Introduction

CS 430 Applications Software Engineering (Development)

https://student.cs.uwaterloo.ca/~cs430

Contents

Overview

Introductions

Collin Roberts

Assisatant Professor (Instructor)

Cheriton School of Computer Science

🏠 https://cs.uwaterloo.ca/contacts/collin-roberts

✉️ cd2rober@uwaterloo.ca

Responsible for:

- Managing the course.
- Lectures, course materials.
- Setting quizzes, assessments.

TAs are still being assigned!

Their contact info will appear on the contacts page.

Course Website

The course website has everything you need, including these slides [1].

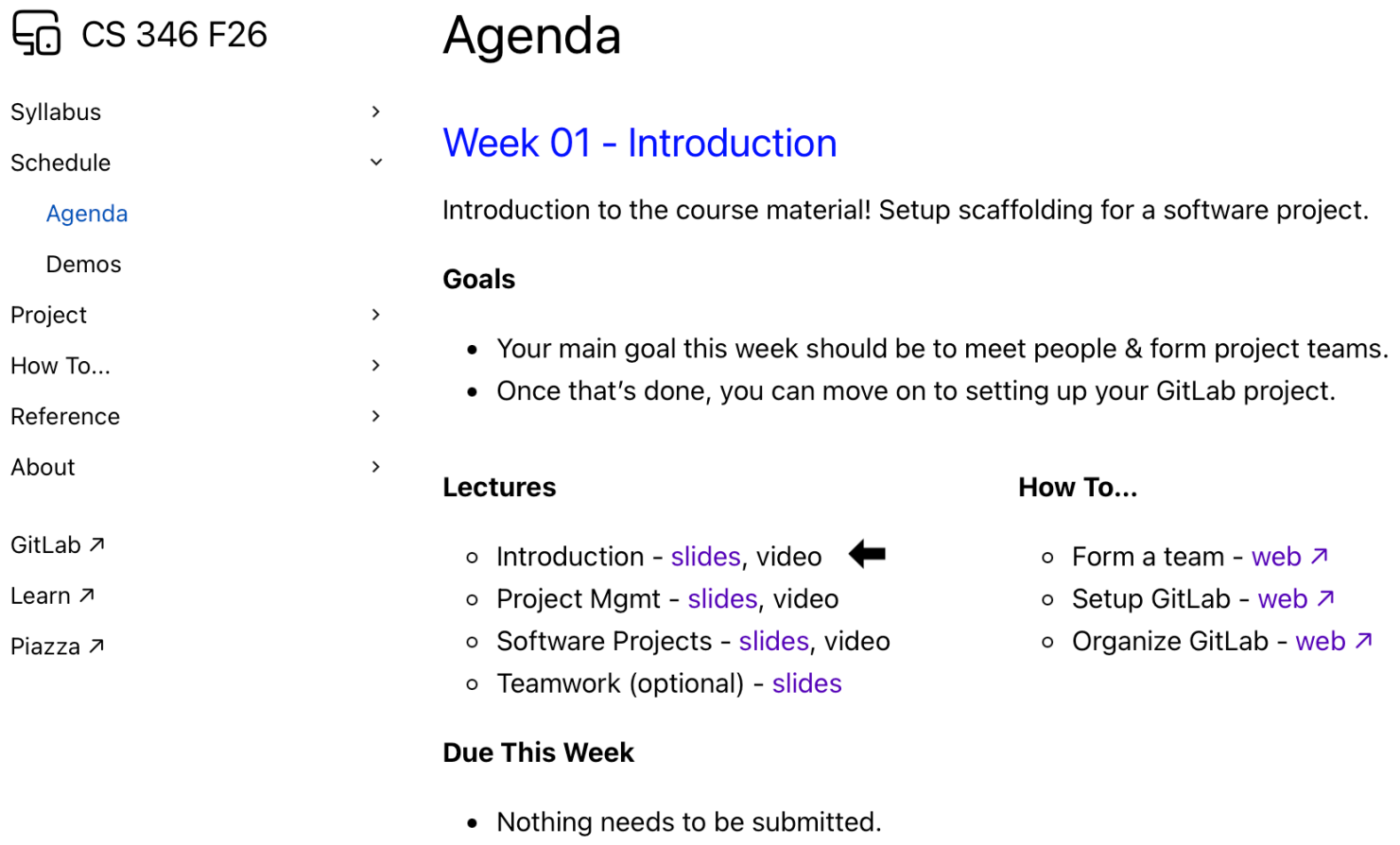

Figure 1: Course syllabus and other course details are posted online.

Description

> The application of software engineering practices to full-stack application design and development. Students will work in project teams to design and build complete, working applications using standard tools. Topics include best practices in design, development, testing, and deployment.

You will be:

- Forming project teams, planning and making decisions together.
- Interacting with your TA to define requirements and features.
- Designing, building, testing an application of your choice.
- Working together as a team to produce software releases.

Main themes:

- **Project practices**: working on a team, planning and logging your work.
- **Engineering practices**: learning best-practices in software development.
- **Programming**: building stable, effective, flexible applications.

Main Topics

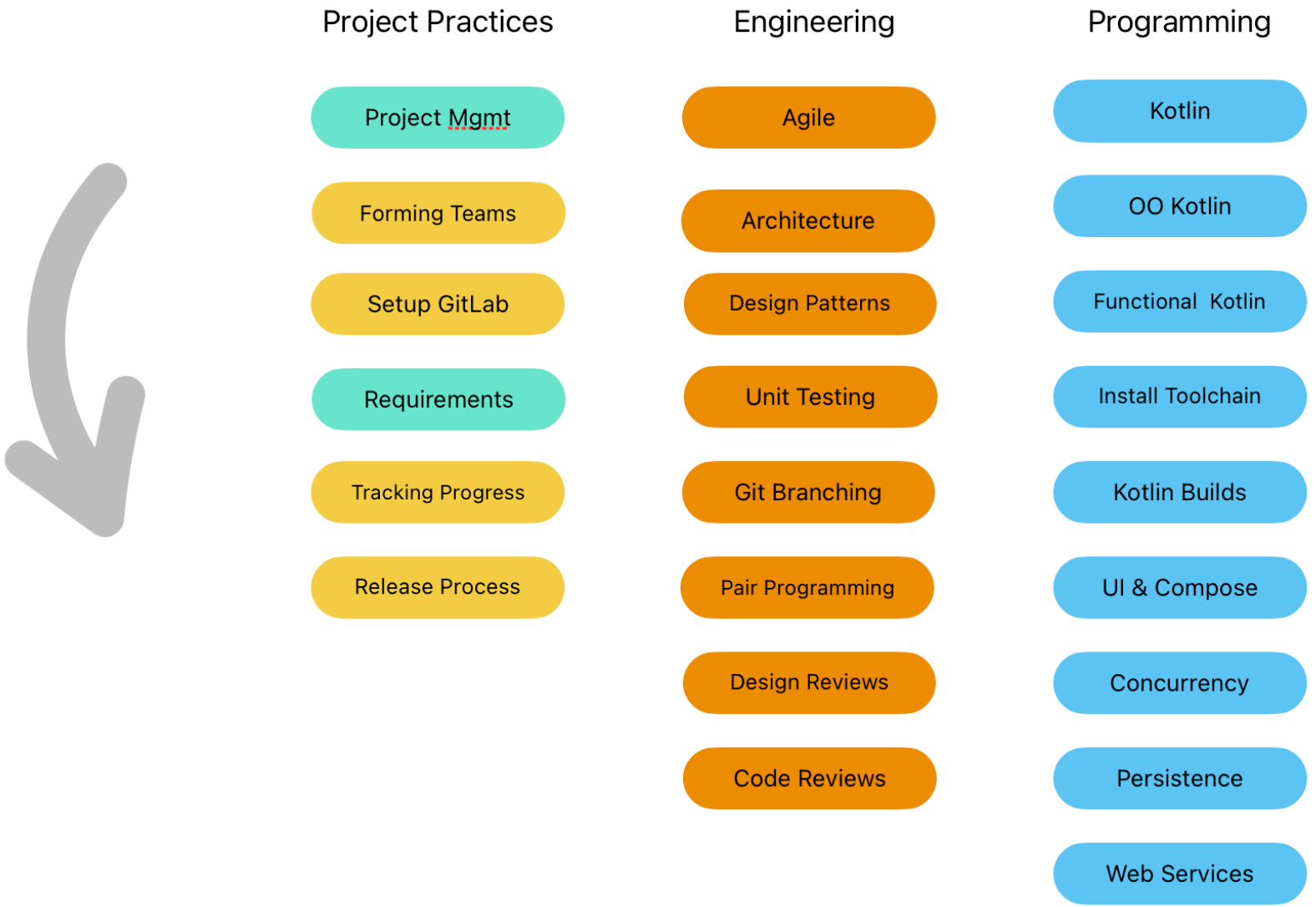

Figure 2: Although this is a CS course, SE and Project Management topics are discussed.

Course Structure

This course has lectures on Tuesdays and check-ins/demos on Thursdays.

- For this course, the goal is to make time available to work on your project, and provide opportunities for you to get help and feedback from course staff.

Activities

What will you actually be doing?

Tuesdays

- Attend the lecture.
- Complete short (15 min) weekly quizzes on that week's lectures.
- Meet to work on your project (or work "virtually" with your team).

In-class (team)

- Tuesday:
  - Lecture.
  - Any time left over is for working on your project.
- Thursday:
  - Team check-ins and demos every week (10 mins).
  - Every three weeks, you will submit a software release i.e., an in-progress version of your application.
  - The remaining time is free to work on your project.

Activities

The course schedule shows the lecture topics and themes for each week.

Assessment

How are you assessed?

Personal (20%)

Your individual grade component.

Component	What is it?	Weight
Quizzes	Weekly, based on lecture topics (10 x 2%)	20%

Team (80%)

The team's grade for group deliverables. Everyone who participates receives the same grade.

Component	What is it?	Weight
Check-ins	Informal demos & discussions (7 x 2%)	14%
Demos	Formal demo & software release (4 x 10%)	40%
Final Submission	Finalized product including documentation	26%

Application Development

Defining Applications

Application development is about designing for end-users.

- Software should be useful, and enjoyable for users to engage.
- It can be impactful and potentially used by millions of users.
- Distinctly different goals compared to system or embedded software.

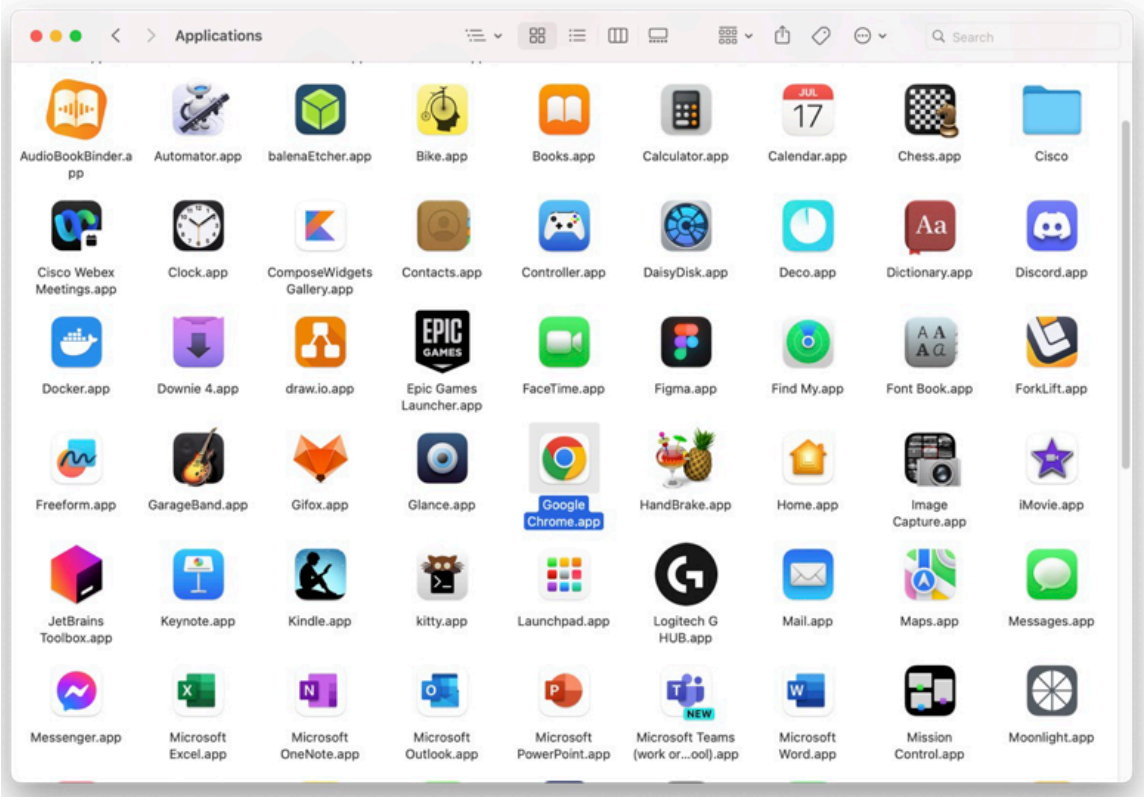

Figure 3: A typical list of applications that you might see installed on a personal computer.

Defining Applications

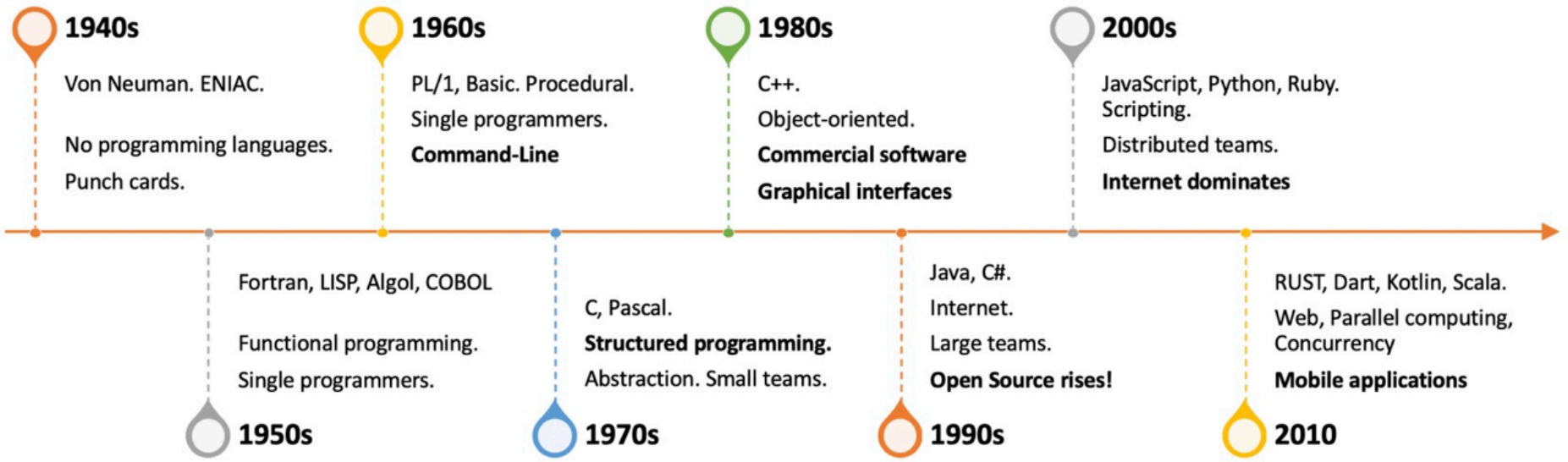

Figure 4: Modern applications can be CLI, desktop, mobile or web.

We have reinvented what “user software” looks like many times over the past 50 years. An application could be:

- a command-line application that runs in a shell.
- a desktop application that runs on macOS, Windows or Linux.
- a mobile application that runs on Android or iOS.
- a single-page application in a web browser.

Defining Applications

Example: Things

Things is an award-winning TODO application for the Apple ecosystem.

Client runs on every device/platform.

- Desktop for serious planning.
- Mobile for checking items.
- Watch for quick reminders.

“Things Cloud” hosts user data.

- Synchronizes data between devices.
- Premium subscription service.

Great example of clean, simple design that was very well-executed. They understand their users.

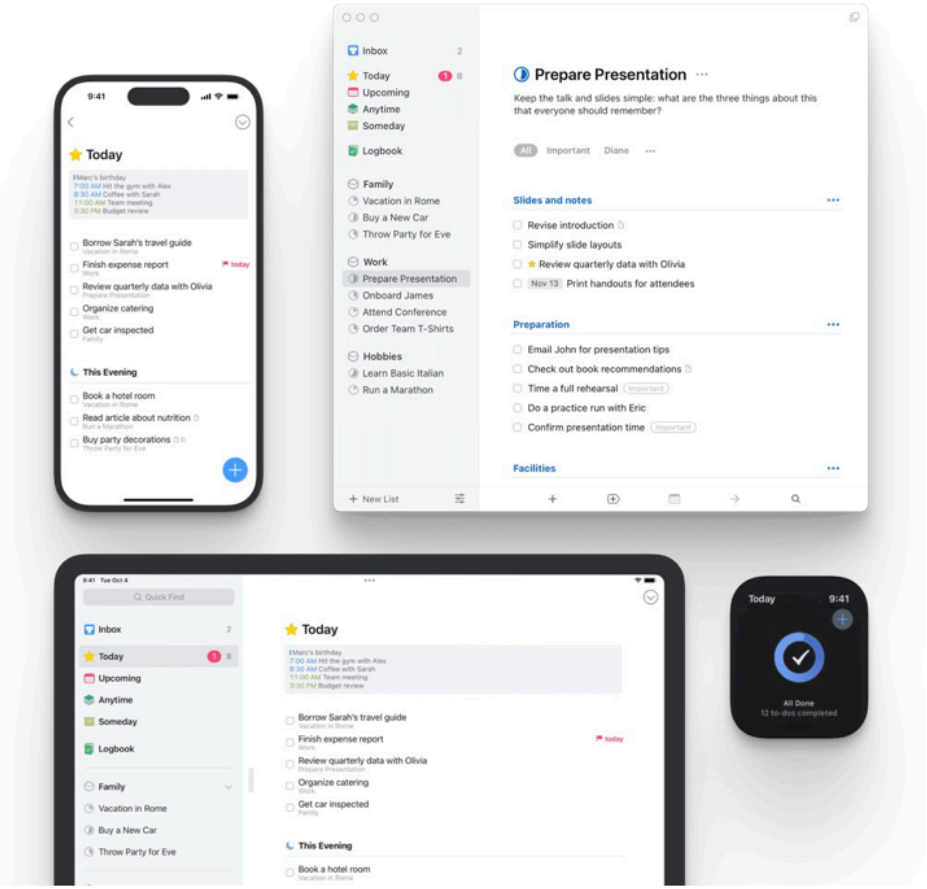

Figure 5: A cross–platform story makes an application more useful and compelling.

Modern Development

How do you build an application like this?

- A programming language alone won't do it.
- We need to leverage OS capabilities ("native" libraries).
- We also need graphics, database connectivity, networking, concurrency, security ("native" or "third-party" libraries).

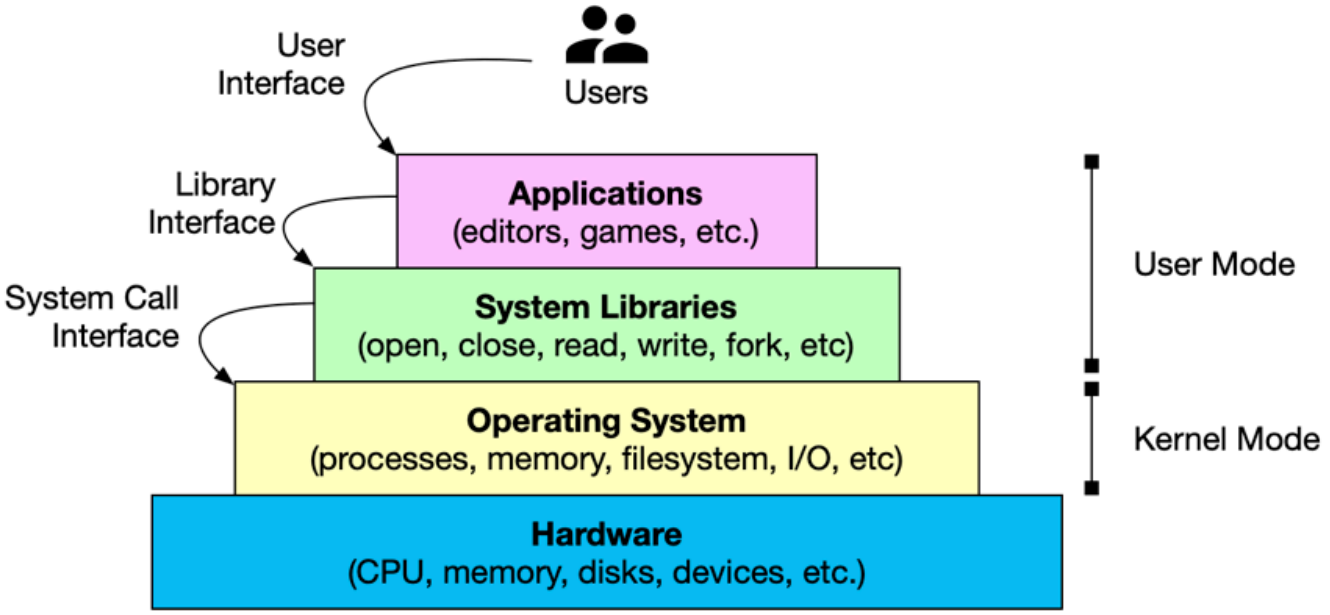

Figure 6: User libraries leverage underlying OS functionality.

Modern Development

We *also* need a suitable programming language.

Our Tech Stack is defined on the course website.

Next Steps

Weekly Agenda

The weekly agenda breaks down each week of the term.

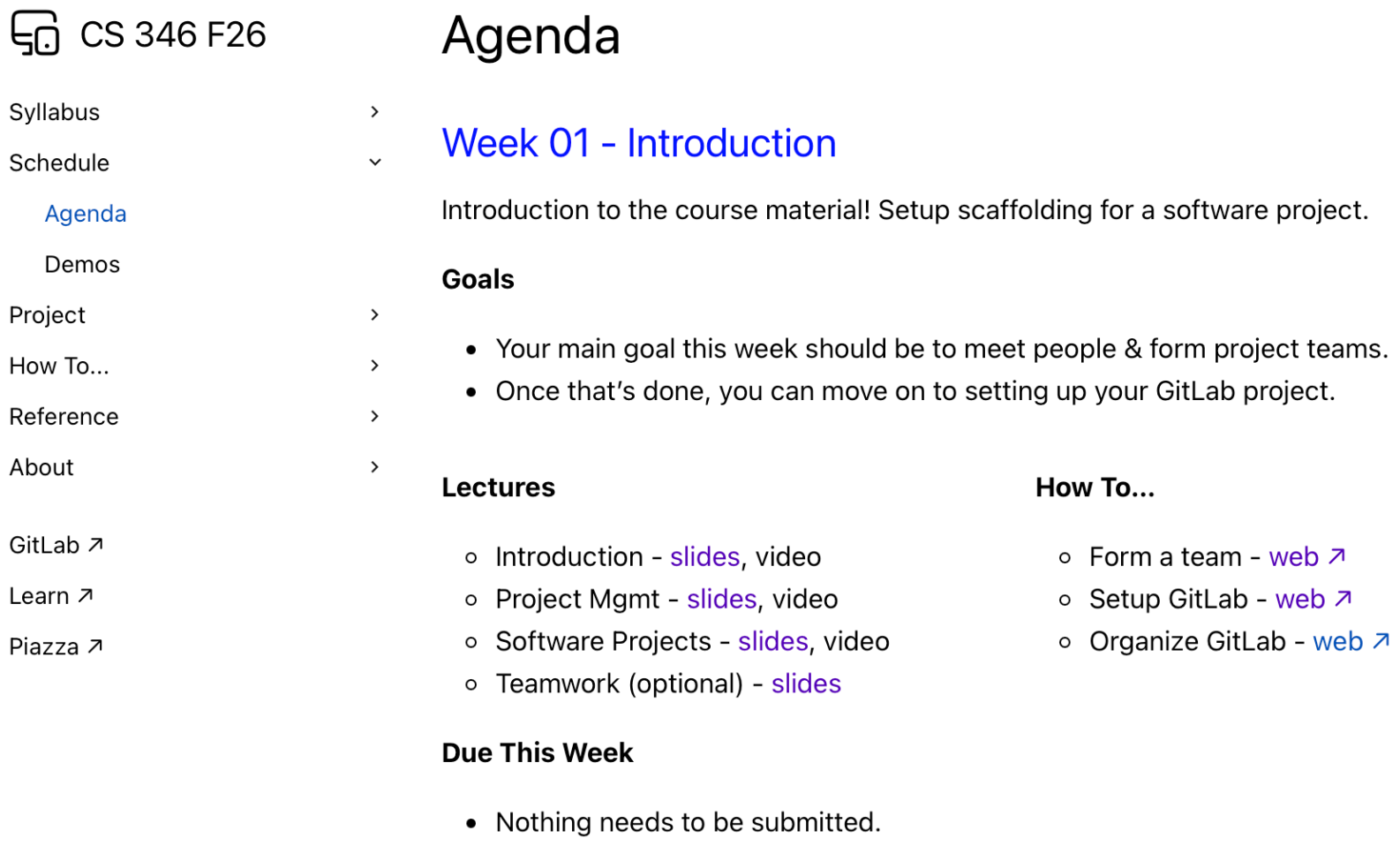

Figure 7: Each week lists goals, lectures and items that are due.

Weekly Agenda

Week 01

- Find a team! See form a team for instructions.
  - See this Piazza post where you can find potential teammates.
- Brainstorm ideas. Think about potential project ideas. See brainstorming.
- Nothing is due.

Week 02

- Setup your Git repo. See setup GitLab and organize GitLab.
- Check-in during Fri lecture. See demo times.
- Quiz 1 is due by EOD Friday.

Week 03

- Demo 1 (proposal) during Fri lecture. See demo times.
- Quiz 2 is due by EOD Friday.

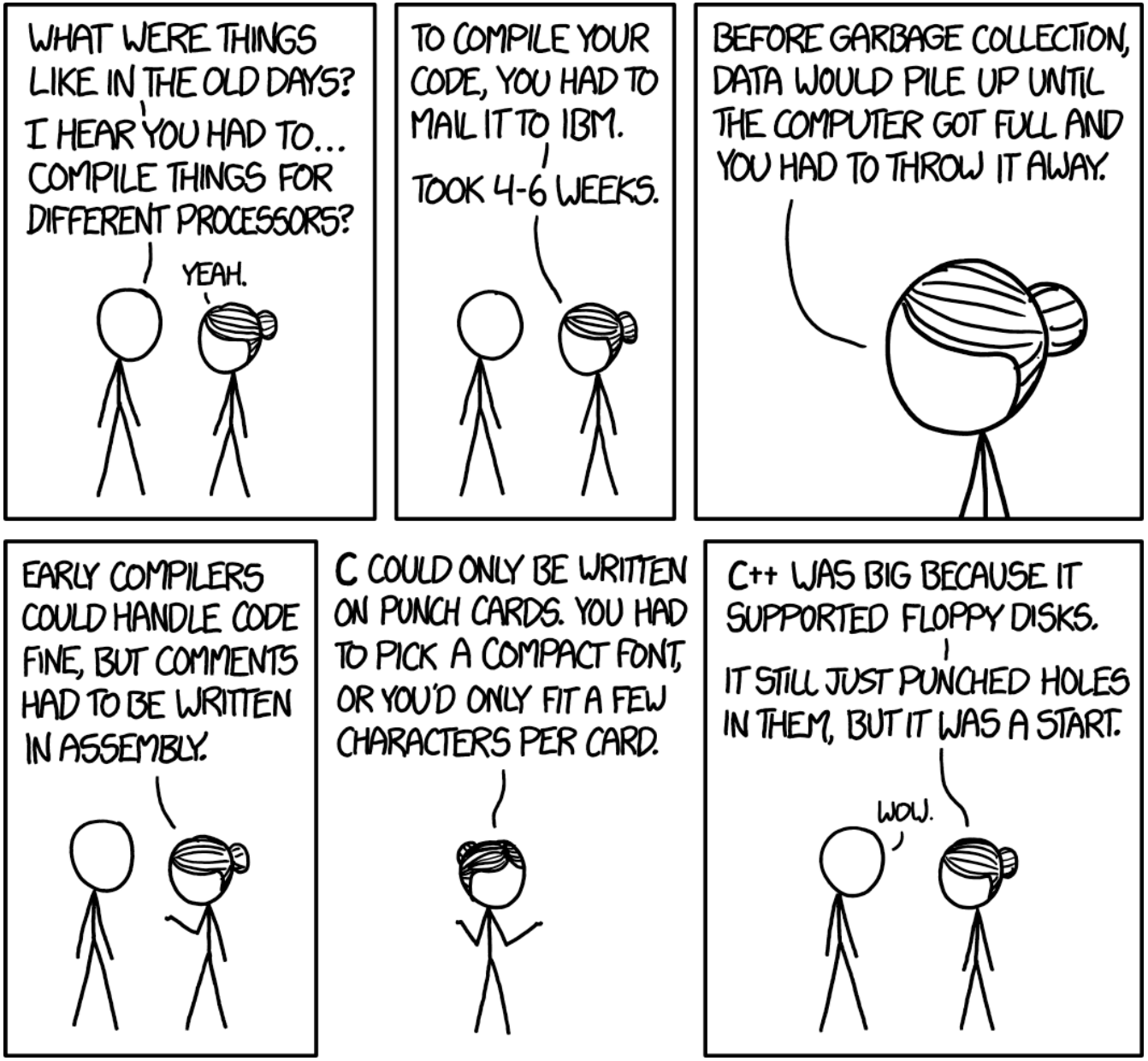

Figure 8: https://xkcd.com/1755

Bibliography

[1] J. Avery, “CS 346 Application Development.” [Online]. Available: https://student.cs.uwaterloo.ca/~cs346