

OO Features of Java

Lecture 02B

4/1/2005

Harbin-OO

1

Java as an OO Language

- Java is considered an OO language
 - Reminder: "Object-Oriented" design generally means "Class-Oriented".
 - Same is true in programming: "OO Programming" is really very much class-oriented
- Java makes OO programming possible, but...
 - You can also write Java programs which violate OO principles

Just because a program is written in Java does NOT mean it is OO!

4/1/2005

Harbin-OO

2

OO Features of Java

- We look at the most basic OO features of Java
 - Much detail is omitted
- These allow us to implement classes, objects, messages, etc.
- This should be Java you already know!
 - We review it to point out the OO features and terminology
- There are other OO and non-OO features of Java that we will review later

4/1/2005

Harbin-OO

3

Classes in Java

- The class is the basic unit of a Java program

```
public class MyClass {  
  // methods, variables, etc.  
}
```

- A .java file typically contains one public class
 - There can be private classes and nested or "inner" classes, too

4/1/2005

Harbin-OO

4

Objects in Java

- Classes are created when the program is designed (written)
- Objects are created when the program runs
- Objects are *instances* of classes
- The *new* operator creates an object
new MyClass(...)
- The newly created object has an internal "name" or reference that unique identifies it

4/1/2005

Harbin-OO

5

References and Objects

- To use a new object later, save its reference

```
variable = new MyClass(...);
```

- The variable now contains a reference which unique identifies the object
- The variable must be declared of an appropriate type (more later), for example
MyClass variable;
- More than one variable can refer to the same object

4/1/2005

Harbin-OO

6

Count The Objects

// Program starts, no MyClass objects yet...

```
variable1 = new MyClass(...);
```

- //now there is one MyClass object

```
variable2 = variable1;
```

- //now how many objects?

```
variable3 = new MyClass(...)
```

- //now how many objects?

4/1/2005

Harbin-OO

7

Objects and References

- Classes, objects, references, and variables are all different things
- This is VERY IMPORTANT to understand.
- Drawing a picture will help

4/1/2005

Harbin-OO

8

Declaring a Variable

- Variables are declared in Java by giving a type followed by a variable name:

Student xiaoWang;

- Such a variable can later refer to a Student object, old or new

4/1/2005

Harbin-OO

9

Declaring vs Creating

- A variable is declared. This does not create or change any object.
- An object is created. This does not create or change any variable.

Student xiaoWang = new Student();

- Three separate operations take place.
 - A new variable is declared. It does not yet refer to any student
 - A new Student object is created.
 - Finally, the reference to the new object is assigned to the new variable

4/1/2005

Harbin-OO

10

Messages in Java

- Messages are implemented in Java by "methods"
- The parameters of the message are the parameters of the method

```
class Employee {  
    public void turnAround(int howMany) {  
    ...
```

The message (method) is "turnAround". The parameter name "howMany". The parameter value is up to the sender of the message.

4/1/2005

Harbin-OO

11

Commands in Java

- Reminder: commands are messages which do not return a value
- In Java, commands are methods with *void* return type

```
class Employee {  
    public void turnAround(int howMany) {  
    ...
```

"void" simply means "there is no return value"

4/1/2005

Harbin-OO

12

Queries in Java

- Reminder: queries are messages which return a value
- In Java, commands are methods with any non-*void* return type

```
class Employee {  
    public String getMyName() {  
    ...  
        return somethingOrOther;  
    }  
}
```

In such a method there will *always* be at least one *return* statement with a value

4/1/2005

Harbin-OO

13

Sending a Message

- Reminder: to send a message, you must know the name of the object.
- In Java, to send a message, you must have a reference to the object. Then you send the message using this syntax:

variable.*methodName(parameters)*

- We say that the method is "called" or "invoked" on the object

4/1/2005

Harbin-OO

14

What's Wrong Here?

- Employee emp;
- emp.turnAround(3);

4/1/2005

Harbin-OO

15

Saving Return Values

- "A query invocation produces a value."
- This is a fancy way of saying "if you call a method with a non-void return type, it will return a value".

Student john;

john.getMyName();

- This is legal, but... the returned value is lost
String age = john.getMyAge(); //save the value

4/1/2005

Harbin-OO

16

Object Attributes

- In Java, attributes (properties) of an object are "instance variables"
- Each object of the class has the same instance variables
- Each object of the class has its own *values* for the instance variables

```
class Student {  
    int age;  
    String name;  
    ...
```

4/1/2005

Harbin-OO

17

Using Instance Variables

- Instance variables are "persistent"
 - Keep their values even when between messages to the object
- An object can always see and modify its own instance variables
- Can one object see or modify the instance variables of another object?
 - Yes, you CAN program that way in Java
 - It is not considered good OO style!

4/1/2005

Harbin-OO

18

Constructors

- A constructor is a special type of method
- Invoked by the *new* operator when an object is created
- The constructor "initializes" the object
- Constructor is neither a command or a query
- A constructor can *never* be invoked except when the object is created
- Examples later

4/1/2005

Harbin-OO

19

Initialization

- A most important duty of a constructor is to initialize instance variables
- Variables can also be initialized when declared

```
class Employee {  
    String name; //constructor should initialize  
    String company = "IBM"; //already initialized  
    ...
```

4/1/2005

Harbin-OO

20

Summary

- Classes are the basic unit of Java program design
- Objects are created by *new*
- Classes, objects, references, and variables are all different things
- Messages correspond to methods
- Parameters and return values of messages correspond to those of methods
- Attributes correspond to instance variables
- Constructors create new objects