New Graphics Engine

Intro

- Attempt to separate the graphics code into Base, Engine, and Devices.
- No Base information in either Engine or Devices.
- Create (and document!) formal graphics API.
- Allows Grid and Base to coexist.
- Can add functionality to Engine more cleanly (e.g., 3D engine).
- Can create and maintain devices more cleanly (e.g., new SVG device).

Status

- Mathematical annotation and vector fonts (to be committed) completely shifted into Engine.
- Most primitives still have duplicated code; Grid and Base run separate (but VERY similar) code.
- Contour lines calculation code in Engine (to be committed), but contour line and label drawing not.
- Persp code not in Engine yet.
- Started naming convention R_GE_* and R_GD_*
- Graphics systems (i.e., Grid and Base) "register" with the Engine. This allows System state per system per device. Also allows Engine to send device "events" to all systems.

Plan

- Base graphics simply being shifted to the side for now.
- Turn Base G* functions into shells which call GE* functions. This leaves RGraphics.h untouched and plot.c untouched. This is working for mathematical annotation and vector fonts.
- Get API ready for 2.0. When 2.0 release? Longer lead in for 2.0? I'm on sabbatical second half of 2003, but travelling first 6 weeks.

Issues

- Most work on the Engine API so far to provide for Grid. Device API needs more work.
- Each version of Grid (and Lattice) will only work with a specific version of R for the forseeable future (as new API evolves).
- Device creation, destruction, and handling done by Engine.
- Registering systems with Engine a bit complex (i.e., probably a poor design). May need revision.
- Display list a problem. List structure changed to allow multiple system states (R version incompatibility). List can contain calls from multiple systems, some of which may not be loaded (run-time incompatibility). Lists can be saved and rerun in separate sessions (R version incompatibility *and* system version incompatibility).
- How should device event capture work?
- Is there more grid function registration I should be doing?
- What can the namespace stuff do for Grid?

Font Specification

Intro

- Allow font family (and lineheight) to be specified

Status

- These are passed to GEText, GEStrWidth, and GEStrHeight. Engine switches to vector fonts if font family is one of Hershey font families. Allows Grid to provide fontfamily gpar (to be committed).
- These are NOT passed to devices yet.

Plan

- Pass these to GEMathText and GEMetricInfo and implement R_GE_VMetricInfo. This would allow vector fonts to be used in mathematical annotation, which would allow mathematical annotation on ALL devices.
- Possibly change to passing graphical parameters in a structure rather than individually before doing pass to devices.
- Pass these to device, which can do what it likes, including ignoring the information. Will require device changes, which will be larger if use graphical parameter structure. Hard to avoid changes to Rdevices.h with this(?)

Double Buffering

- Which devices can/should do it? (X11, Windows, Mac, Quartz)
- Is there a common mechanism/paradigm that I can abstract? e.g., write-offscreen, swap-to-screen.

Graphics QA

- Create PostScript and PDF (because it does not clip itself) per graphics function and generate bitmaps from them to use as test output. Diff against test bitmaps for checking. Can produce XORs to view any differences.

Grid

- Functions return objects. Update of display on grid.edit() still only for single display.

Colour

- Going to produce a package of ColourBrewer palettes.
- Obvious extensions to extend the palettes if required (e.g., Cindy's suggestion of "ramping" between palette items).
- Cindy has offered to produce palettes for other purposes (e.g., palettes with highlight colours).