Let's look at some solutions to the interclass problem.

Submitting the project.
A lot of the extra power we get in Java comes from greater abstraction. This leads to even more need for design.

We need to figure out how we are going to solve things before we actually do it.

In Java we can still go through and begin with a written description of what we will do. We pull that apart to decide what the “pieces” of the solution are and what actions those pieces take.

For the pieces we write classes from which we will create objects. The actions are the methods in those classes.
Each file of a Java program contains a single public top-level class.

Everything except the import statements will go inside of the class.

Structure of a class:
  - public class ClassName {
    - Methods and properties/members
  }

These can be preceded with visibility modifiers. For now we only care about public and private.

The keyword static implies that a method or property is associated with the class, not individual objects.
In Alice you made a method by clicking on a button and giving it a name. You could then add parameters. With functions you also specified the return type.

In Java we write methods in the class and they all have a return type, a name, and parameters in parentheses. The parameter list can be empty.

- returnType methodName(paramType1 param1, paramType2 param2, ...)
A variable declaration has a type followed by the name.

These are just like the variables you used in Alice and they can appear as local variables, properties, or parameters to functions.

The types are a bit different though and we'll talk about those tomorrow. Valid types in Java include:

- `int` – A number that has to be an integer.
- `double` – A number that doesn't have to be an integer.
- `String` – Just like in Alice, this is any set of characters you could type in.
In Alice you made an object from a class using the “Add Object to World” view. You could drag from a class to the world and it would instantiate the class to create an object.

In Java we use the keyword new to create a new object. This is what was done with the Scanner in the example code.

As we will fully discuss next class, you don't have to call new on the “primitive” types like int and double.
What is the most pressing thing you'd like to be able to do in Java?

Interclass Problem – Do one of the following from the book: 7.9, 7.10, 7.11, or 7.12.