Using Classes and Objects

CSC 1051 – Data Structures and Algorithms I

Dr. Mary-Angela Papalaskari
Department of Computing Sciences
Villanova University

Course website:
www.csc.villanova.edu/~map/1051/
Today

Creating Objects
The String Class
The Random and Math Classes
The Java API class library
Declaring Variables, revisited

• Examples of variable declarations:

```java
int count = 0;
double mpg;
String title;
Graphics page;
Color aquamarine;
Scanner scan;
```

• A class name can be used as a type to declare an object reference variable

• The object itself must be created separately
Creating Objects

- We have already seen something like this:

```java
Scanner scan = new Scanner (System.in);
```

This calls the Scanner *constructor*, which is a special method that sets up the object.

Variable refers to a Scanner object.
Creating Objects

• Generally, we use the `new` operator to create an object
• Creating an object is called *instantiation*
• An object is an *instance* of a particular class

```java
title = new String ("Java Software Solutions");
```

This calls the String *constructor*, which is a special method that sets up the object
The String Class is SPECIAL!

• Because strings are so common, we don't have to use the `new` operator to create a `String` object

```java
    title = "Java Software Solutions"
```

• This is special syntax that works only for strings

• Each string literal (enclosed in double quotes) represents a `String` object
Invoking Methods

• We've seen that once an object has been instantiated, we can use the dot operator to invoke its methods

    ans = scan.nextLine();

    numChars = title.length();

• Jargon alert: “A method invocation can be thought of as asking an object to perform a service”
Chapter 3: Using Classes and Objects

- We can create more interesting programs using predefined classes and related objects

- Chapter 3 focuses on:
  - object creation and object references
  - the String class and its methods
  - the Java API class library
  - the Random and Math classes
  - formatting output
  - enumerated types
  - wrapper classes
  - graphical components and containers
  - labels and images
Next

Creating Objects
The String Class
The Random and Math Classes
The Java API class library
String Indexes

- The characters in a string are indexed by their position.

"Hello"

Position 0

Position 4
Example: Using String methods

```java
String a = "stuff ";
String b = a.concat ("more stuff");
String c = a.toUpperCase();
String d = b.replace ( 'r', 'v');
String e = b.substring (2, 7);
char letter = c.charAt(3);
int num = a.length();
int comp = a.compareTo(b);
[a >> b? (use lexicographic order: positive if a>>b, zero if equal, negative if a<<b)]
```

List of some String methods: textbook page 119
public class StringMutation
{
    public static void main (String[] args)
    {
        String phrase = "Change is inevitable";
        String mutation1, mutation2, mutation3, mutation4;

        System.out.println ("Original string: ": phrase + ":\""");
        System.out.println ("Length of string: " + phrase.length());

        mutation1 = phrase.concat (", except from vending machines.");
        mutation2 = mutation1.toUpperCase();
        mutation3 = mutation2.replace ('E', 'X');
        mutation4 = mutation3.substring (3, 30);
    }
}
continued

// Print each mutated string
System.out.println ("Mutation #1: " + mutation1);
System.out.println ("Mutation #2: " + mutation2);
System.out.println ("Mutation #3: " + mutation3);
System.out.println ("Mutation #4: " + mutation4);

System.out.println ("Mutated length: " + mutation4.length());
Output

Original string: "Change is inevitable"
Length of string: 20
Mutation #1: Change is inevitable, except from vending machines.
Mutation #2: CHANGE IS INEVITABLE, EXCEPT FROM VENDING MACHINES.
Mutation #3: CHANGX IS INXVITABLX, XXCXPT FROM VXNDING MACHINXS.
Mutation #4: NGX IS INXVITABLX, XXCXPT F
Mutated length: 27
Quick Check

What output is produced by the following?

```java
String str = "Space, the final frontier.";
System.out.println (str.length());
System.out.println (str.substring(7));
System.out.println (str.toUpperCase());
System.out.println (str.length());
```

Output:

```
26
the final frontier.
SPACE, THE FINAL FRONTIER.
26
```
Today

Creating Objects
The String Class
The Random and Math Classes
The Java API class library
The Random Class

- The Random class is part of the java.util package, so we need to import it:

  ```java
  import java.util.Random;
  ```

- It provides methods that generate pseudorandom numbers
Example: Using Random methods

Random rand = new Random();

int a = rand.nextInt();
    // int range[-2147483648 ... 2147483647]
int b = rand.nextInt(4); // int range [0,1,2,3]
float c = rand.nextFloat(); // float range [0,1)
int d = rand.nextInt(4) + 1; // random [1,2,3,4]

List of some Random methods: page 126

See RandomNumbers.java
Quick Check

Given a `Random` object named `gen`, what range of values are produced by the following expressions?

- `gen.nextInt(25)`
  - Range: 0 to 24
- `gen.nextInt(6) + 1`
  - Range: 1 to 6
- `gen.nextInt(100) + 10`
  - Range: 10 to 109
- `gen.nextInt(50) + 100`
  - Range: 100 to 149
- `gen.nextInt(10) - 5`
  - Range: -5 to 4
- `gen.nextInt(22) + 12`
  - Range: 12 to 33
import java.util.Random;

public class RandomNumbers {
    public static void main (String[] args) {
        Random generator = new Random();
        int num1;
        float num2;

        num1 = generator.nextInt();
        System.out.println ("A random integer: "+ num1);

        num1 = generator.nextInt(10);
        System.out.println ("From 0 to 9: "+ num1);
        
        continued
num1 = generator.nextInt(10) + 1;
System.out.println("From 1 to 10: " + num1);

num1 = generator.nextInt(15) + 20;
System.out.println("From 20 to 34: " + num1);

num1 = generator.nextInt(20) - 10;
System.out.println("From -10 to 9: " + num1);

num2 = generator.nextFloat();
System.out.println("A random float (between 0-1): " + num2);

num2 = generator.nextFloat() * 6;  // 0.0 to 5.999999
num1 = (int)num2 + 1;
System.out.println("From 1 to 6: " + num1);

See RandomNumbers.java
A random integer: 672981683
From 0 to 9: 0
From 1 to 10: 3
From 20 to 34: 30
From -10 to 9: -4
A random float (between 0-1): 0.18538326
From 1 to 6: 3

num2 = generator.nextFloat();
System.out.println("A random float (between 0-1): " + num2);

num2 = generator.nextFloat() * 6; // 0.0 to 5.999999
num1 = (int)num2 + 1;
System.out.println("From 1 to 6: " + num1);

See RandomNumbers.java
Quick Check

Given a Random object named gen, write an expression that produces a random integer in the following ranges:

Range
0 to 12
1 to 20
15 to 20
-10 to 0
Example: counting “snake eyes”

Random gen = new Random();
int trial = 1, count = 0;

// Roll two dice 100,000 times and count how many
// times you roll snake eyes, i.e., two 1’s.

while (trial < 100000)
{
    die1 = gen.nextInt(6) + 1;
    die2 = gen.nextInt(6) + 1;
    if (die1 == 1 && die2 == 1)
        count++;
    // snake eyes
    trial++;
}
The Math Class

• The Math class contains methods that perform various mathematical functions

• These include:
  – absolute value
  – square root
  – exponentiation
  – trigonometric functions

```java
value = Math.cos(90) + Math.sqrt(delta);
```
The Math Class is special!

- The Math class is part of the java.lang package
  - No need to import anything!
- The methods of the Math class are static methods (also called class methods)
  - Static methods are invoked through the class name
  - Do not need to create a Math object

```
value = Math.cos(90) + Math.sqrt(delta);
```

See Quadratic.java
import java.util.Scanner;

public class Quadratic
{
    //-----------------------------------------------------------------
    // Determines the roots of a quadratic equation.
    //-----------------------------------------------------------------
    public static void main (String[] args)
    {
        int a, b, c;  // ax^2 + bx + c
        double discriminant, root1, root2;

        Scanner scan = new Scanner (System.in);

        System.out.print ("Enter the coefficient of x squared: ");
        a = scan.nextInt();

        // continued

See Quadratic.java
System.out.print("Enter the coefficient of x: ");
b = scan.nextInt();

System.out.print("Enter the constant: ");
c = scan.nextInt();

// Use the quadratic formula to compute the roots.
// Assumes a positive discriminant.

discriminant = Math.pow(b, 2) - (4 * a * c);
root1 = ((-1 * b) + Math.sqrt(discriminant)) / (2 * a);
root2 = ((-1 * b) - Math.sqrt(discriminant)) / (2 * a);

System.out.println("Root #1: " + root1);
System.out.println("Root #2: " + root2);
} }

See Quadratic.java
System.out.print("Enter the coefficient of x squared: ")
b = scan.nextInt()
System.out.print("Enter the coefficient of x: ")
c = scan.nextInt()
// Use the quadratic formula to compute the roots.
// Assumes a positive discriminant.

discriminant = Math.pow(b, 2) - (4 * a * c);
root1 = ((-1 * b) + Math.sqrt(discriminant)) / (2 * a);
root2 = ((-1 * b) - Math.sqrt(discriminant)) / (2 * a);
System.out.println("Root #1: "+ root1);
System.out.println("Root #2: "+ root2);
}
}

See Quadratic.java
Class Libraries

• A *class library* is a collection of classes that we can use when developing programs

• The *Java standard class library* is part of any Java development environment

• Its classes are not part of the Java language per se, but we rely on them heavily

• Various classes we've already used (*System*, *Scanner*, *String*) are part of the Java standard class library
The Java API

• The Java class library is sometimes referred to as the Java API

• API stands for Application Programming Interface

• Clusters of related classes are sometimes referred to as specific APIs:
  – The Swing API
  – The Database API
The Java API

• Get comfortable navigating the online Java API documentation
Packages

- For purposes of accessing them, classes in the Java API are organized into *packages*

<table>
<thead>
<tr>
<th>Package</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>java.lang</td>
<td>General support</td>
</tr>
<tr>
<td>java.applet</td>
<td>Creating applets for the web</td>
</tr>
<tr>
<td>java.awt</td>
<td>Graphics and graphical user interfaces</td>
</tr>
<tr>
<td>javax.swing</td>
<td>Additional graphics capabilities</td>
</tr>
<tr>
<td>java.net</td>
<td>Network communication</td>
</tr>
<tr>
<td>java.util</td>
<td>Utilities</td>
</tr>
<tr>
<td>javax.xml.parsers</td>
<td>XML document processing</td>
</tr>
</tbody>
</table>
The import Declaration

• **Without importing** a class, you could use its *fully qualified name*

  `java.util.Scanner`

• Or you can **import the class**, and then use just the class name

  `import java.util.Scanner;`

• Or import all classes in a particular package (use the * * wildcard character)

  `import java.util.*;`
The import Declaration

• All classes of the `java.lang` package are imported automatically into all programs

• It's as if all programs contain the following line:

  ```java
  import java.lang.*;
  ```

• That's why we didn't have to import the `System` or `String` classes explicitly in earlier programs

• The `Scanner` class, on the other hand, is part of the `java.util` package, and therefore must be imported
Homework

• Read Sections 3.1-3.5
  
  – **Always** do all self-review exercises when you review material

• Do Exercises EX 3.1-3.9

Many of the slides in this presentation are adapted from the slides accompanying “Java Software solutions, 7th edition” by Lewis and Loftus. Copyright © 2012 Pearson Education, Inc.