Package ‘grImport’

May 25, 2010

Version 0.6-1

Depends R (>= 2.3.0), methods, grid, XML

SystemRequirements ghostscript

Title Importing Vector Graphics

Author Paul Murrell and Richard Walton.

Maintainer Paul Murrell <p.murrell@auckland.ac.nz>

Description Functions for converting, importing, and drawing PostScript pictures in R plots.

License GPL (>= 2)

Repository CRAN

Date/Publication 2010-05-25 15:57:17

R topics documented:

`drawPath` *Draw a Picture Object*

Description

This is a generic function that does the hard work for `picture` when drawing paths of a `"Picture"` object as part of the current plot.

Usage

```
drawPath(p, trans, ...)
```

Arguments

`p` A `"Picture"` object.

`trans` A function that takes two arguments, `x` and `y`, and can be used to transform the locations on a path into sensible coordinates within the current plot.

`...` Other arguments, typically graphical parameters.

Author(s)

Paul Murrell

See Also

`picture` and `grid.picture`

`explodePaths` *Explode Complex Picture Paths*

Description

This function converts complex paths within a `"Picture"` object (paths that contain more than one move operation) into multiple simple paths (paths with only one move operation).

Usage

```
explodePaths(picture)
```

Arguments

`picture` A `"Picture"` object.

Value

A "Picture" object.

Author(s)

Paul Murrell

See Also

grid.picture

---

grid.picture *Draw a Picture Object*

Description

These function take a "Picture" object and either draw the picture or create a grid graphical object representing the picture.

Usage

```
grid.picture(...)
pictureGrob(picture, x = 0.5, y = 0.5,
            width = 1, height = 1, just = "centre",
            exp = 0.05, xscale = NULL, yscale = NULL,
            distort = FALSE,
            FUN = grobify, ..., name = NULL, gp = gpar())
```

Arguments

picture	A "Picture" object.
x	A single numeric value or unit object specifying an x-value.
y	A single numeric value or unit object specifying a y-value.
width	A single numeric value or unit object specifying a width.
height	A single numeric value or unit object specifying a height.
just	The justification of the picture relative to its (x, y) location. If there are two values, the first value specifes horizontal justification and the second value specifies vertical justification. Possible string values are: "left", "right", "centre", "center", "bottom", and "top". For numeric values, 0 means left alignment and 1 means right alignment.
exp	An expansion factor; determines whether any space is left between the extent of the picture and the bounding rectangle it is drawn within.
xscale	A numeric vector of length two indicating the minimum and maximum on the x-scale.

yscale — A numeric vector of length two indicating the minimum and maximum on the y-scale.

distort — A logical value indicating whether the image should preserve its aspect ratio or distort to fit the area it is being drawn within.

FUN — The function used to convert paths within the picture to grid grobs.

... — For grid.picture arguments to be passed to pictureGrob. For pictureGrob, arguments to be passed to grobify.

name — A character identifier.

gp — An object of class gpar, typically the output from a call to the function gpar. This is basically a list of graphical parameter settings.

Value

A grid gTree.

Author(s)

Paul Murrell

See Also

grobify and grid.symbols

grid.symbols *Draw a Picture Object*

Description

These function take a "Picture" object and either draw the picture at several locations or create a grid graphical object representing the picture (drawn at several locations).

Usage

```
grid.symbols(...)
symbolsGrob(picture, x = unit(0.5, "npc"), y = unit(0.5, "npc"),
            size = unit(1, "npc"), units = "npc", ...,
            name = NULL, gp = gpar())
```

Arguments

picture — A "Picture" object.

x — A numeric vector or unit object specifying x-values.

y — A numeric vector or unit object specifying y-values.

size — A numeric vector or unit object specifying symbol size.

units — The default units to use if x, y, or size are specified as simple numeric values.

`...` For `grid.symbols` arguments to be passed to `symbolsGrob`. For `symbolsGrob`, arguments to be passed to `symbolize`.

`name` A character identifier.

`gp` An object of class `gpar`, typically the output from a call to the function `gpar`. This is basically a list of graphical parameter settings.

Value

A grid gTree.

Author(s)

Paul Murrell

See Also

`symbolize` and `grid.picture`

`grobify` *Convert a Picture into a Grid Grob*

Description

Converts an entire `"Picture"` object, or a path element thereof, into an appropriate grid grob. Not usually called directly by the user; usually called by `grid.picture`.

Usage

```
grobify(object, ...)
```

Arguments

`object` A `"Picture"` object.

`...` Arguments passed on to the relevant grid grob constructor function.

Value

A grid gTree (when given a whole `"Picture"`) or an appropriate grid grob (when given a single picture path).

Author(s)

Paul Murrell

See Also

`grid.picture` and `symbolize`

picture *Draw a Picture Object*

Description

This function take a `"Picture"` object and draw it in the current plot region.

Usage

```
picture(picture, xleft, ybottom, xright, ytop, ...)
```

Arguments

`picture`	A `"Picture"` object.
`xleft`	The location for the left edge of the picture.
`ybottom`	The location for the bottom edge of the picture.
`xright`	The location for the right edge of the picture.
`ytop`	The location for the top edge of the picture.
`...`	graphical parameters.

Details

The positions supplied, i.e., `xleft, ...`, are relative to the current plotting region. If the x-axis goes from 100 to 200 then `xleft` should be larger than 100 and `xright` should be less than 200.

Author(s)

Paul Murrell

See Also

`readPicture` and `grid.picture`

Picture-class *Class "Picture"*

Description

A collection of paths (polylines or polygons) that together describe a picture.

Objects from the Class

Objects can be created by calls of the form `new("Picture", ...)`. ~~ describe objects here ~~

Slots

`paths`: A list of objects of class "`PictureStroke`" (polylines) or "`PictureFill`" (polygons).

`summary`: Object of class "`PictureSummary`"

Methods

[`signature(x = "Picture")`: subset the paths of a picture to produce a new picture (includeing a new, updated summary).

[[`signature(x = "Picture")`: extract either a single path or, more usefully, the sub-paths of a "`PictureText`" object, to produce a new picture (includeing a new, updated summary).

grobify `signature(object = "Picture")`: convert a picture into a grid grob (for use as a one-off image).

symbolize `signature(object = "Picture")`: convert a picture into a grid grob (for use as a plotting symbol).

Author(s)

Paul Murrell

See Also

`readPicture`, `grid.picture`, `grid.symbols` and `PictureStroke-class` `PictureFill-class` `PictureSummary-class`.

`PictureChar-class` *Class "PictureChar"*

Description

A description of a polygon path that describes the outline of a character (and might be part of a larger picture).

Slots

`x`: Object of class "`numeric`"

`y`: Object of class "`numeric`"

`rgb`: Object of class "`character`" that specifies the fill colour for the polygon.

`lwd`: Object of class "`numeric`" that specifies the line width of the polygon (only used if the character outline is stroked).

`char`: Object of class "`character`" that specifies the character that this path represents.

Extends

Class "`PictureOp`", directly.

Methods

- **grobify** `signature(object = "PictureChar")`: convert to a grid grob (for one-off drawing). Use argument `fillText` to fill the text rather than just stroke the outline. In that case, the first path for this character is filled in the appropriate colour and all subsequent paths are filled using a colour controlled by the `bgText` argument (default is white).
- **symbolize** `signature(object = "PictureChar")`: convert to a grid grob (for drawing as a data symbol). Currently draws nothing.
- **drawPath** `signature(object = "PictureChar")`: draws a traced character path as part of the current plot.

Author(s)

Paul Murrell

See Also

`Picture-class`, `PictureStroke-class`, and `grid.picture`, `grid.symbols`.

`PictureFill-class` *Class "PictureFill"*

Description

A description of a polygon path (that might be part of a larger picture).

Slots

- `x`: Object of class `"numeric"`
- `y`: Object of class `"numeric"`
- `rgb`: Object of class `"character"` that specifies the fill colour for the polygon.
- `lwd`: Object of class `"numeric"` that specifies the line width of the polygon (not typically used because the polygon is meant to be filled not stroked).

Extends

Class `"PictureOp"`, directly.

Methods

- **grobify** `signature(object = "PictureFill")`: convert to a grid grob (for one-off drawing). There is a logical argument `use.gc` which can be used to turn off the graphics context coming from the object so that a different one can be specified via the `gp` argument.
- **symbolize** `signature(object = "PictureFill")`: convert to a grid grob (for drawing as a data symbol).
- **drawPath** `signature(object = "PictureFill")`: draws a traced polygonx as part of the current plot.

Author(s)

Paul Murrell

See Also

`Picture-class`, `PictureStroke-class`, and `grid.picture`, `grid.symbols`.

`PictureOp-class` *Class "PictureOp"*

Description

A virtual class from which `"PictureStroke"` and `"PictureFill"` are derived.

Slots

`x`: Object of class `"numeric"`

`y`: Object of class `"numeric"`

`rgb`: Object of class `"character"` giving colour information for the path.

`lty`: Object of class `"numeric"` giving line type information for the path.

`lwd`: Object of class `"numeric"` giving line width information for the path.

Author(s)

Paul Murrell

See Also

`PictureStroke-class` and `PictureFill-class`.

`picturePaths` *Draw Individual Picture Paths*

Description

This function individually draws each path from a `"Picture"` object.

Usage

```
picturePaths(picture, nr, nc,
             col = "black",
             fill = "light grey",
             freeScales = FALSE,
             xscale = NULL, yscale = NULL,
             label = function(n) {
               tg <- textGrob(n, x = 0, y = 0,
                              just = c("left", "bottom"),
                              gp = gpar(fontsize = 6))
               grid.rect(x = 0, y = 0,
                         height = unit(6, "points"),
                         width = grobWidth(tg),
                         just = c("left", "bottom"),
                         gp = gpar(fill = "white"))
               grid.draw(tg)
             }, use.gc = TRUE)
```

Arguments

`picture` A `"Picture"` object.

`nr` Number of rows of paths to draw.

`nc` Number of columns of paths to draw.

`col` Colour of border drawn around each path.

`fill` Background colour drawn behind each path.

`freeScales` A boolean indicating whether each path should be drawn on its own scale. If `FALSE`, all paths are drawn on a common scale. If `TRUE`, `xscale` and `yscale` are ignored.

`xscale` A numeric vector of length two indicating the minimum and maximum on the x-scale.

`yscale` A numeric vector of length two indicating the minimum and maximum on the y-scale.

`label` Function to draw a label on each path. If any value other than a function is specified, no labels are drawn.

`use.gc` A boolean indicating whether to use the graphical parameter information in the path.

Author(s)

Paul Murrell

See Also

`grid.picture`

`PictureStroke-class` *Class "PictureStroke"*

Description

A polyline path (that might be part of a larger picture).

Slots

`x`**:** Object of class `"numeric"`

`y`**:** Object of class `"numeric"`

`rgb`**:** Object of class `"character"` giving the colour of the polyline.

`lwd`**:** Object of class `"numeric"` giving the width of the polyine.

Extends

Class `"PictureOp"`, directly.

Methods

grobify `signature(object = "PictureStroke")`: convert into a grid grob (for one-off drawing). There is a logical argument `use.gc` which can be used to turn off the graphics context coming from the object so that a different one can be specified via the `gp` argument.

symbolize `signature(object = "PictureStroke")`: convert into a grid grob (for drawing as a data symbol)).

drawPath `signature(object = "PictureStroke")`: draws a traced line as part of the current plot.

Author(s)

Paul Murrell

See Also

`Picture-class`, `PictureFill-class`, and `grid.picture`, `grid.symbols`.

`PictureSummary-class` *Class "PictureSummary"*

Description

Summary information about a picture such as the number of paths and bounding box information.

Slots

`numPaths`: Object of class "`numeric`"

`xscale`: Object of class "`numeric`" range of x-values in picture.

`yscale`: Object of class "`numeric`" range of y-values in picture.

Author(s)

Paul Murrell

See Also

`Picture-class`

`PictureText-class` *Class "PictureText"*

Description

A description of a piece of text to draw, at a particular location and a particular size (and it might be part of a larger picture).

Slots

`string`: Object of class "`character`"

`x`: Object of class "`numeric`"

`y`: Object of class "`numeric`"

`w`: Object of class "`numeric`"

`angle`: Object of class "`numeric`"

`rgb`: Object of class "`character`" that specifies the fill colour for the polygon.

`lwd`: Object of class "`numeric`" that specifies the line width of the polygon (only used if the character outline is stroked).

Extends

Class "`PictureOp`", directly.

Methods

- **grobify** `signature(object = "PictureText")`: convert to a grid grob (for one-off drawing). In addition to a `use.gc` argument, there is a `fillText` argument to specify whether to stroke or fill text paths. Filling text paths is not necessarily a good idea because some characters have holes (e.g., the letter 'o'). The `bgText` argument can be used to specify a colour to use to fill these holes; this argument can actually be a named vector of colours so that, e.g., the hole in an 'o' is filled white, but the dot on an 'i' is filled black. Finally, there is a `sizeByWidth` argument which controls whether text is sized based on the traced width or based on the traced height of the original text.
- **symbolize** `signature(object = "PictureText")`: convert to a grid grob (for drawing as a data symbol). Currently draws nothing.
- **drawPath** `signature(object = "PictureText")`: draws a traced piece of text as part of the current plot.

Author(s)

Paul Murrell

See Also

`Picture-class`, `PictureChar-class`, and `grid.picture`, `grid.symbols`.

`PostScriptTrace` *Convert PostScript file to RGML file*

Description

Converts a PostScript file into an RGML file, which is an XML document describing an image that can be read into R.

Usage

```
PostScriptTrace(file, outfilename,
                       charpath=TRUE, charpos=FALSE,
                       setflat=NULL, encoding="ISO-8859-1")
```

Arguments

`file`	The name of the PostScript file.
`outfilename`	The name of the XML document.
`charpath`	A boolean indicating whether text in the PostScript file should be converted to vector outlines, or left as just text.
`charpos`	A boolean indicating whether text in the PostScript file should be treated as individual characters, each with its own location.
`setflat`	A number that controls how many straight lines a curve is broken into. Lower values break a curve into more lines.
`encoding`	A character value giving the encoding for the resulting RGML file.

Details

This function calls ghostscript to do the conversion, so will only work if ghostscript is installed on your system.

If text is converted to outlines, it can be drawn as outlines, or filled using a crude algorithm which may or may not work (see `grid.picture`). On the other hand, if text is left as just text, font information is not stored so the text may not end up looking much like the original.

Ghostscript is only going to emit text in a single-byte encoding, so the RGML file should have an explicit encoding (otherwise XML parsers are likely to assume UTF-8 and barf on any non-ASCII text). The default encoding used is ISO-8859-1 (ISOLatin1), but you can specify another if you know that the source file has text that ghostscript will emit in a different encoding. There may have to be a bit of guesswork, or inspection of the source PostScript file.

References

`http://www.cs.wisc.edu/~ghost/`

`readPicture` *Import an RGML file*

Description

This function reads in an RGML file (produced by `PostScriptTrace`) and creates a `"Picture"` object.

Usage

```
readPicture(rgmlFile, ...)
```

Arguments

`rgmlFile`	The name of the RGML file.
`...`	Arguments that are passed to `xmlTreeParse()` for reading the RGML file.

Value

An object of class `"Picture"`.

Author(s)

Paul Murrell

See Also

`PostScriptTrace`

symbolize *Convert a Picture into a Grid Grob*

Description

Converts an entire "Picture" object, or a path element thereof, into an appropriate grid grob. Not usually called directly by the user; usually called by grid.picture.

Usage

```
symbolize(object, x = unit(0.5, "npc"), y = unit(0.5, "npc"),
          size = unit(1, "npc"), units = "npc", ...)
```

Arguments

object	A "Picture" object.
x	A numeric vector or unit object specifying x-values.
y	A numeric vector or unit object specifying y-values.
size	A numeric vector or unit object specifying symbol size.
units	The default units to use if x, y, or size are specified as simple numeric values.
...	Arguments passed on to the relevant grid grob constructor function.

Value

A grid gTree (when given a whole "Picture") or a "symbolFill" or "symbolStroke" object (when given a single picture path).

Author(s)

Paul Murrell

See Also

grid.picture and symbolize

Index