CS 112 Introduction to Programming

Animations

Yang (Richard) Yang
Computer Science Department
Yale University
208A Watson, Phone: 432-6400
Email: yry@cs.yale.edu
Outline

- Admin and recap
- Animations
Admin

- PS3 questions?
- PS4 to be posted on Wednesday
Method Design Heuristics

1. The main method should read as a concise summary of the overall set of tasks performed by the program.

2. Each method should have a clear set of responsibilities.

3. No method should do too large a share of the overall task.

4. Use method with parameters to remove redundancy with generalization, but do not over generalize

5. Use named constants (final) whenever possible to make the program easier to read/maintain

6. Data should be declared/used at the lowest scope possible (localized data).
Recap

- Java graphics
  - Coordinate system
    - setCanvasSize, setXscale, setYscale
  - Basic drawing shapes
    - line, square, text
  - Color
    - setPenColor(R, G, B)
    - setPenColor(Color)
    - Same method, different parameters are called overloaded methods, which we will cover on Wednesday

- Parameterized drawing, drawing using loops
Book Cover (Color and Loop)

- White 500x600 drawing panel
- Three components at
  - (20, 415), (165, 330), (220, 85) with sizes 150, 120, and 240
  - Each component
    - Yale blue background
    - white “CS112” text left @ 1/2 horizontal, 4/5 vertical
    - 10 brown (red=192, green=128, blue=64) “bricks”
      - 2 pixel between two adjacent bricks

BookCover.java
Recap: Parameterized Drawing

- Write method
  ```java
drawCar(x0, y0, size):
```

- Center: 
  \[(x0 + 0.5 \text{ size}, y0 + 0.25 \text{ size})\]

- Size: 0.5 size, 0.25 size

- Center: 
  \[(x0 + (1.00 - .15 - .10) \text{ size}, y0)\]

- Radius: 0.10 size

- Center: 
  \[(x0 + (1.00 - .30/2) \text{ size}, y0 + .50/2 \text{ size})\]

- Size: 0.15 size, 0.10 size

- Center: 
  \[(x0 + (.15 + .10) \text{ size}, y0)\]

- Radius: 0.10 size
Outline

- Admin and recap
- Animations
Question: Car Launch

- Two cars launching from different heights, at different speeds, compute their positions in the initial 10 sec.

**Car 1: One World Trade Center**
- initial height: 541 m
- speed (210 km/h), with
  - horizontal speed: 50 m/sec
  - vertical speed: 30 m/sec

**Car 2: Empire State**
- initial height: 381 m
- speed (114 km/h), with
  - horizontal speed: 30 m/sec
  - vertical speed: 10 m/sec
In physics, for each dimension (x or y), given initial velocity $v_0$ in m/s, acceleration $a$ in m/s$^2$, and elapsed time $t$ in s:

- The speed of the moving body at time $t$:
  $$ V = v_0 + a \cdot t $$

- The displacement of the moving body at time $t$:
  $$ \text{Displacement} = v_0 \cdot t + \frac{1}{2} a \cdot t^2 $$

- The position of the moving body at time $t$:
  $$ \text{Pos} = \text{initial position} + \text{displacement} $$
Position

- \( p_0 + v_0 \, t + \frac{1}{2} \, a \, t^2 \)

- **Horizontal (x):**
  - \( v_{x0} \, t \)

- **Vertical (y):**
  - \( g \approx 9.81 \, \text{m/s}^2 \), downward
  - \( h_0 + v_{y0} \, t - \frac{1}{2} \, g \, t^2 \)
CarLaunch: Initial Version

// You must have StdAudio.java and race-car.wav in the // same directory and first compile StdAudio.java.
StdAudio.loop("race-car.wav");

// set up the initial state of the two cars
int h1 = 541, v1x = 50, v1y = 30;
int h2 = 381, v2x = 30, v2y = 10;

// Simulate time from 0 to 10 sec.
for (double t = 0; t < 10; t += 0.1) {
    // Compute car 1's position
    double x1 = v1x * t;
    double y1 = h1 + v1y * t - 0.5 * 9.81 * t * t;

    // Compute car 2's position
    double x2 = v2x * t;
    double y2 = h2 + v2y * t - 0.5 * 9.81 * t * t;

    // Use the method defined in Car.java
    StdDraw.picture(x1, y1, "angry-bird-b.png");
    Car.drawCar(x2, y2, CAR2_SIZE);
} // end of for

You can use StdAudio to add sound

One can mix param. drawing and fixed image file
You must have StdAudio.java and race-car.wav in the same directory and first compile StdAudio.java.

```java
StdAudio.loop("race-car.wav");

// set up the initial state of the two cars
int h1 = 541, v1x = 50, v1y = 30;
int h2 = 381, v2x = 30, v2y = 10;

// Simulate time from 0 to 10 sec.
for (double t = 0; t < 10; t += 0.1) {
    // Compute car 1's position
    double x1 = v1x * t;
    double y1 = h1 + v1y * t - 0.5 * 9.81 * t * t;

    // Compute car 2's position
    double x2 = v2x * t;
    double y2 = h2 + v2y * t - 0.5 * 9.81 * t * t;

    // Use the method defined in Car.java
    Car.drawCar(x1, y2, CAR1_SIZE);
    Car.drawCar(x2, y2, CAR2_SIZE);

    StdDraw.clear();
}  // end of for
```
CarLaunch: using 
 StdDraw.show(t) for Timing

// FRAME_T = 60;
int h1 = 541, v1x = 50, v1y = 30;
int h2 = 381, v2x = 30, v2y = 10;

for (double t = 0; t < 10; t += FRAME_T/1000.0) {
    double x1 = v1x * t;
    double x2 = v2x * t;

    double y1 = h1 + v1y * t - 9.81 * t * t / 2;
    double y2 = h2 + v2y * t - 9.81 * t * t / 2;

    Car.drawCar(x1, y1, CAR1_SIZE);
    Car.drawCar(x2, y2, CAR2_SIZE);

    StdDraw.show(FRAME_T); // hold image for 60 ms
    StdDraw.clear(); // now clear up
}
StdDraw.show(t) and Animation

- StdDraw.show is overloaded
  - StdDraw.show(int t)
    - Display on screen, pause for t milliseconds, and **turn on animation mode**: subsequent calls to drawing methods such as line(), circle(), and square() will not be displayed on screen until the next call to StdDraw.show.
  - StdDraw.show()
    - Display on-screen and **turn off animation mode**: subsequent calls to drawing methods such as line(), circle(), and square() will be displayed on screen when called.
On-Screen

- The display of your computer is just a mirror of the content of the Video Memory

http://www.brackeen.com/vga/unchain.html
On-Screen (Video Memory) and (Double) Buffer

In animation mode, drawing is in the (double) buffer.

Next show(t) statement copies to screen.

When finished drawing, the double buffer is copied to video memory.

[Link to more information: http://www.brackeen.com/vga/unchain.html]
Example: AnimationShow.java

- Try running the example in Debug mode
  - Right click first line and toggle breakpoint
  - Execute in Debug mode
  - Execute (step over) each statement
Exercise: Add a Countdown Scene

- Count down from 10 to 0 and then start the race
Exercise: Add a Countdown Scene

- Count down from 10 to 0 and then start the race

```java
public static void sceneStart(int h1, int h2) {
    for (int t = 10; t >= 0; t--) {
        Car.drawCar(0, h1, CAR1_SIZE);
        Car.drawCar(0, h2, CAR2_SIZE);
        StdDraw.text(WIDTH/2, HEIGHT/2, "" + t);
        StdDraw.show(1000);
        StdDraw.clear();
    }
}
```