CSE 557: Impressionist

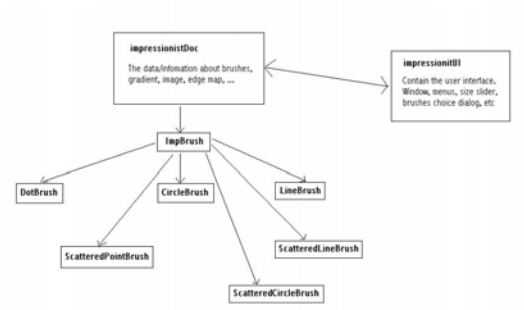
Help Session

Brett Allen

What we'll be going over

- Getting Set Up
- The Skeleton Code
- OpenGL
- Basic FLTK
- How to make a new brush
- Good References for Project 1
- Q&A

The Skeleton Code



The Skeleton Code, part deux

- impressionistDoc
  - This class handles all of the document-related stuff, like loading/saving, etc.
- impressionistUI
  - This class handles all of the UI stuff, such as getting values from sliders, setting up the window, etc.
- PaintView
  - This class handles drawing the side of the window the user paints on.
- OriginalView
  - This class handles the other side of the window.
- ImpBrush
  - This is the virtual class all brushes are derived from.
- PointBrush
  - This is an example brush that draws points.

Meet your new friend: OpenGL

- OpenGL is a great environment for PC 2d/3d graphics applications.
  - It is one among many others, such as DirectX, Glide, Allegro, etc.
- Very easy to start working with
  - It is extremely well documented.
  - Lots of online solutions available – see Google
- We will be using it throughout the quarter.
- Project 1 uses just the basics of OpenGL.
  - Although you're welcome to learn more on your own, the focus of the project is on 2d image manipulation.

How OpenGL Works

- OpenGL draws primitives—lines, vertexes, or polygons—subject to many selectable modes.
- It can be modeled as a *state machine*
  - Once a mode is selected, it stays there until turned off.
- It is procedural—commands are executed in the order they're specified.
- The coordinate system in which it draws is transformed using function calls.
  - glRotate, and why it might be confusing (right now).
  - The matrix stack.

Drawing with openGL

- That said, how to draw an actual primitive?
  - Lets do an example: a filled triangle. (why will you need this later. . .?)

First, set your color:

```
glColor3f( red, green, blue );
```

Now, tell openGL to begin drawing:

```
glBegin( GL_POLYGON );
```

Specify vertices A, B, and C. Since we're drawing in an image, use integers.

```
  glVertex2i( Ax, Ay );
  glVertex2i( Bx, By );
  glVertex2i( Cx, Cy );
```

Close the openGL block.

```
glEnd();
```

Force openGL to draw what you specified *now*.

```
glFlush();   // don't forget this!
```

FLTK

- Stands for Fast Light ToolKit.
  - A *really* handy cross-platform windowing system.
- Completely Event-driven (via callbacks).
  - The window setup code is run, and then the main loop is called. (we'll look at an example in a second)
  - All further events are handed out to callbacks.
- For those who have used Tk before, the structure of it is really similar. (I've been told)

FLTK Example code

- This code is taken/modified directly from fltk.org:

```
#include <put das junk here>
This code is executed in order:
int main(int argc, char **argv) {
  Fl_Window *window = new Fl_Window(300,180);
  Fl_Box *box = new Fl_Box(20,40,260,100,"Hello, World!");
Run functions registered to Fl_Box on the box you created:
  box->box(FL_UP_BOX);
  box->labelsize(36);
  box->labelfont(FL_BOLD+FL_ITALIC);
  box->labeltype(FL_SHADOW_LABEL);
  window->end();
  window->show(argc, argv);
This is where we hand control of our program to FLTK.  Anything that happens now is
     the result of a callback.
  return Fl::run(); }
```

Where to get FLTK Help

- References linked on web page.
  - There are a lot of function calls!!
- Widget-specific code directly commented into ImpressionistUI.cpp!
  - No help session on copying and pasting. . .
- Ask the TA

How to Make a Brush

- Now that we've got all the background, lets make a brush!
  - And because I'm mean, lets make one that isn't required. ☺ Presenting. . .triangleBrush!
- Because we're lazy, lets make a copy of pointBrush.h/cpp and rename them triangleBrush.h/cpp.
- Add them to the impressionist project.
- Go through the code and change all pointBrush labels to triangleBrush.

Brushmaking, continued. . .

- Now, open up impressionistDoc.cpp
- Add triangleBrush.h to the includes
- Scroll down a bit, and add triangleBrush to the selectable brushes. Pick a constant for it.
- Go to ImpBrush.h and add the constant for triangleBrush to the enum.
- Go to impressionistUI.cpp, and add the triangle brush to the brush menu.

Brushmaking, continued again

- Run Impressionist. See the triangle brush.
  - And, well, see the triangle brush make points instead of triangles.
- Open triangleBrush.cpp and go to BrushMove.
  - Here's what's there now:

```
glBegin( GL_POINTS );
      SetColor( source );

      glVertex2d( target.x, target.y );
glEnd();
```

- Triangles need 3 vertices. Lets center ours around the target point where the user clicked.
- How do we do this?

Brushmaking, continued again

- We do it like so:

```
int size = pDoc->getSize();

int Ax,Ay,Bx,By,Cx,Cy;

Ax = target.x - (.5*size);
Bx = target.x + (.5*size);
Cx = target.x;
Ay = target.y - (.5*size);
By = target.y - (.5*size);
Cy = target.y + (.5*size);

glBegin( GL_POLYGON );
  SetColor( source );
  glVertex2i( Ax, Ay );
  glVertex2i( Bx, By );
  glVertex2i( Cx, Cy );
glEnd();
```

Good References

- Books around the lab and on Web
  - The Red/Blue OpenGL Bibles, and Erik's books
- Class Web
  - Lots of references linked there
- Google – of course
- www.fltk.org
- www.opengl.org
- Me

Questions. . .?

- Ask 'em now. . .
- . . . Or email me (allen@cs) later. . .
- . . . Or check the web page for good ways to contact your staff.