(JEE), Software Architecture for Enterprises and a lot of fun

- The description in this frame title says it all.
- This addresses several aspect which we as teachers think are interesting for you.
- The way of working prepares you best for your bachelor project and beyond.
- You will have a say (because you will do some of it yourselves ) in what we will tackle as topics.
- **There will be NO written exam.**

The fine print

However, we need some rules about grading, assessment, fairness and rewarding differences in effort. These are the rules:

- All students will do time writing for all modules in this semester, including SOFA, GRAP, COM7 and this module, JAVAJEE.
- The presence during the scheduled hours is mandatory.
- ~~The use of the fontys venlo provided repository for all work is mandatory~~.
- The use of the sebivenlo repository group at github is mandatory. Make sure you have a github account with a recognisable name. Send it to me, so I can invite you as commiter to the repo.
- The teachers prescribe the technology of reports and any other documentation. Think LaTeX and/or HTML (preferably asciidoc(tor)) (for websites).
- The students will produce course material and present it as well. In many cases this will be a group effort.
- For group work, we will assess contribution through peer assessment and github history.
- You are meant to question everything (except this list) to maximise the learning of all involved. Actually the relevance of your questions will count towards your esteem.

- We want publication quality stuff, meaning *reviews, reviews, reviews*, so that your and our names will be known for the good stuff.

What went before

- First round: 10 projects, little control, useful work. I learned a lot. Very productive, useful for both the student-teachers and student-student. Few projects sprang out, like akka framework, and javafx work shop, work shop wise.
- Second round: github mandatory. Still interesting topics, some of good, some of lesser quality. Bottom line: too much freedom.
- This year: stricter rules. “Whole day” allocated for JEE, including presence (and watching) of teacher (HOM).
  - Standardized platform for “server” side: Docker images.

TOPICS Autumn 2016

At least:

- Critical regard towards all choices possible and made.
  - Considering the theoretical “physics of programming”, like Lazy vs “double work”, big-O notation etc.
- Docker as platform, mandatory for all server side work real soon.
- As platform: Wildfly, rationale: more active development.
- Testing, in particular using mockito, Java8.
- Discussing interesting tricks in (Java) programming.
- Arquillian, again.
- To ORM or not to ORM (let the database do the work, certainly if you pay a hefty premium for the licence)
  - Which we won't: Postgresql (9.5) db is mandatory for all projects.
- JSF where useful, other (even older!! technologies like JSP) where more optimal.
- Javascript on the client side. (No node-js yet), meaning rest-full services on the server.

Topics

1. Intro into nosql, e.g. neo4j and (open) cypher intro.
2. 'Modern' data types in RDBMS, e.g. hstore, json(b) in postgresql. How to easily support with database built in functions.
3. Spring boot
4. web sockets, maybe streaming video in browser?
5. Actor based programming paradigma, e.g. using the AKKA framework.
6. React Js, reactive streams, RxJava
7. Java 9: jigsaw.
8. Java 9: everything but jigsaw, e.g. the flow package (see react js item).
9. Business workflow support with BPMB, using Camunda.
10. use of OAuth2 and Json web token.
11. Lightweight communication for embedded/iot Message Queuing Telemetry Transport (MQTT), interoperability and how to use in java application. Also address authentication and security.
12. b2b authentication. Server to server authentication.
13. Continuous delivery with Jenkins 2.
14. Micro services with jhipster.
15. Rest Assured and wire-mock.

How to get selected

- If you want to have your own topic, present a 5 minute pitch, give it next week.
- For all others: choose your topic from the list.
- Next week we will form groups.

TODO week1

- Get up and running with docker on you machine. Use the Linux Container variant, not windows containers.
  - On Linux, use docker package from distro. Docker can run 'natively' on your machine.
  - For Windows and OSX this implies the use of a virtual machine. Prefer to use VirtualBox.
- See `https://github.com/sebivenlo/dockerstart`