Lecture 2: Objects and classes in Java

http://courses.cs.cornell.edu/cs2110

CMS VideoNote.com, PPT slides, DrJava

CMS. Visit course webpage, click “Links”, then “CMS for 2110”.
Videos of our lectures: Look at https://piazza.com/cornell/fall2017/cs2110/home
Download ppt slides the evening before each lecture, have them available in class. Please don’t ask questions on the piazza about that material the day before the lecture!
Download DrJava (the jar file, not the app). It may require downloading an old version of Java.
Got a Java question? See first if it’s answered on JavaHyperText

Next week’s recitation/discussion: Exception handling

Visit course webpage, click on “Lecture notes”
For the row for recitation 2, click on “tutorial”.
Your job: BEFORE your recitation, look at all videos in the tutorial.
There are about 25-30 minutes of tutorial! This is the longest tutorial you will have to do.
Then, in recitation/discussion, you will have a problem set to do. Can do it with 1-2 other people (up to groups of 3). TA will walk around, helping, answering questions, giving pointers, etc.
The problem set is due on the CMS no later than one week after the recitation (always on Wednesday night). But we encourage you to finish during the recitation and turn it in immediately.
Anything to be submitted is always on the course assignments page!

Assignment A2

Get assignment A2 from course website later today.
Objective:
- Get practice with Java functions.
- Learn about and use JUnit testing

Given to you before A1, but due after A1. Provide flexibility, allow you to get ahead and learn Java early.
Groups. You can do A2 with 1 other person. FORM YOUR GROUP EARLY! Use pinned Piazza Note @5 to search for partner!

Java OO (Object Orientation)

Python and Matlab have objects and classes. Strong-typing nature of Java changes how OO is done and how useful it is. Put aside your previous experience with OO (if any).
This lecture:
First: describe objects, demoing their creation and use.
Second: Show you a class definition and how it contains definitions of functions and procedures that appear in each object of the class.
Third: Talk about keyword null.
Fourth (if there is time). Show you a Java application, a class with a “static” procedure with a certain parameter.

Homework

1. Study material of this lecture.
2. Visit course website, click on Resources and then on Code Style Guidelines. Study
   - 3. Documentation
   - 3.1 Kinds of comments
   - 3.2 Don’t over-comment
   - 3.4 Method specifications
     - 3.4.1 Precondition and postcondition
3. Spend a few minutes perusing slides for lecture 3; bring them to lecture 3.
References to JavaHyperText entries

Objects: B.1 object
Calling methods: method call
Class definition: class
public, private: public private method
Parameter vs argument: parameter, argument
Inside-out rule
Methods may have parameters
Method calls may have arguments

Text mentions fields of an object. We cover these in next lecture

Methods may have parameters
Method calls may have arguments

Function: returns a value; call on it is an expression
Procedure: does not return a value; call on it is a statement

Evaluation of new-expression creates an object

new javax.swing.JFrame()
creates an object and gives as its value the name of the object

If evaluation creates this object, value of expression is

JFrame@25c7f37d

A class variable contains the name of an object

If variable h contains the name of an object, you can call methods of the object using dot-notation:

h.show();
h.setTitle("this is a title");
h.getX() + h.getWidth()

A class variable contains the name of an object

If variable h contains the name of an object, you can call methods of the object using dot-notation:

h.show();
h.setTitle("this is a title");
h.getX() + h.getWidth()

Class definition: a blueprint for objects of the class

Class definition: Describes format of an object (instance) of the class.

/** description of what the class is for */
public class C {
    declarations of methods (in any order)
}

This is a comment
Access modifier
public means C can be used anywhere

Class definition C goes in its own file named C.java
On your hard drive, have separate directory for each Java project you write; put all class definitions for program in that directory. You’ll see this when we demo.
**First class definition**

```java
public class C {
}
```

Then, execution of

```java
C k; k = new C();
```

creates object shown to right

**Class extends (is a subclass of) JFrame**

```java
public class C extends javax.swing.JFrame {
...
}
```

C: subclass of JFrame

JFrame: superclass of C

C inherits all methods that are in a JFrame

```java
getWidth() getHeight() setTitle(String) getTitle() setLocation(int, int) ... area() { return getWidth() * getHeight(); }
```

Object has 2 partitions: one for JFrame methods, one for C methods

**Class definition with a function definition**

```java
public class C extends javax.swing.JFrame {
    public int area() {
        return getWidth() * getHeight();
    }
}
```

**Class definition with a procedure definition**

```java
public class C extends javax.swing.JFrame {
    public void setWtoH() {
        setSize(getHeight(), getHeight());
    }
}
```

**Inside-out rule for finding declaration**

To what declaration does a name refer? Use inside-out rule:

1. Look first in method body, starting from name and moving out; then look at parameters; then look outside method in the object.
2. The whole method is in the object.

**Inside-out rule for finding declaration**

Function : in each object.

 getWidth() calls function which it appears.

Call on procedure

setSize instead of return type
**Using an object of class Date**

```java
/** An instance is a JFrame with more methods */
public class C extends javax.swing.JFrame {
    ...  
    /** Put the date and time in the title */
    public void setTitleToDate() {
        setTitle(new java.util.Date()).toString());
    }
}
```

An object of class java.util.Date contains the date and time at which it was created. It has a function toString(), which yields the data as a String.

**About null**

```java
v1 null
v2 null
```

null denotes the absence of a name. v2.getName() is a mistake! Program stops with a NullPointerException.

You can write assignments like: v1 = null;

and expressions like: v1 == null

**Hello World!**

```java
/** A simple program that prints Hello, world! */
public class myClass {
    /** Called to start program. */
    public static void main(String[] args) {
        System.out.println("Hello, world!");
    }
}
```

We explain static next week. Briefly: there is only one copy of procedure main, and it is not in any object.