Inertia

A simple, yet powerful
user interface environment

History

- Thoughts of a simpler Self or Squeak
- Prototypes have been written C++, Io, Dylan and now Ruby
- Tried a few ideas like ‘Magic Lens’ live editing and window grouping
- Looking for the simplest solution

Features

- No distinction between editing and runtime modes
- Live move, resize and reparenting of all visible objects
- Drag & drop between any objects
- Model view controller with UI styles

No Edit and Run

- To design, press [Alt] then
  - click = select View
  - drag = pick up, move or resize
  - release = drop / reparent
  - right button = context menu
- [Shift] and [Ctrl] modifiers...

No Edit and Run

- [Shift] and [Ctrl] modifiers
  - [Alt] drag = move
  - [Alt] + [Shift] drag = copy
  - [Alt] + [Ctrl] drag = subclass
- To subclass Button, simply
  - drag a Button with [Alt] + [Ctrl]
  - A dialog will prompt for a name

Drag & Drop

- Drag and drop is simple:
  - def drag_accept(origin, object)
    object.instance_of? Color
    end
  - def drag_drop(object)
    @back_color = object
    end

UIStyles

- Defining a new style is simple:
  - class UIStyle::MyStyle

```
class Button < UIStyle
    def draw_content( view )
        view.draw_rect( ... )
    end
end
end
```

Core Classes

- Point
- Event
  - MouseEvent, KeyEvent, ShapeEvent
- Font
- View
  - Screen, Window

Optimizations

- Font optimizations
  - Textured, uses glCallLists(), use draw_strings() to draw multiple lines
- Display lists for
  - Textured fonts, Window shadows
- Soon... textured Windows

Techniques

- Menu buttons, checkboxes, etc.
  - Simply Buttons with ‘Menu’ ui style
- Event dispatching makes use splat
- Heavy use of setter methods
- Hierarchical event system
  - Screen handles grabbing, dragging

Implementation

- Uses SDL with OpenGL & FreeType
- So far development going smoothly
  - About 1200 lines of Ruby in 15 files
- What I like about Ruby

  simple syntax extensive library

  -rprofile setter methods