Software Design and Documentation Documentation

Release Fall 2011

Alex Gaynor

Jul 18, 2017

Contents

Things stuck in paranthese are either parenthetical statements, or my own personal commentary, figure out which from context (or don't, what do I care).

- Class site
- 2010 Class site

CHAPTER 1

September 1st, 2011

First day of class.

- 2 sections combined because larger classes are better pedagogy.
- Syllabus is a contract [of adhesion].
- Documentation
  - **UML** - graphical language
  - "Vision statements" (drug induced?)
- Software Design
  - Design patterns (GoF?)
- "Agile form of the unified process" (para ingles?)
- See the syllabus for a list of worst practices (they're mislabeled as best practices, not sure why).
- Ruby on Rails is an object oriented language. (At this point your eyes should be bloodshot).
- Professors
  - Struman
    - Director of technology for a school district
    - Technical writer
    - Not a programmer
  - Kuchera
    - Manager
    - CS graduate
    - "Performance and database related items" (Oh my.)
    - Pushing "Coupling, cohesion, polymorphism"

CHAPTER 2

September 12th, 2011

Third class session, second I've attended. Second official class was missed due to DjangoCon.

Doing pitches for class projects, I gave one in the first class on a "package status" website for PyPy:

- Something for tracking fraternity rush students.
  - PHP
- Tournament management program, for gaming club
- Web app for new bands to connect with club owners and fans
- making a new graphical programming language

Now picking groups for projects.

Group selected:

Going to build a room management system for the campus. jQuery + Django, PostgreSQL, git. Need to look into whether GIS is appropriate for handling maps of floors (TODO: contact jbronn or malcolm). One of my group members insists I invented the internet, rolling with it.

Motto: Never refresh, never surrender.

CHAPTER 3

September 15th, 2011

Class number three. In the previous class each group created a shield (a crest displaying: strengths, challenges, technologies, and strategies). In this class we are (apparently) presenting them as well the status of the group meetings we held this week.

Professor is currently explaining how email is the devil and you need project websites instead. She's pitching JIRA so I think either she's a coke fiend of Jesper paid her off.

"Subversion is fine for this class, and gives you real world experience" (We're using git)

Discussion about automation and unittest. We already have one click deploys, so that's pretty awesome (thanks ep.io). No unit tests yet, but we'll get there (once I start writing view code). I started sphinx docs for this, so I might also put those on readthedocs.org (with post-commit hook) for more epic automation. We now have a post-commit hook to deploy.

We're being told to use design patterns. But not to design out application around one. I pray to god MTV counts, or Active Record. "You can go back and introduce a design pattern if it's appropriate." Apparently Active Record does count, because I just used it as an answer to question on if anyone has chosen a design pattern. She's now suggesting the "mock" pattern, might use Michael Foord's `mock` library, for some more epicness.

We [the team] need to come up with 4 metrics on which to evaluate each other.

CHAPTER 4

September 19th, 2011

Class is about iterative process. I'm not paying enough attention to say if it's bad or not, just that I've built and shipped real products so I'm not sure I care about "classroom realworld".

Professor is saying nice things about UML, this is my "I don't give a shit" face.