

# CSC108H lab – week 2

This document contains the instructions for the week 2 CSC108H lab. To earn your lab marks, you must actively participate in the lab. *You do not need to finish in the time allotted, you just need to try hard.*

Please return this handout to your TA. We use the same set of lab handouts throughout the week to save paper. We will post each handout at the end of the week.

## 1 Objectives

1. Meet your fellow lab inmates.
2. Make sure you have email forwarding set up properly.
3. Practice reading and using common computer science terms: *variable, method, instantiate, execute, declaration, initializing declaration, syntax, semantics* and *expression*.
4. Understand the difference between a void method and method that returns a value.
5. Practice using JFrames.

## 2 Icebreaker

Here you will introduce yourself to the other students in your lab. Your TA will tell you what to do.

## 3 Using the UTM Computer Lab

Sit down with your partner. The rest of these instructions call you two `s1` and `s2`. Pick which one is which. Execute the following set of instructions twice, once for `s1` and once for `s2`.

1. Log in: enter your user ID and password. Wait while the computer starts up.
2. Click on the Start button, then click on the Course Applications folder at the top, and finally click the DrJava icon. Wait while DrJava starts up; please be patient, and don't close the little window that pops up.
3. While DrJava is loading, open a web browser (like Internet Explorer). Type in the following address into the Address field:

```
http://www.utm.utoronto.ca/myaccount
```

4. We will sometimes send important email to everyone's CSC108H account. Unless this is your primary email account, you should set up "email forwarding". Log into the MyAccount page using your computing ID and password, and go to the Mail Forwarding section partway down the page. In the space, type in your favourite email address, and click **Submit Change**. Any mail sent to your CSC108H account will be forwarded to this email address. Make sure that both `s1` and `s2` do this.
5. After setting your forwarding addresses, type in the following address into the Address field of your web browser:

```
http://www.utm.utoronto.ca/submit
```

6. You will also need to submit your assignment files electronically for this course. For this lab, you will submit Assignment 0, which consists of a file called `message.txt` that contains a comment or message for the instructor of the course. To create this message, use your favorite text editor (like Notepad, for instance) to create your message, and save the file as `message.txt`. Remember where you saved it, so that you can find it when you're ready to submit.
7. Make sure both `s1` and `s2` have created their own `message.txt` files. When you're ready, click on the **Student Login** button, and enter your `userid` and password on the following page. It should list the assignments that you are able to submit for the course so far.
8. Click on **Assignment 0**, and look at the files that this "assignment" requires. There's just one in this case, but in the future there may be several. Click on the **Upload Files** button at the bottom of the page. On the page that comes up, fill in the field with the name of your message file, with full path details. Use the **Browse** button to help you, if you need it. When you've specified the file, click on **Submit Files**, and the submit page will let you know if it submitted correctly or not. Once the first person has submitted correctly, switch places and let the other person try it.
9. (If you want, you can even use DrJava to create your message file. Just type your message into the Definitions pane (top right one), and enter your message. When you're done, click on **File|Save**. A dialog box will appear. In the text field labelled **File Name:**, type "`message.txt`". Then select directory **H:** in the **Save In:** drop-down menu, and click the **Save** button.)
10. Quit DrJava and log out.

## 4 Using DrJava, playing with JFrames

Throughout the term, we will use the terms *driver* and *navigator*. Here are the definitions of the two roles:

**driver:** The person typing at the keyboard.

**navigator:** The person watching for mistakes, and thinking ahead.

Here is the most important rule for this lab:

**The navigator must not touch the keyboard or mouse.** If the navigator does type or click when they are not supposed to, the navigator will get a zero for this lab.

### 4.1 Trying JFrames

This section is hard! Don't worry if you don't finish or don't understand everything. Remember that we will be repeating this information in lecture this week and next.

In particular, don't worry if there are several other students in your lab who finish this quickly. Grab them and make them explain it to you!

Also, if you are one of those fast students, please help the others around you.

`s1` should now log in again. Throughout the lab, you'll be switching back and forth between the driver and navigator roles.

In lecture you learned about `ints` and `doubles`, variables, and how to create and manipulate `JFrames`. In this lab you and your partner will experiment with those concepts.

- These statements create a `JFrame`, make it visible, set the size, and set the title:
  - `import javax.swing.*;`
  - `JFrame j1 = new JFrame();`

```
- j1.show();
- j1.setSize(400, 600);
- j1.setTitle("Look! A window!");
```

- Use a couple of initializing declarations to instantiate four JFrames (call them j1, j2, j3, and j4), and practice calling the methods you saw in class on those objects. Here are (some of) those methods:

```
show, setSize, setTitle           getWidth, getHeight, getX, getY
```

- Discuss the syntactic and semantic differences between the methods in the left group and the methods in the right group. Make a *brief* list of the differences you can think of and show it to your TA.

## 4.2 Positioning JFrames

- Switch roles: s2 drives and s1 navigates.
- Reset the Interactions Pane.
- Use four initializing declarations to create four JFrame variables and corresponding objects, and call method `show` on all of them.
- Set the size of windows 1 and 3 to 100x200 pixels, and the size of windows 2 and 4 200x100 pixels. (Which JFrame method should you call?)

Does the horizontal or vertical coordinate come first in a call to `setSize`? Check your answer by calling `getWidth` and `getHeight` on one of the JFrames.

## 4.3 Working with the screen size.

Each JFrame has a method `setLocation(x, y)` that moves the upper left corner of the JFrame to the pixel with coordinates (x,y).

- Switch roles: s1 drives and s2 navigates.
- Using `setLocation`, move the first window flush against the left side of the screen, centered.
- Using `setLocation`, make the second window flush against the top of the screen, centered.
- The screen size is 1024x768. Figure out how to move the third window flush against the right-hand side of the screen, centered, and the fourth window flush against the bottom of the screen, centered.

## 4.4 Spelling with JFrames

- Switch roles: s2 drives and s1 navigates.
- Reset the Interactions Pane.
- Write down s1's initials.
- Use JFrames to spell the initials on the screen by using a bunch of skinny and wide windows.

Show your JFrames and your Interactions Pane to your TA.