SENG 330 - Object Oriented Design

George Tzanetakis (filling in for Dr. Ernst)

Course People


Professor: Neil Ernst

(*my humble apologies for being absent!*)

- nernst@uvic.ca and http://neilernst.net
- ECS 560
- Office hours after class T/W or by appointment

TA: Omar Elazhary

- omazhary@gmail.com
- Chisel Lab, fifth floor ECS

Preferred form of communication is via Slack, chat or direct message.

Ground rules

This course is collaborative and discussion-based. A few ground rules:

- university policies on harassment, inclusion, and academic misconduct will be strictly enforced.
- do the readings before class they are assigned to.
- I may move some discussions offline for the sake of time.
- devices are to be used for engaging with this class:
    - looking up what I say, Slack chatter, reading the notes, etc.
- I reserve the right to ask you to leave if you are not in class to learn.
- Turn off notifications and audible alerts. I will try to do the same!

Course outline

There is an official course outline on HEAT. Assessment will be

- 2 assignments, 10% each
- 1 project, 35%
- 1 midterm, 15%
- 1 final, 30% (in exam period)
- Project will be in groups formed by the instructor. Neil will do this once class roster settles down.

You must pass final and midterm to pass the class.

Course Setup

- All class notes are/will be on the class Github schedule page - https://github.com/SENG330-17/course (navigate here; schedule at bottom)
- Slack will be the tool for managing communications. It is free, widely used in industry, and pretty decent.
    - (But see the privacy statement on Connex).
    - Go to https://seng330-f17.slack.com/ to get started.
- There are two polls on our Slack #announcement channel about Git and Github knowledge. Please complete those so we know what to expect.
- Assignments and project work are managed by Github Classroom. Essentially, you follow a link and Github creates an assignment/project repository for your Github username.

Course Setup (2)

- VITAL!! Omar needs to know your Github id to link it to your Uvic info (V-number or email). Please DM us in Slack with that.

Lectures will be about a reading I assign; you must do the reading before the class for the lecture to make sense.

Readings are on the Github schedule. Readings may be academic papers, websites, blog posts, and podcasts and videos, among other things.

Project

The project is going to be 35%, and marked on:

- understanding customer requirements.
- quality of the design and related artifacts.
- quality of the eventual product and its presentation.

This is a SENG course, so naturally there is programming; it is *not* a capstone project course, so the programming should be secondary. Budget your time accordingly!

More info about the project will be posted soon.

Course Format

We will begin with an overview of what software engineering is, before getting into the details of design and analysis using OO, including some design patterns and other abstractions.

Lectures will start with a recap of the previous day, a discussion of the reading, and periodically time for group work.

Topics (not necessarily in order)

1. The software development processes
2. Requirements analysis and use cases
3. Classical and Object Oriented Design
4. Design patterns and architectural styles
5. Functional and OO metrics measuring design quality
6. Verification and testing of OO systems
7. Working in teams

Intro: OO Design

- early software (e.g., like that in CSC110, SENG265) are simple enough to understand and often only consist of 1 or 2 files
- as software grew in complexity, need to *abstract* both the structure of the implementation, and the model of the problem
    - Structure: so we can simplify the engineering problem
    - Model: real-world dynamic and complex (e.g. Time Zones)
- Object orientation is one way to handle both—nouns from the world (Customer) become objects in the code (Customer.java)

Intro: OO Design (2)

- model using concepts closer to reality (e.g. objects, data encapsulation)
- OO program = collection of 'message-passing' *objects*
- modular/imperative program = collection of tasks to perform
- functional program = collection of functions operating on data

Question

Can we name some OO, functional, and imperative languages?

Intro: OO tools

- OO is about writing OO *code* using language constructs (for us, Java)
  - inheritance, encapsulation, interface contracts, classes and instances, etc.
- OO is also about designing your software with OO principles:
  - Design Patterns, Architectural Styles, Decomposition, Specialization, SOLID
- the course/project will give you the skills to *model* complex problems and *implement* OO *designs*
  - and know when an OO approach is appropriate and useful.

Summary & Short-term Tasks

- See the schedule on https://github.com/SENG330-17/course. Complete the readings/assignment there.
- Add yourself to Slack (see the Connex page or go to https://seng330-f17.slack.com/)
- Do Assignment 0 via the Github link (on the schedule). Note: create a new Github account or use an existing one. All work is private to the instructors and you/your team.
- Neil will meet you Tuesday.