Objects, Input

Lecture 7, Wed Jan 20 2010

borrowing from slides by Kurt Eiselt

http://www.cs.ubc.ca/~tmm/courses/111-10
News

■ Midterm location announced: FSC 1005
  ■ for both Feb 28 and Mar 22 midterms
■ Assignment 1 out
  ■ due Wed 3 Feb at 5pm, by electronic handin
Recap: Classes, Methods, Objects

- **Class**: complex data type
  - includes both data and operations
  - programmers can define new classes
  - many predefined classes in libraries
- **Method**: operations defined within class
  - internal details hidden, you only know result
- **Object**: instance of class
  - entity you can manipulate in your program
Recap: Declare vs. Construct Object

```java
public static void main (String[] args) {
    String firstname;
    firstname = new String ("Kermit");
}
```

- Variable declaration does not create object
  - creates object reference
- Constructor and new operator creates object somewhere in memory
  - constructors can pass initial data to object
- Assignment binds object reference to created object
  - assigns address of object location to variable
Recap: Declare vs. Construct Object

firstname

bind variable to expression on right side of assignment operator

String object

"Kermit"

expression on right side of assignment operator
Recap: Objects vs. Primitives

- **References**
  - Frog
    - famousFrog
  - Frog
    - favoriteFrog

- **String object**
  - String
    - frogName
    - isMuppet: true
  - "Kermit"

- **Integers**
  - int
    - famousNum: 42
  - int
    - favoriteNum: 42

- **Comparing**
  - Objects vs. Direct Storage
Recap: Objects vs. Primitives

- references

Frog
- famousFrog

Frog
- favoriteFrog

String
- frogName:
  - boolean isMuppet: false

String
- frogName: "Kermit"

int
- famousNum: 42

int
- favoriteNum: 999

- vs. direct storage
Recap: API Documentation

- Online Java library documentation at [http://java.sun.com/javase/6/docs/api/](http://java.sun.com/javase/6/docs/api/)
  - textbook alone is only part of the story
  - let’s take a look!

- Everything we need to know: critical details
  - and often many things far beyond current need

- Classes in libraries are often referred to as Application Programming Interfaces
  - or just API
Recap: Some Available String Methods

```java
public String toUpperCase();
Returns a new string object identical to this object but with all the characters converted to upper case.
```

```java
public int length();
Returns the number of characters in this String object.
```

```java
public boolean equals( String otherString );
Returns true if this string object is the same as otherString and false otherwise.
```

```java
public char charAt( int index );
Returns the character at the given index. Note that the first character in the string is at index 0.
```
Recap: More String Methods

public String replace(char oldChar, char newChar);
Returns a new String object where all instances of oldChar have been changed into newChar.

public String substring(int beginIndex);
Returns new String object starting from beginIndex position

public String substring(int beginIndex, int endIndex);
Returns new String object starting from beginIndex position and ending at endIndex position

- up to but not including endIndex char:
  substring(4, 7)  "o K"

<table>
<thead>
<tr>
<th>Hello</th>
<th>K</th>
<th>e</th>
<th>r</th>
<th>m</th>
<th>i</th>
<th>t</th>
<th>F</th>
<th>r</th>
<th>o</th>
<th>g</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
</tbody>
</table>
Recap: Methods and Parameters

- Methods are how objects are manipulated
  - pass information to methods with parameters
    - inputs to method call
    - tell `charAt` method which character in the String object we're interested in
  - methods can have multiple parameters
    - API specifies how many, and what type
  - two types of parameters
    - explicit parameters given between parens
    - implicit parameter is object itself

```java
String firstname = "Alphonse";
char thirdchar = firstname.charAt(2);
```
Recap: Return Values

- Methods can have **return values**
- Example: `charAt` method result
  - return value, the character 'n', is stored in `thirdchar`

  ```java
  String firstname = "kangaroo";
  char thirdchar = firstname.charAt(2);
  ```

- Not all methods have return values
  - No return value indicated as `void`
Recap: Constructors and Parameters

- Many classes have more than one constructor, taking different parameters
  - use API docs to pick which one to use based on what initial data you have

```java
animal = new String();
animal = new String("kangaroo");
```
Classes, Continued

- A class has a name.
- A class should describe something intuitively meaningful. Why did someone create this class?
- A class describes the data stored inside objects in the class. (Nouns)
- A class describes the legal operations that can be done to the data. (Verbs)
- Example in Book: `java.awt.Rectangle`
# Primitive Types vs. Classes

<table>
<thead>
<tr>
<th>Primitive Types</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-defined in Java</td>
<td>Written by other programmers or by you</td>
</tr>
<tr>
<td>Simplest things, e.g., <code>int</code></td>
<td>Can be arbitrarily complex</td>
</tr>
<tr>
<td>Operators: +, -, ...</td>
<td>Methods</td>
</tr>
<tr>
<td>Values belong to types. E.g., 3 is an <code>int</code>, 3.14159 is a <code>double</code></td>
<td><strong>Objects</strong> belong to classes E.g., you are a UBC Student</td>
</tr>
</tbody>
</table>
Objects Belong to Classes

- Just as 1, 2, and 3 are all integers, you are all **objects** of the class UBCStudent!
  - You each have names, ID numbers, etc.
  - Each is unique person, but all are students
- Social organizations example:
  - Ballroom Dance Club
  - Ski Club
  - CSSS
  - Etc.
- Sometimes called “instances” of a class.
Class Libraries

- Before making new class yourself, check to see if someone else did it already
  - libraries written by other programmers
  - many built into Java
- Examples (built into Java)
  - BigInteger (java.math.BigInteger) lets you compute with arbitrarily big integers.
  - Date (java.util.Date) lets you get the current time.
  - Calendar (java.util.Calendar) does fancy date computations.
Example: BigInteger

```java
import java.math.BigInteger;
// Tell Java to use standard BigInteger package

class GetRichQuick {
    static void main(String[] args) {
        BigInteger salary = new BigInteger("111222334455666777888999");
        BigInteger stockOptions = new
            BigInteger("100000000000000000000000000000000");
        BigInteger profitPerShare = new BigInteger("314159");

        BigInteger optionCompensation =
            stockOptions.multiply(profitPerShare);
        BigInteger totalCompensation = salary.add(optionCompensation);

        System.out.println("Total Compensation = $" +
            totalCompensation.toString());
    }
}
```
BigInteger Constructors

```java
import java.math.BigInteger;
// Tell Java to use standard BigInteger package

class GetRichQuick {
    public static void main(String[] args) {
        BigInteger salary = new BigInteger("111222333444555666777888999");
        BigInteger stockOptions = new BigInteger("100000000000000000000000000000000");
        BigInteger profitPerShare = new BigInteger("314159");

        BigInteger optionCompensation =
            stockOptions.multiply(profitPerShare);
        BigInteger totalCompensation = salary.add(optionCompensation);

        System.out.println("Total Compensation = $" +
            totalCompensation.toString());
    }
}
```
Literals

With the primitive types, how do you create values with that type?
E.g., how do we create integer values?

1. You type some digits, like 3, or 42
2. You combine integer-valued things with operators that work on integers, e.g.,
   \[3 + 42 \times (a - b)\]
**Literals**

- With the primitive types, how do you create values with that type?
  
  E.g., how do we create integer values?
  
  1. You type some digits, like 3, or 42
  2. You combine integer-valued things with operators that work on integers, e.g.,
     
     \[3+42*(a-b)\]

- A bunch of digits are an integer **literal**.

- It’s the basic way to create an integer value
More Literals

- How about a value of type double?
  1. You type a bunch of digits with a decimal point, and optionally the letter e or E followed by an exponent
  2. You can combine doubles with operators that work on doubles.
More Literals

1. You type a bunch of digits with a decimal point, and optionally the letter e or E followed by an exponent.

2. You can combine doubles with operators that work on doubles.

Those are literals!
Long Literals

How about values of type long?

1. You type a bunch of digits followed by the letter l or L

2. You combine previously created longs
Literals – General Pattern

- To create values of a primitive type:
  1. There’s some way to type a literal
  2. There are operators that create values of the given type.
Literals for Classes?

- Classes are like primitive types, except they can be defined any way you like, and they can be much more complex.

- How to create a value (an object) of a given class?
  1. Invent some way to type a literal???
  2. Operators that create objects of that class (methods).
Constructors!

- A constructor is the equivalent of a literal for a class. It’s how you create a new object that belongs to that class.

- Examples:
  
  new BigInteger("999999")
  new Rectangle(10, 20, 30, 40)
  new UBCStudent("Joe Smith", 12345678, …)
Constructor Syntax

- The reserved word `new`
- Followed by the name of the class
- Followed by an open parenthesis (`(`
- Followed by parameters (information needed to construct the object)
- Followed by a closing parenthesis `)`
Using Constructors

- Use a constructor just as you’d use a literal.
- Example:
  - For the int type:
    ```java
    int a = 3;
    ```
  - For the BigInteger class:
    ```java
    BigInteger a = new BigInteger("3");
    ```
# Primitive Types vs. Classes

<table>
<thead>
<tr>
<th>Primitive Types</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-defined in Java</td>
<td>Written by other programmers or by you</td>
</tr>
<tr>
<td>Simplest things, e.g., <code>int</code></td>
<td>Can be arbitrarily complex</td>
</tr>
<tr>
<td>Operators: +, -, ...</td>
<td>Methods</td>
</tr>
<tr>
<td>Values belong to types. E.g., 3 is an <code>int</code>, 3.14159 is a <code>double</code></td>
<td>Objects belong to classes E.g., you are a UBC Student</td>
</tr>
<tr>
<td>Literals</td>
<td>Constructors</td>
</tr>
</tbody>
</table>
What about `String`?

- Is `String` a primitive type? Is it a class?
- `String` is a class, but it’s a special class!
  - Automatically imported
  - Built-in literals, e.g., “This is a String literal.”
  - + operator for concatenation
- But it also has many other methods, that you can call, just like for any ordinary class…
public class StringTest
{
    public static void main (String[] args)
    {
        String firstname;
        String lastname;
        firstname = new String ("Kermit");
        lastname = new String ("the Frog");
        System.out.println("I am not " + firstname + " " + lastname);
    }
}

String Example – Constructor Syntax
String Example - Literal Syntax

```java
public class StringTest {
    public static void main (String[] args) {
        String firstname;
        String lastname;
        firstname= "Kermit";
        lastname= "the Frog";
        System.out.println("I am not " + firstname
            + " " + lastname);
    }
}
```

String is the only class that supports both literals and constructors!
Escape Characters

■ How can you make a String that has quotes?
  ■ String foo = "oh so cool";
  ■ String bar = "oh so \"cool\", more so";

■ Escape character: backslash
  ■ general principle
Keyboard Input

- Want to type on keyboard and have Java program read in what we type
  - store it in variable to use later
- Want class to do this
  - build our own?
  - find existing standard Java class library?
  - find existing library distributed by somebody else?

- Scanner class does the trick
  - java.util.Scanner
  - nicer than System.in, the analog of System.out
import java.util.Scanner;

public class Echo {
    public static void main (String[] args)
    {
        String message;
        Scanner scan = new Scanner (System.in);
        System.out.println("Enter a line of text: ");
        message = scan.nextLine();
        System.out.println("You entered: \\
            + message + \\
        ");
    }
}
public class Echo
{
    public static void main (String[] args)
    {
        String message;
        Scanner scan = new Scanner (System.in);
        System.out.println ("Enter a line of text: ");
        message = scan.nextLine();
        System.out.println ("You entered: \\
                + message + \\
                ");
    }
}

- Import Scanner class from java.util package
Importing Packages

- Collections of related classes grouped into packages
  - tell Java which packages to keep track of with `import` statement
  - again, check API to find which package contains desired class
- No need to import `String`, `System.out` because core `java.lang` packages automatically imported
import java.util.Scanner;

public class Echo {
    public static void main (String[] args) {
        String message;
        Scanner scan = new Scanner (System.in);
        System.out.println ("Enter a line of text: ");
        message = scan.nextLine();
        System.out.println ("You entered: \\
                          + message + \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\
                          \\n

■ Declare string variable to store what user types in
Scanner Class Example

```java
import java.util.Scanner;

public class Echo
{
    public static void main (String[] args)
    {
        String message;
        Scanner scan = new Scanner (System.in);
        System.out.println ("Enter a line of text: ");
        message = scan.nextLine();
        System.out.println ("You entered: " + message + "\"\"\";
    }
}
```

- Use Scanner constructor method to create new Scanner object named scan
  - could be named anything, like `keyboardStuff` or `foo`
import java.util.Scanner;

public class Echo
{
    public static void main (String[] args)
    {
        String message;
        Scanner scan = new Scanner (System.in);
        System.out.println("Enter a line of text: ");
        message = scan.nextLine();
        System.out.println("You entered: "+ message + "\"\"");
    }
}

- Prompt user for input
import java.util.Scanner;

public class Echo
{
    public static void main (String[] args)
    {
        String message;
        Scanner scan = new Scanner (System.in);
        System.out.println ("Enter a line of text: ");
        message = scan.nextLine();
        System.out.println ("You entered: \\
                          \"" + message + \\
                          \""");
    }
}

- **nextLine** method reads all input until end of line
- returns it as one long string of characters
import java.util.Scanner;

public class Echo {
  public static void main (String[] args) {
    String message;
    Scanner scan = new Scanner(System.in);
    System.out.println("Enter a line of text: ");
    message = scan.nextLine();
    System.out.println("You entered: \\
        " + message + \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \\
        \

- Print out the message on the display
Scanner Class Example

- Let’s try running it