Lecture 2: Objects and classes in Java
http://courses.cs.cornell.edu/cs2110

Next week’s recitation!

BEFORE recitation, visit this webpage (part of course webpage)
www.cs.cornell.edu/courses/CS2110/2016fa/online/apiString/01APIString.html

You can get to it from the lecture-notes page of our course webpage, recitation 2.

Read it, watch the short videos — less than 15 minutes!

Come to class with your Cornell id card and your laptop if you have one. You will work in groups of 2-3 (your choosing) on questions dealing with the API specs and classes Character and String, with the TA helping, guiding. At the end, the TA will make a note (using your Cornell id card) that you did it.

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.

Java OO

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.

References to course text and JavaSummary.pptx

Objects: B.1 slide 10-16
Calling methods: B.2-B.3 slide 18
Class definition: B.5 slide 11
public, private: B.5 slide 11, 12
Indirect reference, aliasing: B.6 slide 17
Method declarations: B.7
Parameter vs argument: B.12-B.14 slide 14
Methods may have parameters
Method calls may have arguments

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

Text uses value-producing method for function and void method for procedure. Get used to terminology: function and procedure
Object is associated with a window on your computer monitor

Name of object, giving class name and its memory location (hexadecimal).
Java creates name when it creates object

Object contains methods (functions and procedures), which can be called to operate on the object

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

If evaluation creates this object, value of expression is

A class variable contains the name of an object

Type JFrame: Names of objects of class JFrame

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

First class definition

/** An instance (object of the class) has (almost) no methods */
public class C {
    declarations of methods (in any order)
}

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

This is a comment
Access modifier

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.

Hence, evaluation of new-expression creates an object

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

Procedure calls: h.show(); h.setTitle("this is a title");
Function calls: h.getX() + h.getWidth()
**Class extends (is a subclass of) JFrame**

```java
/** An instance is a subclass of JFrame */
public class C extends javax.swing.JFrame {
}
```

C: subclass of JFrame
JFrame: superclass of C
C inherits all methods that are in a JFrame

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

Easy re-use of program part!

**Inside-out rule for finding declaration**

```java
/** An instance … */
public class C extends javax.swing(JFrame {
    /** Return area of window */
    public int area() {
        return getWidth() * getHeight();
    }
}
```

To what declaration does a name refer? Use inside-out rule:
- Look first in method body, starting from name and moving out.
- Then look at parameters.
- Then look outside method in the object.

The whole method is in the object

**Class definition with a function definition**

```java
/** An instance is a subclass of JFrame with a function area */
public class C extends javax.swing.JFrame {
    /** Return area of window */
    public int area() {
        return getWidth() * getHeight();
    }
}
```

Function area: in each object.
GetWidth calls function getWidth in the object in which it appears.

**Class definition with a procedure definition**

```java
/** An instance is a JFrame with more methods */
public class C extends javax.swing.JFrame {
    /** Set width of window to its height */
    public void setWtoH() {
        setSize(getHeight(), getHeight());
    }
}
```

Call on procedure setSize
It is a procedure because it has void 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

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!

/** 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!");
    }
}

args is an array of String elements

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