Software Development: Building Large Programs

Strategies for Developing Large Programs

- Divide and conquer
  - Split programs into smaller units of functionality
    - Develop and test each piece of functionality
      - Could be that you develop A and B independently, and then combine
      - Or, could develop A and then develop B that uses A
      - KEY POINT: test each component and make sure it works before moving on

Strategies for Developing Large Programs

- Encapsulation
  - Separate units of functionality from each other and control what information is shared
  - Aids in divide and conquer approaches
  - Makes it easier to maintain software
  - Allows functionality to be reused and shared
  - Strategies
    - Write independent functions
    - Organize functionality in classes
    - Build libraries of related functionality

Strategies for Developing Large Programs

- Use Libraries
  - More in a moment…

Avoid "Magic Numbers"

- "Magic numbers" are numbers that are hard coded or fixed within your code.
  - Meaning is not clear
  - e.g.
  - if(num > 18)
  - rect(14, 300, 12, 12);
- Especially problematic if number is repeated many times
- Hard to adjust code
- Difficult to read
- Replace with a constant or variable

Tabs and multiple files

- Avoid writing hundreds or thousands of lines within one file
- Break into modular parts
  - Better organization
  - Easier to understand and manage
- Sketches can have multiple files
- One file per tab

Tabs and multiple files

- Arrow at right of environment controls tabs
  - Create new tabs
  - Rename
  - Delete
- Can hide tabs
- All non-hidden tabs will be included when code is run

Libraries

- Code written by others that you can use in your programs
- Don't reinvent the wheel!
  - Save work
  - Use code that has been heavily debugged
- Some (often minor) costs
  - May or may not have access to the source
  - May not do exactly what you want
  - Can take time to learn the logic of another programmer

Built-in Libraries

- The processing core library is automatically included
  - Defines the functions and predefined variables we've been using
  - e.g. fill(), mouseX
- Must include other libraries with command at top of sketch:

import processing.opengl.*;

- Can also use Sketch->Import Library

Built-in Libraries

- Core libraries include:
  - opengl: supports hardware accelerated, 3D graphics
  - serial: serial communication with external devices
  - network: client server sketches over internet
  - pdf: high quality pdf output
  - xml: parsing xml docs
  - Video
  - sound

Contributed Libraries

- Written by other users
- For download and installation instructions, see Shiffman Chpt. 12
- Write your own and contribute them!

Running a Processing Program

1. Run in the processing environment
2. Create a standalone application
   - File->ExportApplication
   - Supports Windows, Mac and Linux
   - Can run application directly
   - Very few restrictions on what an application can do

Running a Processing Program

3. Export to a webpage
   1. Click on "JAVA" at the right of the processing environment
   2. Select "Add Mode…"
   3. Add the mode p5.js to run code in your browser
      - For more information, see:

        https://p5js.org/get-started/

Running a Processing Program

4. Create an Android application

   1. Click on "JAVA" at the right of the processing environment
   2. Select "Add mode…"
   3. Select "Android Mode"

   http://android.processing.org/