Package: qrlabelr (via r-universe)

June 22, 2026

Type Package

Title Generate Machine- And Human-Readable Plot Labels for Experiments

Version 0.2.1

Maintainer Alexander Wireko Kena <alex.kena24@gmail.com>

Description A no-frills open-source solution for designing plot labels affixed with QR codes. It features 'EasyQrlabelr', a 'BrAPI-compliant' 'shiny' app that simplifies the process of plot label design for non-R users. It builds on the methods described by Wu 'et al.' (2020) <doi:10.1111/2041-210X.13405>.

License GPL (>= 3)

BugReports https://github.com/awkena/qrlabelr/issues

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Imports argonDash (>= 0.2.0), argonR (>= 0.2.0), assertthat (>= 0.2.1), bslib (>= 0.4.2), desplot (>= 1.10), dplyr (>= 1.0.10), ggplot2 (>= 3.4.2), grid, purrr (>= 1.0.1), QBMS (>= 0.9.1), qrencoder (>= 0.1.0), raster (>= 3.6.23), reactable (>= 0.4.3), readxl (>= 1.4.1), shiny, shinycssloaders (>= 1.0.0), shinyjs (>= 2.1.0), shinyWidgets (>= 0.7.6), tools, utils, uuid

Suggests covr, knitr, rlang (>= 1.1.1), rmarkdown, shinytest2 (>= 0.2.1), testthat (>= 3.0.0), vdiffr (>= 1.0.5)

VignetteBuilder knitr

Depends R (>= 4.2.0), shinyBS (>= 0.61.1)

Config/testthat/edition 3

Language en-US

Config/pak/sysreqs cmake libgdal-dev gdal-bin libgeos-dev make libicu-dev libpng-dev libuv1-dev libnetcdf-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev zlib1g-dev

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

Date/Publication 2026-02-08 21:22:26 UTC

RemoteUrl https://github.com/awkena/qrlabelr

RemoteRef HEAD

RemoteSha daf2894db7f9c8d624747baebd1615eb1e4bec50

Contents

`add_border` *Make an enhanced field layout plot with border rows.*

Description

A helper function that adds border rows to the entire perimeter of a field laid out in a rectangular or square grid. Each experimental plot must have a coordinate that is specified by row and column numbers in the grid layout.

Usage

```
add_border(
  x,
  row_id = "ROW",
  col_id = "COLUMN",
  rep_id = "REP",
  trt_id = "TREATMENT",
  title = "Field layout",
  text_sz = 3,
  axis_title_sz = 12,
  xlab = "Column",
  ylab = "Row",
  border_bg = "grey80",
  text_col = "grey10"
)
```

Arguments

x	The input data frame of field book that has row and column coordinates of each plot.
row_id	The string column identifier for ROW in the input field book.
col_id	The string column identifier for COLUMN in the input field book.
rep_id	The string column identifier for REP in the input field book.
trt_id	The string column identifier for TREATMENT in the input field book.
title	The title of the field layout plot.
text_sz	The text size to print treatment names on the tiles.
axis_title_sz	The text size for axis titles.
xlab	A string to label x axis; default is 'Column'.
ylab	A string to label y axis; default is 'Row'.
border_bg	A string specifying the background color for the border rows.
text_col	A string specifying the text color for the border rows.

Value

A 'ggplot2' graphical object of field layout with border rows around the entire perimeter.

Note

This function works best with input field books generated with the 'FielDHub' package

Examples

```
# Plot a field layout with border rows
library(qrlabelr)

set.seed(123)

add_border(x = data.frame(LOCATION = rep("BAMBEY", 12),
                          PLOT = c(1001:1012),
                          ROW = c(rep(1, 6), rep(2, 6)),
                          COLUMN = c(1:6, 1:6),
                          REP = rep(1, 12),
                          TREATMENT = sample(paste0("G-", 1:12))),
text_sz = 3)
```

`create_label` *Create custom machine- and human-readable rectangular plot labels*

Description

Create print-ready customized plot labels affixed with QR codes given the page setup, label dimensions, the number of rows and columns of labels to print per page.

Usage

```
create_label(
  wdt = 2,
  hgt = 1,
  page_wdt = 8.5,
  page_hgt = 11,
  top_mar = 0.625,
  bot_mar = 0.625,
  left_mar = 0.625,
  right_mar = 0.625,
  numrow = 8L,
  numcol = 3L,
  filename = "PlotLabel",
  font_sz = 8,
  Treetag = FALSE,
  family = "sans",
  rounded = TRUE,
  print_across = TRUE,
  rect = TRUE,
  top_left_1 = NULL,
  top_left_2 = NULL,
  top_right_1 = NULL,
  top_right_2 = NULL,
  center_right_1 = NULL,
  center_right_2 = NULL,
  center_right_3 = NULL,
  bottom_left_1 = NULL,
  bottom_left_2 = NULL,
  unique_id = NULL,
  include_qr = TRUE,
  ec_level = 3,
  ...
)
```

Arguments

`wdt`	The label width in inches.
`hgt`	The label height in inches.

page_wdt	The page width in inches.
page_hgt	The page height in inches.
top_mar	The page top margin in inches.
bot_mar	The page bottom margin in inches.
left_mar	The page left margin in inches.
right_mar	The page right margin in inches.
numrow	The number of label rows per page. It should be an integer.
numcol	The number of label columns per page. It should be an integer.
filename	A character prefix or path for the pdf file to be created. Default path is working directory.
font_sz	The font size to use.
Treetag	Set to TRUE if creating a treetag label.
family	The font style to use to print labels.
rounded	Set to TRUE if label has round corners; set to false if label has square corners.
print_across	Set to TRUE to print labels across rows, left to right; else set to FALSE to print labels down columns, top to bottom. Default is TRUE.
rect	Set to TRUE to draw rectangles around labels, else set to FALSE. Default is TRUE.
top_left_1	String for top-left row 1 position on a rectangular label.
top_left_2	String for top-left row 2 position on a rectangular label.
top_right_1	String for top-right row 1 position on a rectangular label.
top_right_2	String for top-right row 2 position on a rectangular label.
center_right_1	String for center-right row 1 position on a rectangular label.
center_right_2	String for center-right row 2 position on a rectangular label.
center_right_3	String for center-right row 3 position on a rectangular label.
bottom_left_1	String for bottom-left row 1 position on a rectangular label.
bottom_left_2	String for bottom-left row 2 position on a rectangular label.
unique_id	A vector containing unique identifiers or strings to generate QR codes.
include_qr	Logical. Set to FALSE to exclude QR codes. Default is TRUE.
ec_level	The error correction level (‘0‘ - ‘3‘, lowest to highest) for QR codes.
...	Additional optional arguments to be supplied.

Value

A PDF file containing experimental plot labels affixed with QR codes, saved to the default or working directory.

See Also

field_label and gp_label

Examples

```
library(qrlabelr)
file <- tempfile()

if(file.exists(file))
# Create rectangular plot labels based on the Avery 94220 template-- the default template
create_label(
font_sz = 10,
filename = file,
print_across = TRUE,
rect = TRUE,
top_left_1 = paste("Plot:", 101:105),
top_left_2 = paste("Row:", c(rep(1, 3), rep(2, 2))),
top_right_1 = paste("Rep:", rep(1, 5)),
top_right_2 = paste("Col:", c(1:3, 1:2)),
center_right_1 = paste("iBLOCK:", c(rep(1, 3), rep(2, 2))),
center_right_2 = paste("Seed:", rep("OFF_NUR", 5)),
center_right_3 = rep("AWk", 5),
bottom_left_1 = paste("Loc:", rep("BAMBEY", 5)),
bottom_left_2 = paste0("G-", 1:5),
unique_id = paste("KUMASI2023_PYT", c(101:105),
                  c(rep(1, 3), rep(2, 2)), c(1:3, 1:2),
                  sep = "_"),
ec_level = 1)
```

`field_label` *Create field plot labels embossed with QR codes*

Description

Create machine- and human-readable plot labels that are well-suited for field experiments.

Usage

```
field_label(
  dat,
  get_unique_id = c("ruid", "uuid", "custom"),
  unique_id = NULL,
  filename = "PlotLabel",
  Year = NULL,
  rname = NULL,
  Trial = "PYT",
  seed_source = FALSE,
  IBlock = FALSE,
  rep_id = "REP",
```

```
  plot_id = "PLOT",
  row_id = "ROW",
  col_id = "COLUMN",
  loc_id = "LOCATION",
  entry_id = "TREATMENT",
  IBlock_id = "IBLOCK",
  seed_source_id = NULL,
  ...
)
```

Arguments

dat	An input data frame of field book that contains plot attributes. To design field plot labels, the imported field book must have LOCATION, PLOT, ROW, COLUMN/RANGE, REP, TREATMENT columns. The order of the columns in the field book is not important, and the columns can be any name the user desires.
get_unique_id	Set to 'ruid' if reproducible and informative unique ids are to be generated from imported field book. Set to 'uuid' if universal unique ids are to be generated from imported field book. Set to 'custom' if imported field book already has unique IDs for each plot.
unique_id	The column identifier for UNIQUE_ID in the imported field book.
filename	A character prefix or path for the pdf file to be created. Default path is working directory.
Year	The year of experiment or trial.
rname	The researcher's name. Initials or initials of first and middle names and the last name.
Trial	The name of the trial to use.
seed_source	Set to TRUE if seed source is included in the imported field book, FALSE if otherwise.
IBlock	Set to TRUE if dat contains incomplete blocks within replications.
rep_id	The column identifier for REP in the imported field book.
plot_id	The column identifier for PLOT in the imported field book.
row_id	The column identifier for ROW in the imported field book.
col_id	The column identifier for COLUMN in the imported field book.
loc_id	The column identifier for LOCATION in the imported field book.
entry_id	The column identifier for ENTRY/TREATMENT in the imported field book.
IBlock_id	The column identifier for IBLOCK in the imported field book. It must be provided if IBlock is set to TRUE.
seed_source_id	The column identifier for SEED_SOURCE in the imported field book. It must be provided if seed_source is set to TRUE.
...	Additional arguments passed to the create_label() function.

Details

The default column identifiers for LOCATION, PLOT, ROW, COLUMN/RANGE, REP, TREATMENT are based on the column IDs of a field book generated using the 'FielDHub' package. If user imports any field book generated with other programs, the user must specify the equivalent column identifiers used for LOCATION, PLOT, REP, ROW, COLUMN, and TREATMENT/ENTRY in the imported field book.

if `get_unique_id = 'ruid'` (i.e. Reproducible Unique IDs), the function concatenates location, year, trial name, plot, row and column IDs. if `get_unique_id = 'uuid'` (i.e. Universal Unique IDs), the function generates UUIDs by time randomly. Note that UUIDs are uninformative and not reproducible.

If input field book already has unique IDs for each plot, the `get_unique_id` argument should be set to 'custom'; and the `unique_id` argument must be specified as a string using the column name in the input field book that denotes plot unique IDs.

if `Year` is NULL, the function uses the current year as defined in the 'sys.time()'.

If the user is printing labels for any incomplete block design, the imported field book must include an IBLOCK column if the users wishes to display intra-blocking information for experimental plots on the label.

Set the `IBlock` argument to TRUE if the field layout has incomplete blocks within replications. The imported field book must include an IBLOCK column if the `IBlock` argument is set to TRUE.

Value

A PDF file containing field plot labels affixed with QR codes, and a data frame of an updated field book; all saved to the default or working directory.

See Also

`create_label` and `gp_label`

Examples

```
library(qrlabelr)
df <- data.frame(LOCATION = rep("BAMBEY", 5),
                 PLOT = 1001:1005,
                 ROW = c(rep(1, 3), rep(2, 2)),
                 COLUMN = c(1:3, 1:2),
                 REP = rep(1, 5),
                 IBLOCK = c(rep(1, 3), rep(2, 2)),
                 TREATMENT = paste0("G-", 1:5),
                 SEED_SOURCE = rep("OFF_NUR", 5))
df$ids <- paste0(df$LOCATION,'2023', '_PYT', '_', df$PLOT, '_', df$ROW, '_',
                  df$COLUMN)
file <- tempfile()

if(file.exists(file))
field_label(
  dat = df,
  wdt = 5,
```

```
  hgt = 2,
  page_wdt = 8.5,
  page_hgt = 11,
  top_mar = 0.75,
  bot_mar = 0.75,
  left_mar = 1.75,
  right_mar = 1.75,
  numrow = 4L,
  numcol = 1L,
  filename = file,
  font_sz = 20,
  Trial = "PYT",
  Year = 2023,
  family = "sans",
  rounded = TRUE,
  IBlock = TRUE,
  get_unique_id = "ruid",
  rname = "AW Kena",
  seed_source = TRUE,
  seed_source_id = "SEED_SOURCE",
  ec_level = 1)
```

`gp_label` *Create a general-purpose (gp) label with text aligned in a landscape orientation.*

Description

This is a wrapper that gives more flexibility to the user to design any general-purpose label affixed with QR codes. It gives nine(9) text positions in landscape orientation that can be filled with human-readable text items as specified by the user. Arguments are passed to the ‘create_label()‘ function.

Usage

```
gp_label(
  dat,
  get_unique_id = c("uuid", "custom"),
  unique_id = NULL,
  filename = "PlotLabel",
  top_left_txt1 = NULL,
  top_left_txt2 = NULL,
  top_right_txt1 = NULL,
  top_right_txt2 = NULL,
  center_right_txt1 = NULL,
  center_right_txt2 = NULL,
  center_right_txt3 = NULL,
  bottom_left_txt1 = NULL,
```

```
  bottom_left_txt2 = NULL,
  top_left_id1 = NULL,
  top_left_id2 = NULL,
  top_right_id1 = NULL,
  top_right_id2 = NULL,
  center_right_id1 = NULL,
  center_right_id2 = NULL,
  center_right_id3 = NULL,
  bottom_left_id1 = NULL,
  bottom_left_id2 = NULL,
  ...
)
```

Arguments

dat	An input data frame or field book that contains plot or label attributes. The order of the columns is not important, and the columns can be any name the user desires.
get_unique_id	Set to 'uuid' if universal unique ids are to be generated.
unique_id	The column identifier in dat containing unique identifiers or strings to generate QR codes. Set to 'custom' if imported field book already has unique IDs for each plot.
filename	A character prefix or path for the pdf file to be created. Default path is working directory.
top_left_txt1	The prefix text for top-left row 1.
top_left_txt2	The prefix text for top-left row 2.
top_right_txt1	The prefix text for the top-right row 1.
top_right_txt2	The prefix text for the top-right row 2.
center_right_txt1	The prefix text for center-right row 1.
center_right_txt2	The prefix text for center-right row 2.
center_right_txt3	The prefix text for center-right row 3.
bottom_left_txt1	The column identifier in dat containing text for bottom-left row 1.
bottom_left_txt2	The column identifier in dat containing text for bottom-left row 2.
top_left_id1	The column identifier in dat containing text for top-left row 1.
top_left_id2	The column identifier in dat containing text for top-left row 2.
top_right_id1	The column identifier in dat containing text for top-right row 1.
top_right_id2	The column identifier in dat containing text for top-right row 2.
center_right_id1	The column identifier in dat containing text for center-right row 1.

center_right_id2
: The column identifier in dat containing text for center-right row 2.

center_right_id3
: The column identifier in dat containing text for center-right row 3.

bottom_left_id1
: The column identifier in dat containing text for bottom-left row 1.

bottom_left_id2
: The column identifier in dat containing text for bottom-left row 2.

...
: Additional arguments passed to the create_label function.

Value

A PDF file containing plot labels affixed with QR codes, and a data frame of an updated field book; all saved to the default or working directory.

See Also

create_label and field_label

Examples

```
library(qrlabelr)
df <- data.frame(LOCATION = rep("BAMBEY", 5),
                 PLOT = 1001:1005,
                 ROW = c(rep(1, 3), rep(2, 2)),
                 COLUMN = c(1:3, 1:2),
                 REP = rep(1, 5),
                 IBLOCK = c(rep(1, 3), rep(2, 2)),
                 TREATMENT = paste0("G-", 1:5),
                 SEED_SOURCE = rep("OFF_NUR", 5))

df$ids <- paste0(df$LOCATION,'2023', '_PYT', '_', df$PLOT, '_', df$ROW, '_',
                 df$COLUMN)
file <- tempfile()

if(file.exists(file))
gp_label(dat = df,
         wdt = 5,
         hgt = 2,
         page_wdt = 8.5,
         page_hgt = 11,
         top_mar = 0.75,
         bot_mar = 0.75,
         left_mar = 1.75,
         right_mar = 1.75,
         numrow = 4L,
         numcol = 1L,
         filename = file,
         font_sz = 20,
         rname = "Adoma",
         get_unique_id = "custom",
```

```
unique_id = 'ids',
family = "sans",
top_left_txt1 = 'Plot:',
top_left_txt2 = 'Row:',
top_right_txt1 = 'Rep:',
top_right_txt2 = 'Col:',
center_right_txt1 = 'iBlock:',
center_right_txt2 = 'Seed:',
center_right_txt3 = 'Adoma',
top_left_id1 = 'PLOT',
top_left_id2 = 'ROW',
top_right_id1 = 'REP',
top_right_id2 = 'COLUMN',
center_right_id1 = 'IBLOCK',
center_right_id2 = 'SEED_SOURCE',
bottom_left_id1 = 'ids',
bottom_left_id2 = 'TREATMENT',
ec_level = 1)
```

`gp_label_portrait` *Create a general-purpose (gp) label with text aligned in a portrait orientation.*

Description

This is a standalone function that gives more flexibility to the user to design any general-purpose label affixed with QR codes. It gives 10 text positions in portrait orientation that can be filled with human-readable text items as specified by the user.

This function creates print-ready customized plot labels affixed with QR codes given the page setup, label dimensions, the number of rows and columns of labels to print per page.

Usage

```
gp_label_portrait(
  dat,
  wdt = 2,
  hgt = 1,
  page_wdt = 8.5,
  page_hgt = 11,
  top_mar = 0.625,
  bot_mar = 0.625,
  left_mar = 0.625,
  right_mar = 0.625,
  numrow = 8L,
  numcol = 3L,
  filename = "PlotLabel",
```

```
  font_sz = 8,
  family = "sans",
  rounded = TRUE,
  print_across = TRUE,
  rect = TRUE,
  bot_txt1 = NULL,
  bot_txt2 = NULL,
  bot_txt3 = NULL,
  cent_txt1 = NULL,
  cent_txt2 = NULL,
  cent_txt3 = NULL,
  cent_txt4 = NULL,
  top_txt1 = NULL,
  top_txt2 = NULL,
  top_txt3 = NULL,
  bot_txt1_id = NULL,
  bot_txt2_id = NULL,
  bot_txt3_id = NULL,
  cent_txt1_id = NULL,
  cent_txt2_id = NULL,
  cent_txt3_id = NULL,
  cent_txt4_id = NULL,
  top_txt1_id = NULL,
  top_txt2_id = NULL,
  top_txt3_id = NULL,
  unique_id = NULL,
  include_qr = TRUE,
  ec_level = 3
)
```

Arguments

dat	An input data frame or field book that contains plot attributes. The order of the columns is not important, and the columns can be any name the user desires.
wdt	The label width in inches.
hgt	The label height in inches.
page_wdt	The page width in inches.
page_hgt	The page height in inches.
top_mar	The page top margin in inches.
bot_mar	The page bottom margin in inches.
left_mar	The page left margin in inches.
right_mar	The page right margin in inches.
numrow	The number of label rows per page. It should be an integer.
numcol	The number of label columns per page. It should be an integer.
filename	A character prefix or path for the pdf file to be created. Default path is working directory.

font_sz	The font size to use.
family	The font style to use to print labels.
rounded	Set to TRUE if label has round corners; set to false if label has square corners.
print_across	Set to TRUE to print labels across rows, left to right; else set to FALSE to print labels down columns, top to bottom. Default is TRUE.
rect	Set to TRUE to draw rectangles around labels, else set to FALSE. Default is TRUE.
bot_txt1	The prefix text for bottom text position 1.
bot_txt2	The prefix text for bottom text position 2.
bot_txt3	The prefix text for bottom text position 3.
cent_txt1	The prefix text for center text position 1.
cent_txt2	The prefix text for center text position 2.
cent_txt3	The prefix text for center text position 3.
cent_txt4	The prefix text for center text position 4.
top_txt1	The prefix text for top text position 1.
top_txt2	The prefix text for top text position 2.
top_txt3	The prefix text for top text position 3.
bot_txt1_id	The column identifier in dat containing text for bottom text position 1.
bot_txt2_id	The column identifier in dat containing text for bottom text position 2.
bot_txt3_id	The column identifier in dat containing text for bottom text position 3.
cent_txt1_id	The column identifier in dat containing text for center text position 1.
cent_txt2_id	The column identifier in dat containing text for center text position 2.
cent_txt3_id	The column identifier in dat containing text for center text position 3.
cent_txt4_id	The column identifier in dat containing text for center text position 4.
top_txt1_id	The column identifier in dat containing text for top text position 1.
top_txt2_id	The column identifier in dat containing text for top text position 2.
top_txt3_id	The column identifier in dat containing text for top text position 3.
unique_id	The column identifier in dat containing unique identifiers or strings to generate QR codes.
include_qr	Logical. Set to FALSE to exclude QR codes. Default is TRUE.
ec_level	The error correction level (‘0‘ - ‘3‘, lowest to highest) for QR codes.

Value

A PDF file containing labels affixed with QR codes, saved to the default or working directory.

See Also

field_label and gp_label

Examples

```
# Create a general-purpose label in a portrait text orientation based on the
# 2 x 1 inch Avery 94220 template for laser-jet printers
library(qrlabelr)
df <- data.frame(LOCATION = rep("BAMBEY", 5),
                 PLOT = 1001:1005,
                 ROW = c(rep(1, 3), rep(2, 2)),
                 COLUMN = c(1:3, 1:2),
                 REP = rep(1, 5),
                 IBLOCK = c(rep(1, 3), rep(2, 2)),
                 TREATMENT = paste0("G-", 1:5),
                 SEED_SOURCE = rep("OFF_NUR", 5))

df$ids <- paste0(df$LOCATION,'2023', '_PYT', '_', df$PLOT, '_', df$ROW, '_',
                 df$COLUMN)
file <- tempfile()

if(file.exists(file))
gp_label_portrait(
dat = df,
wdt = 2,
hgt = 1,
page_wdt = 8.5,
page_hgt = 11,
top_mar = 0.625,
bot_mar = 0.625,
left_mar = 0.625,
right_mar = 0.625,
numrow = 8L,
numcol = 3L,
filename = file,
font_sz = 10,
family = 'sans',
rounded = TRUE,
print_across = TRUE,
rect = TRUE,
bot_txt1 = 'Rubi',
cent_txt2 = 'Rep:',
cent_txt3 = 'R:',
cent_txt4 = 'r:',
top_txt1 = 'P:',
top_txt2 = 'B:',
bot_txt2_id = 'ids',
bot_txt3_id = 'LOCATION',
cent_txt1_id = 'TREATMENT',
cent_txt2_id = 'REP',
cent_txt3_id = 'COLUMN',
cent_txt4_id = 'ROW',
top_txt1_id = 'PLOT',
top_txt2_id = 'IBLOCK',
top_txt3_id = 'SEED_SOURCE',
unique_id = 'ids',
```

```
ec_level = 1)
```

`make_qrcode` *Make QR codes*

Description

A helper function for QR code generation using the 'qrencoder' library for faster QR code generation. It converts the generated QR code into a raster grob image that can be plotted using the 'grid.draw()' function in the 'grid' package.

Usage

```
make_qrcode(my_id, ec_level = 3)
```

Arguments

`my_id`	Unique ID string to be encoded to QR code.
`ec_level`	The error correction level ('0' - '3', lowest to highest).

Value

A QR code as a raster grob image object that can be plotted with the 'grid.draw()' function in the 'grid' package.

Examples

```
library(qrlabelr)
qr <- make_qrcode("KUMASI2023_PYT_101_1_1", ec_level = 1)

# Plot QR code using the grid package
grid::pushViewport(grid::viewport(x = grid::unit(0.5, "npc"),
                                  y = grid::unit(0.5, "npc"),
                                  width = grid::unit(1, "in"),
                                  height = grid::unit(1, "in")))
grid::grid.draw(qr)
# clean up any open graphical device
# grDevices::graphics.off()
```

qrlabelr *Generate Machine- And Human-Readable Plot Labels for Experiments. "_PACKAGE"*

Description

A no-frills open-source solution for designing experimental or trial plot labels affixed with QR codes. 'qrlabelr' is an R package that features 'EasyQrlabelr', a Shiny app that simplifies the complicated process of plot label design for non-R users. It also offers easily 'customizable' functions that enable plot label generation outside the Shiny app. It generates field plot labels that are compatible with the widely used digital data collection mobile app, Field Book. Our software builds on the foundation of an existing open-source program to offer more flexibility in plot label creation steps; guarantees true string fidelity after QR encoding; and provides faster label generation to users..

Available vignettes

This package comes with one vignette to get users up to speed as soon as possible. It provides a more thorough guide on how to use qrlabelr, from the data import stage to the generation of labels and saving it for printing. To access the vignette, run the command: `browseVignettes("qrlabelr")`

Author(s)

Alexander Kena | Ebenezer Ogoe

rcbd *rcbd*

Description

A sample field book generated with the 'FielDHub' package based on a Randomized Complete Block Design layout at two locations.

Usage

```
data(rcbd)
```

Format

A data frame with 144 observations and 7 variables:

`SN double` contains observation serial numbers.

`LOCATION character` contains information on plot location.

`PLOT double` contains the plot number.

`ROW double` contains the row number.

`COLUMN double` contains the column number.

`REP double` contains the replication number.

`TREATMENT character` contains the treatment identifier.

run_app *Launch 'EasyQrlabelr'.*

Description

An interface function to launch the 'EasyQrlabelr' 'shiny' app.

Usage

```
run_app()
```

Value

No return value, called for side effects. Opens the 'EasyQrlabelr' 'shiny' app in your default browser.

Examples

```
library(qrlabelr)
if(interactive())
qrlabelr::run_app()
```

square_lattice *Square Lattice*

Description

A sample data set to create field plot and general_purpose labels.

Usage

```
data(square_lattice)
```

Format

A data frame with 216 observations and 11 variables:

`SN double` contains observation serial numbers.

`LOCATION character` contains information on plot location.

`PLOT double` contains the plot number.

`ROW double` contains the row number.

`COLUMN double` contains the column number.

`REP double` contains the replication number.

`IBLOCK double` contains the incomplete block number.

UNIT `double` contains the unit number.

ENTRY `double` contains the entry number.

TREATMENT `character` contains the treatment identifier.

SEED_SOURCE `character` contains the seed source identifier.

Index