# Homework 2

# Combo-Boxes and Text Controls

due before class on **September 19th**

## Outline

In this assignment, students will create an application that allows them get information about a number, and that allows them to play a trivial guessing-game.

Students will turn in their submissions using iCollege. Students should submit a zip-file containing their project directory. Students should ensure that their code is submitted in a buildable form.

## The NumberInformation Widget

Students will create a NumberInformation widget. On one row, there should be a(n editable) text-line, with a combo-box next to it, and a button at the end of the row next to the combo-box.The first row, there should be a single, read-only text-*area* (that is, a multi-line plain text control). Both the text-line and the text-area should expand horizontally, so that the full width of the NumberInformation widget is used; additionally, the text-field should expand *vertically* so that it takes up all the extra space in the layout. The combo-box should have two options, “Even” and “Prime”; the button should say “Run”.

When the “Run” button is clicked, the computation selected in the combo-box should be performed on the number that the user has entered in the text-line; the result should then be *appended* to the text in the read-only text area, so that the text-area acts as a kind of log.

That is:

- If Even is selected, then you should determine if the number in the text-line is even; if it is, “x is even” should be added (on a new line) to the read-only text area; if it isn’t, then “x is odd” should be added instead.
- If “Prime” is selected, then you should determine if the number is prime; if it is, then “*x* is prime” should be added; if it isn’t, then “*x* is not prime” should be added instead.
- If the user *has not* entered a valid number in the text-line, then no computation should be performed, and “x is not a valid number, please enter a valid number!” should be added.

In each case, *x* should be replaced with the text that the user entered.

## The GuessingGame Widget

Students will create a GuessingGame widget. The guessing game should have four radio buttons, arranged into two columns; the radio-buttons should all be in one button group, so that only one can be selected at a time. Next to the radio buttons, there should be a “Guess” button; beneath the “Guess” button, there should be a read-only text-line.

The objective of this “guessing game” is to guess the “right” button, and keep track of how many times the user guesses correctly. Except we’re going to cheat a little: we don’t need to pick a “correct button” ahead of time. Instead, we’re going to simply get a random number (as with `Math.random()` in Java), and if the random number is less than 0.25, we’ll count it as a correct guess. We’ll also need to keep a count of their correct and incorrect guesses.

When the user clicks on the “Guess” button, the student should determine if the guess is correct or incorrect, and increment the appropriate count; then, the student should update the text in the read-only display, with the count of correct and incorrect guesses.

The student should Keep the “Guess” button and the score-tracking text-line against lined up against the right edge of the control, and the grid of four radio-buttons lined up against the left. If the widget is expanded, the excess space in the center should be left blank (as when an empty, always-growing cell is placed there).

# Submission

Creating two separate widgets and adding them to the same layout is simple in JavaFX, but is a little tricky in some other kits. Since we haven't talked about that in-depth yet, you do *not* need to display both widgets together. You can create both widgets separately – possibly in two entirely separate projects, if that is what is easiest for you! In that case, be sure to include both projects in your submission.

# Graduate Students

The graduate and undergraduate versions of this assignment are the same.