# CIS 110: Introduction to Computer Programming

Lecture 8
Hey (Objects), Listen!
(§ 3.2-3.3)

10/2/2011 CIS 110 (11fa) - University of Pennsylvania 1

---

## Outline

- Review: what is a library?
- The Math class
- The String class
- The Story Thus Far…

10/2/2011 CIS 110 (11fa) - University of Pennsylvania 2

---

## Announcements

- Homework 3 due tonight (11:59:59).
- Exam 1 on Wednesday.
  - See website for locations (based on time + name).
  - No homework or lab this week.
- Changes to exam protocol:
  - **ID required to turn in exam.**
  - Abbreviations for System.out.print(ln):
    - **S.O.PLN** and **S.O.P**
    - **All other code must be written out in full!**

10/2/2011 CIS 110 (11fa) - University of Pennsylvania 3

---

# What is a Library?

10/2/2011 CIS 110 (11fa) - University of Pennsylvania 4

---

## Review: Libraries

- *Libraries* are collections of classes that other people have written for us to use.
  - Motivation: avoid reinventing the wheel!
  - A particular *strength* of Java.
- Example: the DrawingPanel class



10/2/2011 CIS 110 (11fa) - University of Pennsylvania 5

---

## Learning a Programming Language

- Need to learn *two parts*:
  - *Language*: syntax, structure, etc.
  - *Libraries:* commonly-used functionality.
- Java's built-in library: the *Java Class Libraries*.
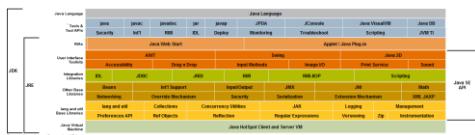


10/2/2011 CIS 110 (11fa) - University of Pennsylvania 6

---

## Learning a Programming Language

- In reality, few people know the entire library.
  - People know where to *look stuff up when needed*.
  - In Java, this is the *javadoc API documentation*.
- For now, we focus on two particular classes:
  - The Math class.
  - The String class.
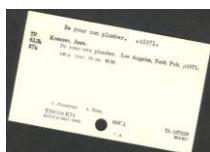
## The Math Class

Demo (MathExamples.java)

## Math Example

- Can we write a method that rounds a decimal to an integer?
  - MyMath.round(3.75) → 4
  - MyMath.round(3.25) → 3

```
public class MyMath {
  public static int round(double d) {
    return (int) (d + 0.5);
  }
}
```

## Calling Methods from Other Classes

- Can't simply call sqrt from another class!

```
public class MathTest {
  public static void main(String[] args) {
    System.out.println(sqrt(3.75)); // Bad!
  }
}
```

- sqrt is *out of scope* in MathTest.

## Dot Notation for Static Methods and Constants

- Solution: clarify where sqrt comes from with *dot notation*.
  - *Syntax:* <class name>.<method name>().

```
public class MathTest {
  public static void main(String[] args) {
    System.out.println(MyMath.sqrt(3.75));
  }
}
```

- Only works for static methods and constants.
  - *Not for local variables!*

## The Math class

- Java already provides round in the Math class.
  - Best to not reinvent the wheel!

```
public class MathTest {
  public static void main(String[] args) {
    System.out.println(Math.sqrt(3.75));
  }
}
```

## Math Methods and Constants Sampler (See p. 150 for details)

```
Math.E                Math.sin(num)
Math.PI               Math.sqrt(num)
Math.abs(num)         Math.toDegrees(num)
Math.ceil(num)        Math.toRadians(num)
Math.exp(num)
Math.floor(num)
Math.log(num)
Math.log10(num)
Math.max(num1, num2)
Math.min(num1, num2)
Math.pow(num1, num2)
Math.random()
Math.round(num)
```

10/2/2011     CIS 110 (11fa) - University of Pennsylvania     13

## The String Class

Demo (StringExamples.java)

10/2/2011     CIS 110 (11fa) - University of Pennsylvania     14

## Recall: Objects Are Different From Primitives

- DrawingPanel is an object!
  - Different syntax for creation and method calls:

```
DrawingPanel p = new DrawingPanel(500, 500);
Graphics g = p.getGraphics();
```

- Remember objects contain *data* and *methods*.
  - e.g., DrawingPanel provides the getGraphics method.

10/2/2011     CIS 110 (11fa) - University of Pennsylvania     15

## Startling Fact: Strings Are Objects!

- Strings try to act like primitives...

```
String msg = "hello world!"
```

- ...but they're really objects!

```
System.out.println(msg.length());
// Prints out 12 --- the length of msg!
```

- What other methods does String provide?

10/2/2011     CIS 110 (11fa) - University of Pennsylvania     16

## Strings as sequence of characters

Each element of the string is a single char.

| msg | 'H' | 'e' | 'l' | 'l' | 'o' | ' ' | 'W' | 'o' | 'r' | 'l' | 'd' | '!' |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
|     | 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   | 10  | 11  |

We refer to individual characters of a string *by index*. The first index is 0.
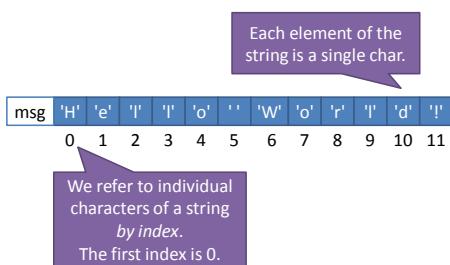
10/2/2011     CIS 110 (11fa) - University of Pennsylvania     17

## Example: charAt(index)

```
System.out.println(msg.charAt(4));
// Prints out 'o' --- the char at index 4
```

| msg | 'H' | 'e' | 'l' | 'l' | 'o' | ' ' | 'W' | 'o' | 'r' | 'l' | 'd' | '!' |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
|     | 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   | 10  | 11  |

msg.charAt(4) → 'o'
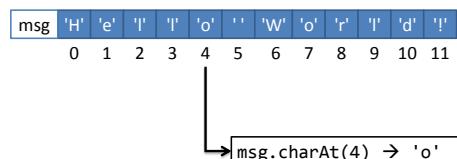
10/2/2011     CIS 110 (11fa) - University of Pennsylvania     18

## Example: substring(start, end)

```
System.out.println(msg.substring(3, 8));
// Prints out "lo Wo": the substring starting
// at index 3 and ending at index (8-1) = 7.
```

| msg | 'H' | 'e' | 'l' | 'l' | 'o' | ' ' | 'W' | 'o' | 'r' | 'l' | 'd' | '!' |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |

```
msg.substring(3, 8) →
"lo Wo"
```
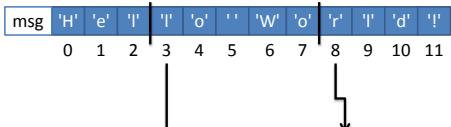
10/2/2011    CIS 110 (11fa) - University of Pennsylvania    19

## Runtime Errors From Bad Arguments

```
System.out.println(msg.charAt(12));
// Raises an exception since 12 is not valid!
```

| msg | 'H' | 'e' | 'l' | 'l' | 'o' | ' ' | 'W' | 'o' | 'r' | 'l' | 'd' | '!' |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |

10/2/2011    CIS 110 (11fa) - University of Pennsylvania    20

## Strings Are Immutable

```
String msg = "Hello World!";
msg.toUpperCase();
System.out.println(msg);      // Prints Hello World!
```

- String are *immutable*!
  - Methods calls on a String do not change that String.
  - Instead, methods *return new Strings* that are the result of the operation.

```
String msg = "Hello World!";
msg = msg.toUpperCase();
System.out.println(msg);      // Prints HELLO WORLD!
```

10/2/2011    CIS 110 (11fa) - University of Pennsylvania    21

## String Methods Sampler
## (See p. 162 for details)

```
charAt(index)
endsWith(text)
indexOf(text)
length()
startsWith(text)
substring(start, stop)
toLowerCase()
toUpperCase()
```

10/2/2011    CIS 110 (11fa) - University of Pennsylvania    22

## And The Story Thus Far…

10/2/2011    CIS 110 (11fa) - University of Pennsylvania    23

## Why Are We Here?

- Learning about *algorithmic thinking* via *computer programming!*
  1. Precision — Mental Model of Computation
  2. Decomposition
  3. Abstraction — Static methods For loops

Class Constants
Parameters
Return values

10/2/2011    CIS 110 (11fa) - University of Pennsylvania    24

## Syntax: Declarations

```
// Class declarations
public class <name> {
   <methods>
}

// Method declaration
public static <type> <name>(<params>) {
   <statements>
}
```

## Syntax: Statements

```
// Static method call
<name>(<params>);

// Static method call (another class)
<class>.<name>(<params>);

// Method call on an object      // For loop
<object>.<name>(<params>);       for (<init>; <test>; <update>) {
                                    <statements>
// Variable declaration          }
<type> <name> = <expr>

// Variable assignment
<name> = <expr>
<name> += <expr>
<name>++
```

## Syntax: Expressions

```
// All method calls if they return values (from previous slide)

// Variable use
<name>

// Literals
0  0.0  'c'  "hello!"

// Operators (+, -, *, /, %)
<expr> + <expr>
```

## Decomposition: Break A Problem Into Sub-problems
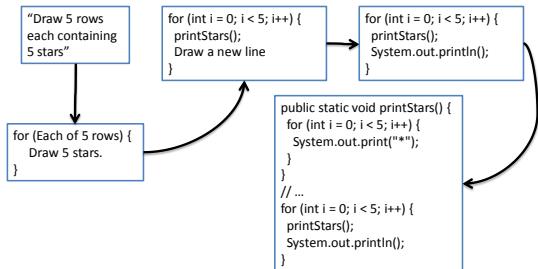
## Generalization: Make a Piece of Code Handle More Cases