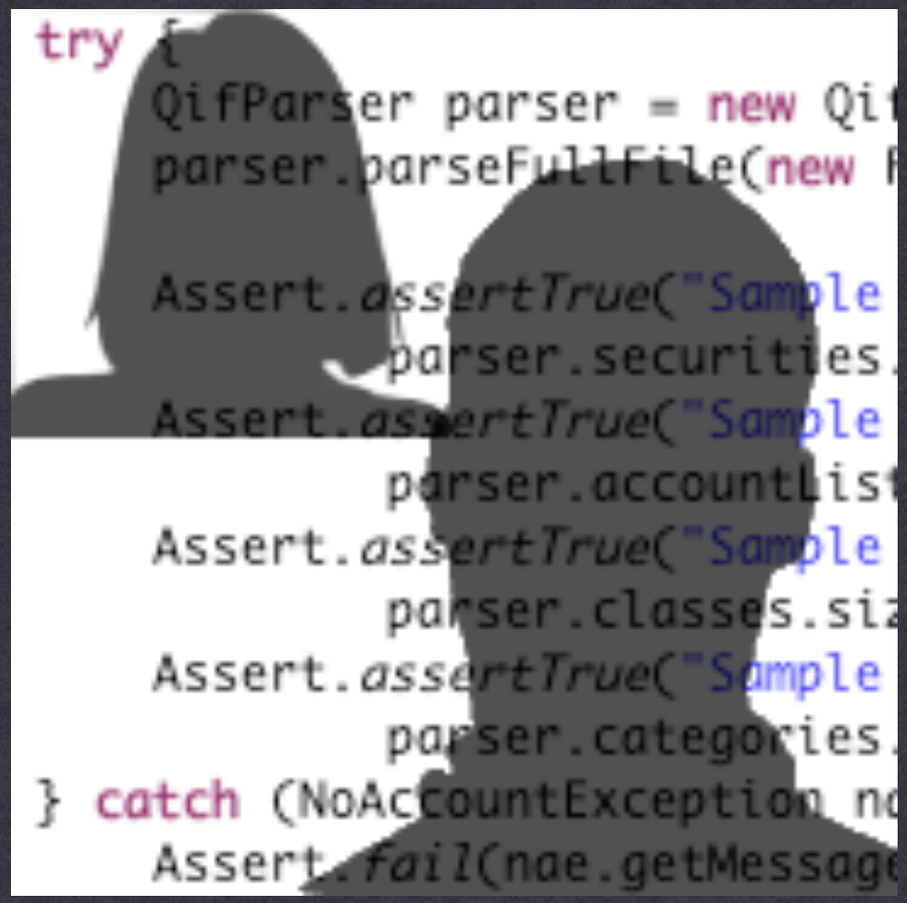

CS 846: Human Aspects of Software Engineering

Reid Holmes

What is Programming?

What is Programming?

> "The process of transforming a mental plan of desired actions for a computer into a representation that can be understood by the computer"
>
> -- Jean-Michel Hoc and Anh Nguyen-Xuan

What is Software Engineering?

What is Software Engineering?

> The establishment and application of scientific, economic, social, and practical knowledge in order to invent, design, build, maintain, research, and improve software that is reliable and works efficiently on real machines.
>
> — Wikipedia Mashup

Why Human Aspects?

Why Human Aspects?

> The vast majority of software is written by humans. Development tools and techniques should help them better understand and perform their daily tasks.
>
> — Me

Topic List

- Program comprehension
  - information needs, code navigation, working sets, code search
- Software evolution
  - refactoring, program differencing, reverse engineering
- Development tools & environments
  - team awareness, delta debugging, visualization, DOI models, task-centric development
- Quantitative and qualitative means of evaluating software engineering research
  - experiments, case studies

Topic List

- Program comprehension
  - information needs, code navigation, working sets, code search
- Software evolution
  - refactoring, program differencing, reverse engineering
- Development tools & environments
  - team awareness, delta debugging, visualization, DOI models, task-centric development
- Quantitative and qualitative means of evaluating software engineering research
  - experiments, case studies

Assessment

- Seminar: 20%
- Seminar Paper Reviews: 10%
- Project: 40%
- Project reviews: 20%
- Participation: 10%

Presentations

- ~1-2 presentations*
  - ~30 minute talk
  - ~30 minute discussion
- 1 paper summary / week
  - Submit by email 0800 day of class
- I will go first (next week)

Paper Summary

- Summary should be ~500-1000 words long.
- Submit by email; subject: “CS846 Review”
  - Why did the authors write this paper?
  - What did they do?
  - How did they validate their work?
  - What are the strengths and weaknesses of the approach and its validation?
  - What did you *learn* from this paper?

Project Reviews

- Assess projects like a program committee
  - Everyone will read and review several papers
  - Reviews organized via Easychair
    - http://easychair.org
- Program committee meeting in the last class
  - Up to you whether we ‘accept’ papers
    - ‘acceptance’ has no bearing on your grade

Project

- The project will be a major part of the course.
- You will build a development tool that addresses a problem you have experienced writing software.
- The output of your project will be a research paper (~6-10 pages).
- Groups are encouraged (up to 3 people).
- Project proposal will be due soon (Jan 28).
  - 1 page description of the problem you are trying to solve.
- 15 minute project presentations (March 25).

Software Tool

- Identify a real problem faced by developers
- Model a solution
- Implement the tool that addresses your model of the problem
- Perform a preliminary evaluation of the tool
  - Users would be great here, but given time constraints qualitative scenarios would work

To Do

- 1) Get an easychair.org account (free)
- 2) Choose 2 papers you would like to present
  - Insert into Google doc (by Jan 16 @ 0800)
- 3) Start thinking about projects

Next Week

- Two very high-level papers:
  - Fred Brooks Jr., No Silver Bullet. IEEE Computer, 1987.
  - W. Wayt Gibbs, Software's Chronic Crisis. Scientific American, 1994.
- Both available online