ALLEGHENY COLLEGE

COMPUTER SCIENCE
Department

Incremental Testing

Shortening the Test Feedback Loop With Incremental Compilation

Hawk Weisman
Department of Computer Science
Allegheny College
weismanm@allegheny.edu
http://hawkweisman.me

The Implement-Test-Fix Loop

The Implement-Test-Fix Loop

1. Write code

The Implement-Test-Fix Loop

1. Write code
2. Run tests

The Implement-Test-Fix Loop

1. Write code
2. Run tests
3. Fix any bugs

The Implement-Test-Fix Loop

1. Write code
2. Run tests
3. Fix any bugs
4. Repeat ad nauseum

Problems

- Running tests often takes a long time
- Waste of programmer time
- Encourages sloppiness

A Potential Solution

- What if tests were only run when the code they test has changed?

A Potential Solution

- What if tests were only run when the code they test has changed?
- Running individual tests is much quicker than running a whole suite

A Potential Solution

- What if tests were only run when the code they test has changed?
- Running individual tests is much quicker than running a whole suite
- Rapid feedback could be possible

Incremental Compilers

- What's an incremental compiler?
  - Compiles changes to source and updates binaries
  - Eliminates re-compilation of unchanged source
  - Much faster compilation

Incremental Compilers

- What's an incremental compiler?
  - Compiles changes to source and updates binaries
  - Eliminates re-compilation of unchanged source
  - Much faster compilation
- Some examples:
  - GNU incremental gcc
  - Zinc incremental Scala compiler
  - Eclipse incremental Java compiler

Incremental Compilers

- What's an incremental compiler?
  - Compiles changes to source and updates binaries
  - Eliminates re-compilation of unchanged source
  - Much faster compilation
- Some examples:
  - GNU incremental gcc
  - Zinc incremental Scala compiler
  - Eclipse incremental Java compiler
- How does it work?
  - Compiler runs as a server
  - Watches source code for changes

Continuous Testing With Incremental Compilers

- On the first test run
  - Determine what tests target what code
  - Targeting information cached at function level

Continuous Testing With Incremental Compilers

- On the first test run
  - Determine what tests target what code
  - Targeting information cached at function level
- When code is changed
  - Get change set from compiler
  - Run only the tests that target changed functions
  - Alert programmer to any failures

Continuous Testing With Incremental Compilers

- On the first test run
  - Determine what tests target what code
  - Targeting information cached at function level
- When code is changed
  - Get change set from compiler
  - Run only the tests that target changed functions
  - Alert programmer to any failures
- After changes
  - If new functions were added, run all tests to determine what target them
  - If tests were added, run them immediately to determine targets

Potential Issues

- Works best for pure functional code
- Integration tests still need to be rerun every time
- Incremental compiler cannot apply all optimizations

Any questions?