Lab: Structures, Variants, and Universe Programs

COMP 50

Fall 2013

This lab has three purposes:

- To give you more practice with `big-bang`
- To help prepare you for your homework problem building and interactive user-interface experiment for Fitts's Law
- To expose you to iterative development and refinement

Please consult the handout *Design Guidelines for `big-bang` Programs*.

Start of lab

Begin by reviewing the design handout mentioned above and the circle-drawing program from last time. To review the program, run

```
drracket /comp/50/www/solutions/lab-circles.rkt
```

and look over the code. If you have questions, ask the course staff.

Managing the design process

In this lab you will be building a `big-bang` program. Please check in with the lab staff at the following points:

1. When you have *drawn scenes from the program*
2. When you have *written a data description* for *world states*
3. When you have a *wish list* for functions (including signature, purpose statement and header for each function)
4. When you have a *tested* implementation that you are ready to refine

Whack-a-mole

You will be developing a game of Whack-a-Mole:

- At random intervals, a mole (represented as a circle) appears on the screen. Click it before it disappears, and you score a point.
- A running score is displayed in one corner of the screen.
- After a certain number of moles, the game ends.

Design requirements:

- The human brain likes the unpredictable. Are there other elements besides the time interval that should vary randomly?
- To maintain a suitable level of fun, it has to be *easy* to adjust factors like how big the circles are, how long they last before disappearing, and how many chances you get before the game is over.

Domain Knowledge: Calling `(random n)`, where `n` is an exact integer, returns a nonnegative integer less than `n`. All candidates are chosen with equal probability.

Design refinements

Real programs evolve over time. Here are some suggestions for refining your program:

- Make it possible for the player to pass a parameter that adjusts the difficulty of the game?
- Maybe circles should come in multiple sizes chosen randomly?
- Maybe the difficulty should adjust itself to the player's past performance? (*Oblivion* did something like this.)
- Would the game be more fun with a better image of a mole than just a circle? How would you tell when a mole got clicked on?

Required refinement: Whack-a-mole with Jumbos

Change the game so that on rare occasions, what pops up is not a mole but Jumbo. If a player clicks on Jumbo, the player *loses* 10 points.

What to submit

Ten minutes before the end of the lab, put the following text at the beginning of a DrRacket file. Use the source code you have been working on:

```
#|
What I did during this lab:
   (you fill in this part)

What I learned during this lab:
   (you fill in this part)
|#
```

The lab staff will help you articulate what you learned.

Finally, *submit this file through the handin server* as `lab3`. You will need to submit it using *two* usernames connected by a + sign, as in

```
Jane.Doe+Richard.Roe
```

You submit using Jane Doe's password.