Lecture 1: Fundamentals & Graphics

CS 51G
Spring 2018
Kim Bruce

TAs: Alejandro Salvador, Joe Brennan, & Cleo Forman
Course web page: http://www.cs.pomona.edu/classes/cs051G

Who we are:

Kim Bruce

Cleo Forman

Joe Brennan

Alejandro Salvador

Read syllabus
On-line more complete
and up-to-date

1st lab Friday

Course Description

- Object-oriented programming in Grace
  - New language designed for teaching novices.
  - 4th offering using Grace
    - More expressive, fewer confusing parts than Java, C++, etc.
- Use locally-written library to explore graphic applications, animations, event-driven programming
  - Fun programming projects
  - Serious skills in solving practical problems

Why 51G (Grace) over 51J (Java)?

- Grace has simpler syntax and semantics
- Same material (and projects), but fewer unnecessary complications in Grace
  - How learn to fly stealth bomber (or any complex skill)?
- Preliminary studies show students do better when learning Grace.
- Text in Grace is free, text in Java is ~$150 new.
- Will learn Java anyway in CS 62
- Students learn more in smaller classes
  - Max of 20 students in 51G

Why Java over Grace?

- You feel professor is not being paid enough and deserves the $5 profit earned for every new Java book sold.
- You think the prof in 51J is better than 51G professor.
- You believe doing things the hard way builds character.
- You think it would be cooler to learn a language designed 23 years ago rather than one designed in the last 5 years.
- It's absolutely necessary for you to learn Java as quickly as possible even if learn less well.
- You feel that learning Java syntax now will give you a leg up when learning Java in CS 62.
- You don't think you can possibly get up for a 9 a.m. class!

Languages In Use

- If want to program
  - Mac or iOS: Objective C or Swift
  - Windows: C#
  - Android: Java
  - Web: Javascript, php, or Dart
  - Systems: C, C++, Rust, or Go
  - Scientific Programming: Fortran, C, Python, Matlab, R, or ...?

Is this course for you?

- Intended for student with no or little programming experience
  - If took AP CS then over-qualified.
    - Take 52 or 54
    - Talk to me if not sure

Labs

- Weekly labs (required!)
  - Start w/quiz *(not this week)*
  - Work on weekly homework, due Monday night at 11 p.m.
  - Most learning takes place in labs.
    - Lectures focused on helping learning ideas used in lab
  - Two times during semester can turn in lab 1 day late
    - Save for emergencies. No other extensions unless prolonged illness
  - Test programs before spring break and final
    - Count 20% of grade each. (Midterm exam 15%, final exam 25%)

How to Succeed

- Read lightly sections to be covered before class
  - Don’t need to understand it all, but get questions
- Annotate printed notes in class
- Afterwards read sections carefully, work out problems at end of chapters & on syllabus
  - Focus in class on examples, not details of syntax.

Academic Honesty

- Has been problem in CS 51 in past
  - Easy to catch. Don't do it!
- See statement on syllabus!
- Close to half of labs will be in pairs.
- Ask instructors and mentors for help!

Algorithms

- Set of instructions to solve general problem
- Humans good at carrying out bad commands
  - Computers are stupid, too literal!
- Learning chess
  - Is learning how pieces move enough?
  - How do you become good at it?

Writing Grace Programs

- Use Chrome web browser
  - http://www.cs.pomona.edu/~kim/minigrace/
  - Write in editor pane.
  - When push run, compiles into javascript & executes
- First program:
  - MakeBox: http://www.cs.pomona.edu/classes/cs051G/demos/MakeBox/MakeBox.grace

MakeBox Program

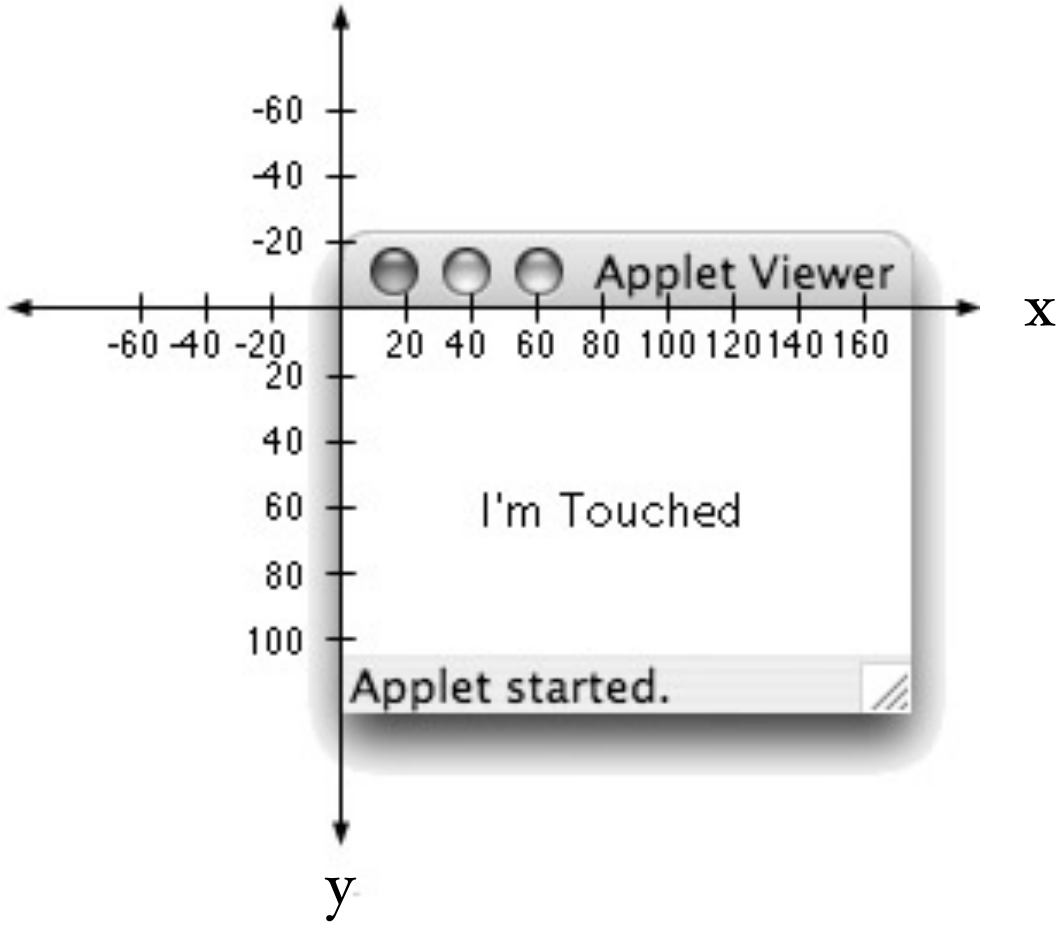
```
dialect “rtobjectdraw”                          Set dialect to use graphics library
object {
    inherit graphicApplicationSize(400 @ 400)   Ready window of size 400 by 400

    windowTitle := "Hello World Program!"                 set title bar

    ...                                                Do something interesting!

    startGraphics
                                                                  Show the window
}
```

Computer Graphics



Origin in upper-left corner of screen
y-coordinates go down!

Write coordinates as 40 @ 60

One pixel = 1 dot on screen
screen roughly 1000 x 2000 pixels

Graphics Primitives

- Create all graphics objects on screen with objectdraw library classes:
  - framedRectAt (100 @ 250)size (20 @ 50) on (canvas),
  - filledRectAt (100 @ 250) size (20 @ 50) on (canvas)
  - framedOvalAt (100 @ 250) size (20 @ 50) on (canvas)
  - filledOvalAt (100 @ 250) size (20 @ 50) on (canvas)
  - textAt (100 @ 250) with (“hello!) on (canvas)
  - lineFrom (100 @ 250) to (120 @ 300) on (canvas)

Mouse Event-Handling

- Methods:
  - method onMousePress (pt: Point) -> Done {...}
  - method onMouseRelease (pt: Point) -> Done {...}
  - method onMouseClick (pt: Point) -> Done {...}
  - method onMouseMove (pt: Point) -> Done {...}
  - method onMouseDrag (pt: Point) -> Done {...}
  - method onMouseEnter (pt: Point) -> Done {...}
  - method onMouseExit (pt: Point) -> Done {...}

Code in {...} is executed when event occurs.
pt is location of mouse when event occurs.

Naming Objects

- Can associate names with objects:
  - def box = filledRectAt (20 @ 20) size (100 @ 100) on (canvas)
- Use names to change properties
  - box.color := colorGen.red
- See UpDown program:
  - http://www.cs.pomona.edu/classes/cs051G/demos/UpDown/UpDown.grace