

# Package ‘RiboCrypt’

February 13, 2025

**Type** Package

**Title** Interactive visualization in genomics

**Version** 1.13.0

**License** MIT + file LICENSE

**Description** R Package for interactive visualization and browsing NGS data.

It contains a browser for both transcript and genomic coordinate view.

In addition a QC and general metaplots are included, among others differential translation plots and gene expression plots. The package is still under development.

**biocViews** Software, Sequencing, RiboSeq, RNASeq,

**Encoding** UTF-8

**LazyData** true

**BugReports** <https://github.com/m-swirski/RiboCrypt/issues>

**URL** <https://github.com/m-swirski/RiboCrypt>

**Depends** R (>= 3.6.0), ORFik (>= 1.13.12)

**Imports** bslib, BiocGenerics, BiocParallel, Biostrings, data.table, dplyr, GenomeInfoDb, GenomicFeatures, GenomicRanges, ggplot2, htmlwidgets, httr, IRanges, jsonlite, knitr, markdown, NGLVieweR, plotly, rlang, RCurl, shiny, shinycssloaders, shinyhelper, shinyjqui, stringr

**Suggests** testthat, rmarkdown, BiocStyle, BSgenome, BSgenome.Hsapiens.UCSC.hg19

**RoxygenNote** 7.2.3

**VignetteBuilder** knitr

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

**git\_branch** devel

**git\_last\_commit** 148e091

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.21

**Date/Publication** 2025-02-12

**Author** Michal Swirski [aut, cre, cph],  
 Haakon Tjeldnes [aut, ctb],  
 Kornel Labun [ctb]

**Maintainer** Michal Swirski <michal.swirski@uw.edu.pl>

## Contents

antisense . . . . .	2
createSeqPanelPattern . . . . .	3
DEG_plot . . . . .	3
distanceToFollowing . . . . .	5
fetch_JS_seq . . . . .	5
fetch_summary . . . . .	6
geneTrackLayer . . . . .	6
getCoverageProfile . . . . .	7
getIndexes . . . . .	7
ggplotlyHover . . . . .	8
matchMultiplePatterns . . . . .	8
matchToGRanges . . . . .	9
multiOmicsPlot_animate . . . . .	9
multiOmicsPlot_list . . . . .	12
multiOmicsPlot_ORFikExp . . . . .	15
organism_input_select . . . . .	18
RiboCrypt_app . . . . .	18
trimOverlaps . . . . .	19

<b>Index</b>	<b>21</b>
--------------	-----------

---

antisense	<i>Get antisense</i>
-----------	----------------------

---

### Description

Get antisense

### Usage

```
antisense(grl)
```

### Value

a GRangesList

---

createSeqPanelPattern *Create sequence panel for RiboCrypt*

---

### Description

Create sequence panel for RiboCrypt

### Usage

```
createSeqPanelPattern(
  sequence,
  start_codons = "ATG",
  stop_codons = c("TAA", "TAG", "TGA"),
  frame = 1,
  custom_motif = NULL
)
```

### Arguments

start\_codons character vector, default "ATG"  
 stop\_codons character vector, default c("TAA", "TAG", "TGA")  
 custom\_motif character vector, default NULL.

### Value

a ggplot object

---

DEG\_plot *Differential expression plots (1D or 2D)*

---

### Description

Gives you interactive 1D or 2D DE plots

### Usage

```
DEG_plot(
  dt,
  draw_non_regulated = FALSE,
  xlim = ifelse(two_dimensions, "bidir.max", "auto"),
  ylim = "bidir.max",
  xlab = ifelse(two_dimensions, "RNA fold change (log2)", "Mean counts (log2)"),
  ylab = ifelse(two_dimensions, "RFP fold change (log2)", "Fold change (log2)"),
  two_dimensions = ifelse("LFC" %in% colnames(dt), FALSE, TRUE),
  color.values = c(`No change` = "black", Significant = "red", Buffering = "purple"),
```

```

`mRNA abundance` = "darkgreen", Expression = "blue", Forwarded = "yellow", Inverse =
  "aquamarine", Translation = "orange4")
)

```

## Arguments

<code>dt</code>	a data.table with results from a differential expression run. Normally from: <code>ORFik::DTEG.analysis(df1, df2)</code>
<code>draw_non_regulated</code>	logical, default FALSE. Should non-regulated rows be included in the plot? Will make the plot faster to render if skipped (FALSE)
<code>xlim</code>	numeric vector or character preset, default: <code>ifelse(two_dimensions, "bidir.max", "auto")</code> (Equal in both + / - direction, using max value + 0.5 of <code>meanCounts(in 1d) / rna(in 2d)</code> column of <code>dt</code> ). If you want ggplot to decide limit, set to "auto". For numeric vector, specify min and max x limit: like <code>c(-5, 5)</code>
<code>ylim</code>	numeric vector or character preset, default: "bidir.max" (Equal in both + / - direction, using max value + 0.5 of <code>LFC(in 1d) / rfp(in 2d)</code> column of <code>dt</code> ). If you want ggplot to decide limit, set to "auto". For numeric vector, specify min and max x limit: like <code>c(-5, 5)</code>
<code>xlab</code>	character, default: <code>ifelse(two_dimensions, "RNA fold change (log2)", "Mean counts (log2)")</code>
<code>ylab</code>	character, default: <code>ifelse(two_dimensions, "RFP fold change (log2)", "Fold change (log2)")</code>
<code>two_dimensions</code>	logical, default: <code>ifelse("LFC" %in% colnames(dt), FALSE, TRUE)</code> Is this two dimensional, like: Ribo-seq vs RNA-seq. Alternative, FALSE: Then Log fold change vs mean counts
<code>color.values</code>	named character vector, default: <code>c("No change" = "black", "Significant" = "red", "Buffering" = "purple", "mRNA abundance" = "darkgreen", "Expression" = "blue", "Forwarded" = "yellow", "Inverse" = "aquamarine", "Translation" = "orange4")</code>

## Value

plotly object

## Examples

```

# Load experiment
df <- ORFik.template.experiment()
# 1 Dimensional analysis
dt <- DEG.analysis(df[df$libtype == "RNA",])
dt$Regulation[1] <- "Significant" # Fake sig level
DEG_plot(dt, draw_non_regulated = TRUE)
# 2 Dimensional analysis
dt_2d <- DTEG.analysis(df[df$libtype == "RFP",], df[df$libtype == "RNA",],
  output.dir = NULL)
dt_2d$Regulation[4] <- "Translation" # Fake sig level
dt_2d$Regulation[5] <- "Buffering" # Fake sig level
DEG_plot(dt_2d, draw_non_regulated = TRUE)

```

---

distanceToFollowing     *Distance to following range*

---

**Description**

Distance to following range

**Usage**

```
distanceToFollowing(gr1, gr12 = gr1, ignore.strand = FALSE)
```

**Arguments**

gr1                    a GRangesList  
 gr12                  a GRangesList, default 'gr1'  
 ignore.strand        logical, default FALSE

**Value**

numeric vector of distance

---

fetch\_JS\_seq             *Fetch Javascript sequence*

---

**Description**

Fetch Javascript sequence

**Usage**

```
fetch_JS_seq(  
  target_seq,  
  nplots,  
  distance = 50,  
  display_dist,  
  aa_letter_code = "one_letter"  
)
```

**Arguments**

target\_seq            the target sequence  
 nplots                number of plots  
 distance              numeric, default 50.  
 display\_dist         display distance  
 aa\_letter\_code       "one\_letter"

**Value**

a list of 2 lists, the nt list (per frame, total 3) and AA list (per frame, total 3)

---

fetch_summary	<i>Fetch summary of uniprot id</i>
---------------	------------------------------------

---

**Description**

Fetch summary of uniprot id

**Usage**

```
fetch_summary(qualifier, provider = "alphafold")
```

**Arguments**

qualifier	uniprot ids
provider	"pdbe", alternatives: "alphafold", "all"

**Value**

a character of json

---

geneTrackLayer	<i>How many rows does the gene track need</i>
----------------	---

---

**Description**

How many rows does the gene track need

**Usage**

```
geneTrackLayer(gr1)
```

**Arguments**

gr1	a GRangesList
-----	---------------

**Value**

numeric, the track row index

---

`getCoverageProfile`      *Get coverage profile*

---

**Description**

Get coverage profile

**Usage**

```
getCoverageProfile(grl, reads, kmers = 1, kmers_type = "mean")
```

**Arguments**

<code>grl</code>	a GRangesList
<code>reads</code>	GRanges
<code>kmers</code>	1
<code>kmers_type</code>	"mean"

**Value**

data.table of coverage

---

`getIndex`      *Get index*

---

**Description**

Get index

**Usage**

```
getIndex(ref_granges)
```

**Arguments**

<code>ref_granges</code>	a GRanges object
--------------------------	------------------

**Value**

integer vector, indices

`ggplotlyHover`*Call ggplotly with hoveron defined*

---

**Description**

Call ggplotly with hoveron defined

**Usage**

```
ggplotlyHover(x, ...)
```

**Arguments**

`x` a a ggplot argument  
`...` additional arguments for ggplotly

**Value**

a ggplotly object

---

`matchMultiplePatterns` *Match multiple patterns*

---

**Description**

Match multiple patterns

**Usage**

```
matchMultiplePatterns(patterns, Seq)
```

**Arguments**

`patterns` character  
`Seq` a DNAStrngSet

**Value**

integer vector, indices (named with pattern hit)



---

matchToGRanges	<i>Match to GRanges</i>
----------------	-------------------------

---

**Description**

Match to GRanges

**Usage**

```
matchToGRanges(matches, ref_granges)
```

**Arguments**

matches	integer vector, indices
ref_granges	GRanges

**Value**

GRanges object

---

multiOmicsPlot_animate	<i>Multi-omics animation using list input</i>
------------------------	---

---

**Description**

The animation will move with a play button, there is 1 transition per library given.

**Usage**

```
multiOmicsPlot_animate(  
  display_range,  
  annotation = display_range,  
  reference_sequence,  
  reads,  
  viewMode = c("tx", "genomic")[1],  
  custom_regions = NULL,  
  leader_extension = 0,  
  trailer_extension = 0,  
  withFrames = NULL,  
  frames_type = "lines",  
  colors = NULL,  
  kmers = NULL,  
  kmers_type = c("mean", "sum")[1],  
  ylabels = NULL,
```

```

lib_to_annotation_proportions = c(0.8, 0.2),
lib_proportions = NULL,
annotation_proportions = NULL,
width = NULL,
height = NULL,
plot_name = "default",
plot_title = NULL,
display_sequence = c("both", "nt", "aa", "none")[1],
seq_render_dist = 100,
aa_letter_code = c("one_letter", "three_letters")[1],
annotation_names = NULL,
start_codons = "ATG",
stop_codons = c("TAA", "TAG", "TGA"),
custom_motif = NULL,
BPPARAM = BiocParallel::SerialParam()
)

```

### Arguments

display_range	the whole region to visualize, a <a href="#">GRangesList</a> or <a href="#">GRanges</a> object
annotation	the whole annotation which your target region is a subset, a <a href="#">GRangesList</a> or <a href="#">GRanges</a> object
reference_sequence	the genome reference, a <a href="#">FaFile</a> or <a href="#">FaFile</a> convertible object
reads	the NGS libraries, as a list of <a href="#">GRanges</a> with or without score column for replicates.
viewMode	character, default "tx" (transcript coordinates, first position is 1, exons are merged into a single sequence) Alternative: "genomic" (genomic coordinates, first position is first position in display_range argument. Introns are displayed).
custom_regions	a <a href="#">GRangesList</a> or NULL, default: NULL. The alternative annotation, like self defined uORFs etc. The vertical annotation bars will have a different color.
leader_extension	integer, default 0. (How much to extend view upstream)
trailer_extension	integer, default 0. (How much to extend view downstream)
withFrames	a logical vector, default NULL. Alternative: a length 1 or same length as list length of "reads" argument.
frames_type	character, default "lines". Alternative: - columns - stacks - area
colors	character, default NULL (automatic colouring). If "withFrames" argument is TRUE, colors are set to to c("red", "green", "blue") for the 3 frames. Alternative: Character vector of length 1 or length of "reads" list argument.
kmers	numeric (integer), bin positions into kmers.

kmers_type	character, function used for kmers sliding window. default: "mean", alternative: "sum"
ylabels	character, default NULL. Name of libraries in "reads" list argument.
lib_to_annotation_proportions	numeric vector of length 2. relative sizes of profiles and annotation.
lib_proportions	numeric vector of length equal to displayed libs. Relative sizes of profiles displayed
annotation_proportions	numeric vector of length 3 (seq displayed), or 2 (seq not displayed). Relative sizes of annotation tracks.
width	numeric, default NULL. Width of plot.
height	numeric, default NULL. Height of plot.
plot_name	= character, default "default" (will create name from display_range name). Alternative: custom name for region.
plot_title	character, default NULL. A title for plot.
display_sequence	character/logical, default c("both", "nt", "aa", "none")[1]. If TRUE or "both", display nucleotide and aa sequence in plot.
seq_render_dist	integer, default 100. The sequences will appear after zooming below this threshold.
aa_letter_code	character, when set to "three_letters", three letter amino acid code is used. One letter by default.
annotation_names	character, default NULL. Alternative naming for annotation.
start_codons	character vector, default "ATG"
stop_codons	character vector, default c("TAA", "TAG", "TGA")
custom_motif	character vector, default NULL.
BPPARAM	how many cores/threads to use? default: BiocParallel::SerialParam(). To see number of threads used for multicores, do BiocParallel::bpparam()\$workers. You can also add a time remaining bar, for a more detailed pipeline.

**Value**

the plot object

**Examples**

```
library(RiboCrypt)
df <- ORFik.template.experiment()[9:10,]
cds <- loadRegion(df, "cds")
mrna <- loadRegion(df, "mrna")
# multiOmicsPlot_animate(mrna[1], annotation = cds[1], reference_sequence = findFa(df),
#                          frames_type = "columns", leader_extension = 30, trailer_extension = 30,
#                          reads = outputLibs(df, type = "pshifted", output.mode = "envirlist",
#                          naming = "full", BPPARAM = BiocParallel::SerialParam()))
```

---

multiOmicsPlot\_list    *Multi-omics plot using list input*

---

## Description

Customizable html plots for visualizing genomic data.

## Usage

```
multiOmicsPlot_list(
  display_range,
  annotation = display_range,
  reference_sequence,
  reads,
  viewMode = c("tx", "genomic")[1],
  custom_regions = NULL,
  leader_extension = 0,
  trailer_extension = 0,
  withFrames = NULL,
  frames_type = "lines",
  colors = NULL,
  kmers = NULL,
  kmers_type = c("mean", "sum")[1],
  ylabels = NULL,
  lib_to_annotation_proportions = c(0.8, 0.2),
  lib_proportions = NULL,
  annotation_proportions = NULL,
  width = NULL,
  height = NULL,
  plot_name = "default",
  plot_title = NULL,
  display_sequence = c("both", "nt", "aa", "none")[1],
  seq_render_dist = 100,
  aa_letter_code = c("one_letter", "three_letters")[1],
  annotation_names = NULL,
  start_codons = "ATG",
  stop_codons = c("TAA", "TAG", "TGA"),
  custom_motif = NULL,
  AA_code = Biostrings::GENETIC_CODE,
  BPPARAM = BiocParallel::SerialParam(),
  summary_track = FALSE,
  summary_track_type = frames_type,
  export.format = "svg"
)
```

## Arguments

`display_range`    the whole region to visualize, a [GRangesList](#) or [GRanges](#) object

annotation	the whole annotation which your target region is a subset, a <a href="#">GRangesList</a> or <a href="#">GRanges</a> object
reference_sequence	the genome reference, a <a href="#">FaFile</a> or <a href="#">FaFile</a> convertible object
reads	the NGS libraries, as a list of <a href="#">GRanges</a> with or without score column for replicates.
viewMode	character, default "tx" (transcript coordinates, first position is 1, exons are merged into a single sequence) Alternative: "genomic" (genomic coordinates, first position is first position in display_range argument. Introns are displayed).
custom_regions	a <a href="#">GRangesList</a> or NULL, default: NULL. The alternative annotation, like self defined uORFs etc. The vertical annotation bars will have a different color.
leader_extension	integer, default 0. (How much to extend view upstream)
trailer_extension	integer, default 0. (How much to extend view downstream)
withFrames	a logical vector, default NULL. Alternative: a length 1 or same length as list length of "reads" argument.
frames_type	character, default "lines". Alternative: - columns - stacks - area
colors	character, default NULL (automatic colouring). If "withFrames" argument is TRUE, colors are set to c("red", "green", "blue") for the 3 frames. Alternative: Character vector of length 1 or length of "reads" list argument.
kmers	numeric (integer), bin positions into kmers.
kmers_type	character, function used for kmers sliding window. default: "mean", alternative: "sum"
ylabels	character, default NULL. Name of libraries in "reads" list argument.
lib_to_annotation_proportions	numeric vector of length 2. relative sizes of profiles and annotation.
lib_proportions	numeric vector of length equal to displayed libs. Relative sizes of profiles displayed
annotation_proportions	numeric vector of length 3 (seq displayed), or 2 (seq not displayed). Relative sizes of annotation tracks.
width	numeric, default NULL. Width of plot.
height	numeric, default NULL. Height of plot.
plot_name	= character, default "default" (will create name from display_range name). Alternative: custom name for region.
plot_title	character, default NULL. A title for plot.

display_sequence	character/logical, default c("both", "nt", "aa", "none")[1]. If TRUE or "both", display nucleotide and aa sequence in plot.
seq_render_dist	integer, default 100. The sequences will appear after zooming below this threshold.
aa_letter_code	character, when set to "three_letters", three letter amino acid code is used. One letter by default.
annotation_names	character, default NULL. Alternative naming for annotation.
start_codons	character vector, default "ATG"
stop_codons	character vector, default c("TAA", "TAG", "TGA")
custom_motif	character vector, default NULL.
AA_code	Genetic code for amino acid display. Default is SGC0 (standard: Vertebrate). See Biostrings::GENETIC_CODE_TABLE for options. To change to bacterial, do: Biostrings::getGeneticCode("11")
BPPARAM	how many cores/threads to use? default: BiocParallel::SerialParam(). To see number of threads used for multicores, do BiocParallel::bpparam()\$workers. You can also add a time remaining bar, for a more detailed pipeline.
summary_track	logical, default FALSE. Display a top track, that is the sum of all tracks.
summary_track_type	character, default is same as 'frames_type' argument
export.format	character, default: "svg". alternative: "png". when you click the top right image button export, what should it export as?

## Value

the plot object

## Examples

```
library(RiboCrypt)
df <- ORFik.template.experiment()[9:10,]
cds <- loadRegion(df, "cds")
mrna <- loadRegion(df, "mrna")
multiOmicsPlot_list(mrna[1], annotation = cds[1], reference_sequence = findFa(df),
  frames_type = "columns", leader_extension = 30, trailer_extension = 30,
  reads = outputLibs(df, type = "pshifted", output.mode = "envirlist",
    naming = "full", BPPARAM = BiocParallel::SerialParam()))
```

---

 multiOmicsPlot\_ORFikExp

*Multi-omics plot using ORFik experiment input*


---

## Description

Customizable html plots for visualizing genomic data.

## Usage

```
multiOmicsPlot_ORFikExp(
  display_range,
  df,
  annotation = "cds",
  reference_sequence = findFa(df),
  reads = outputLibs(df, type = "pshifted", output.mode = "envirlist", naming = "full",
    BPPARAM = BiocParallel::SerialParam()),
  viewMode = c("tx", "genomic")[1],
  custom_regions = NULL,
  leader_extension = 0,
  trailer_extension = 0,
  withFrames = libraryTypes(df, uniqueTypes = FALSE) %in% c("RFP", "RPF", "LSU"),
  frames_type = "lines",
  colors = NULL,
  kmers = NULL,
  kmers_type = c("mean", "sum")[1],
  ylabels = bamVarName(df),
  lib_to_annotation_proportions = c(0.8, 0.2),
  lib_proportions = NULL,
  annotation_proportions = NULL,
  width = NULL,
  height = NULL,
  plot_name = "default",
  plot_title = NULL,
  display_sequence = c("both", "nt", "aa", "none")[1],
  seq_render_dist = 100,
  aa_letter_code = c("one_letter", "three_letters")[1],
  annotation_names = NULL,
  start_codons = "ATG",
  stop_codons = c("TAA", "TAG", "TGA"),
  custom_motif = NULL,
  BPPARAM = BiocParallel::SerialParam(),
  input_id = "",
  summary_track = FALSE,
  summary_track_type = frames_type,
  export.format = "svg"
)
```

**Arguments**

display_range	the whole region to visualize, a <code>GRangesList</code> or <code>GRanges</code> object
df	an ORFik <code>experiment</code> or a list containing ORFik experiments. Usually a list when you have split Ribo-seq and RNA-seq etc.
annotation	the whole annotation which your target region is a subset, a <code>GRangesList</code> or <code>GRanges</code> object
reference_sequence	the genome reference, default <code>ORFik::findFa(df)</code>
reads	the NGS libraries, as a list of <code>GRanges</code> with or without 'score' column for replicates. Can also be a <code>covRle</code> object of precomputed coverage. Default: <code>outputLibs(df, type = "pshifted", output.mode = "envrlist", naming = "full", BPPARAM = BiocParallel::SerialParam())</code>
viewMode	character, default "tx" (transcript coordinates, first position is 1, exons are merged into a single sequence) Alternative: "genomic" (genomic coordinates, first position is first position in <code>display_range</code> argument. Introns are displayed).
custom_regions	a <code>GRangesList</code> or <code>NULL</code> , default: <code>NULL</code> . The alternative annotation, like self defined uORFs etc. The vertical annotation bars will have a different color.
leader_extension	integer, default 0. (How much to extend view upstream)
trailer_extension	integer, default 0. (How much to extend view downstream)
withFrames	a logical vector, default <code>libraryTypes(df, uniqueTypes = FALSE) %in% c("RFP", "RPF", "LSU")</code> Alternative: a length 1 or same length as list length of "reads" argument.
frames_type	character, default "lines". Alternative: - columns - stacks - area
colors	character, default <code>NULL</code> (automatic colouring). If "withFrames" argument is <code>TRUE</code> , colors are set to <code>c("red", "green", "blue")</code> for the 3 frames. Alternative: Character vector of length 1 or length of "reads" list argument.
kmers	numeric (integer), bin positions into kmers.
kmers_type	character, function used for kmers sliding window. default: "mean", alternative: "sum"
ylabels	character, default <code>bamVarName(df)</code> . Name of libraries in "reads" list argument.
lib_to_annotation_proportions	numeric vector of length 2. relative sizes of profiles and annotation.
lib_proportions	numeric vector of length equal to displayed libs. Relative sizes of profiles displayed
annotation_proportions	numeric vector of length 3 (seq displayed), or 2 (seq not displayed). Relative sizes of annotation tracks.



width	numeric, default NULL. Width of plot.
height	numeric, default NULL. Height of plot.
plot_name	character, default "default" (will create name from display_range name).
plot_title	character, default NULL. A title for plot.
display_sequence	character/logical, default c("both", "nt", "aa", "none")[1]. If TRUE or "both", display nucleotide and aa sequence in plot.
seq_render_dist	integer, default 100. The sequences will appear after zooming below this threshold.
aa_letter_code	character, when set to "three_letters", three letter amino acid code is used. One letter by default.
annotation_names	character, default NULL. Alternative naming for annotation.
start_codons	character vector, default "ATG"
stop_codons	character vector, default c("TAA", "TAG", "TGA")
custom_motif	character vector, default NULL.
BPPARAM	how many cores/threads to use? default: BiocParallel::SerialParam(). To see number of threads used for multicores, do BiocParallel::bpparam()\$workers. You can also add a time remaining bar, for a more detailed pipeline.
input_id	character path, default: "", id for shiny to display structures, should be "" for local users.
summary_track	logical, default FALSE. Display a top track, that is the sum of all tracks.
summary_track_type	character, default is same as 'frames_type' argument
export.format	character, default: "svg". alternative: "png". when you click the top right image button export, what should it export as?

**Value**

the plot object

**Examples**

```
library(RiboCrypt)
df <- ORFik.template.experiment()[9,] #Use third library in experiment only
cds <- loadRegion(df, "cds")
multiOmicsPlot_ORFikExp(extendLeaders(extendTrailers(cds[1], 30), 30), df = df,
                        frames_type = "columns")
```

---

organism\_input\_select *Select box for organism*

---

### Description

Select box for organism

### Usage

```
organism_input_select(genomes, ns)
```

### Arguments

genomes	name of genomes, returned from list.experiments()
ns	the ID, for shiny session

### Value

selectizeInput object

---

RiboCrypt\_app *Create RiboCrypt app*

---

### Description

Create RiboCrypt app

### Usage

```
RiboCrypt_app(
  validate.experiments = TRUE,
  options = list(launch.browser = ifelse(interactive(), TRUE, FALSE)),
  all_exp = list.experiments(validate = validate.experiments),
  browser_options = c(),
  init_tab_focus = "browser"
)
```

### Arguments

validate.experiments	logical, default TRUE, set to FALSE to allow starting the app with malformed experiments, be careful will crash if you try to load that experiment!
options	list of arguments, default list("launch.browser" = ifelse(interactive(), TRUE, FALSE))

`all_exp` a data.table, default: `list.experiments(validate = validate.experiments)`. Which experiments do you want to allow your app to see, default is all in your system config path.

`browser_options` named character vector of browser specific arguments:

- `default_experiment` : Which experiment to select, default: first one
- `default_gene` : Which genes to select, default: first one
- `default_libs` : Which libraries to select: first one, else a single string, where libs are separated by "|", like "RFP\_WT\_r1|RFP\_WT\_r2".
- `default_kmer` : K-mer windowing size, default: 1
- `default_frame_type` : Ribo-seq line type, default: "lines"
- `plot_on_start` : Plot when starting, default: "FALSE"

`init_tab_focus` character, default "browser". Which tab to open on init.

**Value**

RiboCrypt shiny app

**Examples**

```
## Default run
# RiboCrypt_app()
## Plot on start
# RiboCrypt_app(browser_options = c(plot_on_start = "TRUE"))
## Init with an experiment and gene (you must of course have the experiment)

#RiboCrypt_app(validate.experiments = FALSE,
#               browser_options = c(plot_on_start = "TRUE",
#                                   default_experiment = "human_all_merged_150",
#                                   default_gene = "ATF4-ENSG00000128272"))
```

---

<code>trimOverlaps</code>	<i>Trim overlaps</i>
---------------------------	----------------------

---

**Description**

Trim overlaps

**Usage**

```
trimOverlaps(overlaps, display_range)
```

**Arguments**

<code>overlaps</code>	GRanges
<code>display_range</code>	GRanges

**Value**

GRanges

# Index

## \* **internal**

- antisense, [2](#)
- createSeqPanelPattern, [3](#)
- geneTrackLayer, [6](#)
- getCoverageProfile, [7](#)
- getIndexes, [7](#)
- ggplotlyHover, [8](#)
- matchMultiplePatterns, [8](#)
- matchToGRanges, [9](#)
- trimOverlaps, [19](#)

antisense, [2](#)

createSeqPanelPattern, [3](#)

DEG\_plot, [3](#)

distanceToFollowing, [5](#)

experiment, [16](#)

FaFile, [10](#), [13](#)

fetch\_JS\_seq, [5](#)

fetch\_summary, [6](#)

geneTrackLayer, [6](#)

getCoverageProfile, [7](#)

getIndexes, [7](#)

ggplotlyHover, [8](#)

GRanges, [10](#), [12](#), [13](#), [16](#)

GRangesList, [10](#), [12](#), [13](#), [16](#)

matchMultiplePatterns, [8](#)

matchToGRanges, [9](#)

multiOmicPlot\_animate, [9](#)

multiOmicPlot\_list, [12](#)

multiOmicPlot\_ORFikExp, [15](#)

organism\_input\_select, [18](#)

RiboCrypt\_app, [18](#)

trimOverlaps, [19](#)