

Package ‘qrcode’

November 11, 2023

Type Package

Title Generate QRcodes with R

Version 0.2.2

Description Create static QR codes in R. The content of the QR code is exactly what the user defines. We don't add a redirect URL, making it impossible for us to track the usage of the QR code. This allows to generate fast, free to use and privacy friendly QR codes.

License GPL-3

URL <https://thierryo.github.io/qrcode/>,
<https://github.com/Thierry0/qrcode>,
<https://doi.org/10.5281/zenodo.5040088>

BugReports <https://github.com/Thierry0/qrcode/issues>

Depends R (>= 4.1.0)

Imports assertthat, stats, utils

Suggests httr, jpeg, knitr, opencv, png, rsvg, testthat (>= 3.0.0)

Config/checklist/keywords two-dimensional barcode; matrix barcode

Config/testthat/edition 3

Encoding UTF-8

Language en-GB

RoxygenNote 7.2.3

NeedsCompilation no

Author Thierry Onkelinx [aut, cre] (Author of the reimplemented functions,
<<https://orcid.org/0000-0001-8804-4216>>),
Victor Teh [aut] (Original author)

Maintainer Thierry Onkelinx <qrcode@muscardinus.be>

Repository CRAN

Date/Publication 2023-11-11 10:53:21 UTC

R topics documented:

add_logo	2
as.character.bits	3
bits	4
bits2int	4
c.bits	5
coordinates	6
generate_svg	7
plot.qr_code	9
print.bits	10
print.qr_code	10
qr_code	11
qr_event	12
qr_wifi	13

Index	14
--------------	-----------

add_logo	<i>Add a logo to a QR code</i>
----------	--------------------------------

Description

First generate a qr_code with a higher ecl level. Then add the logo. The maximum area of logo depends on the difference in ecl level between the version with and without logo. The size of the logo is further restricted by its image ratio. We shrink very wide or tall logos to make sure it still fits on the logo.

Usage

```
add_logo(
  code,
  logo,
  ecl = c("L", "M", "Q", "H"),
  hjust = c("c", "l", "r"),
  vjust = c("c", "b", "t")
)
```

Arguments

code	A qr_code object
logo	the path to a logo image file. Must be either png, svg or jpeg format.
ecl	the required error correction level for the QR code after overlaying the logo. Must be lower than the ecl in the code. Defaults to "L". The difference between the ecl set here and the ecl in code determines the maximum area of the logo. For the largest logo, generate code with ecl = "H" and add the logo with ecl = "L".

hjust	Horizontal position of the logo. The default of "c" indicates the centre of the QR code. Use "r" to align the right side of the logo with the right side of the QR code. Use "l" to align the left side of the logo with the right side of the two vertical finder patterns.
vjust	Vertical position of the logo. The default of "c" indicates the centre of the QR code.. Use "b" to align the bottom of the logo with the bottom of the QR code. Use "t" to align the top of the logo with the bottom side of the two horizontal finder patterns.

as.character.bits *Convert a bits object into a character string*

Description

Convert a bits object into a character string

Usage

```
## S3 method for class 'bits'  
as.character(x, ...)
```

Arguments

x	the bits object
...	currently ignore

Author(s)

Thierry Onkelinx

See Also

Other bits: [bits2int\(\)](#), [bits\(\)](#), [c.bits\(\)](#), [print.bits\(\)](#)

Examples

```
z <- bits(c(FALSE, TRUE, TRUE, FALSE))  
z  
as.character(z)
```

bits *Create a bits object*

Description

Converts a logical vector into a bits object. This remains a logical vector. The main difference is that is printed as a 0 and 1 bit string rather than a FALSE and TRUE vector

Usage

```
bits(x)
```

Arguments

x a logical vector

Author(s)

Thierry Onkelinx

See Also

Other bits: [as.character.bits\(\)](#), [bits2int\(\)](#), [c.bits\(\)](#), [print.bits\(\)](#)

Examples

```
z <- bits(c(FALSE, TRUE))
z
str(z)
```

bits2int *Convert a bits object to an integer and vice versa*

Description

Convert a bits object to an integer and vice versa

Usage

```
bits2int(x)
```

```
int2bits(i, n_bit = 16)
```

Arguments

x the bits object
i the integer
n_bit the number of bits

Author(s)

Thierry Onkelinx

See Also

Other bits: [as.character.bits\(\)](#), [bits\(\)](#), [c.bits\(\)](#), [print.bits\(\)](#)

Examples

```
z <- bits(c(FALSE, TRUE, TRUE, FALSE))
z
y <- bits2int(z)
y
int2bits(y)
int2bits(y, 4)
```

c.bits

Combine bits

Description

The result inherits arguments from the first element.

Usage

```
## S3 method for class 'bits'
c(...)
```

Arguments

... the bits to concatenate

Author(s)

Thierry Onkelinx

See Also

Other bits: [as.character.bits\(\)](#), [bits2int\(\)](#), [bits\(\)](#), [print.bits\(\)](#)

Examples

```
z <- bits(c(FALSE, TRUE))
z
c(z, z, rev(z))
```

coordinates

Extract coordinates from a QR code object.

Description

Selects the dark elements from the `qr_code` object and returns their coordinates. This can be useful when you want to create a QR code with a custom style.

Usage

```
coordinates(x)
```

Arguments

`x` the `qr_code` object.

Value

A data.frame with the column and row number of the dark elements.

Author(s)

Thierry Onkelinx

See Also

Other qr: [generate_svg\(\)](#), [plot.qr_code\(\)](#), [print.qr_code\(\)](#), [qr_code\(\)](#), [qr_event\(\)](#), [qr_wifi\(\)](#)

Examples

```
x <- qr_code("test")
plot(x)
head(coordinates(x))
plot(coordinates(x), pch = 19, cex = 2, asp = 1)
```

generate_svg	<i>Generate the QR code as an svg file</i>
--------------	--

Description

Create the QR code using `qr_code()` and save it as an svg file.

Usage

```
generate_svg(  
  qrcode,  
  filename,  
  size = 300,  
  foreground = "black",  
  background = "white",  
  show = interactive(),  
  ...  
)  
  
## Default S3 method:  
generate_svg(  
  qrcode,  
  filename,  
  size = 300,  
  foreground = "black",  
  background = "white",  
  show = interactive(),  
  ...  
)  
  
## S3 method for class 'qr_code'  
generate_svg(  
  qrcode,  
  filename,  
  size = 300,  
  foreground = "black",  
  background = "white",  
  show = interactive(),  
  ...  
)  
  
## S3 method for class 'qr_wifi'  
generate_svg(  
  qrcode,  
  filename,  
  size = 300,  
  foreground = "black",
```

```

    background = "white",
    show = interactive(),
    ...,
    fontsize = 15
)

## S3 method for class 'qr_logo'
generate_svg(
  qrcode,
  filename,
  size = 300,
  foreground = "black",
  background = "white",
  show = interactive(),
  ...
)

```

Arguments

qrcode	a <code>qr_code</code> object as generated by <code>qr_code</code> .
filename	Where to store the QR code as svg file. Silently overwrites existing files. Tries to create the path, when it doesn't exist.
size	width of the svg file in pixels. Defaults to 300.
foreground	Stroke and fill colour for the foreground. Use a valid CSS colour . Defaults to "black".
background	Fill colour for the background. Use a valid CSS colour . Defaults to "white".
show	Open the file after creating it. Defaults to TRUE on <code>interactive()</code> sessions, otherwise FALSE.
...	Currently ignored.
fontsize	The size of the font in pixels.

Value

invisible NULL

Author(s)

Thierry Onkelinx

See Also

Other qr: `coordinates()`, `plot.qr_code()`, `print.qr_code()`, `qr_code()`, `qr_event()`, `qr_wifi()`

Examples

```

code <- qr_code("HELLO WORLD")
generate_svg(
  qrcode = code, filename = tempfile(fileext = ".svg"), show = FALSE
)

```

plot.qr_code	<i>Plot the QR code This function plots to QR code to the open device.</i>
--------------	--

Description

Plot the QR code This function plots to QR code to the open device.

Usage

```
## S3 method for class 'qr_code'  
plot(x, col = c("white", "black"), y, ...)  
  
## S3 method for class 'qr_logo'  
plot(x, col = c("white", "black"), y, ...)
```

Arguments

x	the qr_code object
col	Define the colours. The first element refers to FALSE and the second TRUE. Defaults to c("white", "black").
y	currently ignored
...	currently ignored

Author(s)

Thierry Onkelinx

See Also

[opencv::ocv_qr_detect\(\)](#) for reading QR codes.

Other qr: [coordinates\(\)](#), [generate_svg\(\)](#), [print.qr_code\(\)](#), [qr_code\(\)](#), [qr_event\(\)](#), [qr_wifi\(\)](#)

Other qr: [coordinates\(\)](#), [generate_svg\(\)](#), [print.qr_code\(\)](#), [qr_code\(\)](#), [qr_event\(\)](#), [qr_wifi\(\)](#)

Examples

```
qr <- qr_code("HELLO WORLD")  
plot(qr)  
  
# Test the QR code with the opencv package  
if (requireNamespace("opencv")) {  
  png("test.png")  
  plot(qr)  
  dev.off()  
  opencv::ocv_qr_detect(opencv::ocv_read('test.png'))  
  unlink("test.png")  
}
```

print.bits	<i>Print a bits vector Display the logical vector as a bit string where FALSE is shown as 0 and TRUE as 1.</i>
------------	--

Description

Print a bits vector Display the logical vector as a bit string where FALSE is shown as 0 and TRUE as 1.

Usage

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

Arguments

x	the object to print
...	currently ignored

Author(s)

Thierry Onkelinx

See Also

Other bits: [as.character.bits\(\)](#), [bits2int\(\)](#), [bits\(\)](#), [c.bits\(\)](#)

Examples

```
z <- bits(c(FALSE, TRUE))  
print(z)
```

print.qr_code	<i>Print the qr_code object</i>
---------------	---------------------------------

Description

Please use `plot(x)` for a better quality image

Usage

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

Arguments

x the qr_code object
... currently ignored

Author(s)

Thierry Onkelinx

See Also

Other qr: [coordinates\(\)](#), [generate_svg\(\)](#), [plot.qr_code\(\)](#), [qr_code\(\)](#), [qr_event\(\)](#), [qr_wifi\(\)](#)

Examples

```
qr_code("HELLO WORLD")
```

qr_code *Generate the QR code*

Description

A **QR code** is a two-dimensional barcode developed by the Denso Wave company.

Usage

```
qr_code(x, ecl = c("L", "M", "Q", "H"))
```

Arguments

x the input string
ecl the required error correction level. Available options are "L" (7%), "M" (15%), "Q" (25%) and "H" (30%). Defaults to "L".

Value

The QR code as a logical matrix with "qr_code" class.

Author(s)

Thierry Onkelinx

See Also

Other qr: [coordinates\(\)](#), [generate_svg\(\)](#), [plot.qr_code\(\)](#), [print.qr_code\(\)](#), [qr_event\(\)](#), [qr_wifi\(\)](#)

Examples

```
qr_code("https://www.r-project.org")
qr <- qr_code("https://cran.r-project.org/package=rcode", ecl = "M")
qr
plot(qr)
# the qr_code object is a logical matrix
str(qr)
qr[1:10, 1:10]
```

qr_event	<i>Generate a QR code for an event</i>
----------	--

Description

Generate a QR code for an event

Usage

```
qr_event(start, end, title, ..., ecl = c("L", "M", "Q", "H"))
```

Arguments

start	the required start time as POSIXct.
end	the required end time as POSIXct.
title	the required title of the event.
...	optional arguments as defined in the details.
ecl	the required error correction level. Available options are "L" (7%), "M" (15%), "Q" (25%) and "H" (30%). Defaults to "L".

Details

Optional arguments. Other arguments are silently ignored.

- description
- location
- organiser
- url

See Also

Other qr: [coordinates\(\)](#), [generate_svg\(\)](#), [plot.qr_code\(\)](#), [print.qr_code\(\)](#), [qr_code\(\)](#), [qr_wifi\(\)](#)

qr_wifi	<i>Generate QR code with wifi login information</i>
---------	---

Description

Generate QR code with wifi login information

Usage

```
qr_wifi(  
  ssid,  
  encryption = c("WPA", "WEP", ""),  
  key = "",  
  hidden = FALSE,  
  ecl = c("L", "M", "Q", "H")  
)
```

Arguments

ssid	The SSID of the network.
encryption	The encryption standard. Options are "WPA", "WEP" and "". The latter implies no encryption. Defaults to "WPA".
key	The key for the encryption.
hidden	Use FALSE for a visible SSID. Use TRUE for a hidden SSID. Defaults to FALSE.
ecl	the required error correction level. Available options are "L" (7%), "M" (15%), "Q" (25%) and "H" (30%). Defaults to "L".

See Also

Other qr: [coordinates\(\)](#), [generate_svg\(\)](#), [plot.qr_code\(\)](#), [print.qr_code\(\)](#), [qr_code\(\)](#), [qr_event\(\)](#)

Index

- * **bits**
 - as.character.bits, 3
 - bits, 4
 - bits2int, 4
 - c.bits, 5
 - print.bits, 10
- * **qr**
 - coordinates, 6
 - generate_svg, 7
 - plot.qr_code, 9
 - print.qr_code, 10
 - qr_code, 11
 - qr_event, 12
 - qr_wifi, 13
- add_logo, 2
- as.character.bits, 3, 4, 5, 10
- bits, 3, 4, 5, 10
- bits2int, 3, 4, 4, 5, 10
- c.bits, 3–5, 5, 10
- coordinates, 6, 8, 9, 11–13
- generate_svg, 6, 7, 9, 11–13
- int2bits (bits2int), 4
- interactive(), 8
- opencv::ocv_qr_detect(), 9
- plot.qr_code, 6, 8, 9, 11–13
- plot.qr_logo (plot.qr_code), 9
- print.bits, 3–5, 10
- print.qr_code, 6, 8, 9, 10, 11–13
- qr_code, 6, 8, 9, 11, 11, 12, 13
- qr_code(), 7
- qr_event, 6, 8, 9, 11, 12, 13
- qr_wifi, 6, 8, 9, 11, 12, 13