Introduction to Jam’s Video Game Package

Scope of the Video Game Package

- Goals of Jam’s Video Game Package
  - Simple to use
  - Reasonably fast
  - Designed to be examined and modified for academic purposes in computer science courses

- What Jam’s package is not designed for
  - High speed animation/user input
  - Scripting games
  - Complex games

Basic Design of the Video Game Package

The Plan

- Scope of Video Game Package
- Basic Design of the Video Game Package
- Steps to making a game
- How the Pong was made
- Limitations of the Video Game Package
Basic Design of the Video Game Package

Tracker

The Tracker is what makes the BallSprite move

Basic Design of the Video Game Package

• Point2D.Double getLocation()
• double getScaleFactor()
• double getRotationAddition()
• void advanceTime()

Basic Design of the Video Game Package

Sprite
Tracker

Has instance variables, and mutator and accessor methods for:
• Shape
• Location
• Size
• Rotation
• Color
Basic Design of the Video Game Package

AnimationCanvas is a JPanel
- with a collection of Sprites
- that paints itself by painting all of its Sprites

GameLoop adds:
- animation
- interaction via
  - Keyboard
  - Mouse

JFrame or Applet used to:
- Put the game on the screen
- Start the game
- Provide game control
Basic Design of the Video Game Package

1. Extend GameLoop.java
   - In Constructor
     a. Make Sprites
     b. Make Trackers
     c. Attach Trackers to Sprites
     d. Add Sprites to AnimationCanvas
   - In advanceFrame provide game logic
2. Make a JFrame or Applet
3. Add extended GameLoop to JFrame or Applet
4. setFrameRate to start the game

Steps to Making a Game

1. Extend GameLoop.java
   - In Constructor
     a. Make Sprites
     b. Make Trackers
     c. Attach Trackers to Sprites
     d. Add Sprites to AnimationCanvas
   - In advanceFrame provide game logic
2. Make a JFrame or Applet
3. Add extended GameLoop to JFrame or Applet
4. setFrameRate to start the game

Basic Design of the Video Game Package

1. Extend GameLoop.java
   - In Constructor
     a. Make Sprites
     b. Make Trackers
     c. Attach Trackers to Sprites
     d. Add Sprites to AnimationCanvas
   - In advanceFrame provide game logic
2. Make a JFrame or Applet
3. Add extended GameLoop to JFrame or Applet
4. setFrameRate to start the game

How Pong was Made

1. BallSprite was written to extend Sprite
2. WallSprite was written to extend Sprite
3. ProjectileTracker was written to implement a Tracker
4. PongLoop was written to extend GameLoop
5. Pong was written to use PongLoop
How Pong was Made

Some of the more complex parts of Pong that we’ll talk about in more detail later:
- Loops
- Conditionals
- Event handling
- Inheritance & Interfaces
- Collections
- Collision detection

How Pong was Made

What you will need to know shortly:
- Basic classes in the package
- Basic structure of the package
- Basic steps to making the game
- General idea about how Pong.java and PongLoop.java work
- Enough familiarity to make minor modifications to BallSprite and PongLoop

How Pong was Made

What you don’t need to understand yet:
- All of the classes in the package
- All of the code in the basic classes
- How to make your own classes like Pong and PongLoop from scratch

How Pong was Made

What you’ll be doing in the next homework:
(in addition to problems from Big Java)
- Altering BallSprite
- Altering ProjectileTracker

…and it will be fun (when it works)
How Pong was Made

Let’s take a look at two classes:

1. Open up a web browser
2. Go to the course website and download pong.jar to the desktop (from code link)
3. Open up Eclipse
4. Import pong.jar
5. Look in tipgame for BallSprite.java
6. Look in tipgame for ProjectileTracker.java