



**CS1110. Lecture 2, 2 Sep 2010. Objects & classes**

**Reading for this lecture:** Sec. 1.3. **Study this section, practice** what is taught using DrJava over the weekend.

**PLive:** Activities 3-3.1, 3-3.2, 3-3. (not 3-3.3), 3-4.1, 3-4.2.

**Reading for Tuesday, 7 Sep.** Sections 1.4, (p. 41); 13.3.1 (p. 376).

**Quote for the day:** Computational thinking: a fundamental skill for everyone ... [It] is ... choosing an appropriate representation for a problem or modeling the relevant aspects of a problem to make it tractable.  
**Jeannette Wing**

Not receiving emails from us from the CMS?

- You are not registered in the CMS. Email Maria Witlox [mwitlox@cs.cornell.edu](mailto:mwitlox@cs.cornell.edu) and ask her to register you. Needs your netid.
- Your email is bouncing. Your Cornell system is not set up correctly or the place to which you forward us is having trouble. Best thing to do: email yourself, at [netid@cornell.edu](mailto:netid@cornell.edu), see what happens, and fix it.

**AEWs** 1-credit AEW sections for CS1110.  
 Two hrs per week. Not remedial. See course website for link.  
 Wed 7:30-9:25pm;  
 Mon 2:30-4:25.

**Quiz on Tuesday.** Everyone should get 100.

- What is a type?
- How do you execute (carry out, perform) the assignment statement?
- Be able to execute an assignment statement.

2

**Two aspects of a programming language**

- Organization – structure
- Procedural – commands to do something

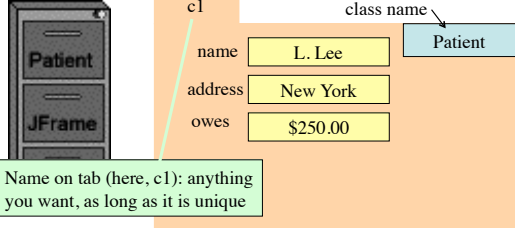
Example: Recipe book

- Organization: Several options; here is one:
  - Appetizers
  - list of recipes
  - Beverages
  - list of recipes
  - Soups
  - list of recipes
  - ...
- Procedural: Recipe: sequence of instructions to carry out

Parts to this course

- structural**  
objects  
classes  
methods  
inheritance
- procedural**  
assignment,  
return,  
if-statement  
iteration (loops)  
recursion
- miscellaneous**  
GUIs  
exception handling  
Testing/debugging

**A class is a file-drawer.** Contents: manila folders, each containing the same kind of information



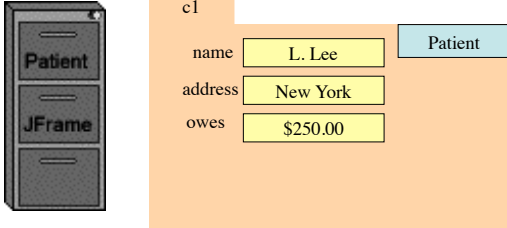
Name on tab (here, c1): anything you want, as long as it is unique

**manila folder:** an **object** or **instance** of the class

name, address, owes: **variables**, called **fields** of the folder

4

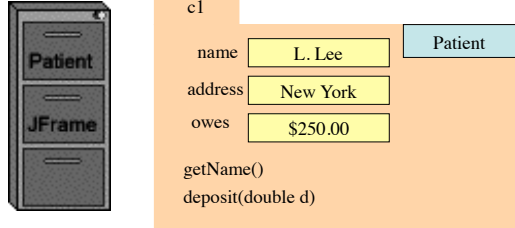
**A class is a file-drawer.** Contents: manila folders, each containing the same kind of information



**Instructions to be carried out by different people:**  
 change the name, get the name, bill the patient, receive money from patient, insert teeth xrays into the folder, ...

3

**A class is a file-drawer.** Contents: manila folders, each containing the same kind of information



**Instructions to be carried out by different people: methods.**  
 Assume getName is a **function**: it returns a value.  
 Assume deposit is a **procedure**: it does a task, doesn't return value

pat `c1`  
variable contains the name of the folder

`c1`  
name `L. Lee` Patient  
address `New York`  
owes `$250.00`  
getName()  
deposit(double d)

`pat.getName()` function call. Its value is "L. Lee"

`pat.deposit(250.0);` procedure call. Subtract 250.0 from field `owes`.

7

pat `c1`  
variable contains the name of the folder

`c1`  
name `L. Lee` Patient  
address `New York`  
owes `$250.00`  
getName()  
deposit(double d)

`new Patient()` An expression: create a new folder (put it in file-drawer `Patient`) and give as the value of the expression the name of the folder.

`pat= new Patient();` A statement: evaluate `new Patient()` and store its value (the name of the new folder) in variable `pat`.

8

j `a0`  
variable contains name of folder

An object (manila folder) of class `Javax.swing.JFrame` is associated with a window on your computer monitor. It has (among others) these functions:  
`getHeight()` `getWidth()` `getX()` `getY()`  
`getTitle()` `isResizable()`

We will demo the use of most of these methods

and these procedures:  
`show()` `hide()`  
`setTitle()` `setSize(int, int)`  
`setLocation(int, int)` `setResizable(boolean)`

In groups of 2, draw an object (manila folder) of this class, and put the name `a0` on its tab.

9

j `a0`  
variable contains the name of the folder

`j= new javax.swing.JFrame();`  
`j.show();`  
...

Expression `new JFrame()`  
Create new folder and put in file drawer `JFrame`.

Statement `jf= new JFrame();`  
Create new folder, as above, and place its name in variable `jf`.

Thereafter, use  
`jf. method-name ( arguments, if any )`  
to call methods of folder (object) `jf`.

- Read section 1.3.
- Practice what we did in class in DrJava.
- Try the self-review exercises on page 40.

10

**package:** A collection of classes that are placed in the same directory on your hard drive. Think of it as a room that contains file cabinets with one drawer for each class.

package `java.io` classes having to do with input/output  
package `java.net` classes having to do with the internet  
package `java.awt` classes having to do with making GUIs  
package `javax.swing` newer classes having to do with GUIs

To reference class `JFrame` in package `javax.swing`, use:  
`javax.swing.JFrame`

Instead: `import javax.swing.*;`

Then use simply `JFrame`

11

### Comments from last semester

I understand classes and objects fairly well, and I thought the file drawer/file folder analogy was very helpful.

I think I'm definitely prepared for 2110. The folder/file drawer analogy was actually very helpful for a first-time Java programmer in understanding them.

I did learn the concept before coming to this class, CS1110 is really what made me understand how objects and classes work.

The folder was a great way to learn objects and classes. It simplified a very complex concept.

Teaching methods were terrible. ... boxes and folders made the subject more confusing than it should be.

I'm still a bit dubious about the whole file folders and cabinets thing.

12