1.

What are the locations of the two robots if I do the following.
robot1.translate(1,5); robot 1 =
robot1.translate(2,3); robot 1 = (4,11)
robot2.translate(5,5); robot 2 = (7,9)

Is it possible to manipulate robot2 by calling a method on robot1? Why or why not?
Is it possible to manipulate all of the robots by calling Robot.translate(x,y);? Why or why not?

It is not possible to manipulate robot2 by calling a method on robot1 because a method for robot1 is only supposed to manipulate robot1.

It is not possible to manipulate all of the robots despite sharing the same class. I think it is just how JAVA works.

2. Say that you've written two different main methods in the same folder. In one main method, you create a robot named 'robbie'. What happens if you try to add lines of code pointing to 'robbie' in the other main method?

   No, because it is not possible to have 2 main methods in the same class.

3. You've written a main method that contains the following variables a,b, and c.
   a. public static void main(String[] args)
      {
      int a,b,c = 2;
      int z = equation(a,b);
      }
      // You later write a method that does the following.
      public int equation(int a, int b)

      {
      int z = a + b;
      return z;
      }
      You realize that you want to change the code to do the following:
      public int equation(int a, int b)

      {
      int z = (a + b) * c;

return z;
}

Will the above code compile? Why or why not?

The above code will not compile because the main method has already declared "z" to be an integer, and the public equation method declares "z" again.

Lab

1. Constructors

   Froilan Zarate: Driver

   Alan Xiao: Scribe

   a. Scribe: record what appears on the screen there. What methods showed up?

      Inherited from Object
      - Boolean equals(object)
      - Class<?>getClass()
      - Int hashCode
      - Void notify()
      - Void notifyAll()
      - String toString()
      - Void wait(long)
      - Void wait(long,wait)
      - Void wait()

      String sayGoodbye

      String sayHello

   b. Select one of the methods. Which method did you select and what did it do?

      String sayHello

      This caused a pop up to appear with String sayHello() and it tells you it returned with the String sayHello() appearing again. There is an inspect button, get button, and a close button as well.

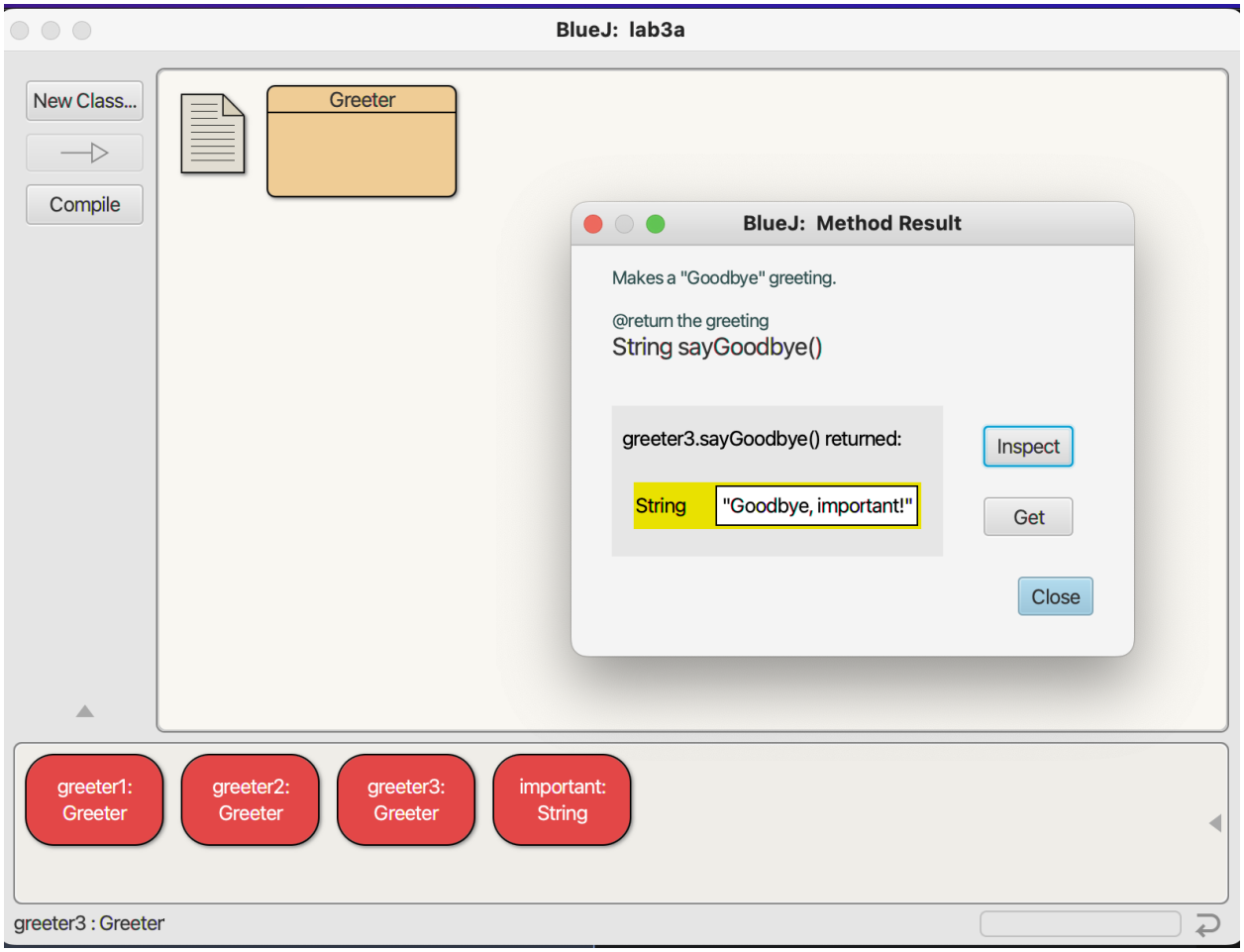
   c. Repeat all of the above steps but instead of the 'new Greeter()' you will choose 'new Greeter(string aName)'

      Inherited from Object
      - Boolean equals(object)
      - Class<?>getClass()
      - Int hashCode
      - Void notify()
      - Void notifyAll()
      - String toString()
      - Void wait(long)
      - Void wait(long,wait)
      - Void wait()

      String sayGoodbye

      String sayHello

d. Scribe: How did the result of step e look different from when you created the second Greeter? Why was this the case?
    i. Instead of “Hello, World” the new string said “Hello, [insert our string]” The first one was the default answer because the string was given to the computer for the first one, while the second one called the string that it forced us to put in.

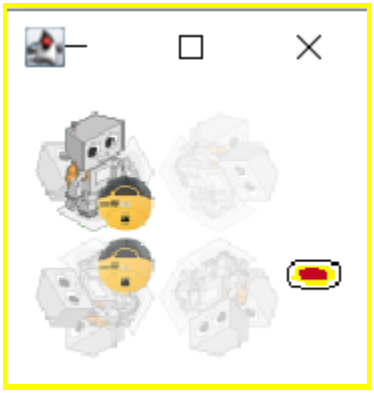


2. Lab 03B: Mor robots
   Driver: Alan Xiao
   Scribe: Froilan Zarate
    a. What method will you use when you right click on the red box representing the robot?
        i. void moveForward()
    b. What methods would you use to turn the robot around?
        i. void turnRight() or void turnLeft()
    c. What does it need to be created?
        i. int x, int y, String imageFile
    d. Where would you put ‘1, 2, and roomba.png’ as options?
        i. int x can be 1 or 2.
        ii. int y can be 1 or 2.

iii. roomba.png must be String imageFile

e. Can you make a robot out of any image?

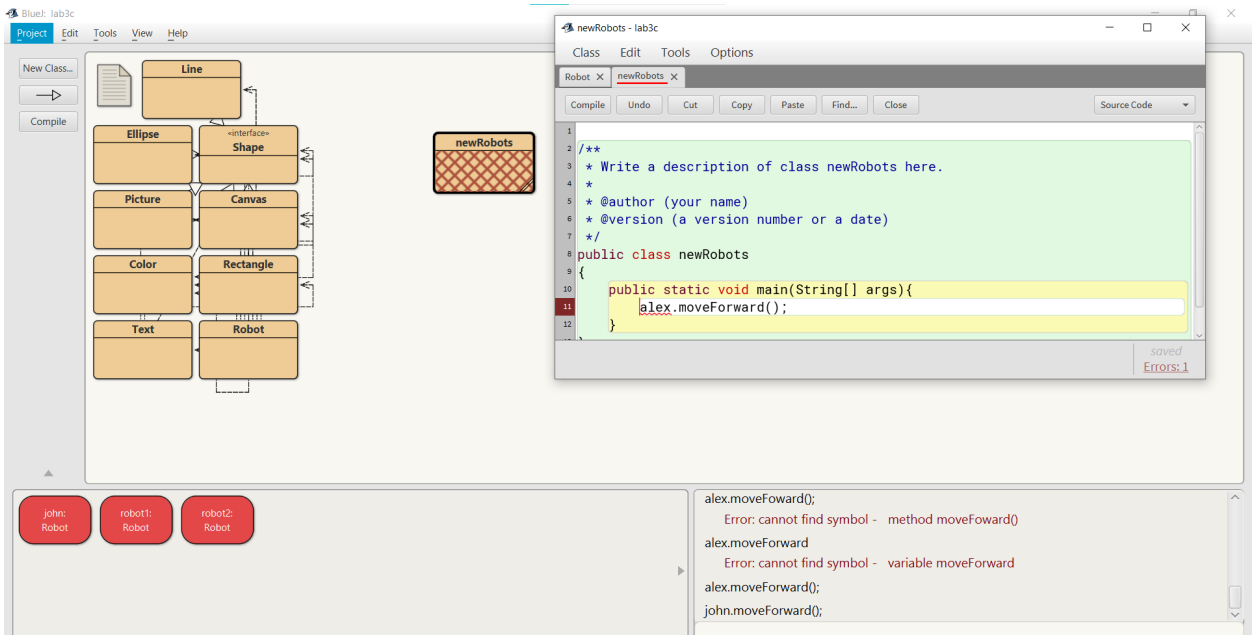
i. Yes, as long as it's in the "lab03b" folder.



3. Use Code Pad
   Driver: Alan Xiao
   Scribe: Froilan Zarate

   a. What happens when you compile it?

      i. It did not compile because "alex" was not declared as an object. It was only declared in the code pad.



Write into your lab document the kinds of scopes that you've encountered so far.

- The kind of scopes that we encountered so far are the classes in the main methods, Code Pad, and Workbench objects (red blocks).