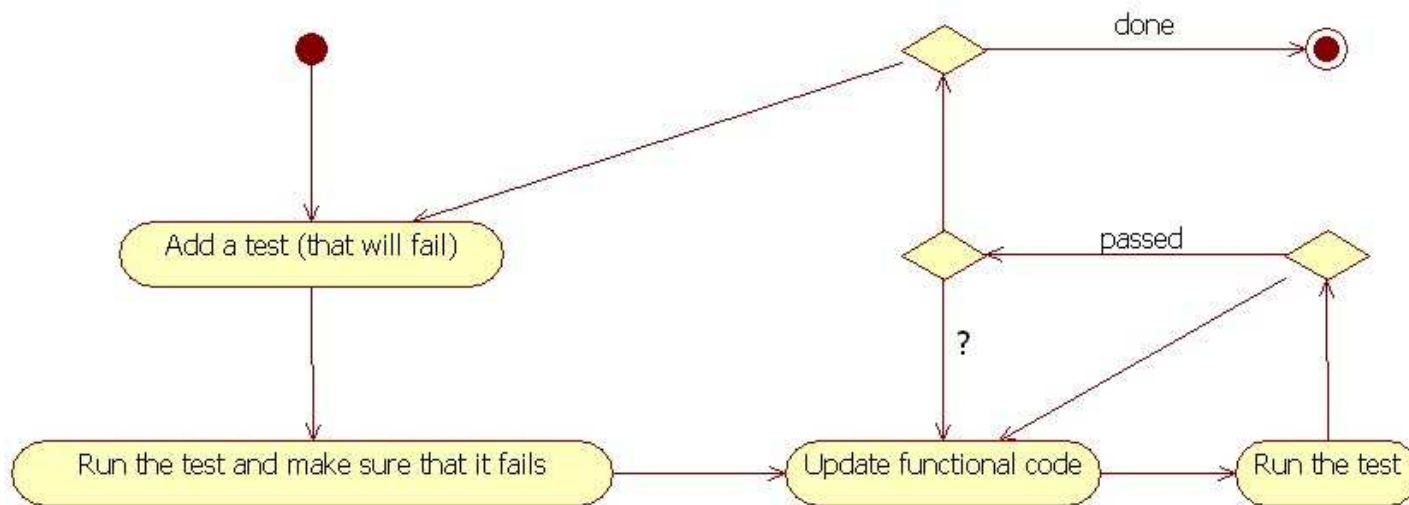


CSE70: Lecture 3

- Projects started
 - 2 days to the first milestone deadline
- Feedback form is ready
 - Try to login and to submit feedback
- For Today:
 - Read Scott Ambler's article on TDD
 - Read Chapter 6 of "The practice of Programming" by Kernighan & Pike (in the reader)
- Today
 - TDD, Testing
 - JUnit, JMock, Ant

Test Driven Design

- Test First Design (TFD)



- TD + Design or Development?
- Test Driven Design (TDD)
 - TFD + ...

TDD aspects

- TDD
 - Does pair programming support TDD?
 - Does TDD require a testing framework?
 - How does TDD affects the developers' behavior?
- TDD and Traditional Testing
 - Different testing perspective?
 - High coverage (100%?)
 - “If it's worth building, it's worth testing. If it's not worth testing, why building it?”
- TDD and Documentation
 - Executable specification
- Enables incremental development

What is testing?

- “Program testing can be used to show the presence of bugs, but never to show their absence!” (Dijkstra)
- “Testing is a determined, systematic attempt to break a program that you think is working” (Kernighan & Pike)
- Do we always need testing?

Test (when writing the code)

- Boundary condition testing
 - *off-by-one* errors
- Test conditions (pre- and post-)
 - Use assertions
 - Program defensively
 - *null pointers, out of range, division by zero, ...*
- Check error returns
- Effort vs payoff

Systematic testing

- What is systematic testing?
- Test incrementally
- Test simple parts first
 - Build a *test scaffold*
- Know what output to expect
- Verify conservation properties
- Compare independent implementations
- Measure test coverage
 - More on this on Thursday!

Scaffolding

- Unit Testing Framework (JUnit)
- Mocking Framework (JMock)
- Automation Tool (Ant)

Unit Testing

- Test for validating *units* of code
 - What are units of code?
- JUnit (Gamma & Beck)
 - Framework for unit testing
 - fixtures
 - validation
 - runners

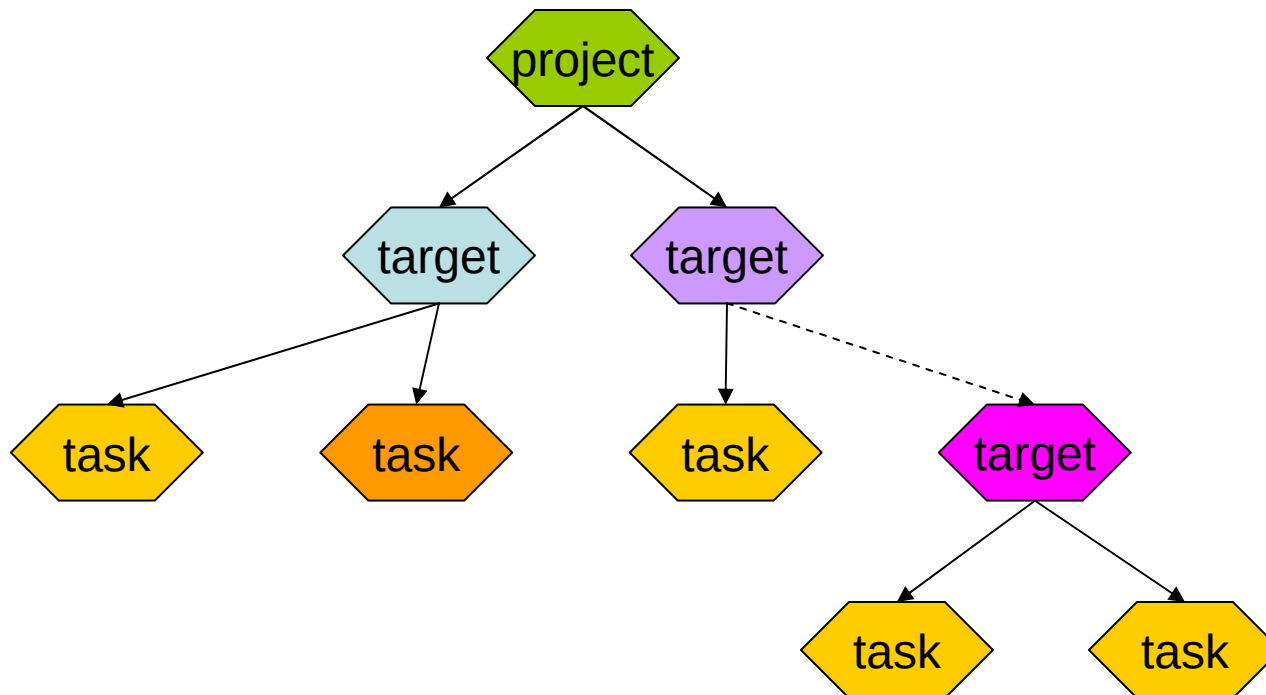
Mocking

- Mock Objects
- Why *mock objects*?
- JMock
 - Set up a *Mockery*
 - Create *mock objects*
 - Defined *expectations*
 - Execute test statements

Automation

- Support the execution of common tasks
 - Build
 - Test
- Ant
 - Build tool
 - Extensible
 - Targets and tasks
 - Defined in *build.xml*

Build file



Summary

- TDD (paper by Scott Ambler)
 - TFD + refactoring
- Testing (reader)
 - Testing while writing the code
 - Systematic Testing
- Tools
 - JUnit, JMock, Ant
- Next time...