Package 'qrcode'

October 13, 2021

Type Package

Title Generate QRcodes with R

Version 0.1.4

Description Create QRcode in R.

License GPL-3

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

BugReports https://github.com/ThierryO/qrcode/issues

Depends R (>= 3.0.0)

Imports R.utils, assertthat, stats, stringr, utils

Suggests httr, testthat (>= 3.0.0)

Config/testthat/edition 3

Encoding UTF-8

Language en-GB

RoxygenNote 7.1.2

NeedsCompilation no

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

Maintainer Thierry Onkelinx <thierry.onkelinx@inbo.be>

Repository CRAN

Date/Publication 2021-10-13 14:42:07 UTC

R topics documented:

`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))
```

DataStringBinary *Function to convert input data string to binary polynomial*

Description

Convert input data string to binary polynomial

Usage

```
DataStringBinary(dataString, qrInfo)
```

Arguments

dataString	input data string.
qrInfo	dataframe that store all the required info to generate qrcode.

Author(s)

Victor Teh

See Also

Other legacy: ECgenerator(), formatPolyGen(), polynomialGenerator(), qrFillUpMatrix(), qrInitMatrix(), qrInterleave(), qrMask(), qrVersionInfo(), qrcode_gen(), versionPolyGen()

`ECgenerator` *Error correction code generator Generate error correction code based on the input polynomial.*

Description

Error correction code generator Generate error correction code based on the input polynomial.

Usage

```
ECgenerator(GenPoly, DataPoly, DCWordCount, ECWordCount)
```

Arguments

`GenPoly`	generated polynomial to calculate error correction code word
`DataPoly`	input data polynomial
`DCWordCount`	data code word count
`ECWordCount`	error code word count

Value

Error code word polynomial

Author(s)

Victor Teh

See Also

Other legacy: `DataStringBinary()`, `formatPolyGen()`, `polynomialGenerator()`, `qrFillUpMatrix()`, `qrInitMatrix()`, `qrInterleave()`, `qrMask()`, `qrVersionInfo()`, `qrcode_gen()`, `versionPolyGen()`

`formatPolyGen` *Function to calculate and generate format polynomial*

Description

Function to calculate and generate format polynomial

Usage

```
formatPolyGen(formatString, polyString)
```

Arguments

formatString	QRcode format binary string
polyString	polynomial to create ECL for formatString

Author(s)

Victor Teh

See Also

Other legacy: DataStringBinary(), ECgenerator(), polynomialGenerator(), qrFillUpMatrix(), qrInitMatrix(), qrInterleave(), qrMask(), qrVersionInfo(), qrcode_gen(), versionPolyGen()

---

generate_svg *Generate the QR code as an svg file*

Description

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

Usage

```
generate_svg(
  qrcode,
  filename,
  size = 100,
  foreground = "black",
  background = "white",
  show = interactive()
)
```

Arguments

qrcode	a qr_code object as generated by qr_code.
filename	Where to store the filename. Silently overwrites existing files. Tries to create the path, when it doesn't exist.
size	size of the svg file in pixels.
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 interactive() sessions, otherwise FALSE.

Value

invisible NULL

Author(s)

Thierry Onkelinx

See Also

Other qr: `plot.qr_code()`, `print.qr_code()`, `qr_code()`

Examples

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

---

plot.qr_code *Plot the QR code*

Description

Plot the QR code

Usage

```
## S3 method for class 'qr_code'
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

Other qr: `generate_svg()`, `print.qr_code()`, `qr_code()`

Examples

```
qr <- qr_code("HELLO WORLD")
plot(qr)
```

polynomialGenerator *Function to generate polynomial*

Description

Function to generate polynomial

Usage

```
polynomialGenerator(ECcount)
```

Arguments

ECcount	error correction code word count

Value

polynomail to generate Error correction code

Author(s)

Victor Teh

See Also

Other legacy: DataStringBinary(), ECgenerator(), formatPolyGen(), qrFillUpMatrix(), qrInitMatrix(), qrInterleave(), qrMask(), qrVersionInfo(), qrcode_gen(), versionPolyGen()

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

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 *Print the qr_code object*

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: generate_svg(), plot.qr_code(), qr_code()

Examples

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

qrcode_gen *QRcode generator*

Description

Create QRcode in R. Capable to generate all variant of QRcode, version 1 to 40 and Error correct level of "L","M","Q" and " H". Not all reader in market can support all QRcode version, `qrcode_gen()` has a software limit to version 10 which is tested working in most reader.

Usage

```
qrcode_gen(
  dataString,
  ErrorCorrectionLevel = "L",
  dataOutput = FALSE,
  plotQRcode = TRUE,
  wColor = "White",
  bColor = "black",
  mask = 1,
  softLimitFlag = TRUE
)
```

Arguments

`dataString`	input string for the QRcode.
`ErrorCorrectionLevel`	Error Correction Level. The available options are `"L"`, `"M"`, `"Q"` and `"H"`. Default value as `"L"`.
`dataOutput`	option to export data as matrix. Default value is `FALSE`.
`plotQRcode`	option to plot QRcode. Default value is `TRUE`.
`wColor`	color of the white module(white squre) in QRcode. Default value `"white"`.
`bColor`	color of the black module(black squre) in QRcode. Default value `"black"`.
`mask`	mask for QRcode to increase decodability. Available values are `0` to `7`.
`softLimitFlag`	flag to limit the QRcode version to 10. Default value `TRUE`.

Value

A matrix that represent the QRcode. 1 as black module and 0 as white module.

Author(s)

Victor Teh

See Also

Other legacy: `DataStringBinary()`, `ECgenerator()`, `formatPolyGen()`, `polynomialGenerator()`, `qrFillUpMatrix()`, `qrInitMatrix()`, `qrInterleave()`, `qrMask()`, `qrVersionInfo()`, `versionPolyGen()`

Examples

```
qrcode_gen("www.r-project.org")

#User may change the color of the module
qrcode_gen("www.r-project.org", bColor = "Green3")
```

`qrFillUpMatrix` *Function to fill up the data bits*

Description

Fill up the predefined QRcode matrix with the input binary string.

Usage

```
qrFillUpMatrix(allBinary, data, version)
```

Arguments

`allBinary`	all data in binary in character format.
`data`	matrix data created by `qrFillUpMatrix`
`version`	version of the QRcode.

Value

matrix filled up with the data bits

Author(s)

Victor Teh

See Also

Other legacy: `DataStringBinary()`, `ECgenerator()`, `formatPolyGen()`, `polynomialGenerator()`, `qrInitMatrix()`, `qrInterleave()`, `qrMask()`, `qrVersionInfo()`, `qrcode_gen()`, `versionPolyGen()`

qrInitMatrix *Function to initialize QRcode in matrix for different version*

Description

Create a basic structre of QRcode in matrix format. Each element in QRcode will be marked as different value.

Usage

```
qrInitMatrix(version)
```

Arguments

version	version number of the target QRcode

Author(s)

Victor Teh

See Also

Other legacy: DataStringBinary(), ECgenerator(), formatPolyGen(), polynomialGenerator(), qrFillUpMatrix(), qrInterleave(), qrMask(), qrVersionInfo(), qrcode_gen(), versionPolyGen()

qrInterleave *Function to interleave the Data Code and Error Correction Core*

Description

Function to interleave the Data Code and Error Correction Core

Usage

```
qrInterleave(poly, dataPoly, qrInfo)
```

Arguments

poly	error correction code word polynomial
dataPoly	input data code word polynomial
qrInfo	dataframe that store all the required info to generate QRcode. Via qrVersionInfo

Value

Interleaved polynomial readied to fill up the QRcode matrix

Author(s)

Victor Teh

See Also

Other legacy: DataStringBinary(), ECgenerator(), formatPolyGen(), polynomialGenerator(), qrFillUpMatrix(), qrInitMatrix(), qrMask(), qrVersionInfo(), qrcode_gen(), versionPolyGen()

---

qrMask *Apply mask to the QRcode matrix*

Description

Apply mask to the QRcode matrix

Usage

```
qrMask(data, qrInfo, mask)
```

Arguments

data	QRcode matrix
qrInfo	dataframe that store all the required info to generate QRcode. Via qrVersionInfo()
mask	mask for QRcode to increase decodability. Available value are 0 to 7.

Details

QRcode stardard specify 8 masks as listed below.

- M0, (row + column) %% 2 == 0
- M1, (row) %% 2 == 0
- M2, (column) %% 3 == 0
- M3, (row + column) %% 3 == 0
- M4, (row%/%2 + column%/%3) %% 2 == 0
- M5, ((row * column) %% 2) + ((row * column) %% 3) == 0
- M6, (((row * column) %% 2) + ((row * column) %% 3)) %% 2 == 0
- M7, (((row + column) %% 2) + ((row * column) %% 3)) %% 2 == 0

Author(s)

Victor Teh

See Also

Other legacy: DataStringBinary(), ECgenerator(), formatPolyGen(), polynomialGenerator(), qrFillUpMatrix(), qrInitMatrix(), qrInterleave(), qrVersionInfo(), qrcode_gen(), versionPolyGen()

qrVersionInfo *Function to identify the version of the QRcode based on input string*

Description

Function to identify the version of the QRcode based on input string

Usage

```
qrVersionInfo(dataString, ECLevel = c("L", "M", "Q", "H"))
```

Arguments

dataString	dataString is the input string.
ECLevel	Error Correction Level. In QRcode standard, the are 4 levels "L", "M", "Q" and "H" which represent 7%, 15%, 20% and 30% data recovery capability.

Value

1 row dataframe that include all required info to generate QRcode.

Author(s)

Victor Teh

See Also

Other legacy: DataStringBinary(), ECgenerator(), formatPolyGen(), polynomialGenerator(), qrFillUpMatrix(), qrInitMatrix(), qrInterleave(), qrMask(), qrcode_gen(), versionPolyGen()

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: `generate_svg()`, `plot.qr_code()`, `print.qr_code()`

Examples

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

`versionPolyGen` *Function to calculate and generate version polynomial*

Description

Function to calculate and generate version polynomial

Usage

```
versionPolyGen(versionString, polyString)
```

Arguments

`versionString`	version in binary string
`polyString`	polynomial in binary string, specified in the standard to calculate version ECL.

Value

version polynomial.

Author(s)

Victor Teh

See Also

Other legacy: `DataStringBinary()`, `ECgenerator()`, `formatPolyGen()`, `polynomialGenerator()`, `qrFillUpMatrix()`, `qrInitMatrix()`, `qrInterleave()`, `qrMask()`, `qrVersionInfo()`, `qrcode_gen()`

Index