TDD: A SOLID Experience

Tarald Sponnich

Alcor Academy – Running

What I've learned throughout the Course

- The principles and importance of refactoring
- Refactor readability over design (Pareto)
- The existence and theory of Code Smells
- Identifying and treatment of Code Smells
- Coupling & Cohesion

- Aaaaand ->

SOLID Principles++

- Single responsibility
- Open / Close
- Liskov Substitution
- Interface Segregation
- Dependency Inversion
- Balanced Abstraction
- Least Astonishment

SOLID++

Single responsibility:

- "A class should have only one reason to change"

Open/Close:

- Plugin ability

Liskov Substitution:

- Keeping promises of contract

Interface Segregation:

- General purpose Interface < Several specific Interfaces

Dependency Inversions:

- Put a contract between objects you want to couple

SOLID++

Balanced Abstraction:

- *"all code constructs grouped by a higher-level construct should be on the same level of abstraction"*

Least Astonishment:

- "People are not only part of the system – they are the system".

Reflections

- The knowledge aquired through this course is a potential cornerstones as a developer

- I will give an effort to implement the teachings through my work and/or practices.

"You can't build a great building on a weak foundation. You must have a solid foundation if you're going to have a strong superstructure."

Gordon B. Hinckley

Questions?

Thank you!

Thanks to Alcor Academy for a great learning experience!

Contact info:
Tarald Sponnich – Bouvet AS
Email: tarald.sponnich@bouvet.no
Github: https://github.com/Tarald93
LinkedIn: https://www.linkedin.com/in/tarald-sponnich-08849114

Sources:

- Martin Fowler, Kent Beck (Refactoring: Improving the Design of Existing Code)
- Alex Yampolsky (https://uxplanet.org/the-principle-of-least-astonishment-bc3f67991510)
- Alcor Academy (Lectures)