Computer Science 101 S1

Lecture 6

Contents

Objects
String objects

Coursebook: **§7**

Java - supports *object oriented* programming

The world is made up of real world objects e.g. students, dogs, cars, cats, books.

Objects are the things our programs deal with.

Each object has:

data - storing the current state of the object.

and

code - which operates on the object's data i.e what can be done with that object type.

Objects – Example

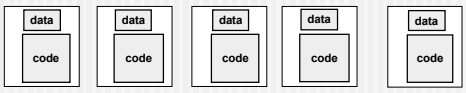
Consider a system managing university students.
A student object has:

data e.g.
id, name, age, contact number, address, completed courses, faculty, grades, stage, current courses, …

code which operates on the data e.g.
add a new course, add grades to a course, change contact number, change address, set faculty, …

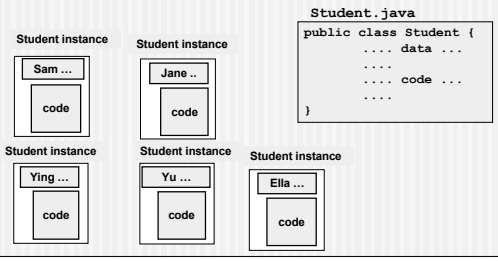
Visualising objects (instances)

Each object (also called instance) consists of two things:
- data (the current state of the object)
- code (which operates on the data in the object)



Class and instances of the class

Each kind of object (instance) is represented by a Java class, which is stored as a Java source file.



7

Primitive and Object types

All data in Java is either:

a **primitive** type **there are exactly 8:**

or an **object** type **infinitely many...**

8

Declaring object variables

Just as before, we must specify the *type* of the variable (which will be the name of a class) and give it a *name*:

```
<Classname> <identifier>;
```

For example:

```
String name;
Rectangle box;
Student sam;
```

type of the variable (class name)

variable name (identifier)

9

Initialising objects

Remember that to initialise a primitive variable, we simply assign some literal value to the variable: `int i = 10;`

To initialise an object variable, we must construct a **new** object using the keyword `new`. The general form of this is:

```
<identifier> = new <Classname>();
```

For example:

```
String name = new String("Hello World");
Rectangle box = new Rectangle(10, 20, 30, 40);
```

10

Initialising objects

When we construct an object:

- we must use the keyword `new`
- we sometimes have to pass information (called *arguments*) about the new object to initialise it
- the `new ClassName()` expression creates a new object and returns the memory address of this object
- the variable we assign the object to stores the memory address of the object

```
Rectangle box = new Rectangle(10, 20, 30, 40);
```

box 10001

10001 *(address in memory)*

10 20

30 40

11

Visualising objects

It is common to visualise the object variable as pointing to the actual object in memory

box 10001 → 10001: 10 20 / 30 40

12

Object vs. primitive variables

Primitive variables

```
int price = 1500;
```

price 1500

Object variables

```
String name = new String("Adriana");
```

name 1010011

1010011 "Adriana"

13

Assigning object variables

Primitive variables

```
int a = 10;
int b = a;
```

a 10

b 10

Object variables

```
String word = new String("dog");
String animal = word;
```

1010011 "dog"

word 1010011

animal 1010011

14

Strings

We previously saw String literals in our System.out.println() statements:

```
System.out.println("Hi World");
```

[we now know Strings are objects but we didn't use the `new` keyword to create the String object]

Strings are very special objects in Java.

They are the only type of object that can be used without first having to be explicitly created with `new`.

They are treated in this special way because they are such common objects

15

String objects and literals

When the compiler comes across a String literal for the first time, it automatically constructs a new String object:

"Hello" —(automatically constructed)→ **new String("Hello")**

So the code:

```
String s = "Interesting";
System.out.println("Hi World");
```

is exactly the same as:

```
String s = new String("Interesting");
System.out.println(new String("Hi World"));
```

16

Dot notation – instance methods

What can we do with our objects once we have constructed them?

We get things to happen in our programs by calling methods on the objects we have created. The general form of this is:

```
object.methodName();
```

or

```
object.methodName( parameters );
```

These are called *instance* methods.

17

Dot notation

Instance methods make a request to the specified object telling it to carry out the task given by the method name e.g.

```
String s = new String("A few words");

int letters = s.length();
```

1010011 "A few words"

S 1010011

letters 11

- the method call `s.length()` tells the String object referred to by the variable `s` to report the number of characters it has.
- the number that is reported (said to be *returned* by the method) is then assigned to the integer variable `letters`
- after the assignment statement, the variable `letters` will store the value 11

18

String instance methods

greeting 1011011

1011011

H	e	l	l	o		W	o	r	l	d
0	1	2	3	4	5	6	7	8	9	10

greeting.charAt(0) → 'H'

greeting.charAt(8) → 'r'

Give the output

```
char c1 = greeting.charAt(6);
char c2 = greeting.charAt(10);
String word = c1 + "oo" + c2;
System.out.println(word);
```

19

String instance methods

greeting → 1011011

1011011

H	e	l	l	o		W	o	r	l	d
0	1	2	3	4	5	6	7	8	9	10

```
greeting.indexOf('o')   → 4
greeting.indexOf("Z")   → -1
greeting.indexOf("or")  → 7
```

Give the output

```
int position = greeting.indexOf('r');
char c1 = greeting.charAt(position + 2);
char c2 = greeting.charAt(position - 1);
String word = c1 + c2 + "ll";
System.out.println(word);
```

20

String instance methods

greeting → 1011011

1011011

H	e	l	l	o		W	o	r	l	d
0	1	2	3	4	5	6	7	8	9	10

```
greeting.substring(0, 2)   → "He"
greeting.substring(3, 7)   → "lo W"
greeting.substring(8)      → "rld"
```

Give the output

```
String bit1 = greeting.substring(4, 7);
String bit2 = greeting.substring(0, 1);
String bit3 = greeting.substring(9);
String word = bit1 + bit2 + bit3;
System.out.println(word);
```

21

Example

We will develop the following simple program:

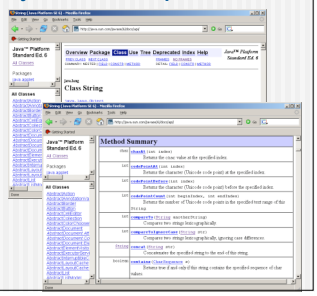
```
c:\NamesProg> java NamesApp
Enter your name: Adriana Ferraro
Hello A. Ferraro
```

22

Java API

http://java.sun.com/javase/6/docs/api/

You can find more information about what methods are available for a given type of object using the Java API (Application Programming Interface):



23

What you need to know

- Objects are made up of data, and code which operates on that data.
- There are eight primitive types, infinite object types.
- Declare instances. Initialise instances using the keyword, `new`.
- Visualising instances in memory.
- Assigning to instance variables.
- String instance methods: length(), charAt(), indexOf(), substring().
- The Java API.

24

Ex01 – Write code which creates the instances shown

```
String word1, word2, word3;
word1 = new String("Bit");
word2 = new String("Byte");
word3 = word2;
```

word1 → 1010011 → "Bit"

word2 → 1011111 → "Byte"

word3 → 1011111

```
public class Ex01 {
  public void start() {
    String word1 =
    String word2 =
    String word3 =
  }
}
```

25

Ex02 – User enters "SNIP", output?

```
public class Words {
   public void start() {
      String middle = "**";
      char letter1, letter2;
      System.out.print("Enter 4 letter word: ");
      String userWord = Keyboard.readInput();
      letter1 = userWord.charAt(0);
      letter2 = userWord.charAt(3);
      System.out.println(letter1 + middle +
                                          letter2);
   }
}
```

```
> java Ex02
Enter 4 letter word: SNIP
S**P
```

26

Ex03 – User enters "POST", output?

```
> java Ex03
Positions: 2, -1
```

```
public class FindLetters {
   public void start() {
     String search1 = "S";
     String search2 = "t";
     int position1, position2;
     System.out.print("Enter 4 letter word: ");
     String userWord = Keyboard.readInput();
     position1 = userWord.indexOf(search1);
     position2 = userWord.indexOf(search2);
     System.out.println("Positions: " +
                  position1 + ", " + position2);
   }
}
```

27

Ex04 – Give the output?

```
public class GetBits {
   public void start() {
     String words = "BIP BOP BAP";
     String part1, part2;
     int position = words.indexOf("O");
     part1 = words.substring(0, position);
     part2 = words.substring(8);
     System.out.println(part1 + part2);
   }
}
```

```
> java Ex04
BIP BBAP
```