CSSE 220

Introduction to Unit Testing

Check out UnitTesting projects from SVN

Outline

- Console Input
- Unit testing code with JUnit

Reading keyboard input from the console

CONSOLE INPUT WITH JAVA.UTIL.SCANNER

Console input with Scanner

- Creating a Scanner object
  - import java.util.Scanner;
  - Scanner inputScanner = new Scanner(System.in);
- Defines methods to read from keyboard
  - inputScanner.nextInt();
  - inputScanner.nextDouble();
  - inputScanner.nextLine();
  - inputScanner.next();
- Exercise: Look at UnitTesting/src/ConsoleWorker.java. Add missing methods to read from console

Test-driven Development,
unit testing and JUnit

WRITING CODE TO TEST YOUR CODE

Unit Testing

- Using code that you write to test other code
  - Focused on testing individual pieces of code (units) in isolation
    - Individual methods
    - Individual classes

- Why would software engineers do unit testing?

Unit Testing With JUnit

- JUnit is a unit testing *framework*
  - A *framework* is a collection of classes to be used in another program.
  - Does much of the work for us!
- JUnit was written by
  - Erich Gamma
  - Kent Beck
- Open-source software
- Now used by **millions** of Java developers

JUnit Example

- `BankAccountTester` in Big Java shows how to write tests in plain Java (pg. 103)

- Look at `JUnitMoveTester` in today's repository
  - Shows the same test in JUnit
  - Let's look at the comments and code together...

Interesting Tests

- Test "boundary conditions"
  - Intersection points: -40°C == -40°F
  - Zero values: 0°C == 32°F
  - Empty strings
- Test known values: 100°C == 212°F
  - But not too many
- Tests things that might go wrong
  - Unexpected user input: "zero" when 0 is expected
- Vary things that are "important" to the code
  - String length if method depends on it
  - String case if method manipulates that

Unit test *shout*, *whisper*, and *holleWerld* using "interesting" test cases

EXERCISE

Review for written portion of Exam 1

EXAM 1 REVIEW - WRITTEN