Java syntax

Lecture 6

Comparing primitives

- To compare two primitives use ==

```
int a = 3;
byte b = 3;
if (a == b)  // true
```

Comparing object references

- Use ==

```
Foo a = new Foo();
Foo b = new Foo();
Foo c = a;
if (a == b)  // false
if (a == c)  // true
if (b == c)  // false
```

Comparing objects

- To compare objects use equals – you need to implement a special method, which compares the state of both objects, the fields that you think are important

String s1=new String("Fred");

String s2=new String("Fred");

if(s1==s2) //false

if(s1.equals(s2)) //true

Java operators

- Assignment
- Equivalence
- Auto increment
- Boolean operators
- Short circuiting – lazy &&, || operators (can check that an object is not null and then call methods of this object in the same conditional expression)
- String concatenation
- No sizeof() operator
- Casting and promotion

Casting and promotion

- Can cast from any primitive value to any other primitive value. There is no casting on boolean
- if you perform any mathematical or bitwise operations on primitive data types that are smaller than an **int** (that is, **char**, **byte**, or **short**), those values will be *promoted* to **int** before performing the operations, and the resulting value will be of type **int**. So if you want to assign back into the smaller type, you must use a cast.
- In general, the largest data type in an expression is the one that determines the size of the result of that expression; if you multiply a **float** and a **double**, the result will be **double**; if you add an **int** and a **long**, the result will be **long**.

Flow control

- If, else, if else
- For, comma operator
- For each
- While
- Do while
- Break and continue

For each loop in Java

```
for (int cell : locationCells) {
}
```

For each element in locationCells array assign element to a variable *cell* and perform the body of the loop with this variable

Avoiding common mistakes

while(x=5)

Expects boolean data type, but x=5 is not boolean

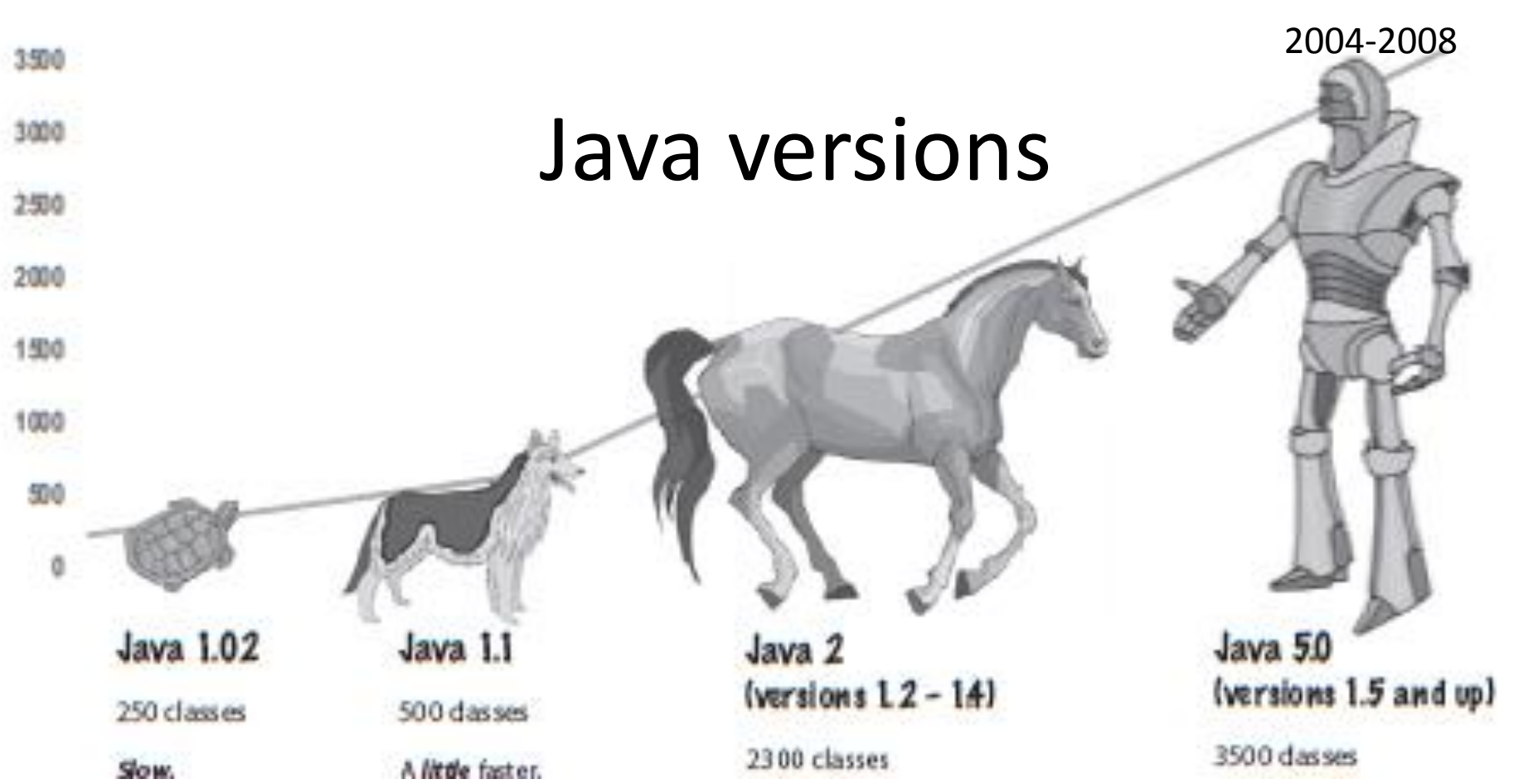
Do … while

do { statement(s) } **while** (expression);

```
//password checking
boolean validPassword=false;
do
{
        String password=getUserPassword();
        if(database.isValid(password))
                validPassword = true;
}while (!validPasword)
```

Java versions

2004-2008



Slow.
Cute name and logo. Fun to use. Lots of bugs. Applets are the Big Thing

A little faster.
More capable, friendlier. Becoming very popular. Better GUI libraries.

Much faster.
Can (sometimes) run at native speeds. Serious, powerful. Comes in 3 flavors: J2ME, J2SE, J2EE. Becomes the language of choice for enterprise (WEB and mobile) applications

More power (known as “tiger”).
Easier to develop with. Changes in language, adding features popular in other languages

Java latest versions

- **Java 6 (“Mustang”) (from 2006)**

Dramatic performance improvements

The first release to offer a standardized framework for scripting languages. Easier user input with System.console(). Buffer-oriented, non-blocking NIO library. Compiler API. Swing double-buffering (eliminating the gray-area effect). Improved JVM garbage collection algorithms.

- **Java 7 (“Dolphin”) (from 2011)**

Enhancements in the JVM to support Non-Java languages: multi-language virtual machine. Strings in switch. Binary Integer literals (no bit manipulation needed). Xrender graphic card support of 2D engines. Enhanced support of Socket Direct Protocol

What we are using

- Textbook for Java 6 (3-rd edition)
- Lab – Java 6

Your Object-Oriented design toolbox

Concepts

Abstraction

Encapsulation

Object

Object class (type)

Properties

Methods

Association

Composition

Skills

UML diagrams

Class definition

Composition relationship

Association relationship

Your Java toolbox

Language Concepts

Variable

Primitives

Reference variables

Class and Instance

Object constructor

Instance variables and local variables

Arrays

Static variables and methods

Programming Skills

Using classes

Defining classes

Implementing classes

Setting association relationships

Manual tests