Introduction to Software Development Principles and Practices (SWPP)

M1522.000100

Chung-Kil Hur

(Credit: Byung-Gon Chun)

All You Ever Wanted to Know about How to Build Large-Scale Software ☺

SWPP, CSE, SNU

Who am I?

- Prof. Chung-Kil (Gil) Hur [허충길]
  - Education: KAIST (B.S.), Univ. of Cambridge (Ph.D.)
  - Software Foundations Lab
    http://sf.snu.ac.kr
  - Research Topics
    - Software Verification
    - Low-level Language Semantics (C/C++/LLVM/Rust)
    - Relaxed-Memory Concurrency
  - Our collaborators
    - [UK] U of Cambridge, Microsoft Research Cambridge
    - [Germany] Max Planck Institute for Software Systems
    - [France] INRIA
    - [USA] Princeton, Yale, UPenn, Utah, Google.
  - Publications
    - The top-two conferences in Programming Languages:
      POPL(5), PLDI(6) (last 9 years in a row, 4th in the world)
  - (High school) Bronze medal in Intl' Math Olympiad (IMO) 1994.

Teaching Staff

- Instructor: Chung-Kil Hur
  - Email: gil.hur@sf.snu.ac.kr
  - Office: Bldg. 302, Rm. 426
  - Office hours: Anytime by appointment

- TAs
  - Dongjoo Kim
  - Yonghyun Kim
  - Sung-Hwan Lee
  - Minki Cho
  - Email at swpp@sf.snu.ac.kr

- Course Web
  https://github.com/snu-sf-class/swpp201901

Goals for Today

- What is this course about?
- How does this class operate?
- Interactive is important!
  - Ask questions!

This Course is About

- Principles + Practices
  of building large-scale software systems

- An hands-on course on large-scale software systems: project-oriented
  - This semester's theme is web services

This Course is About

- Building large software systems that actually work is hard. This course covers techniques for dealing with the complexity of software systems

- We will focus on the technology of software development principles and software engineering for the individual and small team
  - Specifications, principles of design and software architecture, testing, abstraction, modularity, design patterns, software development process, etc.

This Course is About

- The students are expected to apply the principles to systems in practice by working on semester-long group projects on web services

- You can think that each team is creating its own startup. The students applies software engineering principles to build their software products.

Class Components
[Tentative: the percentage can change]

Class participation	5%
Warm up practice (GIT, Django, React/Redux, Jest, Selenium)	20%
Milestone 1 (Basic Features)	35%
Milestone 2 (Fancy Features)	40%

Course Materials

- **There is no required textbook in this class.**

- If you want to read more about the topics covered in the class, I recommend to read the following books.
  - "Engineering Software as a Service: An Agile Approach Using Cloud Computing", by Armando Fox and David Patterson
  - “Software Engineering. A Practitioner’s Approach (6th ed.) ”, by Roger Pressman
  - "Code Complete”, by Steve McConnell
  - "Design Patterns: Elements of Reusable Object-Oriented Software”, by Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides
  - “Extreme Software Engineering. A Hands-On Approach”, by Daniel H. Steinberg, Daniel W. Palmer
  - “Structure and Interpretation of Computer Programs (SICP) (2nd ed.)”, by Harold Abelson, Gerald Jay Sussman

  …

Course Structure

- Lecture – Tue, Thu 3:30-4:45 PM
  - Project presentations
- Practice session – Tue 6:30-8:20 PM
  - Project presentations
  - Step-by-step guidance on software development principles
- Don’t miss practice sessions: lectures and practice sessions go hand in hand

Course Timeline

- ~ 10 Mar (1 week)
  - Learing Git & Github
  - Individual assignments & exam
- ~ 31 Mar (3 weeks)
  - Learning tools (Django,React/Redux,Jest,…)
  - Individual assignments
  - Team project design and ideas.
- ~ 6 May (5 weeks)
  - Milestone 1: Basic Features
- ~ 17 June (6 weeks)
  - Milestone 2: Fancy Features

Finale: Poster & Demo Session

Main Project

- ~~This semester's theme – web services~~
  - ~~Facebook~~

- Build "new" services you've dreamed about while learning software development principles and practices
  - Amazon, Ebay, Paypal, Uber, Airbnb, Facebook, Twitter, LinkedIn, …

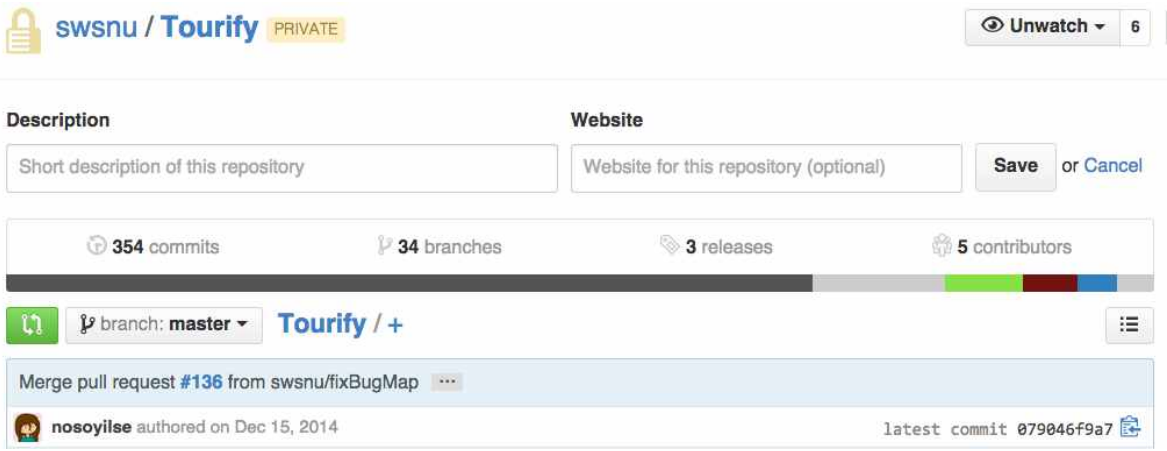
Main Project

- Group: a team of 4 students (no exception)
  - W.h.p. 2-student group(s) fail to finish
- Start forming teams this week!

- Development environment
  - Backend: Python, Django
  - Frontend: HTML5/Javascript, React with Redux

Main Project

- Agile software development process
- Git for version control
- Github for project management
  - Milestones
  - Issues
  - Pull requests
  - Code review



- Testing infra – unit tests/integration tests

Back-end

- The back-end of your project will run on a server in the Amazon EC2 or on Heroku. It is likely that it will use a database (Mongo, MySQL, etc.) and a model-view-controller architecture.

Front-end

- The front-end for your project will run in a browser. HTML5 and Javascript are your friends.

Warm-up Individual Projects

- Warmup Project1 – Frontend using Javascript, Backend using RoR

Timeliness

- Hard deadlines

- Catastrophic events
  - Major illness, death in family, …
  - Consult your academic advisor to come up with a plan to get back on track
  - Consult with me about this class

Cheating

- What is cheating?
  - Sharing code: by copying, retyping, looking at, or supplying a file
  - Coaching: helping your friend to write a programming assignment, line by line
  - Copying code from pervious course or from elsewhere in the Internet
  - Especially, be careful about copying code since we may open your project code! Be alert about code licenses.
- Penalty for cheating
  - F or D- & retaking this course is permanently disallowed

Welcome!
We will have lots of fun!