

# Package ‘Rcssplot’

October 12, 2022

**Title** Styling of Graphics using Cascading Style Sheets

**Version** 1.0.0

**Author** Tomasz Konopka [aut, cre]

**Maintainer** Tomasz Konopka <tokonopka@gmail.com>

**Description** Provides a means to style plots through cascading style sheets.  
This separates the aesthetics from the data crunching in plots and charts.

**Depends** R (>= 3.4.0)

**Imports** methods, graphics, grDevices, stats, utils

**Suggests** knitr, rmarkdown, testthat

**License** GPL-2

**URL** <https://github.com/tkonopka/Rcssplot>

**LazyData** true

**VignetteBuilder** knitr

**RoxygenNote** 7.0.2

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2019-12-13 20:50:02 UTC

## R topics documented:

abline . . . . .	2
arrows . . . . .	3
axis . . . . .	4
barplot . . . . .	5
box . . . . .	5
boxplot . . . . .	6
contour . . . . .	7
ctext . . . . .	8
grid . . . . .	9
hist . . . . .	9

jpeg	10
legend	11
lines	11
matplot	12
mtext	13
par	13
parplot	14
pdf	15
plot	15
png	16
points	17
polygon	17
print.Rcss	18
printRcss	19
Rcss	19
RcssChange	20
RcssChangePropertyValue	21
RcssCompulsoryClass	22
RcssDefaultStyle	22
RcssGetCompulsoryClass	23
RcssGetDefaultStyle	23
RcssGetPropertyValue	24
RcssGetPropertyValueOrDefault	25
RcssOverload	26
RcssProperty	26
RcssValue	27
RcssWatch	28
rect	29
stripchart	29
text	30
title	31
<b>Index</b>	<b>32</b>

---

abline

*Add a styled straight line to a plot*


---

### Description

This is a wrapper for R's abline function. See R's documentation for `graphics::abline` for further details.

**Usage**

```

abline(
  a = NULL,
  b = NULL,
  h = NULL,
  v = NULL,
  reg = NULL,
  coef = NULL,
  Rcsc = "default",
  Rcscclass = NULL,
  ...
)

```

**Arguments**

a, b	coefficient (intercept and slope) for line
h, v	horizontal, vertical positions for line
reg	an object with a coef method
coef	vector with intercept and slope for line
Rcsc	style sheet object. Leave "default" to use a style defined via RcscSetDefaultStyle().
Rcscclass	character, style class
...	Further parameters, see documentation of abline()

**Examples**

```

# draw a set of horizontal lines and a vertical line
plot(c(0, 1), c(0, 1), type="n")
abline(h=seq(0, 1, by=0.2))
abline(v=0.8)

```

---

arrows

*Add styled arrows to a plot*


---

**Description**

This is a wrapper for R's arrows function. See R's documentation for graphics::arrows for further details.

**Usage**

```

arrows(x0, y0, x1 = x0, y1 = y0, Rcsc = "default", Rcscclass = NULL, ...)

```

**Arguments**

<code>x0, y0</code>	coordinates of <i>*from*</i> endpoint
<code>x1, y1</code>	coordinates of <i>*to*</i> endpoint
<code>Rcss</code>	style sheet object. Leave "default" to use a style defined via <code>RcssSetDefaultStyle()</code>
<code>Rcssclass</code>	character, style class
<code>...</code>	Futher parameters, see documentation of <code>graphics::arrows</code>

**Examples**

```
# draw an arrow
plot(c(0, 1), c(0, 1), type="n")
arrows(0.2, 0.2, x1=0.8, y1=0.5)
```

---

<code>axis</code>	<i>Add a styled axis to a plot</i>
-------------------	------------------------------------

---

**Description**

This is a wrapper for R's `axis` function. See R's documentation for `graphics::axis` for further details.

**Usage**

```
axis(side, Rcss = "default", Rcssclass = NULL, ...)
```

**Arguments**

<code>side</code>	integer specifying what side of the plot to draw the axis. The codes are 1: bottom, 2: left, 3: top, 4: top. vertices
<code>Rcss</code>	style sheet object. Leave "default" to use a style defined via <code>RcssSetDefaultStyle()</code>
<code>Rcssclass</code>	sub class of style sheet
<code>...</code>	Further parameters, see documentation of <code>graphics::axis</code>

**Examples**

```
# draw separate axes on an empty plot
plot(c(0, 1), c(0, 1), type="n", axes=FALSE, xlab="x-axis", ylab="")
axis(1)
axis(3)
```

---

barplot	<i>Draw a styled barplot</i>
---------	------------------------------

---

**Description**

This is a wrapper for R's barplot function. See R's documentation for graphics::barplot for further details.

**Usage**

```
barplot(height, Rcss = "default", Rcssclass = NULL, ...)
```

**Arguments**

height	numeric vector giving bar lengths
Rcss	style sheet object. Leave "default" to use a style defined via RcssSetDefaultStyle()
Rcssclass	character, style class
...	Further parameters, see documentation of graphics::barplot

**Examples**

```
# draw a complete barplot  
barplot(1:5)
```

---

box	<i>Add a styled box around a plot</i>
-----	---------------------------------------

---

**Description**

This is a wrapper for R's box function. See R's documentation for graphics::box for further details.

**Usage**

```
box(which = "plot", Rcss = "default", Rcssclass = NULL, ...)
```

**Arguments**

which	character specifying where to draw a box; see documentation of box()
Rcss	style sheet object. Leave "default" to use a style defined via RcssSetDefaultStyle()
Rcssclass	sub class of style sheet
...	Further parameters, see documentation of graphics::box

**Examples**

```
# draw a box around an existing plot
plot(c(0, 1), c(0, 1), type="n", frame=FALSE)
box(lwd=3)
```

---

 boxplot

*Draw a styled boxplot*


---

**Description**

This is a wrapper for R's boxplot function. See R's documentation for `graphics::boxplot` for further details.

**Usage**

```
boxplot(x, Rcss = "default", Rcssclass = NULL, ...)
```

**Arguments**

x	data for boxplot; either single numeric vector or a list of numeric vectors; see documentation of <code>boxplot()</code>
Rcss	style sheet object. Leave "default" to use a style defined via <code>RcssSetDefaultStyle()</code>
Rcssclass	character, style class
...	Further parameters, see documentation of <code>graphics::boxplot</code>

**Examples**

```
# draw a complete boxplot
dataset <- list(A=rpois(30, 10), B=rpois(30, 20))
boxplot(dataset, col=c("#dd0000", "#dd8888"))
```

---

contour	<i>Draw a styled contour</i>
---------	------------------------------

---

## Description

This is a wrapper for R's contour function. See R's documentation for `graphics::contour` for further details.

## Usage

```
contour(  
  x = seq(0, 1, length.out = nrow(z)),  
  y = seq(0, 1, length.out = ncol(z)),  
  z,  
  Rcss = "default",  
  Rcssclass = NULL,  
  ...  
)
```

## Arguments

x	numeric vector; locations of grid lines
y	numeric vector; locations of grid lines
z	matrix of values
Rcss	style sheet object. Leave "default" to use a style defined via <code>RcssSetDefaultStyle()</code>
Rcssclass	character, style class
...	Further parameters, see documentation of <code>graphics::contour</code>

## Examples

```
# draw a complete contour plot  
dataset <- outer(1:10, 1:10)  
contour(z=dataset)
```

---

`ctext`*Write styled text into a plot corner*

---

### Description

This can be suitable for placing a label in a multi-panel figure. Note the automatic placement does not work when a plot is generated with logarithmic scales.

### Usage

```
ctext(  
  label,  
  x = NULL,  
  y = NULL,  
  adj = NULL,  
  cex = NULL,  
  Rcss = "default",  
  Rcssclass = NULL,  
  ...  
)
```

### Arguments

<code>label</code>	character, text for corner label
<code>x, y</code>	numeric, positions for manual placement
<code>adj</code>	numeric of length 2, argument <code>adj</code> for text
<code>cex</code>	numeric, argument <code>cex</code> for text
<code>Rcss</code>	style sheet object
<code>Rcssclass</code>	character, style class
<code>...</code>	additional argument, passed to <code>text()</code>

### Examples

```
plot(1:10, 1:10)  
ctext("A")
```

---

```
grid                                #' Draw a styled grid
```

---

### Description

This is a wrapper for R's grid function. See R's documentation for `graphics::grid` for further details.

### Usage

```
grid(Rcss = "default", Rcssclass = NULL, ...)
```

### Arguments

Rcss	style sheet object. Leave "default" to use a style defined via <code>RcssSetDefaultStyle()</code>
Rcssclass	character, style class
...	Further parameters, see documentation of <code>graphics::grid</code>

### Examples

```
# add a grid to an existing plot
plot(c(0, 10), c(0, 10), type="n", xaxs="i", yaxs="i", las=1)
grid(nx=10, ny=5, col="#777777")
```

---

```
hist                                Draw a styled histogram
```

---

### Description

This is a wrapper for R's hist function. See R's documentation for `graphics::hist` for further details.

### Usage

```
hist(x, Rcss = "default", Rcssclass = NULL, ...)
```

### Arguments

x	numeric vector
Rcss	style sheet object. Leave "default" to use a style defined via <code>RcssSetDefaultStyle()</code>
Rcssclass	character, style class
...	Further parameters, see documentation of <code>graphics::hist</code>

## Examples

```
# draw a complete histogram
dataset <- rpois(400, 6)
hist(dataset, breaks=seq(0, max(dataset)))
# only obtain the bin counts, without plotting
histdata <- hist(dataset, breaks=seq(0, 2+max(dataset), by=2), plot=FALSE)
histdata
```

---

jpeg

*Create a styled jpg figure*

---

## Description

This is a wrapper for R's jpeg function. See R's documentation for grDevices::jpeg for further details

## Usage

```
jpeg(file, Rcss = "default", Rcssclass = NULL, ...)
```

## Arguments

file	character string with file name
Rcss	style sheet object. Leave "default" to use a style defined via RcssSetDefaultStyle()
Rcssclass	character, style class
...	Further parameters, see documentation of grDevices::jpeg

## Examples

```
# send content of graphics to a jpg file
# to run this, un-comment the jpeg() and dev.off() lines
# jpeg(file="example-file.jpg")
barplot(1:5)
# dev.off()
```

---

legend *Add a styled legend to a plot*

---

**Description**

This is a wrapper for R's legend function. See R's documentation for graphics::legend for further details.

**Usage**

```
legend(x, y = NULL, legend, Rcss = "default", Rcssclass = NULL, ...)
```

**Arguments**

x, y	position of the legend
legend	character vector with labels (text appears in the legend)
Rcss	style sheet object. Leave "default" to use a style defined via RcssSetDefaultStyle()
Rcssclass	character, style class
...	Further parameters, see documentation of graphics::legend

**Examples**

```
# add a legend to an existing plot
plot(1:8, 1:8, col=rep(c(1,2), each=4), pch=19)
legend(7, 3, c("A", "B"), pch=19, col=1:2)
```

---

lines *Add styled line segments to a plot*

---

**Description**

This is a wrapper for R's lines function. See R's documentation for graphics::lines for further details.

**Usage**

```
lines(x, y = NULL, Rcss = "default", Rcssclass = NULL, ...)
```

**Arguments**

<code>x, y</code>	coordinates for start and end points for lines
<code>Rcss</code>	style sheet object. Leave "default" to use a style defined via <code>RcssSetDefaultStyle()</code>
<code>Rcssclass</code>	character, style class
<code>...</code>	Further parameters, see documentation of <code>graphics::lines</code>

**Examples**

```
# add lines to an existing plot area
plot(c(0, 10), c(0, 10), type="n")
lines(c(1,8), c(2, 2), lwd=3, col="black")
lines(c(1, 7, NA, 4, 9), c(1, 6, NA, 1, 3), lwd=1, col="blue")
lines(c(8, 3), c(7, 9), lwd=3, lty=2, col="red")
```

---

matplot

---

*Add styled line segments to a plot*


---

**Description**

This is a wrapper for R's `matplot` function. See R's documentation for `graphics::matplot` for further details.

**Usage**

```
matplot(x, y, Rcss = "default", Rcssclass = NULL, ...)
```

**Arguments**

<code>x, y</code>	vectors or matrices of data for plotting. The number of rows should match. If one of them are missing, the other is taken as <code>y</code> and an <code>x</code> vector of <code>1:n</code> is used. Missing values (NAs) are allowed.
<code>Rcss</code>	style sheet object. Leave "default" to use a style defined via <code>RcssSetDefaultStyle()</code>
<code>Rcssclass</code>	character, style class
<code>...</code>	Further parameters, see documentation of <code>graphics::lines</code>

**Examples**

```
# draw scatter based on column in a matrix
dataset = cbind(A=rnorm(20), B=rnorm(20))
matplot(dataset)
```

---

mtext	<i>Write styled text into a plot margin</i>
-------	---

---

**Description**

This is a wrapper for R's mtext function. See R's documentation for graphics::mtext for further details.

**Usage**

```
mtext(text, Rcsc = "default", Rcscclass = NULL, ...)
```

**Arguments**

text	characters to print on the plot
Rcsc	style sheet object. Leave "default" to use a style defined via RcscSetDefaultStyle()
Rcscclass	character, style class
...	Further parameters, see documentation of graphics::mtext

**Examples**

```
# draw text into a margin
plot(c(0, 1), c(0, 1), type="n", xlab="", ylab="")
mtext(side=1, "bottom x-axis label", line=2.5)
mtext(side=2, "left y-axis label", line=2.5)
mtext(side=3, "top x-axis label")
mtext(side=4, "right y-axis label")
```

---

par	<i>Set styled parameters for base graphics</i>
-----	--

---

**Description**

This is a wrapper for R's par function. See R's documentation for graphics::par for further details.

**Usage**

```
par(Rcsc = "default", Rcscclass = NULL, ...)
```

**Arguments**

Rcss	style sheet object. Leave "default" to use a style defined via RcssSetDefaultStyle()
Rcssclass	character, style class
...	Further parameters, see documentation of graphics::par

**Examples**

```
# set properties for plot
par(ps=8, mar=c(3, 8, 3, 1))
plot(c(0, 1), c(0, 1), type="n", frame=FALSE)
text(rep(0.5, 2), c(0.2, 0.5), c("abc", "def"))
par(ps=12)
text(0.5, 0.8, "xyz")
```

---

 parplot

*combination of par and plot*


---

**Description**

The sequence of par() and plot() occurs so frequently that it a shortcut is helpful.

**Usage**

```
parplot(x, y, Rcss = "default", Rcssclass = NULL, ...)
```

**Arguments**

x, y	coordinates for points on the plot
Rcss	style sheet object, leave "default" to use a style defined via RcssDefaultStyle()
Rcssclass	character, style class
...	Further parameters, passed to plot()

**Examples**

```
parplot(x=1:4, y=c(1,3,2,4))
```

---

pdf *Create a styled pdf figure*

---

**Description**

This is a wrapper for R's pdf function. See R's documentation for grDevices::pdf for further details

**Usage**

```
pdf(file, Rcss = "default", Rcssclass = NULL, ...)
```

**Arguments**

file	character string with file name
Rcss	style sheet object. Leave "default" to use a style defined via RcssSetDefaultStyle()
Rcssclass	character, style class
...	Further parameters, see documentation of grDevices::pdf

**Examples**

```
# send content of graphics to a pdf file
# to run this, un-comment the pdf() and dev.off() lines
# png(file="example-file.pdf")
barplot(1:5)
# dev.off()
```

---

plot *Create a styled plot*

---

**Description**

This is a wrapper for R's plot function. See R's documentation for graphics::plot for further details.

**Usage**

```
plot(x, y, Rcss = "default", Rcssclass = NULL, ...)
```

**Arguments**

x, y	coordinates for points on the plot
Rcss	style sheet object. Leave "default" to use a style defined via RcssSetDefaultStyle()
Rcssclass	character, style class
...	Further parameters, see documentation of graphics::plot

## Examples

```
# draw a new empty plot area - unit square
plot(c(0, 1), c(0, 1), type="n", xlab="", ylab="")
# draw a plot area, automatically add some points
plot(runif(20), rpois(20, 100))
```

---

png

*Create a styled png figure*

---

## Description

This is a wrapper for R's png function. See R's documentation for `grDevices::png` for further details.

## Usage

```
png(file, Rcss = "default", Rcssclass = NULL, ...)
```

## Arguments

file	character string with file name
Rcss	style sheet object. Leave "default" to use a style defined via <code>RcssSetDefaultStyle()</code>
Rcssclass	character, style class
...	Further parameters, see documentation of <code>grDevices::png</code>

## Examples

```
# send content of graphics to a png file
# to run this, un-comment the png() and dev.off() lines
# png(file="example-file.png")
barplot(1:5)
# dev.off()
```

---

points *Add styled points to a plot*

---

**Description**

This is a wrapper for R's points function. See R's documentation for graphics::points for further details.

**Usage**

```
points(x, y = NULL, Rcss = "default", Rcssclass = NULL, ...)
```

**Arguments**

x, y	coordinates for points on the plot
Rcss	style sheet object. Leave "default" to use a style defined via RcssSetDefaultStyle()
Rcssclass	character, style class
...	Further parameters, see documentation of graphics::points

**Examples**

```
# draw a set of points onto an existing plot
plot(c(0, 1), c(0, 1), type="n")
points(runif(10), runif(10))
points(runif(10), runif(10), col="blue", pch=19)
```

---

polygon *Draw a styled polygon on a plot*

---

**Description**

This is a wrapper for R's polygon function. See R's documentation for graphics::polygon for further details.

**Usage**

```
polygon(x, y = NULL, Rcss = "default", Rcssclass = NULL, ...)
```

**Arguments**

x, y	coordinates for polygon vertices
Rcss	style sheet object. Leave "default" to use a style defined via RcssSetDefaultStyle()
Rcssclass	character, style class
...	Further parameters, see documentation of graphics::polygon

**Examples**

```
# draw a multi-sided shape on an existing plot
plot(c(0, 10), c(0, 10), type="n", xlab="", ylab="")
polygon(c(1, 4, 7, 7, 1), c(1, 1, 4, 8, 8), col="blue")
```

---

```
print.Rcss           Show basic information about an Rcss object
```

---

**Description**

Display selectors encoded in an Rcss object. For more detailed information about the object, see function printRcss()

**Usage**

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

**Arguments**

x	style sheet object
...	Further parameters are ignored

**Examples**

```
# define a custom style, display it
custom.style <- Rcss(text="points { cex: 2; }")
custom.style
```

---

printRcss                      *Display properties encoded in an Rcss object*

---

### Description

Display properties encoded in an Rcss object, including any subclasses.

### Usage

```
printRcss(Rcss, selector = NULL, verbose = FALSE)
```

### Arguments

Rcss	style sheet object
selector	character string with name of selector to print
verbose	logical. If TRUE, function prints all information about the selector, including subclasses. If FALSE, function omits detailed information about subclasses.

### Examples

```
# define a custom style
custom.style <- Rcss(text="points { pch:2; } points.A { pch: 3; }")

# printing details for a selector, concise and verbose
printRcss(custom.style, "points")
printRcss(custom.style, "points", verbose=TRUE)
```

---

Rcss                              *Create an Rcss style object*

---

### Description

Creates a style sheet object using definition specified in an Rcss file. When a file is not specified, creates a base object object without any styling.

### Usage

```
Rcss(file = NULL, text = NULL)
```

### Arguments

file	filename containing Rcss definitions. If set to NULL, function returns a basic Rcss object. If multiple files, function reads each one and produces a joint style.
text	character, a string with Rcss

**Details**

See also related functions `RcssGetDefaultStyle()` and `RcssOverload()`.

**Value**

Rcss object

**Examples**

```
# define a custom style
custom.style <- Rcss(text="plot { pch:19; col: 2 }")

# display the custom style
printRcss(custom.style, "plot")

# use the custom style in a chart
plot(1:4, 1:4, Rcss=custom.style)
```

---

RcssChange

*Modify an Rcss style sheet object*

---

**Description**

Creates a new Rcss style sheet object from the input, modifying one or more properties.

**Usage**

```
RcssChange(
  selector,
  propertylist = NULL,
  property = NULL,
  value = NULL,
  Rcssclass = NULL,
  Rcss = "default"
)
```

**Arguments**

selector	name of one selector ("text", "plot", "axis", etc.)
propertylist	list with property/value pairs to update
property	name of a single property. This is only used when propertylist is set to NULL
value	new values associated with property above. This is only used when propertylist is set to NULL
Rcssclass	subclass of style sheet. Leave NULL to change base property. Provide one character value to edit one subclass. Provide a vector to edit a subclass of a ...
Rcss	style sheet object

**Value**

always returns an Rcss object. Note: when changing the default style, this will return a new style without actually affecting the default style. To change how the default works in practice, assign this return value to RcssDefaultStyle

**Examples**

```
style1 <- Rcss(text="points { cex: 1; pch: 19; }")
printRcss("points", Rcss=style1, verbose=TRUE)
style2 <- RcssChange("points", list(cex=2), Rcss=style1)
printRcss("points", Rcss=style2, verbose=TRUE)
```

---

RcssChangePropertyValue

*Modify an Rcss style sheet object*

---

**Description**

Creates a new Rcss style sheet object from the input, modifying one or more properties.

**Usage**

```
RcssChangePropertyValue(
  Rcss,
  selector,
  Rcssclass = NULL,
  propertylist = NULL,
  property = NULL,
  value = NULL
)
```

**Arguments**

Rcss	style sheet object
selector	name of one selector ("text", "plot", "axis", etc.)
Rcssclass	subclass of style sheet. Leave NULL to change base property. Provide one character value to edit one subclass. Provide a vector to edit a subclass of a ...
propertylist	list with property/value pairs to update
property	name of a single property. This is only used when propertylist is set to NULL
value	new values associated with property above. This is only used propertylist is set to NULL

**Details**

Equivalent to RcssChange: use RcssChange instead

**Examples**

```
# use RcssChange instead
```

---

RcssCompulsoryClass     *Vector holding set a compulsory Rcssclass*

---

**Description**

These style class (or classes) are applied in all functions of the Rcss family.

**Usage**

```
RcssCompulsoryClass
```

**Format**

An object of class NULL of length 0.

---

RcssDefaultStyle     *Default Rcssplot style sheet*

---

**Description**

This style sheet will be applied in all functions of the Rcss family.

**Usage**

```
RcssDefaultStyle
```

**Format**

An object of class NULL of length 0.

---

`RcssGetCompulsoryClass`*Get current state of compulsory Rcssclass*

---

**Description**

Fetches the value of the RcssCompulsoryClass object defined in parent environments.

**Usage**

```
RcssGetCompulsoryClass(Rcssclass = NULL)
```

**Arguments**

`Rcssclass` character vector, set of additional compulsory classes. When NULL, function returns the current set of compulsory classes defined in parent environments. When non-NULL, functions returns the concatenation of the current set and new set.

**Examples**

```
# retrieve the current compulsory class
class.null <- RcssGetCompulsoryClass()

# augment the current compulsory class with more labels
class.A <- RcssGetCompulsoryClass("A")
class.A
class.B <- RcssGetCompulsoryClass("B")
class.B

# when the object RcssCompulsoryClass is set, this augments a vector
RcssCompulsoryClass <- c("X", "Y")
class.XYZ <- RcssGetCompulsoryClass("Z")
class.XYZ
```

---

`RcssGetDefaultStyle` *Get default Rcssplot style object*

---

**Description**

Fetches the value of the RcssDefaultStyle object defined in parent environments.

**Usage**

```
RcssGetDefaultStyle(Rcss = "default")
```

**Arguments**

Rcss            Rcss object, replacement default style object. When set to "default", the function returns a copy of the default object defined in parent environment. When set to Rcss object, the function ignores the default and returns the set object back.

**Examples**

```
# retrieve the current default style
style.now <- RcssGetDefaultStyle()
```

---

RcssGetPropertyValue    *Extract a value for an Rcss property*

---

**Description**

Extract a value for a property from an Rcss style sheet object. Returns a list with two items. "Defined" is a boolean that indicates the property is defined in the style sheet. "Value" gives the actual value of the property.

**Usage**

```
RcssGetPropertyValue(Rcss, selector, property, Rcssclass = NULL)
```

**Arguments**

Rcss            style sheet object  
 selector        name of selector of interest (e.g. "plot", "axis", "text", etc.)  
 property        name of property of interest (e.g. "col", "pch", etc.)  
 Rcssclass       subclass of style sheet

**Details**

Equivalent to RcssProperty; use RcssProperty instead.

**Examples**

```
# use RcssProperty or RcssValue instead
```

---

`RcssGetPropertyValueOrDefault`*Extract a value for an Rcss property*

---

### Description

If the requested property is defined within an Rcss object, this function will return the associated value. If the property is not defined, the function returns a default value that can be passed into the function and is set NULL otherwise. See also `RcssGetPropertyValue()`.

### Usage

```
RcssGetPropertyValueOrDefault(  
  Rcss,  
  selector,  
  property,  
  default = NULL,  
  Rcssclass = NULL  
)
```

### Arguments

<code>Rcss</code>	style sheet object
<code>selector</code>	name of selector of interest (e.g. "plot", "axis", "text", etc.)
<code>property</code>	name of property of interest (e.g. "col", "pch", etc.)
<code>default</code>	value to return if the desired property is not defined in Rcss
<code>Rcssclass</code>	subclass of style sheet

### Details

Equivalent to `RcssValue()`; use `RcssValue()` instead

### Examples

```
# use RcssValue instead
```

---

RcssOverload	<i>Overloads base graphics functions by their Rcssplot wrappers</i>
--------------	---

---

**Description**

Rcssplot graphics functions have 'Rcss' prefixes, e.g. Rcssstext(). This function can be invoked to overload base-graphics functions by their Rcss wrappers. i.e. After executing this function, you can execute e.g. text() and automatically use the Rcss capabilities.

**Usage**

```
RcssOverload()
```

**Details**

Warning: this function creates masking objects in your current environment for many base-graphics functions. See documentation for details.

**Examples**

```
# this function is deprecated - do not use it
suppressWarnings(RcssOverload())
```

---

RcssProperty	<i>Extract information about property and its value</i>
--------------	---

---

**Description**

Extract information about property and its value

**Usage**

```
RcssProperty(selector, property, Rcssclass = NULL, Rcss = "default")
```

**Arguments**

selector	character, name of selector, e.g. 'points'
property	character, name of property, e.g. 'col'
Rcssclass	character or vector, subclass in Rcss
Rcss	Rcss object

**Value**

list with two ites. Component "defined" is a boolean that indicates whether the property is defined in the style. Component "value" gives the actual value associated to the property.

**Examples**

```
style1 <- Rcss(text="points { cex: 2; }")
# cex is defined, col is not defined
RcssProperty("points", "cex", Rcss=style1)
RcssProperty("points", "col", Rcss=style1)
```

---

RcssValue

*Extracts a value from an Rcss object*


---

**Description**

If the selector and property are defined in the Rcss object, this function will return the value stored in the Rcss object. Otherwise, the function will return a default value. See also related functions `RcssGetPropertyValueOrDefault`, which is the same, except that `RcssValue` is shorter to write and takes the Rcss object as its last argument.

**Usage**

```
RcssValue(
  selector,
  property,
  default = NULL,
  Rcssclass = NULL,
  Rcss = "default"
)
```

**Arguments**

selector	character, name of selector, e.g. 'points'
property	character, name of property to get, e.g. 'col'
default	value to return if selector/property are not defined
Rcssclass	character or vector, subclass in Rcss
Rcss	Rcss object

**Value**

a value from the Rcss object

**Examples**

```
style1 <- Rcss(text="custom { key: 100 }")
RcssValue("custom", "key", default=1, Rcss=style1)
RcssValue("custom", "key2", default=0, Rcss=style1)
```

---

RcssWatch

*development tool for adjusting Rcss and R graphics code*


---

**Description**

This is a macro script that loads R code and a default Rcss style, and then executes a function. This process is repeated indefinitely.

**Usage**

```
RcssWatch(f, files = NULL, ...)
```

**Arguments**

<code>f</code>	function or character of function name, executed at each iteration
<code>files</code>	character, paths to R and Rcss files
<code>...</code>	other arguments, passed to function <code>f</code>

**Examples**

```
# Note: the examples below draw a charat once and exit.
# To enable quick re-drawing, RcssWatch must be provided with file paths

# draw and redraw a bar plot
RcssWatch(plot, x=1:4, y=1:4)

# alternative syntax, using a function name as a string
custom.barplot <- function(x=1:4, main="") { barplot(x, main=main) }
RcssWatch("custom.barplot", main="Custom")

# for more interesting behavior, specify a files with styles and R source
```

---

rect	<i>Draw styled rectangles on a plot</i>
------	---

---

**Description**

This is a wrapper for R's `rect` function. See R's documentation for `graphics::rect` for further details.

**Usage**

```
rect(xleft, ybottom, xright, ytop, Rcss = "default", Rcssclass = NULL, ...)
```

**Arguments**

<code>xleft, ybottom, xright, ytop</code>	vector of coordinates for rectangles' vertices
<code>Rcss</code>	style sheet object. Leave "default" to use a style defined via <code>RcssSetDefaultStyle()</code>
<code>Rcssclass</code>	character, style class
<code>...</code>	Further parameters, see documentation of <code>graphics::rect</code>

**Examples**

```
# draw rectangles on an existing plot
plot(c(0, 10), c(0, 10), type="n", xlab="", ylab="")
rect(4.5, 1, 5.5, 3)
rect(c(1, 7.5), c(6, 6), c(2.5, 9), c(8, 8))
```

---

stripchart	<i>Draw styled strip chart</i>
------------	--------------------------------

---

**Description**

This is a wrapper for R's `stripchart` function. See R's documentation for `graphics::stripchart` for further details.

**Usage**

```
stripchart(x, Rcss = "default", Rcssclass = NULL, ...)
```

**Arguments**

x	list of numeric vectors
Rcss	style sheet object. Leave "default" to use a style defined via RcssSetDefaultStyle()
Rcssclass	character, style class
...	Further parameters, see documentation of graphics::stripchart

**Examples**

```
# draw a complete strip-chart plot
dataset <- list(A=c(1,9,3,8), B=c(3,4,2,9,2), C=rpois(8, 10))
stripchart(dataset)
stripchart(dataset, method="jitter", vertical=TRUE, pch=19)
```

---

text

---

*Add styled text to a plot*


---

**Description**

This is a wrapper for R's text function. See R's documentation for graphics::text for further details.

**Usage**

```
text(
  x,
  y = NULL,
  labels = seq_along(x),
  Rcss = "default",
  Rcssclass = NULL,
  ...
)
```

**Arguments**

x, y	coordinates where to write labels
labels	characters to print on the plot
Rcss	style sheet object. Leave "default" to use a style defined via RcssSetDefaultStyle()
Rcssclass	character, style class
...	Further parameters, see documentation of graphics::text

**Examples**

```
# add text to an existing plot
plot(c(0, 1), c(0, 1), type="n")
text(0.1, 0.1, "A")
text(c(0.2, 0.7), c(0.8, 0.6), c("B", "C"))
```

---

title	<i>Add styled annotation to a plot</i>
-------	--

---

**Description**

This is a wrapper for R's title function. See R's documentation for `graphics::title` for further details.

**Usage**

```
title(
  main = NULL,
  sub = NULL,
  xlab = NULL,
  ylab = NULL,
  Rcss = "default",
  Rcssclass = NULL,
  ...
)
```

**Arguments**

main	plot title
sub	plot sub title
xlab, ylab	labels on axes
Rcss	style sheet object. Leave "default" to use a style defined via <code>RcssSetDefaultStyle()</code>
Rcssclass	character, style class
...	Further parameters, see documentation of <code>graphics::title</code>

**Examples**

```
# add a title
plot(c(0, 1), c(0, 1), type="n", xlab="", ylab="")
title("This is the title")
title(sub="This is a bottom title")
```

# Index

## \* datasets

RcssCompulsoryClass, [22](#)  
RcssDefaultStyle, [22](#)

abline, [2](#)  
arrows, [3](#)  
axis, [4](#)

barplot, [5](#)  
box, [5](#)  
boxplot, [6](#)

contour, [7](#)  
ctext, [8](#)

grid, [9](#)

hist, [9](#)

jpeg, [10](#)

legend, [11](#)  
lines, [11](#)

matplot, [12](#)  
mtext, [13](#)

par, [13](#)  
parplot, [14](#)  
pdf, [15](#)  
plot, [15](#)  
png, [16](#)  
points, [17](#)  
polygon, [17](#)  
print.Rcss, [18](#)  
printRcss, [19](#)

Rcss, [19](#)  
RcssChange, [20](#)  
RcssChangePropertyValue, [21](#)  
RcssCompulsoryClass, [22](#)

RcssDefaultStyle, [22](#)  
RcssGetCompulsoryClass, [23](#)  
RcssGetDefaultStyle, [23](#)  
RcssGetPropertyValue, [24](#)  
RcssGetPropertyValueOrDefault, [25](#)  
RcssOverload, [26](#)  
RcssProperty, [26](#)  
RcssValue, [27](#)  
RcssWatch, [28](#)  
rect, [29](#)

stripchart, [29](#)

text, [30](#)  
title, [31](#)