Laurea University of Applied Sciences

Object-Oriented Programming with Java

Student's workbook

Compiled by Antonius Camara

Table of Contents

How to use this workbook

This workbook contain guidelines and instructions to support your studies during the course.

This workbook is not intended to be a “Textbook” that contains detailed descriptions and explanations about the course theory. For that, you should use proper resources of your preference (books, electronic books, online manuals, and online tutorials).

In the different sections of this workbook you will find notes, guidelines and explanations from the teacher related to different topics and activities of the course. You should refer to this document when planning your studies and project work.

The teacher will keep updating this workbook during the course, so you should check in Optima for new versions. The teacher will also notify you at the class when a new version is published.

Learning resources recommended by the teacher

The table below lists some learning resources recommended by the teacher. Some of the course homework refers to some of the se resources. Feel free to search and use other resources of your preference. If you find some interesting resource that you think could be added to this list, please suggest it to the teacher.

Resource	Type	What's in it	Link
Viope course "Object-Oriented Programming with Java (Spring 2017)"	Online course	Programming exercises supported by Viope's automatic code checking	
Beginning Java Programming: The Object-Oriented Approach, Baesens Bart, Wiley, 2015	Electronic Book (Laurea's EBL)	Very good introduction to the principles and concepts of Object-Orientation	http://www.laurea.eblib.com/patron/FullRecord.aspx?p=1895071
Beginning Java 8 APIs, Extensions and Libraries, Kishori Sharan, Apress 2014	Electronic Book (Laurea's EBL)	Very good descriptions and details about Swing, Layout Managers, Swing components	http://www.laurea.eblib.com/patron/FullRecord.aspx?p=1964735
Java – How to Program, 8th Edition, Paul Deitel, Pearson, 2010	Hard copy from Laurea's library	Comprehensive coverage of all topics. Very good explanations about event handling with lots of examples	
Java 2 - Ohjelmoinnin peruskirja, Kosonen Pekka, Docendo, 2008	Hard copy from Laurea's library	If you prefer to read in Finnish. Broad coverage of several topics	

How to use Viope in this course

Viope is offered in this course as an online learning resource where you can practice your programming skills from home. The Viope course is configured in such a way that Viope's own example answers are shown after you successfully submit a working solution for an exercise. This way you can compare the code you created with Viope's standard solution.

In addition to this, the teacher has created a discussion forum in Optima where you can post the solution you coded. If someone else has already published a solution for one exercise, you can add your solution as an answer to that first solution. This way, students can find and compare different solutions for a same exercise under a same discussion thread.

If you are the first one to post a solution for one exercise, add as a title of your posting the number of the exercise and a short name describing it. For example: **2-1 Object introduction**

Object-Oriented Programming – Key Concepts

During the contact sessions the teacher will explain several concepts related to Object-Oriented Programing Principles.

As part of your independent studies (homework), you should review these concepts by reading and studying it from the learning resources you have chosen for yourself.

The following are the concepts you should study and learn:

- What is “Object-Orientation”?
- Objects and Classes
- Attributes and Behaviors (Operations, Methods)
- Constructors
- Encapsulation and Information Hiding
- Instance variable, Class variable, Final variable, Variable scope
- Access modifiers, Getters, and Setters
- Instance methods vs. Class methods
- Polymorphism and Inheritance
- The “this” keyword
- Method Overloading and Method Overriding
- The superclass “Object” and its methods (Java)
- Packages (Java)
- Interfaces
- Object-Oriented programming vs. Procedural programming

Hint from the teacher:

Many of the concepts above are nicely described in the electronic book “Beginning Java Programming”:

- chapter 4: Moving Toward Object-Oriented Programming
- chapter 7: Delving Further into Object-Oriented Concepts

Installing Eclipse Neon and WindowBuilder

Installing Eclipse (Neon)

Download and install Eclipse Neon from https://www.eclipse.org/downloads/. Follow the standard installation process for a fresh Neon installation. If you are presented with a list of different configurations to be installed, chose “Eclipse IDE for Java Developers”.

Installing WindowBuilder

WindowBuilder is an easy to use Java GUI designer. It is installed as a plug-in under the Eclipse IDE.

Eclipse’s WindowBuilder project page is located at https://eclipse.org/windowbuilder/. There you can find general information about this plug-in and have access to other important links such as documentation and download.

To install WindowBuilder use the instructions located at http://help.eclipse.org/kepler/index.jsp?topic=/org.eclipse.wb.doc.user/html/installation/index.html.

If this is the first time you are installing WindowBuilder, follow the steps under section “Installing software by adding a new update site location”.

The URL of the product’s site location can be found from the table “Update Sites” located at http://www.eclipse.org/windowbuilder/download.php. Make sure to select the link corresponding to the Eclipse version you are using (Neon).

When following the installation instructions, note that after you copy/paste the update site URL into the Install/Available Software window, you also need to press Enter, so that the window can update the list of items to be installed. If you just copy/paste the update site URL without pressing enter, the list of items to be installed will not be updated. If you are using Eclipse Neon, the list of items to update will only contain one check box for “WindowBuilder”. Tick that box and continue the update process.

Programming GUI's with Swing and WindowBuilder

Key Concepts

Designing GUIs in Java/Swing require you to understand several concepts. During the course's contact sessions the teacher will explain the concepts. You should also independently study these concepts to ensure you understand its foundations and that you are able to apply it in your projects:

- GUI Libraries in Java (AWT, Swing, SWT, JavaFX)
- Swing Containers
- Swing Components
- Layout Managers
- Event Handling
- Eclipse plugin WindowBuilder

The following books/chapters contain good instructions and explanations about these topics. Feel free anyway to use any other resources of your preference.

- Electronic book "Beginning Java 8 APIs, Extensions and Libraries", chapter 1 (Introduction to Swing) and chapter 2 (Swing components).
- Electronic book "Beginning Java Programming" chapter 11 (Designing Graphical Interfaces)

Steps to start programming your application

1. Create a new Java project in Eclipse
2. Create a Class for the home screen of your application using the template code provided in this workbook (File name here)
3. Edit the template code to adjust the home screen properties according to your project

Which layout manager to use in your project?

You can freely decide which layout managers to use in your project. You are encouraged to explore the capabilities of different layout managers to decide which one best suits your needs. If you search the internet, you will find different discussions about the applicability and usefulness of the Swing layout managers. For example:

- http://zetcode.com/tutorials/javaswingtutorial/swinglayoutmanagement/
- http://stackoverflow.com/questions/1832432/which-swing-layouts-do-you-recommend

You don't even need to stick to only one Layout Manager as you can create a hierarchy of Containers each using a different Layout Manager.

Under this course's perspective however, you are not expected to master the details of layout building and layout management in Java. This is something you will learn gradually if you continue programming in Java. For this course, you are only required to understand the GUI programming key concepts and build simple GUIs.

For this purpose is enough you use WindowBuilder to position UI components in a frame configured with a "null layout manager" (Absolute positioning). The end result can be interfaces containing components that will not automatically resize and reposition themselves when you re-size the application's windows, but this will be an acceptable behavior in your project.

Using Github in this course

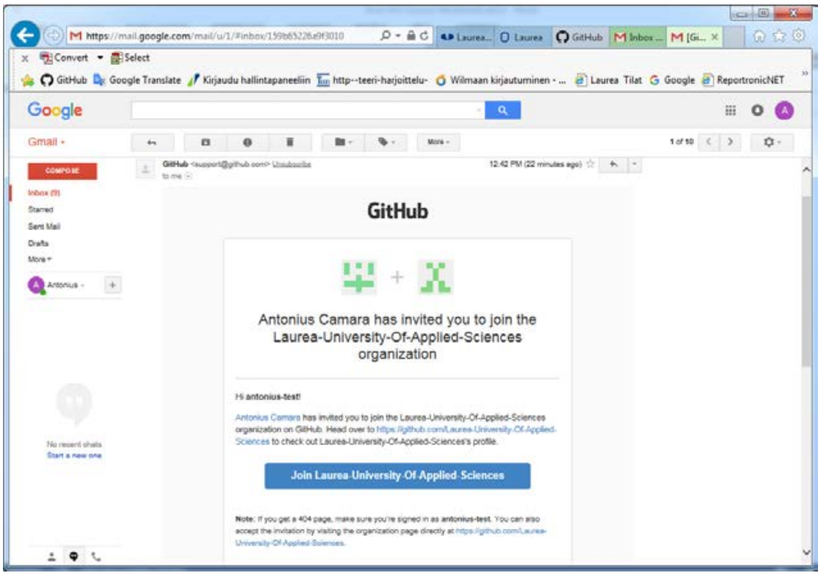
Accessing code used in the workshops (contact session)

If you don't have a GitHub account, create one. Go to www.github.com and signup.

Send an email to the teacher informing the username of your GitHub account.

The teacher will add you to the team "R0317-Spring2017" under the organization "Laurea-University-Of-Applied-Sciences" (In GitHub)

After the teacher adds you to the team, you will receive an email informing that the teacher has added you to "Laurea-University-Of-Applied-Sciences". On that email, click on the button "Join Laurea-University-Of-Applied-Sciences". See screen shot:



GitHub

Antonius Camara has invited you to join the Laurea-University-Of-Applied-Sciences organization

Hi **antonius-test**!

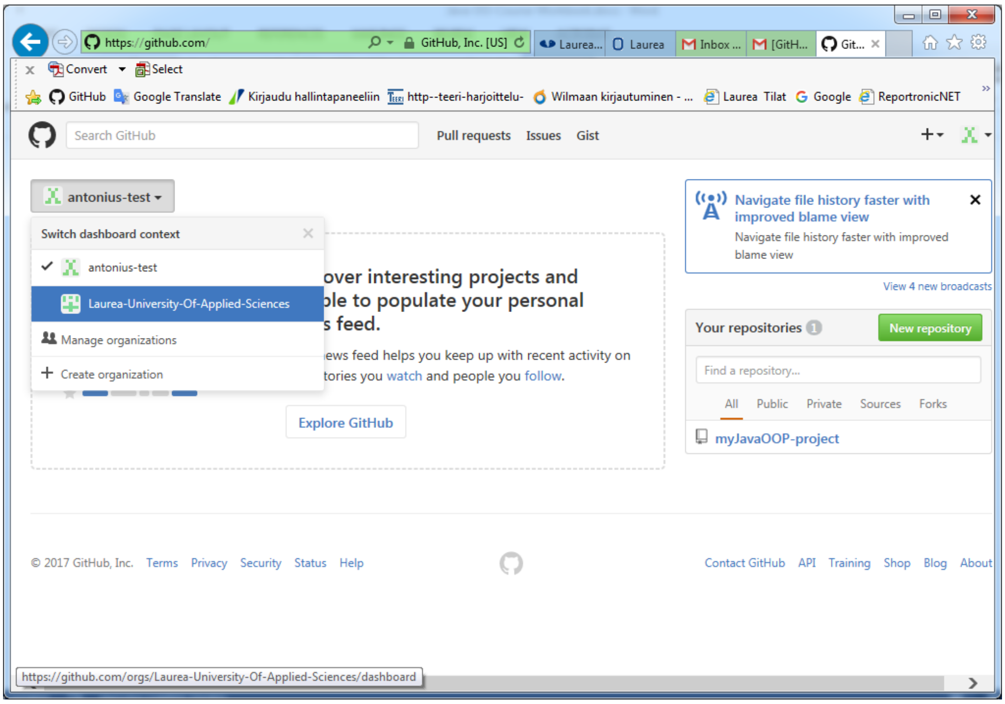
Antonius Camara has invited you to join the Laurea-University-Of-Applied-Sciences organization on GitHub. Head over to https://github.com/Laurea-University-Of-Applied-Sciences to check out Laurea-University-Of-Applied-Sciences's profile.

Join Laurea-University-Of-Applied-Sciences

Note: If you get a 404 page, make sure you're signed in as **antonius-test**. You can also accept the invitation by visiting the organization page directly at https://github.com/Laurea-University-Of-Applied-Sciences.

After these steps are completed you should be able to access the repository "Object-Oriented-Java". Example code and exercises used the course's workshop can be found under this repository. Note that you only have "Read" access to this repository, therefore you will not be able to upload files or edit files. The only operations you can do are "Download" or "Clone". At this stage it is enough you just download the files you are interested in.

To access this repository, make sure to switch to the "Laurea-University-Applied-Sciences" dashboard context. See the screen shot below.



over interesting projects and
ple to populate your personal
s feed.

ews feed helps you keep up with recent activity on
tories you watch and people you follow.

Navigate file history faster with improved blame view

Navigate file history faster with improved blame view

Creating your project repository at Github

Follow the guidelines provided in class to create your project repository at www.github.com. The following steps will need to completed:

- Create your github account
- Install Github Desktop App in your computer
- Create your project folder, java package, and java class files under your Eclipse workspace (see ppt slide from course material)
- Using Github Desktop App (https://guides.github.com/introduction/getting-your-project-on-github/) create a local repository at the package folder of your Eclipse project
- Using Github Desktop App, publish and synchronize your local repository to a public repository at www.github.com

Implementing your course project (Java desktop application)

The workshops provided by the teacher (theory, exercises, source code) gives the needed resources to support you in the project implementation. You should use the guidelines provided in the workshops' power-point presentations to design and implement your project.

To implement your project you will need:

- An IDE installed in your computer (ex: Eclipse)
- A database server installed in your computer (ex: MySQL server, either as a stand-alone installation or via the XAMPP installation)
- A local folder in your computer where your project files are being kept. This will be the project folder created by Eclipse under your eclipse workspace
- Github Desktop App installed in your computer
- A local github repository created at the same folder where you have your project files under the Eclipse's workspace
- A public repository under your account at www.github.com. This public repository is synchronized with your local project repository. The Github Desktop App is used to publish/synchronize your project at www.github.com

When implementing the project you should observe the requirements and evaluation criteria described in the corresponding file. You should use the evaluation criteria to evaluate your own project (self-evaluation) and evaluate the project of another student in the class (peer evaluation).

Project evaluation criteria, self-evaluation, and peer-evaluation

Projects will be evaluated based on a given evaluation criteria (separate document published by the teacher). The evaluation criteria contains a list of several evaluation targets and the corresponding amount of points the student can get from each evaluation target.

As part of the evaluation process, you should:

- Make a self-evaluation of your project
- Evaluate the project of another student (peer-evaluation)

Note that your self-evaluation and the peer-evaluation are included as targets in the evaluation criteria.

The evaluation process should progress according to the following workflow:

Teacher	You	Student evaluating your project
Publish a list containing pairs: (Student to be evaluated, Evaluator). From this list you will find out the name of the student (project) you should evaluate. This list ensures that every student will be evaluated at least by one other student.		
Activate a discussion forum "**Announce your project**" in Optima where the students can inform details about their projects. Evaluators can use this forum to contact the colleague being evaluated		
	Add an entry in the discussion forum (Announce your project) with only your family name as a subject. In this entry you should provide: - an estimated date when your project will be ready at Github - the repository name where your project will be available	
	When you have completed your project you should do your self-evaluation and upload it to the "**Project Evaluations**" return box in Optima. Name the file as: **YourLastName-Self.docx**	
	Inform your evaluator that he/she can now evaluate your project	
		Read the project's description on Github, download the code and database scripts. Create the

		database in your local environment and test the application. Fill-in the evaluation form and upload it to your “Project evaluations” return box. Name the file as **LastNameOfEvaluatedStudent-Peer.docx**
	If the student you were supposed to evaluate have not added any info on the “**Announce your project**” discussion forum, contact the teacher. The teacher will assign you another project to be evaluated	
The teacher will review the self-evaluations, peer-evaluations and perform the final evaluation for all students in the course		