Driving Design with TFD

David Starr
http://www.pluralsight.com/

Benefits to Design

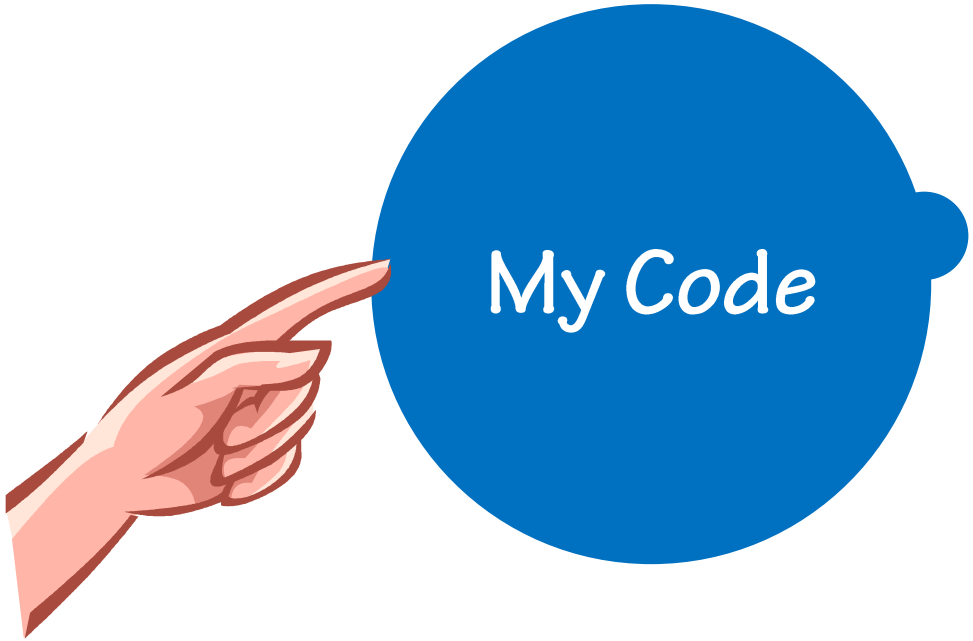
Singletons

Coupling

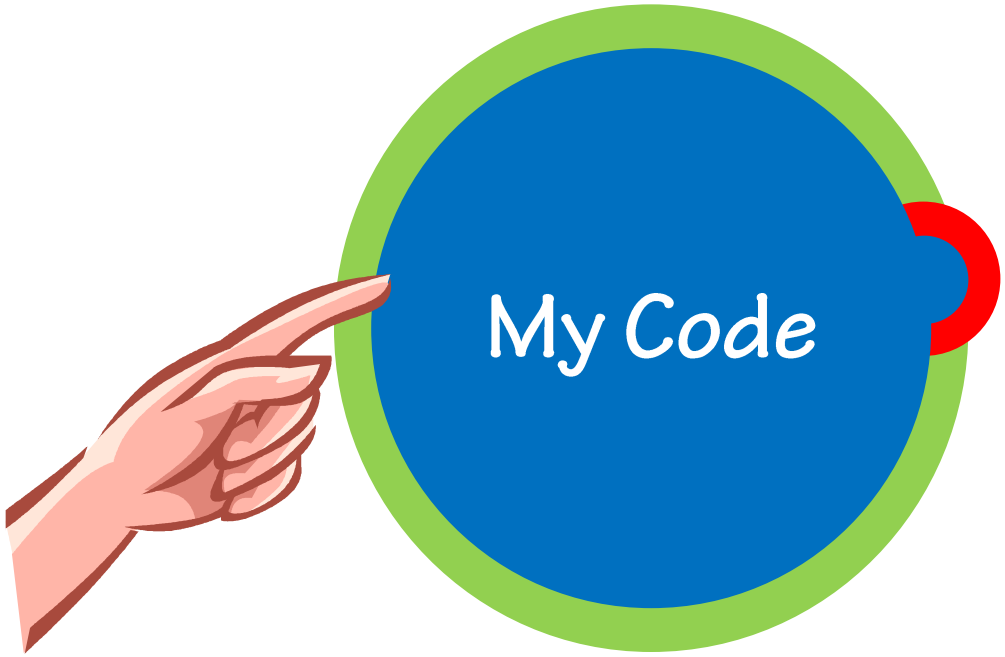
Concern Separation

Dependency Inversion

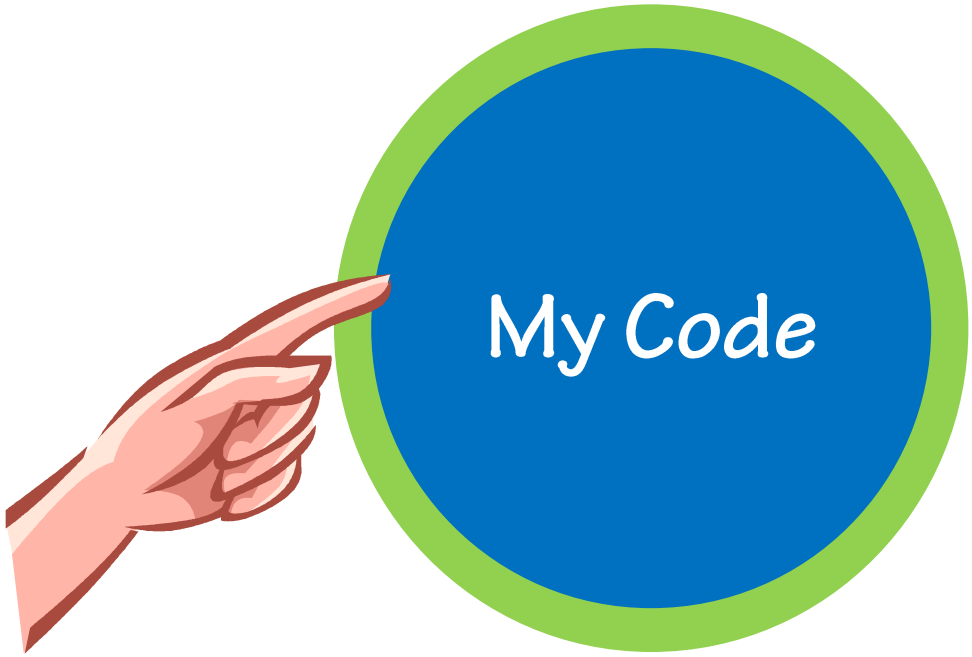
Change Design Without Fear



Change Design Without Fear



Change Design Without Fear



Writing tests first means writing testable code

Writing testable code happens to result in well-designed code

Singletons

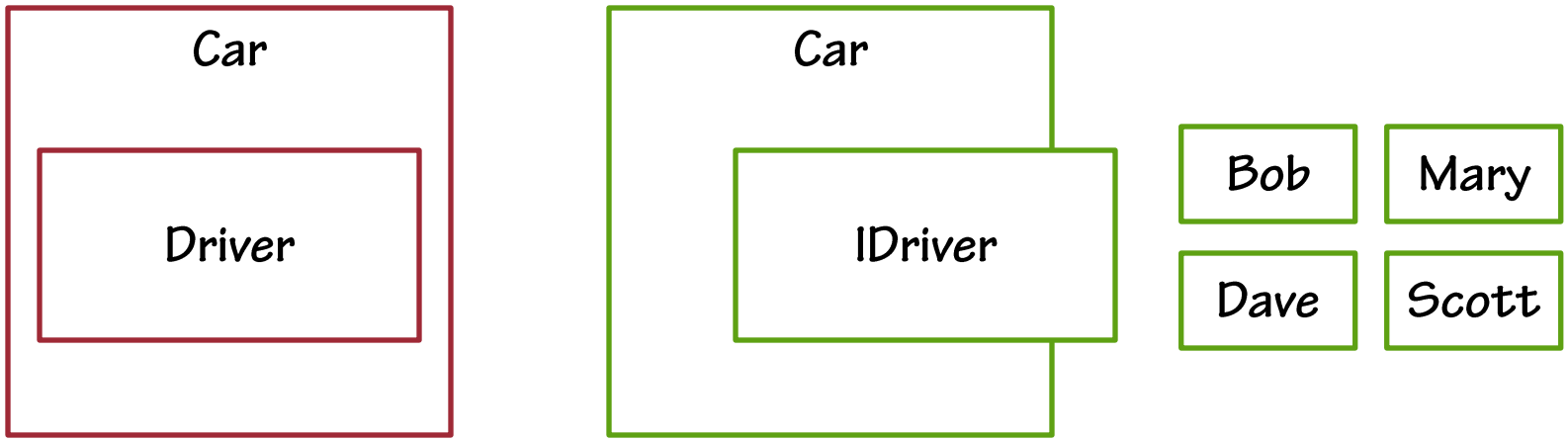
- **Often provide global access to resources**
  - Hiding implementation “details”
  - Actually hiding other dependencies
- **The singleton pattern itself violates Single Responsibility Principle**
  - The functionality of the class
  - The creation and management of the singleton instance
- **Promote tight coupling**
  - Unable to supply alternative implementations
- **State is a problem for the singleton as long as the program is running**

Coupling

- Given two lines of code, A and B, they are coupled when B must change behavior only because A changed.
- Limits ability for design to change

Dependency Inversion

- *High-level modules should not depend on low-level modules. Both should depend on abstractions.*
- *Abstractions should not depend upon details. Details should depend upon abstractions.*



Separation of Concerns

A classes has one and only one purpose

- Clear intent for the next programmer
- Simpler code implementation
- Fewer dependencies
- Fewer defects

Summary

- Benefits to Design
- Singletons
- Coupling
- Concern Separation
- Dependency Inversion