

The GDD Package

October 19, 2008

Version 0.1-13

Title GD device for creating bitmap graphics as jpeg, png or gif files using gd library

Author Simon Urbanek <Simon.UrbaneK@r-project.org>

Maintainer Simon Urbanek <Simon.UrbaneK@r-project.org>

Depends R (>= 1.9.0)

Description Platform and X11 independent device for creating bitmaps (png, gif and jpeg).

License GPL version 2 or newer

SystemRequirements libgd (>= 2.0.29 <http://www.boutell.com/gd/>), recommended: freetype2, fontconfig and msttcorefonts

URL <http://www.rosuda.org/R/GDD/>

R topics documented:

GDD	1
GDDfont	2
Index	4

GDD

Create a new GD graphics device for creating bitmap files

Description

GDD initializes a new GD graphics device.

Usage

```
GDD(file = "plot", type = "png", width = 400, height = 300, ps = 12, bg = "white")
```

Arguments

<code>file</code>	prefix of the file(s) that will be created. The final filename will be of the form <code><file>[<sequence>].<type-extension></code> . The only case when no extension is appended is when the file already has that extension.
<code>type</code>	file format used when producing the output. The actual files are created either on close (such as via <code>dev.off()</code>) or when a new page is requested. Possible file formats currently include "png", "png24", "png8", "gif" and "jpeg".
<code>width</code>	width of the plot area in pixels.
<code>height</code>	height of the plot area in pixels.
<code>ps</code>	initial point size.
<code>bg</code>	plot background. For all types except for "jpeg", you can use "transparent" as the packground color. JPEG files don't support transparent background. Using non-white, fully transparent color can be useful to avoid clashes with existing colors when mapping the transparency.

Value

The (invisible) return value is NULL if the device couldn't be created or the raw device number if successful.

Known bugs

- Symbols are not supported yet (we need symbol font + mapping for this).
- Alpha channel is not fully supported (no one complained about this, yet).

Those bugs are rather due to lack of spare time than anything else - technically all of them are fairly easy to solve, given manpower. Volunteers are encouraged to contact me ;).

Note: Consider using the `Cairo` package instead of `GDD`, it has better rendering, symbol support and the back-end is more actively maintained.

Examples

```
## Not run:
GDD(type="jpeg", w=800, h=600)
plot(rnorm(100), rnorm(100))
dev.off() # creates a file "plot.jpeg" with the above plot
## End(Not run)
```

Description

`.GDD.font` shows the current font mapping or performs a font look-up.

Usage

```
.GDD.font (name = NULL)
```

Arguments

name	If this parameter is <code>NULL</code> then the current mapping table is returned. Otherwise this parameter specifies the name of the font to look up.
------	--

Value

If the name is `NULL` then the return value is a vector of strings. Each entry is a path to the font file or `NA` denoting that such font could not be found. First entry corresponds to the font 1 (plain), second to 2 (bold), etc. (see `font` parameter of `par` for details).

Font handling

GDD uses FreeType library for drawing text where available. However, FreeType support of GD requires full path to the font file to be specified. Therefore GDD maps R font IDs into file paths. This is done using the `basefont.config` file in the `fonts` directory of the GDD package. That file defines which files will be used by the GDD device.

The supported entries are "base.norm" for regular base font, "base.bold" for bold base font, "base.ital" for italic base font and "base.bit" for bold, italic base font. Since 0.1-8 you can also specify the symbol font using "symbol" entry (see the default `basefont.mapping` file for examples). The mapping file is processed in a greedy manner, i.e. the first match is used.

Some systems have the fontconfig library which makes finding font files somewhat easier. On such systems the actual font specification in the mappings file can be of the form `` instead of the path, e.g. `<Arial:bold>`.

The `.GDD.font` function can be used to perform fontconfig lookup or to return the currently used font files. If a font name is specified as the `name` parameter, the corresponding font file is located. If there is no fontconfig library or the lookup was unsuccessful, `NULL` is returned.

If the `name` parameter is set to `NULL`, then `.GDD.font` simply returns the list of font files currently used.

Examples

```
# return currently used fonts
.GDD.font()

# look up font closest to Times
# (returns non-NULL only if fontconfig support is enabled
# in libgd and some Times-like fonts exists)
.GDD.font("Times")
```

Index

*Topic **device**

GDD, [1](#)

GDDfont, [2](#)

.GDD.font (*GDDfont*), [2](#)

GDD, [1](#)

GDD.fonts (*GDDfont*), [2](#)

GDDfont, [2](#)

par, [3](#)