Java & OOP

7. Principles driven Project

What is there?

- Small retail project
  - Product & Catalog - already done
  - A shopping basket
  - Discounts etc.

What is there?

- Principles
  - Command Query Separation
  - Tell Dont Ask
  - Combined method

Project Setup

- New class to represent a Line Item in the bill
  - `SaleItem` with `product` and `quantity` as properties
  - A method to calculate the line price
- New Class to represent a basket
  - `ShoppingBasket` which contains a list of `SaleItem` objects
  - A method to add an item with `product code` and `quantity` as params
  - Updates and provides the running total sale amount

Let us code

1. Command Query Separation

A method shuld be either command or query

- Command - tell the object to do something
- Query - ask object for the state

1. Command Query Separation

- Command
  - Affects properties as a result of command execution
  - Does not return anything `void`
  - All setters / mutators are commands
- Query
  - returns the value of a property
  - Does not change/mutate any properties
  - All getters/accessors are queries

1. Command Query Separation

- `addItem` method of the `ShoppingBasket` is a command as well as query
- Let us fix it and before that...

Some more features

- Add a discount logic 5 or more items 5% and 10 or more items 10%
- With the CQS pattern's hangover lets do it

2. Tell Dont Ask

You tell the object what to do, dont ask for information and decide for yourself

- Just too many getter/setter pairs make the class anaemic
- Push all the logic into `ShoppingBasket` and that is where they belong!

3. Combined method

- Think of all the calls in the sequence
- Want to write a book on how to use the class?
- Imagin riding a manual transmission vs automatic transmission vehicle
- Combine the methods!

Point to ponder

- Why no getter/setter for `lineItems` in `ShoppingBasket`?
- Exposing a object reference property can lead to mutation
- make `LineItem` construction safe

Next

Stateful Vs Stateless