Software Engineering Project Workshop (SENG202)

Matthias Galster

Team setup

July 15, 2020

UC
UNIVERSITY OF CANTERBURY
Te Whare Wānanga o Waitaha
CHRISTCHURCH NEW ZEALAND

Teams

- See Learn

Meet your team

Questions

- Where are you from?
- What is the biggest software you have written so far?
- Have you programmed outside university?
- Have you worked in a team before, how was it?
- What is the worst thing you have heard about SENG202 (if any)?

Rules for how team will work (1)

- Discuss and set expectations
  - Grade, learning, previous experiences, strengths + weaknesses, etc.
  - May revisit expectations later in the course
- Agree on methods to communicate and share work
  - E-mail, Slack, Google Docs, Trello, etc.
- Define team timetable and schedule meetings
  - When are team members busy
  - Consider other courses, vacation, assignments, other commitments
  - At least one scheduled meeting per week (e.g., **every** Friday at 3:00pm)
    - Once confirmed email time/location to Patricia by coming Friday (5:00pm)
    - **5% penalty** for Phase 1 if submitted late to Patricia

Hint

- How to book rooms available for SENG202 and SENG302
  - Erskine 132, Erskine 243, Erskine 325, Erskine 334
- Contact CSSE administrators (admin@cosc) and provide
  - Team number and which room you would like
  - Day, start and end time of booking
  - Whether this is a one off or repeated booking for the semester
- Library also has bookable rooms

Leave rooms in tidy state, do not move computers or unplug anything or take devices with you

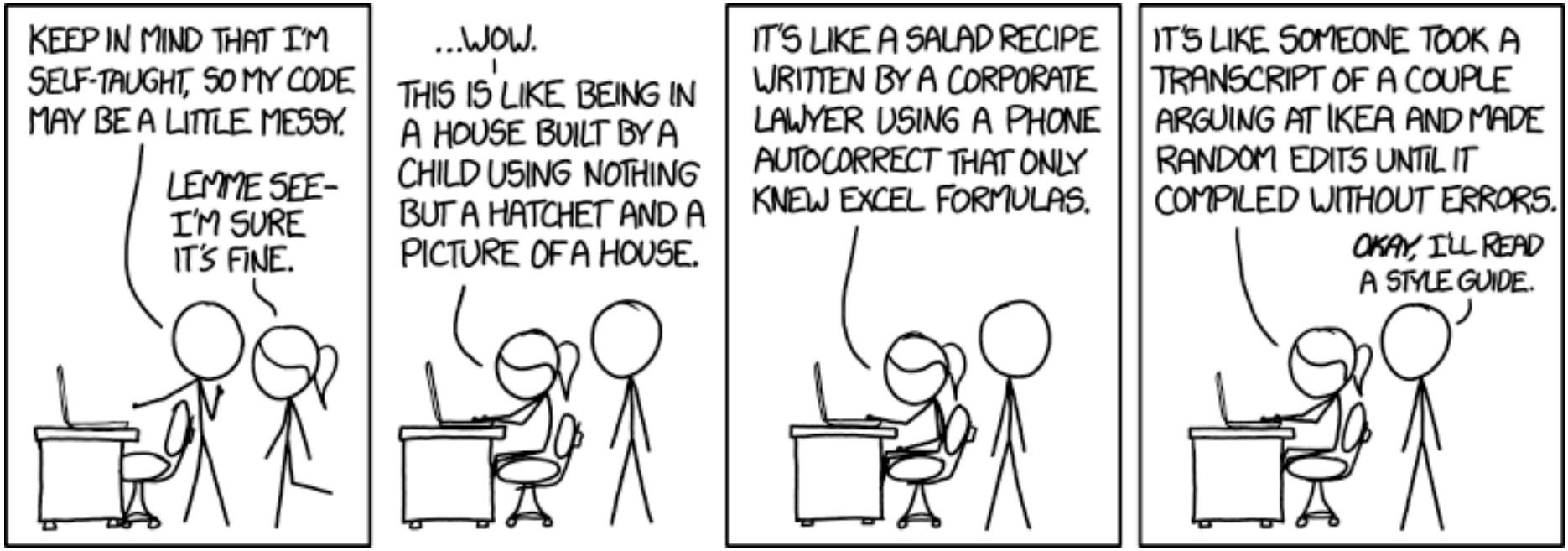
Rules for how team will work (2)

- Decide on a yellow card policy
  - Steps to take when a group member does not meet responsibilities
  - At what stage do you call for staff intervention?
  - May revisit yellow card policy later in the course
- Define roles and responsibilities (may change)
  - Who will take notes, submit deliverable, etc.
  - May revisit roles and responsibilities later in the course
- Decide on coding standards and conventions
  - Checkstyle (or alternative) is your friend

Hint

Coding conventions (e.g., for naming variables, methods, classes, indentation rules, code layout, Javadoc, etc.) should be defined. E.g., you could decide that there should be double lines in between methods, single lines between class variables / properties starting from the header, spaces around operands, spaces after commas, correct indentation at all times, four spaces instead of tabs, same line opening brace, bracers close on their own line at correct indentation, no wild card imports (e.g. import java.util.*), strings are encapsulated with "double quotes", descriptive method and variable names should be used, camel case method and variable names, no methods over 100 lines, statements may be grouped by single empty lines, comment where necessary for understanding, discrete use of java collections, all GUI classes that extend GUI classes have the same suffix as their parent. You may find this odd at first but in the long term common coding conventions significantly increase the readability of source code written by different team members (who may otherwise have used confusing or conflicting coding styles). Most organizations in industry have coding standards in place. A more detailed example of a very popular coding style definition can be found at https://google.github.io/styleguide/javaguide.html. A summary of this standard can be found at http://www.infoq.com/news/2014/02/google-java-coding-standards. Coding standards can be supported by tools, e.g., checkstyle (a plugin for Eclipse) or be loaded directly into some IDE's.

Coding conventions



Logging – Clockify setup

- Join the class workspace
  - Link sent via email
  - Identify your team project

- Set up your profile and options
  - Name : First name + surname + (userId), e.g., Jane Doe (jdo14)
    - Not just first name, etc.
  - Time Zone: (UTC+12:00) Pacific/Auckland
  - Date format: YYYY-MM-DD
  - Time format: 24-hour