is executed after <a href="#">originalHeatmapOutput</a> , the value for it is automatically decided
<code>response</code>	It is only used when <code>output_ui_float = TRUE</code> and <code>response = "brush"</code> or <code>response = "brush-output"</code> , so that single clicking or hovering won't have any effect, in other word, there is only response from brushing. If <a href="#">HeatmapInfoOutput</a> is executed after <a href="#">originalHeatmapOutput</a> , the value for it is automatically decided
<code>internal</code>	Internally used.

**See Also**

[originalHeatmapOutput](#), [subHeatmapOutput](#).

**Examples**

```
# See examples on the help page of originalHeatmapOutput()
```

---

```
htPositionsOnDevice Get heatmap positions on the graphics device
```

---

**Description**

Get heatmap positions on the graphics device

**Usage**

```
htPositionsOnDevice(ht_list = get_last_ht(), unit = "inch", valueOnly = FALSE,
  include_annotation = FALSE, calibrate = TRUE)
```

**Arguments**

ht_list	A <a href="#">HeatmapList-class</a> object returned by <a href="#">draw, Heatmap-method</a> or <a href="#">draw, HeatmapList-method</a> . If it is omitted, it uses the last generated heatmap.
unit	The unit.
valueOnly	Whether only return the numeric values.
include_annotation	Internally used.
calibrate	Internally used.

**Details**

ht\_list must have been already updated by draw() function. The function needs to be executed under a graphics device where the heatmap is written.

**Value**

It returns a [DataFrame](#) object of the position of every heatmap slice.

**Examples**

```
if(dev.interactive()) {
  m = matrix(rnorm(100), 10)
  ht = Heatmap(m, row_km = 2, column_km = 2)
  ht = draw(ht)
  pos = htPositionsOnDevice(ht)

  InteractiveComplexHeatmap:::redraw_ht_vp(pos)
}
```

**Description**

Interactive heatmaps as a Shiny app

**Usage**

```
htShiny(ht_list = get_last_ht(), title = NULL,
        description = NULL, hline = TRUE, html = NULL,

        # parameters passed to InteractiveComplexHeatmapOutput()
heatmap_id = NULL, title1 = "Original heatmap", title2 = "Selected sub-heatmap",
width1 = ifelse(layout == "1|(2-3)", 800, 450),
height1 = ifelse(layout == "1-(2|3)", 700, 350),
width2 = 400,
height2 = 350,
width3 = ifelse(layout == "(1-2)|3", 800, 400),
layout = ifelse("brush" %in% response, "(1-2)|3", "1-3"), compact = FALSE,
action = "click", cursor = TRUE, response = c(action, "brush"),
brush_opt = list(stroke = "#f00", opacity = 0.6),
output_ui_float = FALSE,

# specific for sub-heatmap
show_cell_fun = TRUE, show_layer_fun = TRUE,

save = NULL, app_options = list())
```

**Arguments**

ht_list	A <a href="#">Heatmap-class</a> or a <a href="#">HeatmapList-class</a> object. If it is not specified, the last generated heatmap is used. The heatmap object should better be already updated by <code>draw()</code> function.
title	Title of the app.
description	Description of the app. The content will be wrapped by a <code>p</code> tag and inserted before the interactive heatmap widget.
hline	Whether to add the horizontal line (by <code>hr</code> tag) after description.
html	HTML fragment inserted below the heatmap. The value can be a string or be wrapped by <a href="#">HTML</a> .
heatmap_id	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
title1	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
title2	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
width1	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .



height1	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
width2	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
height2	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
width3	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
layout	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
compact	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
action	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
cursor	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
response	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
brush_opt	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
output_ui_float	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
show_cell_fun	Whether show graphics made by cell_fun on the main heatmap?
show_layer_fun	Whether show graphics made by cell_fun on the main heatmap?
save	The value can be set to a folder name so that the shiny app is saved into several files.
app_options	All pass to the options argument in <a href="#">shinyApp</a> .

### Details

With any Heatmap/HeatmapList object, directly send to htShiny() to create a Shiny app for the heatmap(s):

```
htShiny(ht_list)
```

If the heatmaps are already drawn, ht\_list can be omitted and the last heatmap object is retrieved automatically:

```
Heatmap(...) + other_heatmaps_or_annotations # or other functions that internally use Heatmap()
htShiny()
```

### Value

A Shiny app object.

### See Also

- [https://jokergoo.shinyapps.io/interactive\\_complexheatmap/](https://jokergoo.shinyapps.io/interactive_complexheatmap/)
- [https://jokergoo.shinyapps.io/interactive\\_complexheatmap\\_vertical/](https://jokergoo.shinyapps.io/interactive_complexheatmap_vertical/)
- [https://jokergoo.shinyapps.io/interactive\\_densityheatmap/](https://jokergoo.shinyapps.io/interactive_densityheatmap/)
- [https://jokergoo.shinyapps.io/interactive\\_oncoprint/](https://jokergoo.shinyapps.io/interactive_oncoprint/)
- [https://jokergoo.shinyapps.io/interactive\\_enrichedheatmap/](https://jokergoo.shinyapps.io/interactive_enrichedheatmap/)
- [https://jokergoo.shinyapps.io/interactive\\_upsetp/](https://jokergoo.shinyapps.io/interactive_upsetp/)

- [https://jokergooo.shinyapps.io/interactive\\_pheatmap/](https://jokergooo.shinyapps.io/interactive_pheatmap/)
- [https://jokergooo.shinyapps.io/interactive\\_heatmap/](https://jokergooo.shinyapps.io/interactive_heatmap/)
- [https://jokergooo.shinyapps.io/interactive\\_heatmap\\_2/](https://jokergooo.shinyapps.io/interactive_heatmap_2/)
- [https://jokergooo.shinyapps.io/interactive\\_tidyheatmap/](https://jokergooo.shinyapps.io/interactive_tidyheatmap/)

There are also many examples that can be get with [htShinyExample](#).

## Examples

```
# use last generated heatmap
if(interactive() && dev.interactive()) {
  m = matrix(rnorm(100), 10)
  Heatmap(m)
  htShiny()
}

# by providing a heatmap/heatmap list
if(interactive()) {
  m = matrix(rnorm(100), 10)
  rownames(m) = 1:10
  colnames(m) = 1:10

  ht = Heatmap(m)
  ht = draw(ht)
  htShiny(ht)
}

# vertical heatmap list
if(interactive()) {
  m1 = matrix(rnorm(100), 10)
  rownames(m1) = 1:10
  colnames(m1) = 1:10
  ht1 = Heatmap(m1, row_km = 2, column_km = 2)

  m2 = matrix(sample(letters[1:10], 100, replace = TRUE), 10)
  ht2 = Heatmap(m2)

  ht_list = draw(ht1 + ht2)
  htShiny(ht_list)

  ht_list = ht1 %v% ht2
  htShiny(ht_list)
}

# compact mode
if(interactive()) {
  m = matrix(rnorm(100), 10)
  Heatmap(m)
  htShiny(compact = TRUE)
}
```

---

htShinyExample	<i>Examples of interactive complex heatmaps</i>
----------------	---

---

**Description**

Examples of interactive complex heatmaps

**Usage**

```
htShinyExample(which)
```

**Arguments**

`which` An index of which example to use. The list of all examples can be obtained by executing `htShinyExample` with no argument.

**Details**

In every example, there is a Shiny app opened, which also includes source code that generates this app.

**Value**

A Shiny app object.

**Examples**

```
# list all examples
htShinyExample()

if(interactive()) {
  htShinyExample(4.2)
}
```

---

ht_shiny	<i>Interactive heatmaps as a Shiny app</i>
----------	--

---

**Description**

Interactive heatmaps as a Shiny app

**Usage**

```
ht_shiny(...)
```

**Arguments**

... All goes to [htShiny](#).

**Value**

A Shiny app object.

**Examples**

```
# There is no example  
NULL
```

---

interactivate	<i>Generic function for interactivate an object in an interactive Shiny app</i>
---------------	---

---

**Description**

Generic function for interactivate an object in an interactive Shiny app

**Usage**

```
interactivate(x, ...)
```

**Arguments**

x An object.  
... Other arguments.

**Examples**

```
# There is no example  
NULL
```

---

interactivate.DESeqDataSet

*Visualize DESeq2 result in an interactive Shiny app*


---

### Description

Visualize DESeq2 result in an interactive Shiny app

### Usage

```
## S3 method for class 'DESeqDataSet'
interactivate(x, res = DESeq2::results(x), seed = 123, ...)
```

### Arguments

x	A <a href="#">DESeqDataSet</a> class object. It is normally returned by <a href="#">DESeq</a> .
res	The object returned by <a href="#">results</a> .
seed	Random seed. It is mainly set for the random colors of annotations.
...	Other arguments.

### Examples

```
if(interactive()) {
  require(airway)
  data(airway)
  se = airway

  require(DESeq2)
  dds = DESeqDataSet(se, design = ~ dex)
  keep = rowSums(counts(dds)) >= 10
  dds = dds[keep, ]
  dds$dex = relevel(dds$dex, ref = "untrt")
  dds = DESeq(dds)

  interactivate(dds)
}
```

---

interactivate.kde

*Interactive Shiny application for 2D density distribution*


---

### Description

Interactive Shiny application for 2D density distribution

**Usage**

```
## S3 method for class 'kde'  
interactivate(x, ...)
```

**Arguments**

x                    a kde object generated by [kde](#).  
...                  Other arguments.

**Examples**

```
if(interactive()) {  
  require(ks)  
  lt = readRDS(system.file("extdata", "2d_density_xy.rds", package = "InteractiveComplexHeatmap"))  
  data = cbind(lt$x, lt$y)  
  fit = kde(data)  
  interactivate(fit)  
}
```

---

interactivateDensity2D

*Interactive Shiny application for 2D density distribution*

---

**Description**

Interactive Shiny application for 2D density distribution

**Usage**

```
interactivateDensity2D(x, y, ...)
```

**Arguments**

x                    A numeric vector.  
y                    A numeric vector.  
...                  All pass to [kde](#).

**Examples**

```
if(interactive()) {  
  lt = readRDS(system.file("extdata", "2d_density_xy.rds", package = "InteractiveComplexHeatmap"))  
  interactivateDensity2D(lt$x, lt$y)  
}
```

---

 InteractiveComplexHeatmapModal

*Interactive complex heatmap modal dialog*


---

## Description

Interactive complex heatmap modal dialog

## Usage

```
InteractiveComplexHeatmapModal(
  input, output, session, ht_list, heatmap_id = NULL,

  # parameters passed to InteractiveComplexHeatmapOutput()
  title1 = "Original heatmap", title2 = "Selected sub-heatmap",
  width1 = ifelse(layout == "1|(2-3)", 800, 450),
  height1 = ifelse(layout == "1-(2|3)", 700, 350),
  width2 = 370,
  height2 = 350,
  width3 = ifelse(layout == "(1-2)|3", 800, 370),
  layout = ifelse("brush" %in% response, "(1-2)|3", "1-3"), compact = FALSE,
  action = "click", cursor = TRUE, response = c(action, "brush"),
  brush_opt = list(stroke = "#f00", opacity = 0.6),
  output_ui = TRUE, output_ui_float = FALSE,

  # parameters passed to makeInteractiveComplexHeatmap()
  click_action = NULL, brush_action = NULL,

  # other configurations
  js_code = "", close_button = TRUE, cancel_action = c("remove", "hide"))
```

## Arguments

input	Passed from the Shiny server function.
output	Passed from the Shiny server function.
session	Passed from the Shiny server function.
ht_list	A <a href="#">Heatmap-class</a> or a <a href="#">HeatmapList-class</a> object.
heatmap_id	ID of the plot. If it is not specified, an internal ID is assigned.
title1	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
title2	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
width1	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
height1	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
width2	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
height2	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .

width3	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
layout	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
compact	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
action	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
cursor	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
response	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
brush_opt	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
output_ui	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
output_ui_float	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
click_action	Pass to <a href="#">makeInteractiveComplexHeatmap</a> .
brush_action	Pass to <a href="#">makeInteractiveComplexHeatmap</a> .
js_code	Additional JavaScript code that is put after the interactive heatmap UI. The value can be a text or a function that takes "heatmap ID" as the argument and returns the formatted JavaScript code.
close_button	Whether to add a close button at the end of the widget. If it is FALSE, the widget can be closed by clicking outside of the widget.
cancel_action	Whether to remove the UI from HTML or just hide it when the UI is closed.

### Details

It creates an interactive heatmap "modal dialog" according to a certain action.  
The function is normally put inside [observe](#) or [observeEvent](#).

### Value

No value is returned.

### Examples

```
if(interactive()) {
  require(ComplexHeatmap)

  ui = fluidPage(
    actionButton("show_heatmap", "Generate_heatmap"),
  )

  server = function(input, output, session) {
    m = matrix(rnorm(100), 10)
    ht = Heatmap(m)

    observeEvent(input$show_heatmap, {
      InteractiveComplexHeatmapModal(input, output, session, ht)
    })
  }
  shiny::shinyApp(ui, server)
}
```



---

 InteractiveComplexHeatmapOutput

*UI for the interactive complex heatmaps*


---

## Description

UI for the interactive complex heatmaps

## Usage

```
InteractiveComplexHeatmapOutput(heatmap_id = NULL,
  title1 = "Original heatmap", title2 = "Selected sub-heatmap",
  title3 = if(output_ui_float) NULL else "Output",
  width1 = ifelse(layout == "1|(2-3)", 800, 450),
  height1 = ifelse(layout == "1-(2|3)", 700, 350),
  width2 = 400,
  height2 = 350,
  width3 = NULL,
  layout = ifelse("brush" %in% response, "(1-2)|3", "1-3"), compact = FALSE,
  action = "click", cursor = TRUE,
  response = c(action, "brush"),
  brush_opt = list(stroke = "#f00", opacity = 0.6),
  output_ui = default_output_ui(heatmap_id),
  output_ui_float = FALSE, containment = FALSE,
  internal = FALSE,
  ...)
```

## Arguments

heatmap_id	ID of the plot. If it is not specified, an internal ID is assigned.
title1	Title of the original heatmap.
title2	Title of the sub-heatmap.
title3	Title of the output.
width1	Width of the original heatmap.
height1	Height of the original heatmap.
width2	Width of the sub-heatmap.
height2	Height of the sub-heatmap.
width3	Width of the output div.
layout	One of "1 2)-3", "1-(2 3)", "1-2-3", "1 2 3", "1 (2-3)". If brush is not set with the argument response, which means there is no sub-heatmap panel, the code 2 can be omitted.
compact	If the value is TRUE, there will be no sub-heatmap, and output floats at the mouse position when click/hover on the original heatmap.

action	Which action for selecting single cells on the heatmap? Value should be <code>click</code> , <code>hover</code> or <code>dblclick</code> .
cursor	When moving mouse on heatmap, whether to show the cursors on the four sides?
response	Which action needs to be responded on the server side? Value should be in <code>click/hover/dblclick</code> , <code>brush</code> and <code>brush-output</code> . <code>brush</code> responds in two places which are the sub-heatmap and the output components and <code>brush-output</code> only responds in the output component.
brush_opt	A list of parameters passed to <code>brushOpts</code> . Do not set an ID for the brush. An internal brush ID is automatically set.
output_ui	A <code>htmlOutput</code> or other <code>*Output</code> object (defined in <code>shiny</code> or other related packages). If it is set to <code>NULL</code> , there is no output component in the app.
output_ui_float	Whether the UI defined by <code>output_ui</code> floats at the mouse positions.
containment	Whether the resizing is restricted in a certain parent div? Value can be <code>TRUE/FALSE</code> or a JQuery selector.
internal	Internally used.
...	Pass to the UI container which is wrapped by <code>fluidPage</code> .

### Details

This function generates HTML fragment for the interactive UI. See the example in [makeInteractiveComplexHeatmap](#) page.

layout is defined as follows (1 for the original heatmap, 2 for the selected sub-heatmap and 3 is for the output:

- "(1-2) | 3": Heatmap and sub-heatmap are in a same row, and output is in a second row. This is the default layout.
- "1 | (2-3)": Heatmap is in a single row, while sub-heatmap and output are in a second row.
- "1-2-3": All three components are in a same row.
- "1 | 2 | 3": Each component is in a single row.
- "1-(2 | 3)": Being different from the other four layouts, this is a two-column layout. Heatmap is in a single column. Sub-heatmap and output are vertically aligned and the two are in the second column.

The hover event is implemented with <https://github.com/websanova/mousestop>.

### Value

A UI that can be used in Shiny.

### Examples

```
# There is no example
NULL
```

---

InteractiveComplexHeatmapWidget  
*Interactive complex heatmap widget*

---

## Description

Interactive complex heatmap widget

## Usage

```
InteractiveComplexHeatmapWidget(
  input, output, session, ht_list, heatmap_id = NULL, output_id,

  # parameters passed to InteractiveComplexHeatmapOutput()
  title1 = "Original heatmap", title2 = "Selected sub-heatmap",
  width1 = ifelse(layout == "1|(2-3)", 800, 450),
  height1 = ifelse(layout == "1-(2|3)", 700, 350),
  width2 = 370,
  height2 = 350,
  width3 = ifelse(layout == "(1-2)|3", 800, 370),
  layout = ifelse("brush" %in% response, "(1-2)|3", "1-3"), compact = FALSE,
  action = "click", cursor = TRUE, response = c(action, "brush"),
  brush_opt = list(stroke = "#f00", opacity = 0.6),
  output_ui = TRUE, output_ui_float = FALSE,

  # parameters passed to makeInteractiveComplexHeatmap()
  click_action = NULL, brush_action = NULL,

  # other configurations
  js_code = "", close_button = TRUE, cancel_action = c("remove", "hide"))
```

## Arguments

input	Passed from the Shiny server function.
output	Passed from the Shiny server function.
session	Passed from the Shiny server function.
ht_list	A <a href="#">Heatmap-class</a> or a <a href="#">HeatmapList-class</a> object.
heatmap_id	ID of the plot. If it is not specified, an internal ID is assigned.
output_id	Where the heatmap is put.
title1	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
title2	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
width1	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
height1	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
width2	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .

height2	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
width3	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
layout	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
compact	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
action	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
cursor	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
response	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
brush_opt	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
output_ui	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
output_ui_float	Pass to <a href="#">InteractiveComplexHeatmapOutput</a> .
click_action	Pass to <a href="#">makeInteractiveComplexHeatmap</a> .
brush_action	Pass to <a href="#">makeInteractiveComplexHeatmap</a> .
js_code	Additional JavaScript code that is put after the interactive heatmap UI. The value can be a text or a function that takes "heatmap ID" as the argument and returns the formatted JavaScript code.
close_button	Whether to add a close button at the end of the widget.
cancel_action	Whether to remove the UI from HTML or just hide it when the UI is closed.

### Details

It creates an interactive heatmap widget according to a certain action. The UI is placed to the output ID that user defined.

The function is normally put inside [observe](#) or [observeEvent](#).

### Value

No value is returned.

### Examples

```
if(interactive()) {
  require(ComplexHeatmap)

  ui = fluidPage(
    actionButton("show_heatmap", "Generate_heatmap"),
    htmlOutput("heatmap_output")
  )

  server = function(input, output, session) {
    m = matrix(rnorm(100), 10)
    ht = Heatmap(m)

    observeEvent(input$show_heatmap, {
      InteractiveComplexHeatmapWidget(input, output, session, ht,
        output_id = "heatmap_output")
    })
  }
}
```

```
    })  
  }  
  shiny::shinyApp(ui, server)  
}
```

---

is\_in\_sub\_heatmap      *Test whether it is in sub heatmap*

---

### Description

Test whether it is in sub heatmap

### Usage

```
is_in_sub_heatmap()
```

### Details

Normally, it is used in cell\_fun/layer\_fun.

### Examples

```
# There is no example  
NULL
```

---

makeInteractiveComplexHeatmap  
*Process heatmaps on the sever side*

---

### Description

Process heatmaps on the sever side

### Usage

```
makeInteractiveComplexHeatmap(input, output, session, ht_list,  
  heatmap_id = shiny_env$current_heatmap_id,  
  click_action = NULL, hover_action = NULL,  
  dblclick_action = NULL, brush_action = NULL, res = 72,  
  show_cell_fun = TRUE, show_layer_fun = TRUE)
```

**Arguments**

input	Passed from the Shiny server function.
output	Passed from the Shiny server function.
session	Passed from the Shiny server function.
ht_list	A <a href="#">Heatmap-class</a> or a <a href="#">HeatmapList-class</a> object.
heatmap_id	The corresponding heatmap ID from the UI. If there is only one interactive heatmap in the app, this argument does not need to be specified and it will use the current one used in <a href="#">InteractiveComplexHeatmapOutput</a> .
click_action	Additional actions on the server side when receiving a click event on the UI. This self-defined function should accept two or four arguments. If it is two arguments, they should be df and output and if it is four arguments, they should be df, input, output and session.
hover_action	Additional actions at the server side when receiving a hover event on the UI.
dblclick_action	Additional actions at the server side when receiving a dblclick event on the UI.
brush_action	Additional actions at the server side when receiving a brush event on the UI.
res	Resolution of the plot, pass to <a href="#">renderPlot</a> .
show_cell_fun	Whether show graphics made by cell_fun on the main heatmap?
show_layer_fun	Whether show graphics made by cell_fun on the main heatmap?

**Value**

No value is returned.

**Examples**

```

if(interactive()) {
  ht = Heatmap(m)
  ht = draw(ht)

  ui = fluidPage(
    InteractiveComplexHeatmapOutput()
  )

  server = function(input, output, session) {
    makeInteractiveComplexHeatmap(input, output, session, ht)
  }

  shiny::shinyApp(ui, server)
}

```

---

originalHeatmapOutput *UI for the original heatmap*

---

### Description

UI for the original heatmap

### Usage

```
originalHeatmapOutput(heatmap_id, title = NULL,  
  width = 450, height = 350,  
  action = "click", cursor = TRUE,  
  response = c(action, "brush"),  
  brush_opt = list(stroke = "#f00", opacity = 0.6),  
  containment = FALSE, internal = FALSE)
```

### Arguments

heatmap_id	ID of the plot.
title	Title of the original heatmap.
width	Width of the original heatmap.
height	Height of the original heatmap.
action	Which action for selecting single cells on the heatmap? Value should be click, hover or dblclick.
cursor	When moving mouse on heatmap, whether to show the cursors on the four sides?
response	Which action needs to be responded on the server side? Value should be in click/hover/dblclick, brush and brush-output. brush responds in two places which are the sub-heatmap and the output components and brush-output only responds in the output component.
brush_opt	A list of parameters passed to <a href="#">brushOpts</a> . Do not set an ID for the brush. An internal brush ID is automatically set.
containment	Whether the resizing is restricted in a certain parent div? Value can be TRUE/FALSE or a JQuery selector.
internal	Internally used.

### See Also

[subHeatmapOutput](#), [HeatmapInfoOutput](#).

**Examples**

```

if(interactive()) {
  require(shinydashboard)
  m = matrix(rnorm(100), 10)
  ht = Heatmap(m)

  body = dashboardBody(
    fluidRow(
      box(title = "Original heatmap", width = 4, solidHeader = TRUE, status = "primary",
        originalHeatmapOutput("ht")
      ),
      box(title = "Sub-heatmap", width = 4, solidHeader = TRUE, status = "primary",
        subHeatmapOutput("ht")
      ),
      box(title = "Output", width = 4, solidHeader = TRUE, status = "primary",
        HeatmapInfoOutput("ht")
      )
    )
  )
  ui = dashboardPage(
    dashboardHeader(),
    dashboardSidebar(),
    body
  )
  server = function(input, output, session) {
    makeInteractiveComplexHeatmap(input, output, session, ht, "ht")
  }
  shinyApp(ui, server)
}

```

---

rand\_mat

*A random matrix*


---

**Description**

A random matrix

**Usage**

```
data(rand_mat)
```

**Details**

Following code was used to generate rand\_mat:

```

set.seed(123)
rand_mat = cbind(rbind(matrix(rnorm(20*20, mean = 1, sd = 0.5), nr = 20),
  matrix(rnorm(20*20, mean = 0, sd = 0.5), nr = 20),
  matrix(rnorm(20*20, mean = 0, sd = 0.5), nr = 20)),

```



```
      rbind(matrix(rnorm(20*20, mean = 0, sd = 0.5), nr = 20),
            matrix(rnorm(20*20, mean = 1, sd = 0.5), nr = 20),
            matrix(rnorm(20*20, mean = 0, sd = 0.5), nr = 20)),
      rbind(matrix(rnorm(20*20, mean = 0.5, sd = 0.5), nr = 20),
            matrix(rnorm(20*20, mean = 0.5, sd = 0.5), nr = 20),
            matrix(rnorm(20*20, mean = 1, sd = 0.5), nr = 20))
    ) + matrix(rnorm(60*60, sd = 0.5), nr = 60)
  colnames(rand_mat) = paste0("C", 1:60)
  rownames(rand_mat) = paste0("R", 1:60)
```

### Author(s)

Zuguang Gu <z.gu@dkfz.de>

### Examples

```
data(rand_mat)
rand_mat
```

---

record_observation	<i>Record the observation object</i>
--------------------	--------------------------------------

---

### Description

Record the observation object

### Usage

```
record_observation(obs, heatmap_id = shiny_env$current_heatmap_id)
```

### Arguments

obs	Observation object returned by <a href="#">observe</a> or <a href="#">observeEvent</a> .
heatmap_id	The Heatmap ID.

### Examples

```
# There is no example
NULL
```

---

selectArea	<i>Select an area in the heatmap</i>
------------	--------------------------------------

---

### Description

Select an area in the heatmap

### Usage

```
selectArea(ht_list = get_last_ht(), pos1 = NULL, pos2 = NULL, mark = TRUE, verbose = TRUE,
           ht_pos = NULL, include_annotation = FALSE, calibrate = TRUE)
```

### Arguments

ht_list	A <a href="#">HeatmapList-class</a> object returned by <a href="#">draw, Heatmap-method</a> or <a href="#">draw, HeatmapList-method</a> . If it is omitted, it uses the last generated heatmap.
mark	Whether to mark the selected area as a rectangle.
pos1	If the value is NULL, it can be selected by click on the heatmap (of course, the heatmap should be on the interactive graphics device). If it is set, it must be a <a href="#">unit</a> object with length two which corresponds to the x and y position of the point.
pos2	Another point as pos1, together with pos1 defines the selected region.
verbose	Whether to print messages.
ht_pos	A value returned by <a href="#">htPositionsOnDevice</a> .
include_annotation	Internally used.
calibrate	Internally used. Mainly works for Rstudio desktop IDE.

### Details

The regions can be selected interactively or selected manually by setting pos1 and pos2.

### Value

A [DataFrame](#) object with row indices and column indices corresponding to the selected region.

### Examples

```
if(dev.interactive()) {
  m = matrix(rnorm(100), 10)
  rownames(m) = 1:10
  colnames(m) = 1:10

  ht = Heatmap(m)
  ht = draw(ht)
  selectArea(ht)
```

```
set.seed(123)
ht = Heatmap(m, row_km = 2, column_km = 2)
ht = draw(ht)
selectArea(ht)
}
```

---

selectPosition	Select a position in the heatmap
----------------	----------------------------------

---

### Description

Select a position in the heatmap

### Usage

```
selectPosition(ht_list = get_last_ht(), pos = NULL, mark = TRUE, verbose = TRUE,
  ht_pos = NULL, calibrate = TRUE)
```

### Arguments

ht_list	A <a href="#">HeatmapList-class</a> object returned by <a href="#">draw, Heatmap-method</a> or <a href="#">draw, HeatmapList-method</a> . If it is omitted, it uses the last generated heatmap.
mark	Whether to mark the selected position as a point.
pos	If the value is NULL, it can be selected by click on the heatmap (of course, the heatmap should be on the interactive graphics device). If it is set, it must be a <a href="#">unit</a> object with length two which corresponds to the x and y position of the point.
verbose	Whether to print messages.
ht_pos	A value returned by <a href="#">htPositionsOnDevice</a> .
calibrate	Internally used. Mainly works for Rstudio desktop IDE.

### Details

The regions can be selected interactively or selected manually by setting pos.

### Value

A [DataFrame](#) object with row indices and column indices corresponding to the selected position.

**Examples**

```
if(dev.interactive()) {  
  m = matrix(rnorm(100), 10)  
  rownames(m) = 1:10  
  colnames(m) = 1:10  
  
  ht = Heatmap(m)  
  ht = draw(ht)  
  selectPosition(ht)  
}
```

---

subHeatmapOutput	<i>UI for the sub-heatmaps</i>
------------------	--------------------------------

---

**Description**

UI for the sub-heatmaps

**Usage**

```
subHeatmapOutput(heatmap_id, title = NULL,  
  width = 400, height = 350, containment = FALSE, internal = FALSE)
```

**Arguments**

heatmap_id	ID of the plot.
title	Title of the sub-heatmap.
width	Width of the sub-heatmap.
height	Height of the sub-heatmap.
containment	Whether the resizing is restricted in a certain parent div? Value can be TRUE/FALSE or a JQuery selector.
internal	Internally used.

**See Also**

[originalHeatmapOutput](#).

**Examples**

```
# See examples on the help page of originalHeatmapOutput()
```

# Index

all\_column\_indices, 2  
all\_row\_indices, 3

brushOpts, 18, 23

DataFrame, 7, 26, 27  
DESeq, 13  
DESeqDataSet, 13

fluidPage, 18

getPositionFromBrush, 3, 4  
getPositionFromClick, 4, 4  
getPositionFromDblclick, 4, 5  
getPositionFromHover, 4, 5

HeatmapInfoOutput, 6, 6, 23  
ht\_shiny, 11  
HTML, 8  
htmlOutput, 6, 18  
htPositionsOnDevice, 7, 26, 27  
htShiny, 8, 12  
htShinyExample, 10, 11, 11

interactivate, 12  
interactivate.DESeqDataSet, 13  
interactivate.kde, 13  
interactivateDensity2D, 14  
InteractiveComplexHeatmapModal, 15  
InteractiveComplexHeatmapOutput, 8, 9,  
15, 16, 17, 19, 20, 22  
InteractiveComplexHeatmapWidget, 19  
is\_in\_sub\_heatmap, 21

kde, 14

makeInteractiveComplexHeatmap, 3–5, 16,  
18, 20, 21

observe, 16, 20, 25  
observeEvent, 16, 20, 25

originalHeatmapOutput, 6, 23, 28

plotOutput, 3–5

rand\_mat, 24  
record\_observation, 25  
renderPlot, 22  
results, 13

selectArea, 26  
selectPosition, 27  
shinyApp, 9  
subHeatmapOutput, 6, 23, 28

unit, 4–6, 26, 27