TDD by Example - Chapter 13: Make It

Problem of Duplication

- Data Duplication in Tests
  - $10 in fake implementation duplicates $5 + $5 in test
  - Need to remove duplication to mark test done

Working Forward Instead of Backward

- Approach
  - Previous fake implementations replaced constants with variables
  - This time, unclear how to work backward
  - Decide to work forward despite speculation

Expression and Sum Classes

- Money.plus() Method
  - Should return Expression, specifically a Sum, not Money
  - Sum represents sum of two Money objects
- Test for plus() Returning Sum
  - Verify Money.plus() returns Sum object
  - Sum contains augend and addend fields
- Sum Class Implementation
  - Fields: Money augend, Money addend
  - Constructor initializes fields
  - Implements Expression interface

Bank.reduce() Method

- Handling Sum Expressions
  - If Sum currencies are same and target currency matches
  - Return Money with amount as sum of augend and addend amounts
- Test for Reducing Sum
  - Sum of $3 and $4 reduces to $7 USD
- Implementation Details
  - Initial implementation uses casting and public fields
  - Refactor to move reduce logic into Sum class
  - Sum.reduce(String) returns Money with summed amount

Reducing Money Expressions

- Test for Reducing Money
  - Reducing Money.dollar(1) to USD returns Money.dollar(1)
- Initial Implementation
  - Bank.reduce() checks if source is Money with instanceof
  - Returns source cast to Money if so
  - Otherwise reduces Sum
- Polymorphism Introduction
  - Add reduce(String) method to Expression interface
  - Money implements reduce(String) returning itself
  - Sum implements reduce(String) summing amounts
  - Bank.reduce() calls source.reduce(to) polymorphically
  - Eliminates explicit type checks and casts

Currency Conversion (Preview)

- Next Steps
  - Implement currency exchange in Bank.reduce()
  - Plans to handle rates and conversion between currencies

Summary of Development Process

- Key Points
  - Did not mark test done until duplication eliminated
  - Worked forward to realize implementation
  - Wrote test to force Sum object creation
  - Implemented constructor quickly
  - Moved code to appropriate classes once tests passed
  - Introduced polymorphism to remove class checking