Compile, execute, debugging

THE ECLIPSE PLATFORM

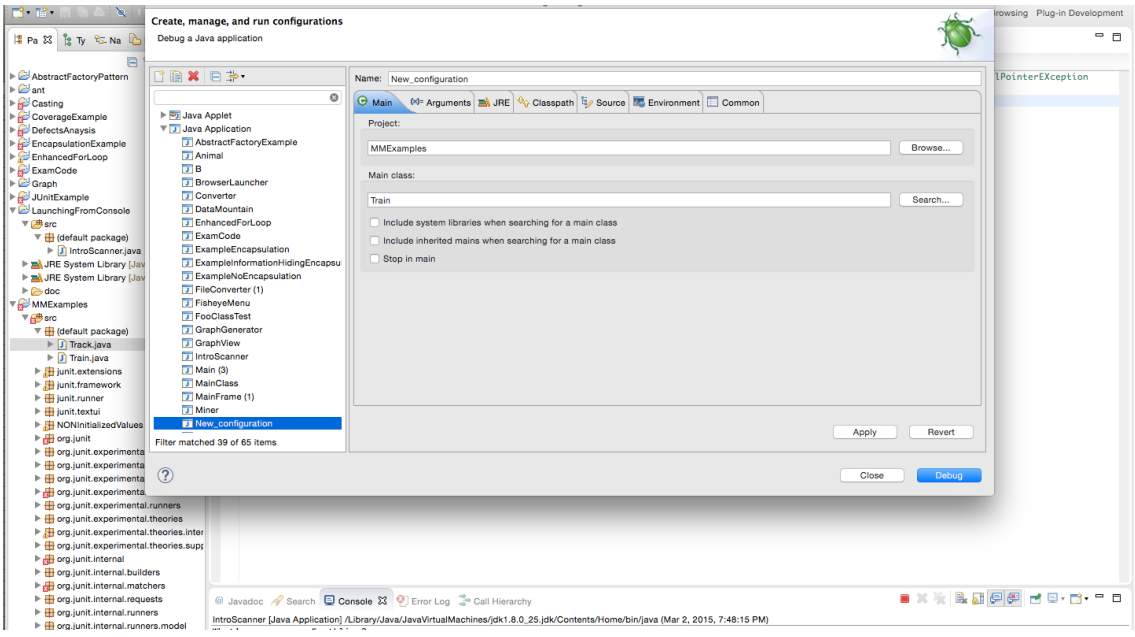
JSE8

- ECLIPSE LUNA

Compile, execute, debug

- Create a java project
- Import the file into your project
  - You can see the mistakes. Debug the class
- Fix the mistakes and click the run button
  - Nothing happens as you forgot to enter the variables
- Configure the execution tab (do not forget to include parameters)
- Configure the debug tab (do not forget to include parameters)

Debugging



Javadoc

The special comments in the Java source code that are delimited by the /** ... */ delimiters are processed by the Javadoc tool to generate the API docs

Use @ tags to create the structure of the comment

```
/**
 * Returns an Image object that can then be painted on the screen.
 * The url argument must specify an absolute {@link URL}. The name
 * argument is a specifier that is relative to the url argument.
 * <p>
 * This method always returns immediately, whether or not the
 * image exists. When this applet attempts to draw the image on
 * the screen, the data will be loaded. The graphics primitives
 * that draw the image will incrementally paint on the screen.
 *
 * @param  url  an absolute URL giving the base location of the image
 * @param  name the location of the image, relative to the url argument
 * @return      the image at the specified URL
 * @see         Image
 */
public Image getImage(URL url, String name) {
      try {
          return getImage(new URL(url, name));
      } catch (MalformedURLException e) {
          return null;
      }
}
```

Commenting your code

- There are two ways to comment your code
- Including your comments in // or /* */
- Using additionally the Javadoc commands together with /* */
- If you type /* and go next line you get automatically the frame

```
/*
 *
 */
```

Commenting your code

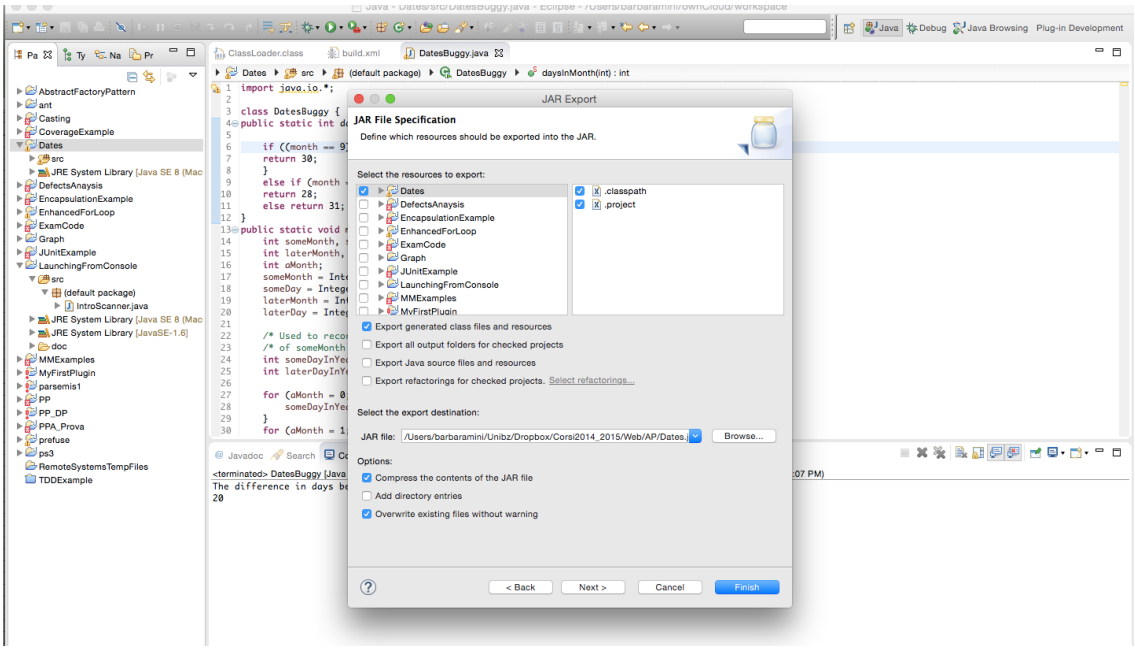
- If you write /** and next line before a method or a class, you also get automatically the tags about the method or the class (e.g., parameters and return type)
- Of course you can also embed other info
- When you generate the javadoc for that file you get an html file that describe the API of your file.
- Before generating the javadoc you must have created a doc folder

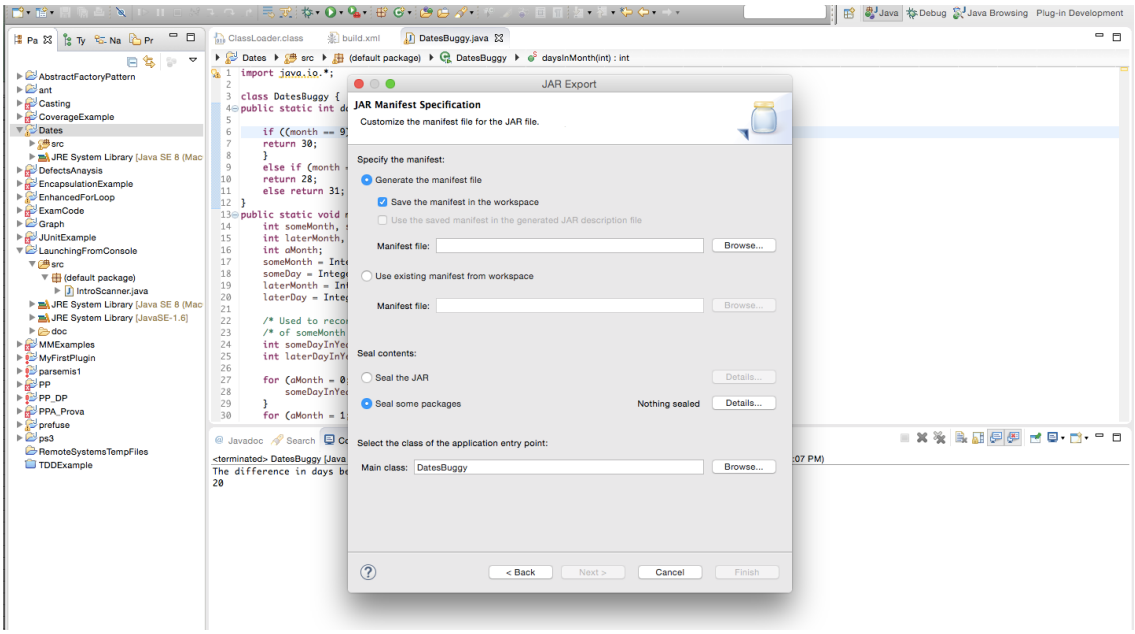
Create your first API

- First create a folder where to store the html files
- Then execute “Generate Javadoc” from the Project tab

Your first jar file

- The jar command compresses your project into a portable file
- You can easily send your jar file to anyone
- You can put .java and/or .class file
- you need to include a manifesto file
- In Eclipse start by right-clicking on the project name in explorer
- Then select export


JAR Export
JAR File Specification
Define which resources should be exported into the JAR.
Select the resources to export:
Dates
DefectsAnaysis
EncapsulationExample
EnhancedForLoop
ExamCode
Graph
JUnitExample
LaunchingFromConsole
MMExamples
MyFirstPlugin
.classpath
.project
Export generated class files and resources
Export all output folders for checked projects
Export Java source files and resources
Export refactorings for checked projects. Select refactorings...
Select the export destination:
JAR file: /Users/barbaramini/Unibz/Dropbox/Corsi2014_2015/Web/AP/Dates.
Browse...
Options:
Compress the contents of the JAR file
Add directory entries
Overwrite existing files without warning
< Back
Next >
Cancel
Finish


JAR Export
JAR Manifest Specification
Customize the manifest file for the JAR file.
Specify the manifest:
Generate the manifest file
Save the manifest in the workspace
Use the saved manifest in the generated JAR description file
Manifest file:
Browse...
Use existing manifest from workspace
Manifest file:
Browse...
Seal contents:
Seal the JAR
Details...
Seal some packages
Nothing sealed
Details...
Select the class of the application entry point:
Main class: DatesBuggy
Browse...
< Back
Next >
Cancel
Finish

Create a jar file

In the Package Explorer, you can optionally pre-select one or more Java elements to export.

- Select the project you want to export and follow the wizard
- In the JAR File Specification page, select the resources that you want to export in the Select the resources to export field.
- Select the appropriate checkbox to specify whether you want to Export generated class files and resources or Export Java source files and resources.
- In the Select the export destination field, either type or click Browse to select a location for the JAR file.
- Select or clear the Compress the contents of the JAR file checkbox.
- Click Next to use the JAR Packaging Options page to set advanced options, create a JAR description, or change the default manifest.

Executing jar files

- Command line
  - > java –jar file.jar

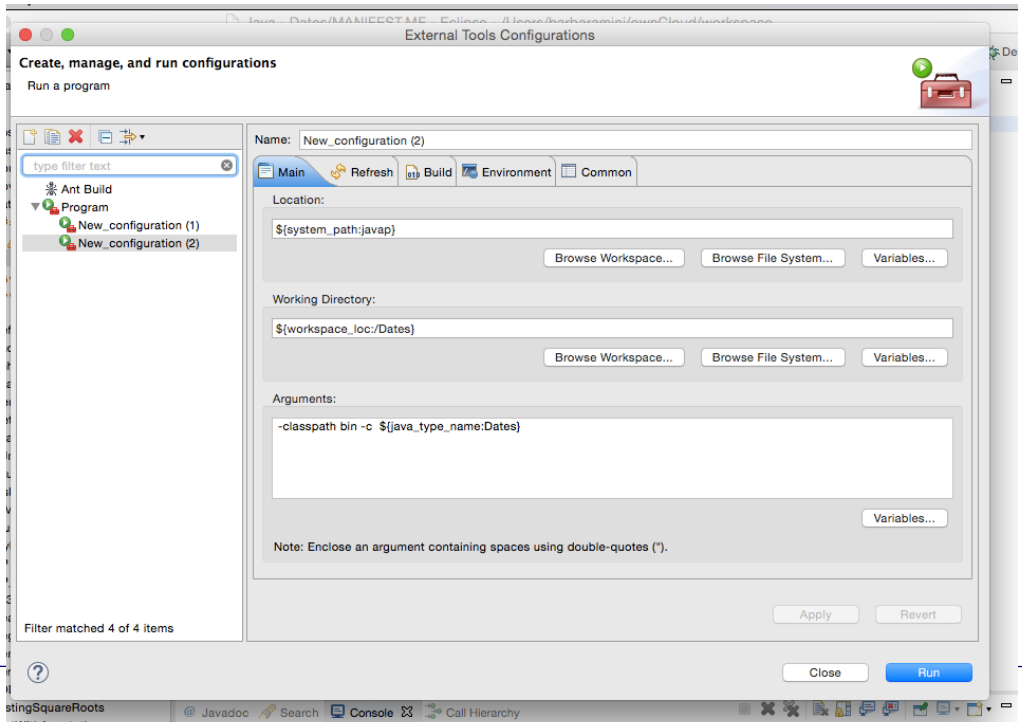
Jar and runnable jar execution

```
BarbaraMini:AP barbaramini$ ls
Analyzer.class                  JavaPlatform.pdf
Analyzer.java                   JavaPlatformNotes.pdf
Dates.jar                       JavaReferenceFrontPage.png
DatesBuggy.java                 Quizes.html
DatesRunnable.jar               Quizes.html.bak
Introduction.pdf                SummerSemesterDRAMALAB.pdf
BarbaraMini:AP barbaramini$ java -jar Dates.jar
no main manifest attribute, in Dates.jar
BarbaraMini:AP barbaramini$ java -jar DatesRunnable.jar
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 0
        at DatesBuggy.main(DatesBuggy.java:17)
BarbaraMini:AP barbaramini$ java -jar DatesRunnable.jar 1 12 3 4
The difference in days between 1/12 and 3/4 is:
20
BarbaraMini:AP barbaramini$
```

javap in Eclipse

- Go to the run tab
- Select External Tolls/ Configuration
- Define the parameters

javap Configuration



javap in Eclipse

- Run the configuration
- Look at the console :-)