



## CS 101

### Problem Solving and Object-Oriented Programming

#### Lab 5 - Racing Mohos

Code due: November 6/7/8/9, 11:55 PM

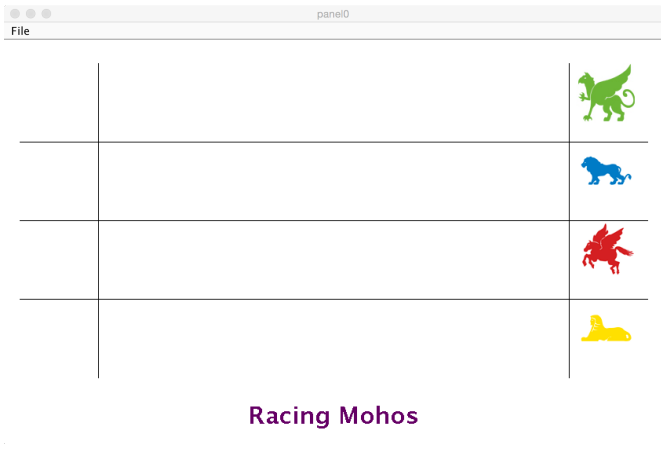
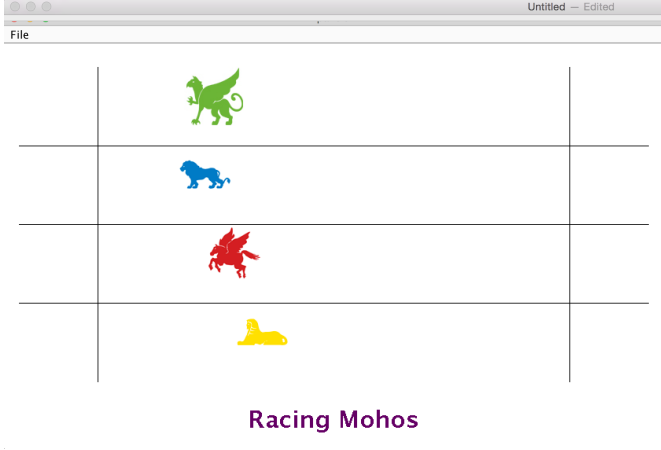
#### Goals:

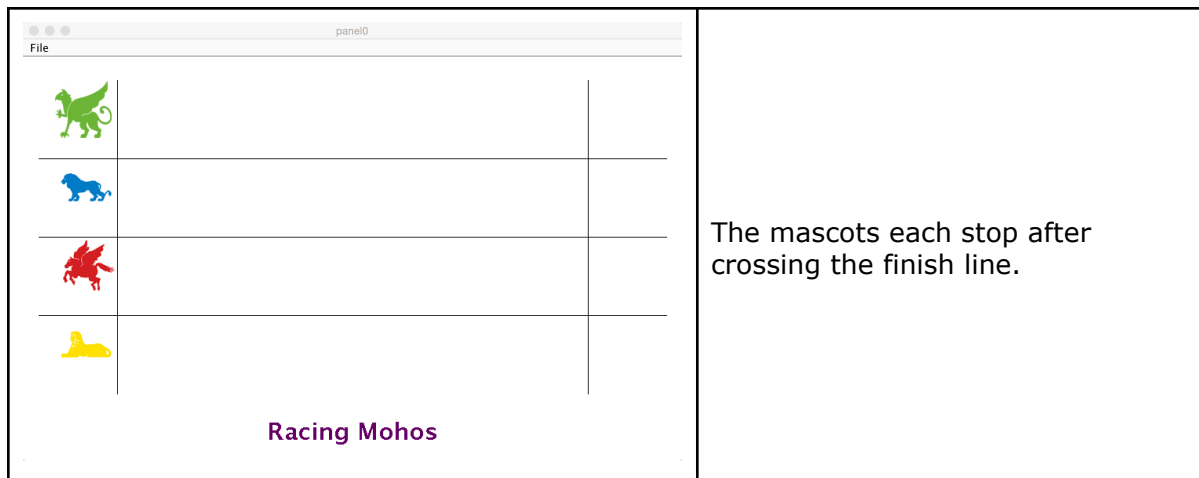
- To work with active objects
- To gain experience reading APIs

#### Introduction to the Assignment

In the RacingMohos program, mascots race across the screen as shown below. They start at the right edge of the screen and move independently and at random speeds across, stopping when they cross the finish line.

Here are some snapshots from this program:

|   |   |
|---|---|
|  <p>Racing Mohos</p>  | <p>The start of the race</p>  |
|  <p>Racing Mohos</p> | <p>When the user clicks the mouse, the race starts with each mascot moving at a random speed across the screen.</p> |



## The Program Design

This program contains 3 classes: RacingMohos, Mascot and Picture. RacingMohos is the controller class that draws the starting screen and handles the mouse click that starts the race.

Mascot is the ActiveObject class that manages the motion of an individual mascot. The RacingMohos class will construct 4 Mascot objects, one for each of the racers.

Finally, Picture is the same class you used last week. Copy it from lab 4 into this project.

In this lab handout, we have added a final section that shows the API for classes and methods that you may find useful when writing this program. You should be able to use this information to determine how to call the methods and constructors you need to use.

## Writing the Program

Create a new BlueJ project called Lab5 in your dev/cs101/ folder. The icons for the mascots are available at:

- [http://www.mtholyoke.edu/~blerner/green\\_griffin.jpg](http://www.mtholyoke.edu/~blerner/green_griffin.jpg)
- <http://www.mtholyoke.edu/~blerner/lion.png>
- <http://www.mtholyoke.edu/~blerner/pegasus.png>
- <http://www.mtholyoke.edu/~blerner/sphinx.png>

### Step 1: Displaying the racing lanes

For the first step, you should draw the lines that make up the race track and place the text at the bottom of the window. Your window should be 840 pixels wide and 560 pixels tall. Each lane should be 100 pixels tall. The icons are all 75 pixels tall and 75 pixels wide.

### Step 2: Displaying the mascots

For the second step, use the Picture class as you did last time to construct Pictures and then turn them into VisibleImages. These should appear on the screen when the program starts.

### Step 3: Creating an animation

When the user clicks the mouse, you should create 4 mascots and start them racing. Your Mascot class should extend ActiveObject. The Mascot constructor needs 2 parameters: the VisibleImage that represents the mascot and the x location of the finish line so that the mascot will know when to stop running.

Initially, just pick a constant speed for your mascots to run. Recall that the constructor for an active object will need to save some information in instance variables so the run method will be able to use the information. It should then start the animation by calling ActiveObject's start() method, which has the signature:

```
public void start()
```

When you call start(), Java does some work to set up for the animation to execute and then calls the run method that you define, which should have this signature:

```
public void run()
```

The run method will contain a while loop. In each iteration of the while loop, it will move the mascot a short distance to the left and then pause. The while loop should end when the mascot has moved completely beyond the finish line.

#### **Step 4: Running the racers at different speeds**

It's not an interesting race if all the mascots run at the same speed. As the movie on the website shows, each racer's speed changes as the race proceeds as well. The easiest way to randomize the speed is to randomize the distance that the mascot moves each time through the while loop. To do this, your Mascot class should construct a RandomIntGenerator that ranges from some minimum distance to some maximum distance that makes for an interesting race.

#### **Step 5: Adding the text**

As a finishing touch, Add the Racing Mohos text to the bottom of the screen. The API section at the end of this document includes a list of methods that you may find useful when adding the text. To center text precisely, you should first create the text and set the font to what you want. Then, call getWidth() on the text so that you can use that information in your centering algorithm to move the text to the correct location (**Hint - Hard coding the position of the text is not a good idea! Make sure that the text can dynamically adjust itself to the center of the window**).

#### **Extra credit: Announcing the winner**

For extra credit, change the text that appears in the window to indicate which racer won the race. You will need to make some changes to both classes to get this to work. Hint: the Mascot class will need to call a new method in the RacingMohos class. This means that the Mascot will need to have a variable that refers to the RacingMohos class, which, in turn, means that the RacingMohos object will need to pass itself as a parameter to the Mascot constructor. There is a special keyword "this" that allows an object to refer to itself. You will need to pass "this" to the Mascot constructor. You can learn more about how to use "this" on pages 165-6 of our text.

## Grading

|            |  |
|------------|--|
| 20         | Design   |
| 5          | Step 1: Displaying the racing lanes            |
| 5          | Step 2: Displaying the mascots                 |
| 10         | Step 3: Creating an animation                  |
| 10         | Step 4: Running the racers at different speeds |
| 5          | Step 5: Adding the text                        |
| 5          | Compiles Without Error                         |
| 5          | Comments                                       |
| 4          | Constant declarations                          |
| 4          | Indentation                                    |
| 5          | Variable names                                 |
| 5          | Variable scopes                                |
| 5          | If statements and conditions                   |
| 5          | While loops                                    |
| 5          | Method design                                  |
| 2          | Folder Naming                                  |
| <b>100</b> | <b>Total</b>                                   |
| <b>5</b>   | <b>Extra credit: Announcing the winner</b>     |

## API

### ActiveObject Class

```
public void start()
```

Causes the ActiveObject to start its animation

```
public void pause(int millis)
```

Suspend the animation for millis milliseconds

### Color Class

```
public Color (int redness, int greenness, int blueness)
```

Creates an RGB color with the given red, green and blue values.

### Line Class

```
public Line (int startX, int startY, int endX, int endY,  
            DrawingCanvas canvas)
```

Creates and draws a line with the given endpoints.

```
public Line (Location start, Location end, DrawingCanvas canvas)
```

Creates and draws a line with the given endpoints.

### **Picture Class**

```
public Picture (String url)
```

Reads a picture from a URL. It does not display the picture.

```
public VisibleImage createVisibleImage (int left, int top, DrawingCanvas canvas)
```

Creates and displays the picture at the given location.

### **RandomIntGenerator class**

```
public RandomIntGenerator (int min, int max)
```

Creates an object that will generate random numbers between min and max.

```
public int nextValue()
```

Returns a random number in the range used when the RandomIntGenerator was constructed.

### **Text Class**

```
public Text (String text, int x, int y, DrawingCanvas canvas)
```

Constructs and displays text at the given location.

```
public Text (String text, Location origin, DrawingCanvas canvas)
```

Constructs and displays text at the given location.

```
public double getWidth()
```

Returns the width of the text.

```
public void moveTo (int x, int y)
```

Move the Text to a specific x, y coordinate

```
public void moveTo (Location point)
```

Move the Text to a specific location

```
public void setBold(boolean bool)
```

To make text bold, pass true. To make text non-bold, pass false.

```
public void setColor (Color c)
```

Changes the color the text is displayed with

```
public void setFontSize(int size)
```

Changes the font size used by the text.

```
public void setText (String text)
```

Changes the string that is displayed in the text.

### **VisibleImage class**

```
double getWidth()
```

Returns the width of the visible image

```
double getX()
```

Returns the x value of the left edge of the visible image

```
public void move (int dx, int dy)
```

Moves the visible image right by dx pixels and down by dy pixels.

### **WindowController Class**

```
public void resize (int width, int height)
```

Changes the window size to the size passed in.

### **Turning in Your Work**

Your work will be automatically collected from your dev/cs101/Lab5 folder at the time that it is due. Please be sure your files are in the right place.

Over the course of the semester, you may have up to 5 late days total. Use them wisely! To request a late day, fill out the Google form on the course website so that we do not collect your assignment when it is due.