Multi-Currency Money in TDD

Introduction
- Start with multi-currency money object from WyCash
- Example report with instruments, shares, price, and total value
- Need to add currencies and specify exchange rates

Requirements for Multi-Currency Money
- Add amounts in different currencies and convert with exchange rates
- Multiply amount (price per share) by number (shares) to get amount

Development Approach
- To-Do List Methodology
  - Create a to-do list of tests and tasks
  - Mark current working item bold and finished items crossed off
  - Add new tests as they arise
- Start with Tests, Not Objects
  - Begin by writing tests first
  - Imagine perfect operation interface before implementation

Example: Multiplication Test
- Initial Test Code
  - Create Dollar object with amount 5
  - Multiply by 2 and assert amount equals 10
- Compilation Errors and Fixes
  - Define Dollar class
  - Add constructor Dollar(int amount)
  - Add stub times(int multiplier) method
  - Add amount field
- Running the Test
  - Initial test fails with amount 0
  - Change amount to constant 10 to pass test (temporary)

Test-Driven Development Cycle
- Add a little test
- Run all tests and fail
- Make a little change
- Run tests and succeed
- Refactor to remove duplication

Dependency and Duplication Issues
- Dependency between code and test is problematic
- Duplication is symptom of dependency
- Eliminating duplication reduces dependency and eases testing
- Duplication exists between test data and code data (e.g., values 5 and 2)

Removing Duplication
- Move multiplication logic from test to times() method
- Change times() to use stored amount and multiplier parameter
- Use *= operator to simplify code

Summary of Achievements
- Listed needed tests
- Wrote test code as a story without focusing on framework details
- Made test compile with stubs
- Made test pass with initial naive code
- Generalized code by replacing constants with variables
- Prepared for next tasks such as fixing side effects and rounding