Java Crash Course
Part III

School of Business and Economics
Institute of Information Systems
HU-Berlin WS 2005

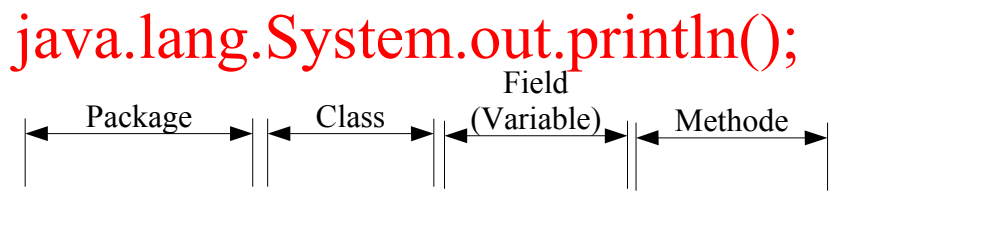
Sebastian Kolbe
skolbe@wiwi.hu-berlin.de

What you should know so far

- How to compile and start a Java application
- Java-Syntax and concepts
  - Variables
  - Operators
  - Control structures (if-then-else, loops)
  - Conventions in Java
    (capital letter for classes, lowercase letter for methods,...)
  - Modifiers for classes, variables and methods
    (public, protected, private, static)
- Classes and objects
  - General idea
  - Java Syntax

Predefined functions in Java

- Java brings a huge database of predefined functions in several libraries
- Hierarchical organisation



- Classes with: `java.[..]` belong to the standard Java SDK package
- Using other packages as java.lang requires import of package: `import java.util.*;`

- See Java-Documentation

http://coltrane.wiwi.hu-berlin.de/lehre/2005w/isi/docs/api/index.html
http://www.java.sun.com/

Exceptions and Errors

- Unexpected states in a program can end your program or interfere proper operation
- Java knows for this case a special procedure called **exception**
- Interrupts the program flow and allows to correct this state
- Example:
  - Opening a file (if file is not readable -> Exception)
  - Accessing array elements that don't exist -> Exception
  - Calculating a value and dividing by 0 -> Exception
- Statements that needs to be called in every case can be putted in a *finally* clause

Exceptions in Java

- Exceptions can be caught or thrown
- Statements that potentially disturb program flow can be isolated by keyword: try

```
try {
   String parameter1 = args[2];
}
catch (ArrayIndexOutOfBoundsException exp ) {
   System.out.println ( “Wrong parameter” );
}
```

- If you don't want to treat the exception in the local method you can throw it (to the calling method)

```
public int calculation( int arg1 )
        throws IOException
{ ... }
```

Example

- Calculating the Fibonacci sequence
  - Every number is the sum of the two previous numbers
  - $n_i = n_{i-1} + n_{i-2}$
  - $n_{i+1} = n_i + n_{i-1}$
  - with $n_0 = 0$, $n_1 = 1$
  - Example: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34

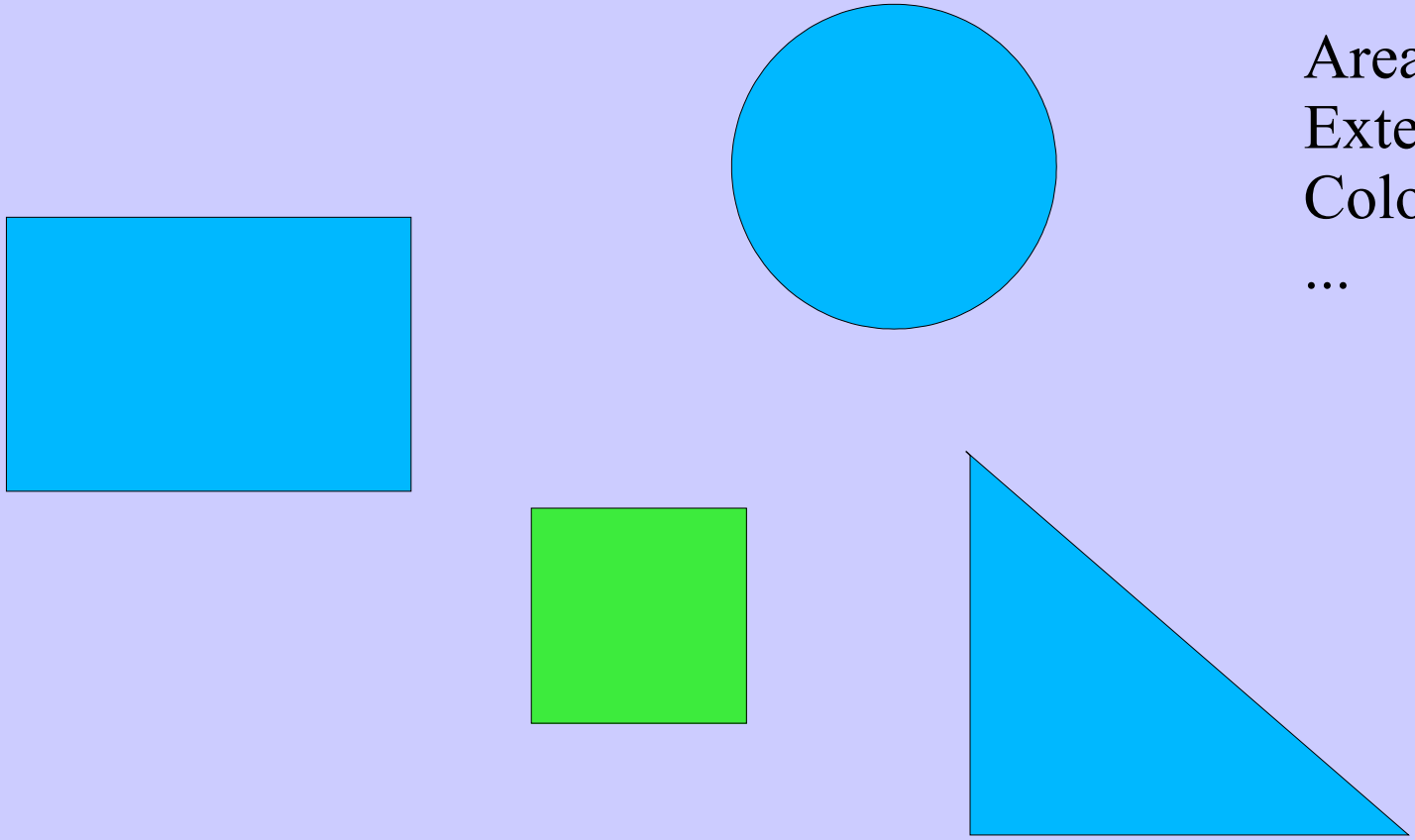
  - Leonardo da Pisa (a.k.a. Fibonacci) wanted to mathematically simulate (model) the growth of a rabbit population (Year 1202)
  - http://de.wikipedia.org/wiki/Fibonacci-Folge

Inheritance

- A subclass inherits behavior (variables and methods) from its ancestors
- From the subclass point of view the ancestors are called superclasses
- Advantages:
  - reusing of code
  - Cleaner design
    - Encapsulation of existing functionality
    - Only adding cumulative functionality
- Disadvantages:
  - sometimes more code to write (framework)
  - complex structures are hardly to understand
    -> **USE COMMENTS & document your program**

A more complex example

Think about geometric figures...



Area
Extend
Color
...

They all have things in common...
and other things uncommon

Creating hierarchies