

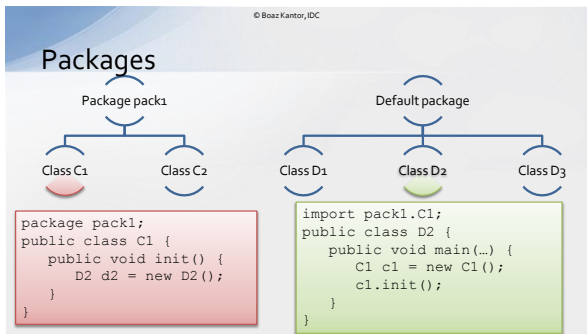
Boaz Kantor
Introduction to Computer Science,
Fall semester 2010-2011
IDC Herzliya



Data & Variables

"It's elementary, my dear Riker. Sir."
– Data, Star Trek ("Lonely Among US")

© Boaz Kantor, IDC



Data

- Primitive
 - Held in primitive data types
 - Actually stored in memory as numbers
 - Assignment statements can modify data
- Complex
 - Held within objects
 - Stored as large chunks of binary data
 - Methods provide operations



© Boaz Kantor, IDC

Primitive Data

	8 bit	16 bit	32 bit	64 bit
Integer number	byte	short	int	long
Decimal number			float	double
Text		char		

Boolean boolean

More info:

<http://download.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html>

© Boaz Kantor, IDC



Casting (Widening & Narrowing)

- It's ok to move data from one type to another
- If the target data type is smaller, data may be lost!
- Java is a Polish mother**
- She's concerned that you lose data
- Therefore, if such possibility exists, tell Java that you are aware of the risk (and that you'll call her later)

© Boaz Kantor, IDC



Examples (1)

```
public class Transportation {
    public static void main(String[] args) {
        short smart = 2; // 2 passengers in a smart
        int wrangler = 5; // 5 passengers in a jeep
        int c4 = 5; // 5 passengers in a Citroen
        long train = 1000; // 1,000 people on a train

        System.out.println("Passenger count:\nsmart: " + smart +
            "\nwrangler: " + wrangler +
            "\nc4: " + c4 +
            "\ntrain: " + train);
    }
}
```

© Boaz Kantor, IDC

```
Passenger Count:
smart: 2
wrangler: 4
c4: 5
train: 1000
```

Examples (2): avoid data loss

```
short smart = 2;
int wrangler = 5;
int c4 = 5;
long train = 1000;
```



Compiler:
"Type mismatch: cannot convert from long to int"

```
c4 = wrangler;           // move people from jeep to c4
wrangler = smart;        // move people from smart to jeep
c4 = train;           // move people from train to c4
c4 = (int)train;          // move people from train to car
```

© Boaz Kantor, IDC

Examples (3): change type

```
int students = 43;
int studentsPerTable = 2;
int tablesNeeded = students / studentsPerTable;
System.out.println("Tables needed: " + tablesNeeded);

float actualTablesNeeded = students / studentsPerTable;
System.out.println("Actual tables needed: " + actualTablesNeeded);

actualTablesNeeded = (float)students / studentsPerTable;
System.out.println("Really actual tables needed: " + actualTablesNeeded);

int remainderTable = (int)(actualTablesNeeded -
    (int)actualTablesNeeded + 0.5);
// ...or any other creative solution (using %, for instance)
System.out.println("Ok last time, I promise. Tables needed: " +
    (tablesNeeded + remainderTable));
```

© Boaz Kantor, IDC

```
Tables needed: 21
Actual tables needed: 21.0
Really actual tables needed: 21.5
Ok last time, I promise. Tables needed: 22
```

Classes & Objects

- A class is a complex data type which offers:
 - Internal variables ("state")
 - Functionality ("methods")
- Some classes provide only methods, some provide both



© Boaz Kantor, IDC

Definition Classes

- Provide both state and methods
- You can create many objects, each with its own state
- You can invoke an object method
- The method can change the object's state
- The method can return a value
- You can work with that value



© Boaz Kantor, IDC

Example

- How would you represent a 2 dimensional point?
int X, Y? float X, Y?
- Consider a definition class called Point which "encapsulates" both X and Y.
- It also provides you with some functionality to change its state or inquire about it.

© Boaz Kantor, IDC

Example

```
Point p1 = new Point(0, 0);
Point p2 = new Point(10, 10);
int x1, y1;

x1 = p1.getXCoordinate();
y1 = p1.getYCoordinate();
System.out.println("p1 is at (" + x1 + ", " + y1 + ")");
System.out.println("p2 is at (" + p2.getXCoordinate() + ", " +
    p2.getYCoordinate() + ")");

p1.moveOnXAxis(-10);
p2.moveOnYAxis(20);
x1 = p1.getXCoordinate();
y1 = p1.getYCoordinate();
System.out.println("p1 is at (" + x1 + ", " + y1 + ")");
System.out.println("p2 is at (" + p2.getXCoordinate() + ", " +
    p2.getYCoordinate() + ")");
```

© Boaz Kantor, IDC

```
p1 is at (0, 0)
p2 is at (10, 10)
p1 is at (-10, 0)
p2 is at (10, 30)
```

Container Classes

- Don't have state!
- Just a bunch of stand-alone methods, packed into a single class, only because it made sense.
- Example: Math, a bunch of mathematics-related methods, with no state
- Usage:

```
int g = 10;
float x = Math.sqrt(g);
```

© Boaz Kantor, IDC

[illegible]

Example, Class String

- Interesting methods:
 - `indexOf`, `substring`, `toUpperCase`, `toLowerCase`, `replace`, more (see API)

```
String input = "Seatec Astronomy";
String output;
int spaceIndex = input.indexOf(" ");
String firstWord = input.substring(0, spaceIndex);
```

Astronomy Seated
STEVEN SEAGAL

<code>Return a new character sequence that is a subsequence of this sequence.</code>
--

String	substring(int beginIndex)
--------	---------------------------

	Returns a new string that is a substring of this string
--	---

```
String substring(int beginIndex, int endIndex)
```

	Returns a new string that is a substring of this string
<code>charAt()</code>	Retrieves the character at the specified index

```
char[] toCharArray()
```

```
    "ven ").toUpperCase()  
    perCase();
```

```
perCase();
```

© Boaz Kantor, IDC

[illegible]

Example, Class Random, Scanner

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

public class RandomGame {
    public static void main(String[] args) {
        Random randomGenerator = new Random();
        Scanner inputScanner = new Scanner(System.in);
        System.out.println("Enter your minimum maximum: ");
        int minimum = inputScanner.nextInt();
        int maximum = inputScanner.nextInt();
        int random = randomGenerator.nextInt(maximum - minimum + 1) + minimum;
        System.out.println("Result: " + random);
    }
}
```

© Boaz Kantor, IDC

Do we have time to..

- Write a game?
- Solve exercise 2?

© Boaz Kantor, IDC
