Summer Term 2011

Softwareentwicklung in der Geoinformatik Praxis

Java

Basic Functionality – Object-oriented Programming

Bernd Resch

Universität Salzburg

06 April 2011

Overview

- Organisational
- Assignment 1 Retrospect
- Introduction
- Object-oriented Programming…

Organisational ::: Dates

- Wed, 16 March 2011 09.00-12.00 ➔ HS 3.004
- Wed, 06 April 2011 09.00-12.00 ➔ HS 3.004
- Wed, 13 April 2011 09.00-12.00 ➔ HS 3.004
- Wed, 04 May 2011 09.00-12.00 ➔ HS 3.004
- Wed, 18 May 2011 09.00-12.00 ➔ HS 3.004
- Wed, 01 June 2011 09.00-12.00 ➔ HS 3.004
- Wed, 15 June 2011 09.00-12.00 ➔ HS 3.004

- *Fri, 15 July 2011* *End-term Assignment Due Date*

Assignment 1 Retrospect

Java ::: Assignment 1 Retrospect

- Create a class called “CircleArea”, which calculates the area of a circle:
  - The radius of the circle is 4 units

Java ::: Assignment 1 Retrospect

- Create a class called “CircleAreaUserInput”, which calculates the area of a circle:
  - The radius of the circle shall be entered by the user

  ➔ BufferedReader

  ➔ Exception Handling (readLine() method)

  ➔ String to Integer conversion

Java ::: Assignment 1 Retrospect

- Read user input:
  - *BufferedReader br = **new** BufferedReader(**new** InputStreamReader(System.in));*
  - *br.readLine();*
- Define pi: *java.lang.Math.PI* (import or fully qualified name)

Java ::: Assignment 1 Retrospect

- Exception handling:

```
try {
    user_input = br.readLine();
} catch (IOException e) {
    e.printStackTrace();
}
```

- NB: be cautious on where to place the exception handling ➔ intended behaviour of the program!
- Close BufferedReader object - *br.close();*
- Convert radius from String to number:

```
double radius = Double.valueOf(user_input);
int radius = Integer.valueOf(user_input);
int radius = Integer.parseInt(user_input);
```

Java ::: Assignment 1

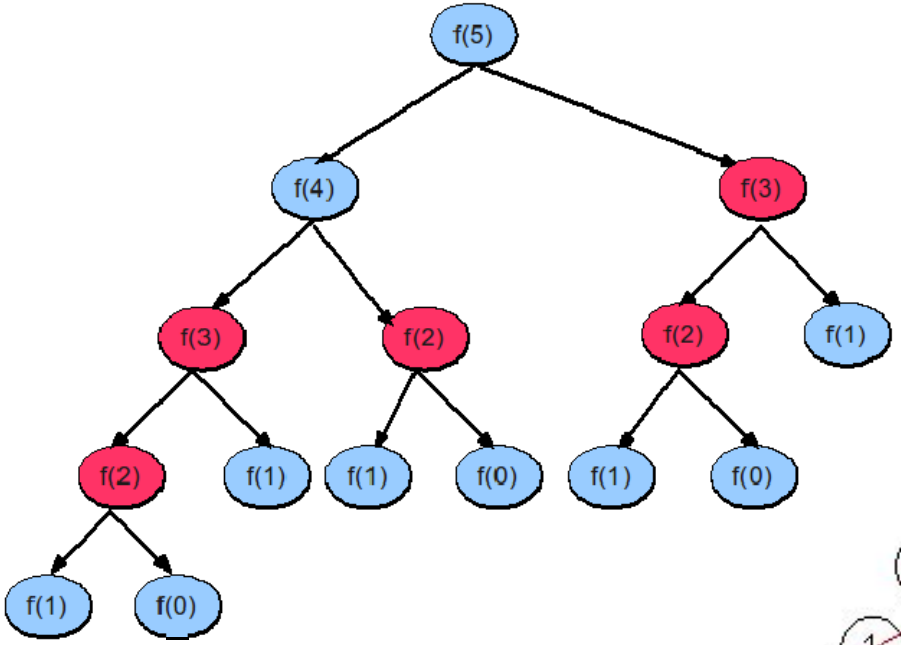
- Create a class called “Fibonacci”, which prints the Fibonacci sequence:
  - Fibonacci: the printed number is the sum of the two previous numbers
  - Definition: the first two numbers are 0 and 1
  - Print 20 iterations

Java ::: Assignment 1 Retrospect

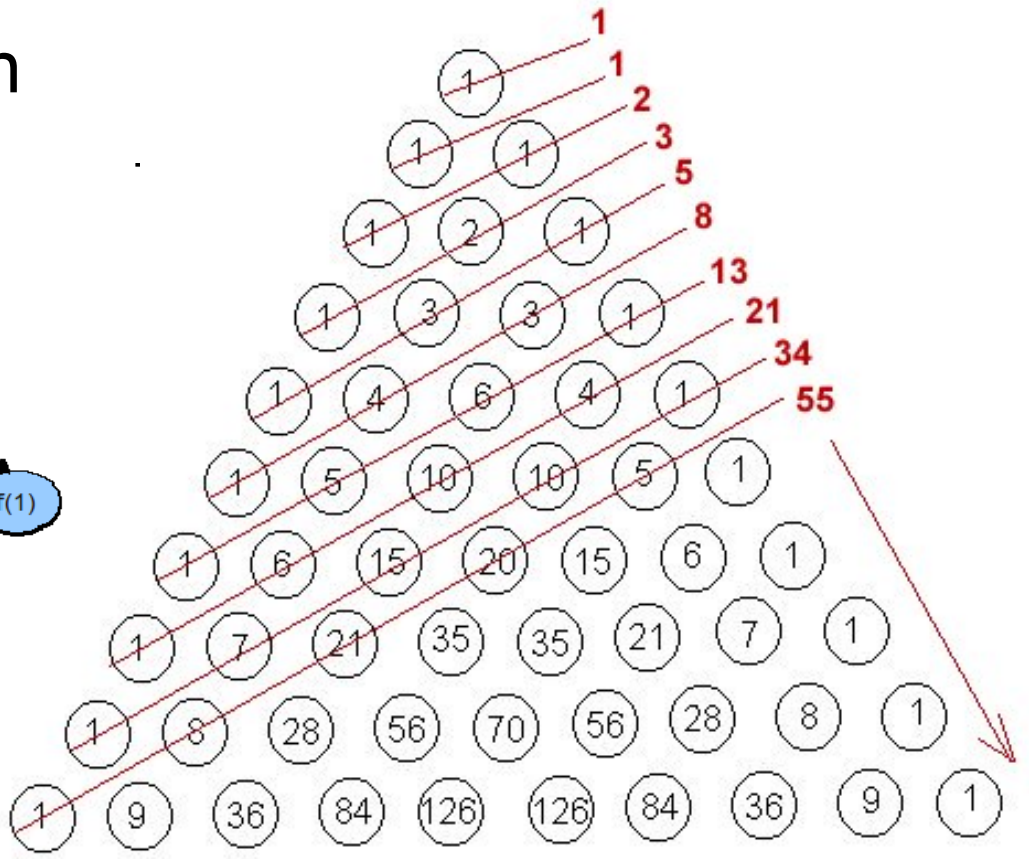
- 1.) Temporal storage of values
  - Integer variables
  - Integer array

Java ::: Assignment 1 Retrospect

- 2.) Recursive solution



Quelle: http://navaneethkn.wordpress.com



Quelle: http://www.tutornext.com

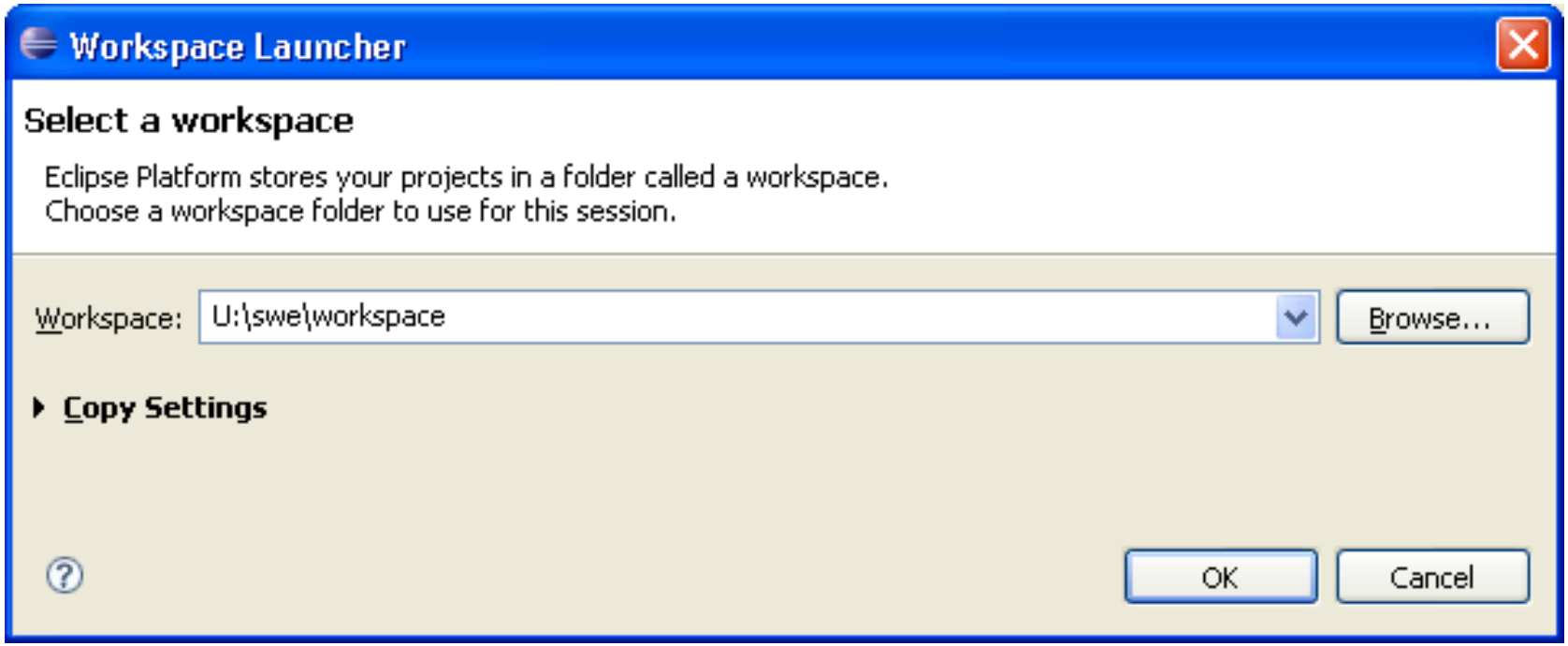
Introduction to Eclipse

Introduction to Eclipse

- IDE – Integrated Development Environment
- One-click:
  - Compliation
  - Execution
  - Debugging

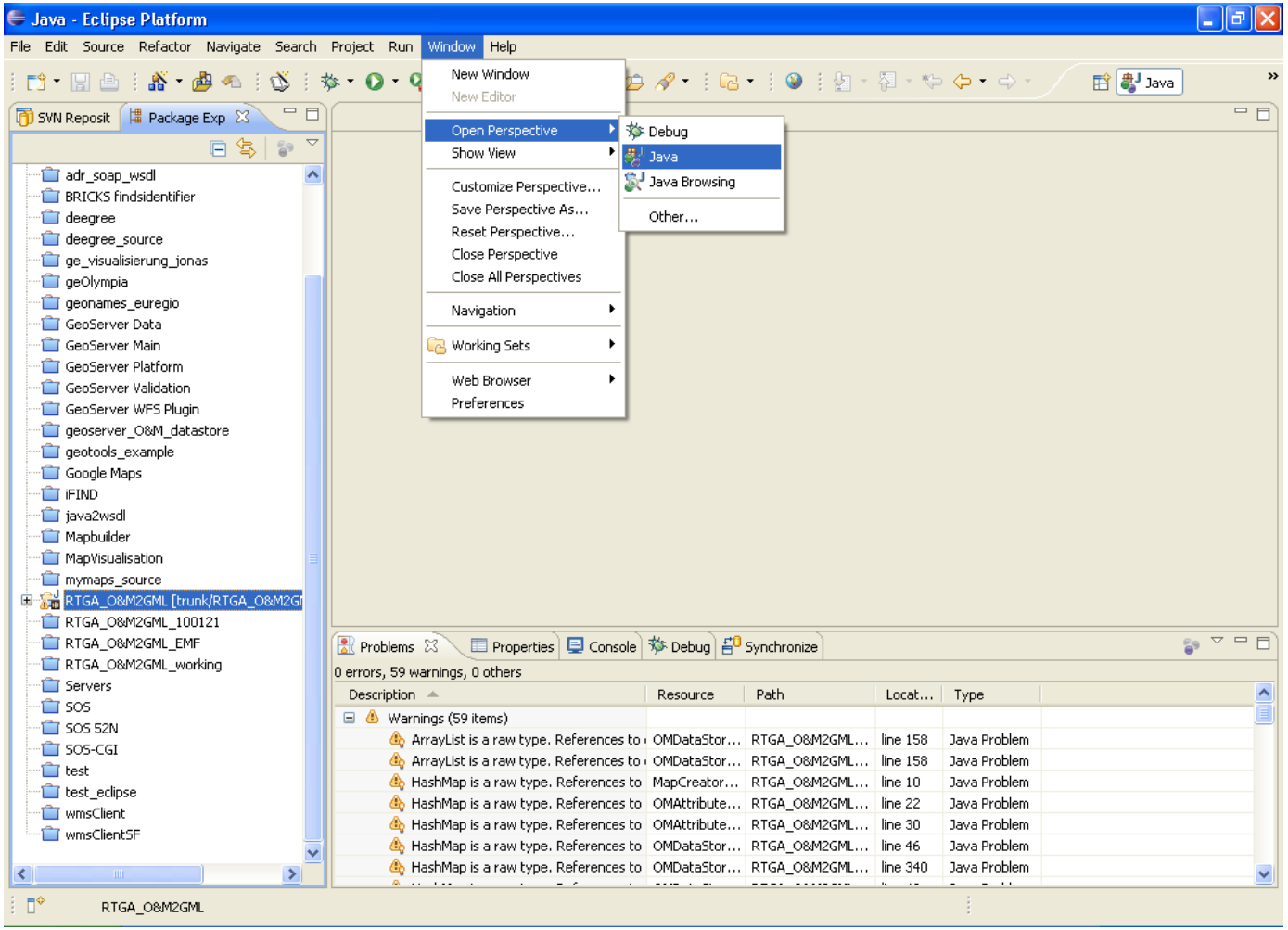
Introduction to Eclipse

- Start-up – Workspace Launcher
  - U:\swe\workspace



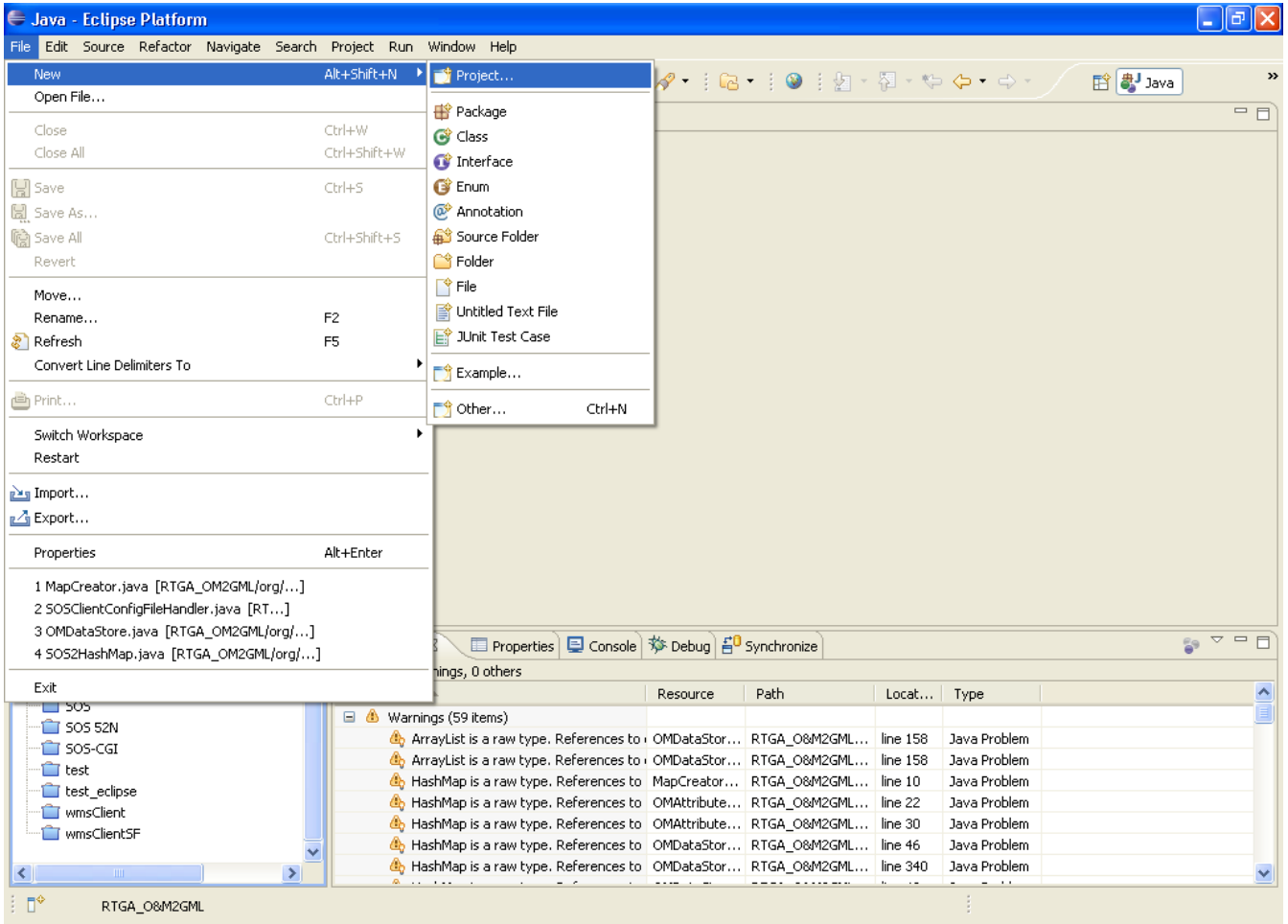
Introduction to Eclipse

- Switch to „Java Perspective“



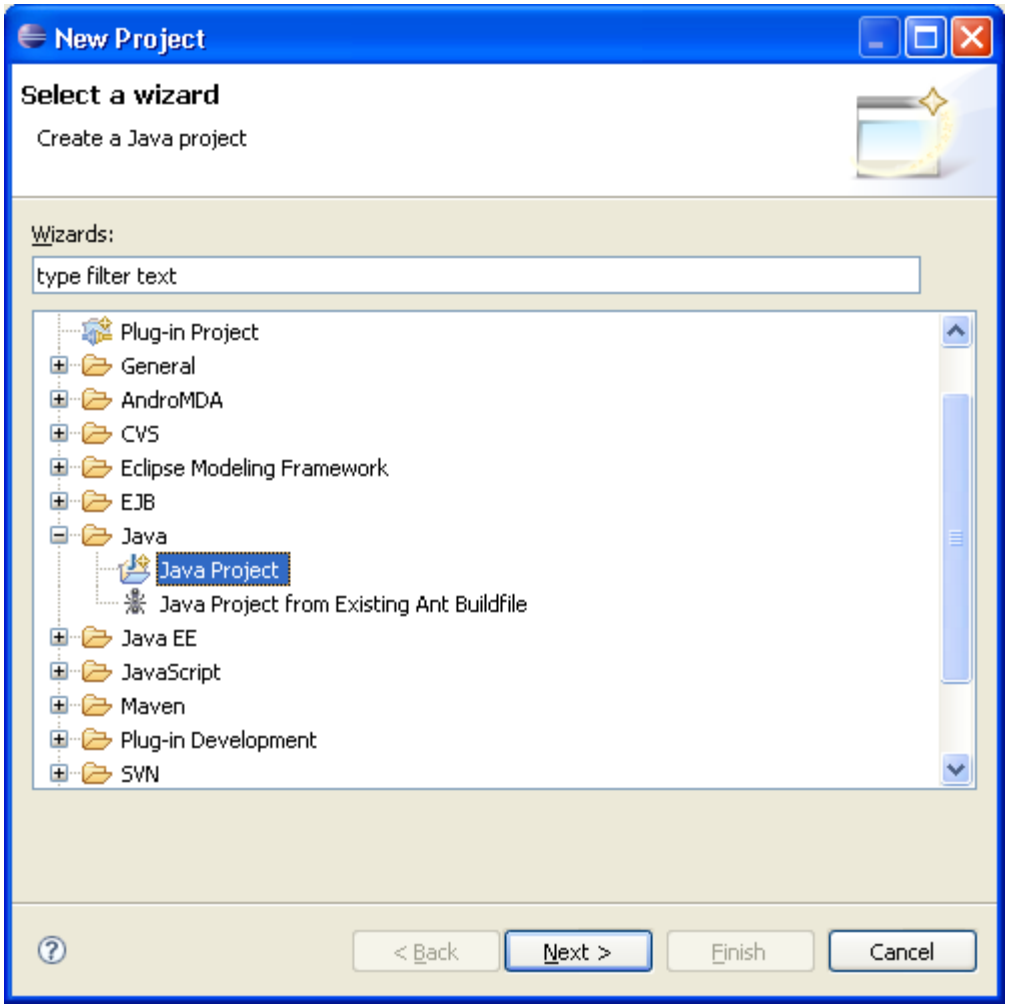
Introduction to Eclipse

- Create a new project



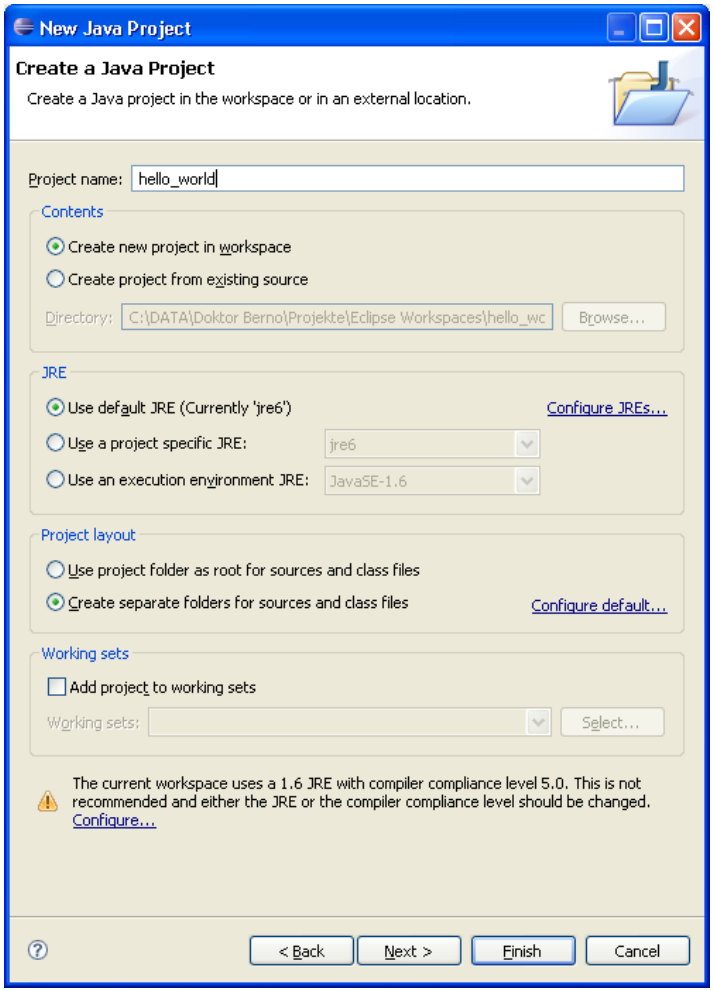
Introduction to Eclipse

- Create a new project



Introduction to Eclipse

- Create a new project
- “unit2”



Java Programming Conventions

- Class names start with capital letters
- Method names start with lower case letters
- No umlauts and symbols
- Inline documentation (// , /* … */)
- Java is case sensitive!

Java Programming in Eclipse

- Let's get started…

- Java tutorials:

➔ http://www.tutorialspoint.com/java/index.htm

➔ http://download.oracle.com/javase/tutorial

➔ http://home.cogeco.ca/~ve3ll/jatutor0.htm

Java Programming

Java ::: Exercise

- Last unit: simple (procedural) programming
  This unit: object-oriented programming

Java ::: Exercise

- Create an executable class called “CubeSimple”, which
  - Declares 3 global integer variables (length, height, breadth)
  - Assigns values to these variables (10, 10, 10) in the main() method
  - Calculates the volume of the cube and prints the result to the screen

Java ::: Exercise

- Create a non-executable class called “Cube”, which
  - Declares 3 global integer variables (length, height, breadth)
  - Defines the 2 methods *getVolume()* and *getSurfaceArea()* (returning the according values)
  - Defines a method *setLBH(int, int, int)* setting the variables length, height, breadth

Java ::: Exercise

- Create a **constructor** for the class "Cube":
  - Cube()
  - In the constructor, set the values for length, breadth and height to 10

Java ::: Exercise

- Create a **constructor** for the class "Cube":
  - Cube(int l, int b, int h)
  - In the constructor, assign the input parameters to length, breadth and height

Java ::: Exercise

- Create an executable class “CubeObjectOperations”, which
  - Calls the method *setLBH(100,100,100)* of the first Cube object before doing the calculations

Java ::: Exercise

- Create an executable class “CubeObjectOperations”, which
  - Declares 3 global integer variables (length, height, breadth)
  - Assigns values to these variables (10, 20, 30) in the *main()* method
  - Creates 2 instances (objects) of the class “Cube” (one with each of the constructors)
  - Prints the volumes and surfaces of the Cube objects to the screen
  - Calls the method *setLBH(100,100,100)* of the first Cube object before doing the calculations

Java ::: Assignment 2 Geotools

- Create a class "CoordinatesTransformer", which transforms the WGS84 position *47.788901 N, 13.060209 E* to UTM33N
  - Use GeoTools 2.7.0 (http://www.geotools.org)
    ➔ geotools-2.7.0-bin.zip
  - GT2 contains several jar archives, which you will have to include into your project

Java ::: Assignment 2 Geotools

- Hints:
  - EPSG properties file (containing WKTs) http://svn.osgeo.org/geotools/tags/AFTER_JTS1.6/plugin/epsg/src/org/geotools/referencing/crs/epsg.properties
  - WGS84 = EPSG:4326
    UTM33N = EPSG:62633
  - The use of WKTs (Well-known Texts) for CRSs is recommended

Java ::: Assignment 2 Geotools

- Step 1: find a possible solution/approach
  - Create a position object
  - Create a source and a target CRS object
  - Transform the position

Java ::: Assignment 2 Geotools

- Step 2: ask an Internet search engine (ready-to-use code?)
- Step 3: download GeoTools 2.7.0
  - Integration into project's build path in Eclipse: Project → Properties → Java Build Path → Add JARs
- [Step 4: find the appropriate classes/jars (extraction)]
- Step 5: download epsg_properties.txt
- Step 6: coding… ➔ Javadocs!

Java ::: Assignment 2 Geotools

- Possible solution:
  - Create a position object (class *DirectPosition2D*)
  - Create the source and target WKTs (String variables)
  - Create the according CRS objects using these WKTs (class *CoordinateReferenceSystem*)
  - Create a *MathTransform* object (method *CRS.findMathTransform(sourceCRS, targetCRS)*);
  - Transform the position (method *math.transform()*);

Java ::: Assignment 2

- Put all the java files in a package named "assignment2_<name1>_<name2>"
- Submission by email:
  - Java code files
  - Including inline documentation of key parts
  - Short description of approach (directly in the email)
- Due 10 May 2011

Summer Term 2011

Softwareentwicklung in der Geoinformatik Praxis

Java

Basic Functionality – Object-oriented Programming

Bernd Resch

Universität Salzburg

06 April 2011