Remembering How To Fly

Putting Theory into Practice

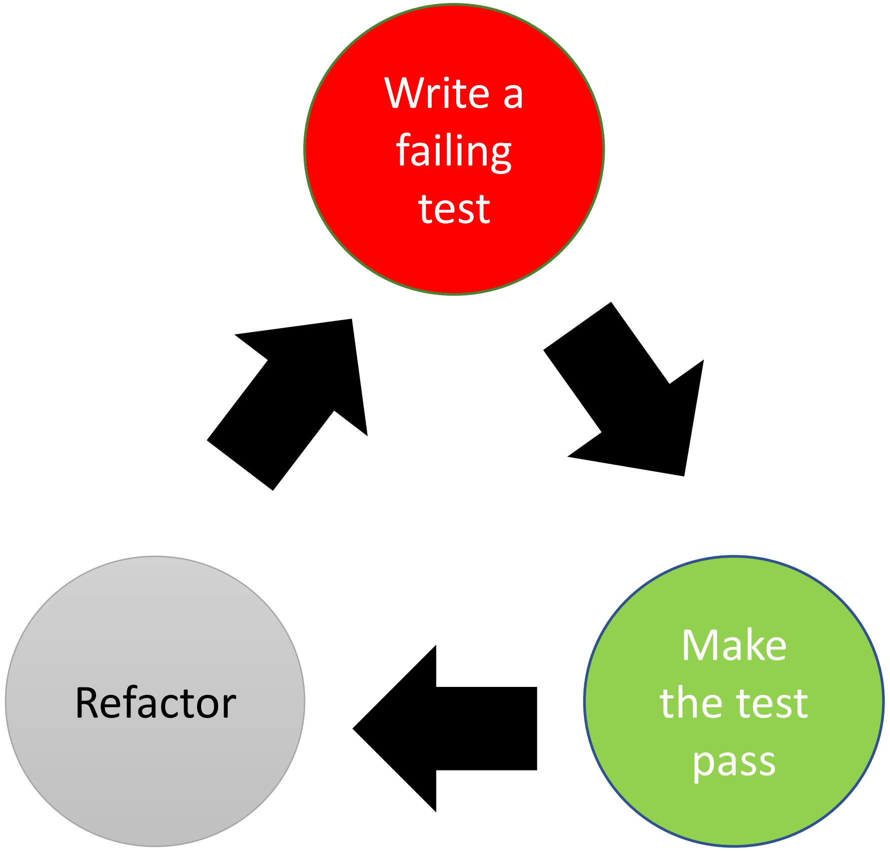
Leanne Matuszyk

Looking Back

Walking

Test Driven Development

- Cycling through:
  1. Fake implementation.
  2. Obvious implementation.
  3. Triangulation with the next test.
- Focus on one behaviour at a time.
- Small increments of functionality.



Transformation Priority Premise

- What is the obvious implementation?
- A table of code transformations, ordered by complexity.
- Small increments in complexity.

Transformation	Initial code	Transformed code
{} -> nil		`return null`
nil => constant	`return null`	`return “1”`
Constant =>	`Return “1”`	`return “1” + “2”`
…	…	…

Mob Programming

- The whole team working together.
- Roles:
  1. Driver
  2. Navigator
  3. The mob
- Rotating roles
- Increases code quality on the fly.
- Quality control (no pull-requests)
- Knowledge transfer

Object Calisthenics

- Strength in beauty.
- A set of rules:
  - One level of indentation per method.
  - Do not use ELSE.
  - Wrap all primitives and strings.
  - …
- The rules help us to write more readable and maintainable code.

Looking Back

Running

Refactoring

- 80-20 Rule
  - 80% value from **improving readability**
  - 20% value from **design change.**
- When refactoring, stay in the **green.**
- Use your IDE to help you.
- Commit as often as possible.
- Parallel change:
  1. Expand
  2. Migrate
  3. Contract

Code Smells

- Provides a common vocabulary to identify problems in the code.
- Smells:
  - Long parameter list
  - Primitive obsession
  - Inappropriate intimacy
  - …
- Smells can be identified and then fixed.

SOLID++

- **S**ingle responsibility
- **O**pen/Closed
  - Open for extension / Closed for modification
- **L**iskov substitution
- **I**nterface Segregation
- **D**ependency Inversion
- **+** Balanced Abstraction
- **+** Least Astonishment

Looking Back

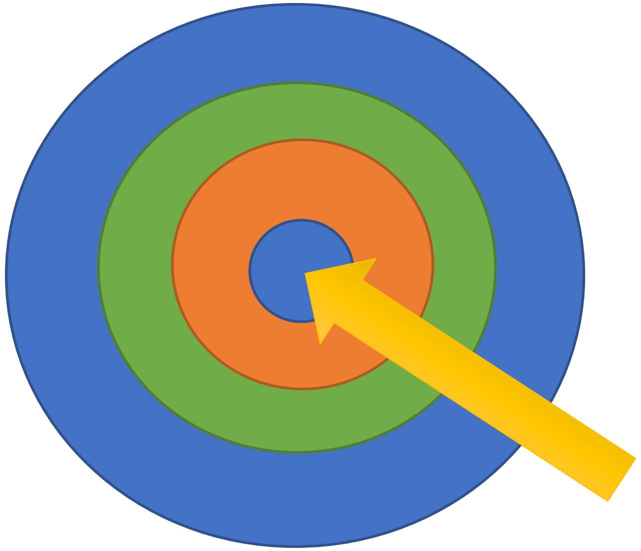
Flying

Test Doubles

- Dummy
  - Needed for a parameter list.
  - Does nothing.
- Stub
  - Responds to calls with pre-programmed output.
- Fake
  - Hand made stubs.
- Mock
  - Can verify if these objects have been used correctly.
- Spy
  - Hand made mocks.

Onion Architecture

- Outside-In
- Modular, loosely coupled architecture



Direction of dependency

Using What We Learned

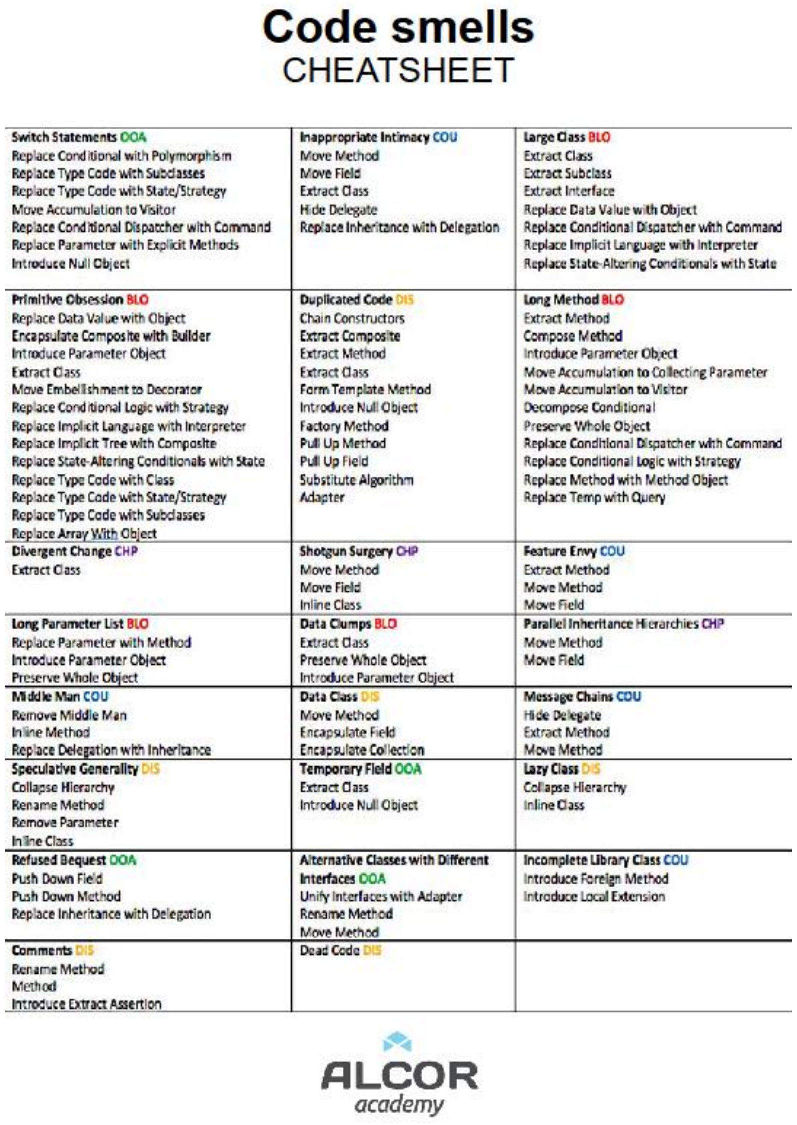
- There were so many great concepts in this course.
- How can we remember them?
- How do we put these ideas into practice?

Practice Makes Perfect

- Use TDD as much as possible.
- Include test doubles where necessary.
- Try working in a mob and keep practicing.
  - It felt a bit awkward at the start.
  - But once we got the hang of it, I felt we made real progress.
  - I compared my own solution to a Kata with that of the mob.
  - The mob won.
- IDE agility:
  - Ask coworkers to stop you if you do something manually that could be done with a short-cut.

Cheat sheets

- Print them out:
  - TPP
  - Object Calisthenics Rules
  - Code smells
- Hang them up in the office.
- Look at them periodically!



Code smells
CHEATSHEET

Switch Statements OOA Replace Conditional with Polymorphism Replace Type Code with Subclasses Replace Type Code with State/Strategy Move Accumulation to Visitor Replace Conditional Dispatcher with Command Replace Parameter with Explicit Methods Introduce Null Object	**Inappropriate Intimacy** COU Move Method Move Field Extract Class Hide Delegate Replace Inheritance with Delegation	**Large Class** BLO Extract Class Extract Subclass Extract Interface Replace Data Value with Object Replace Conditional Dispatcher with Command Replace Implicit Language with Interpreter Replace State-Altering Conditionals with State
Primitive Obsession BLO Replace Data Value with Object Encapsulate Composite with Builder Introduce Parameter Object Extract Class Move Embellishment to Decorator Replace Conditional Logic with Strategy Replace Implicit Language with Interpreter Replace Implicit Tree with Composite Replace State-Altering Conditionals with State Replace Type Code with Class Replace Type Code with State/Strategy Replace Type Code with Subclasses Replace Array With Object	**Duplicated Code** DIS Chain Constructors Extract Composite Extract Method Extract Class Form Template Method Introduce Null Object Factory Method Pull Up Method Pull Up Field Substitute Algorithm Adapter	**Long Method** BLO Extract Method Compose Method Introduce Parameter Object Move Accumulation to Collecting Parameter Move Accumulation to Visitor Decompose Conditional Preserve Whole Object Replace Conditional Dispatcher with Command Replace Conditional Logic with Strategy Replace Method with Method Object Replace Temp with Query
Divergent Change CHP Extract Class	**Shotgun Surgery** CHP Move Method Move Field Inline Class	**Feature Envy** COU Extract Method Move Method Move Field
Long Parameter List BLO Replace Parameter with Method Introduce Parameter Object Preserve Whole Object	**Data Clumps** BLO Extract Class Preserve Whole Object Introduce Parameter Object	**Parallel Inheritance Hierarchies** CHP Move Method Move Field
Middle Man COU Remove Middle Man Inline Method Replace Delegation with Inheritance	**Data Class** DIS Move Method Encapsulate Field Encapsulate Collection	**Message Chains** COU Hide Delegate Extract Method Move Method
Speculative Generality DIS Collapse Hierarchy Rename Method Remove Parameter Inline Class	**Temporary Field** OOA Extract Class Introduce Null Object	**Lazy Class** DIS Collapse Hierarchy Inline Class
Refused Bequest OOA Push Down Field Push Down Method Replace Inheritance with Delegation	**Alternative Classes with Different Interfaces** OOA Unify Interfaces with Adapter Rename Method Move Method	**Incomplete Library Class** COU Introduce Foreign Method Introduce Local Extension
Comments DIS Rename Method Method Introduce Extract Assertion	**Dead Code** DIS	

ALCOR
academy

Tip of the Week

- In our team, we have a weekly knowledge exchange meeting.
- I will introduce a quick segment:
  - Code smell of the week
  - Object calisthenics rule of the week
  - …

Physical Reminders

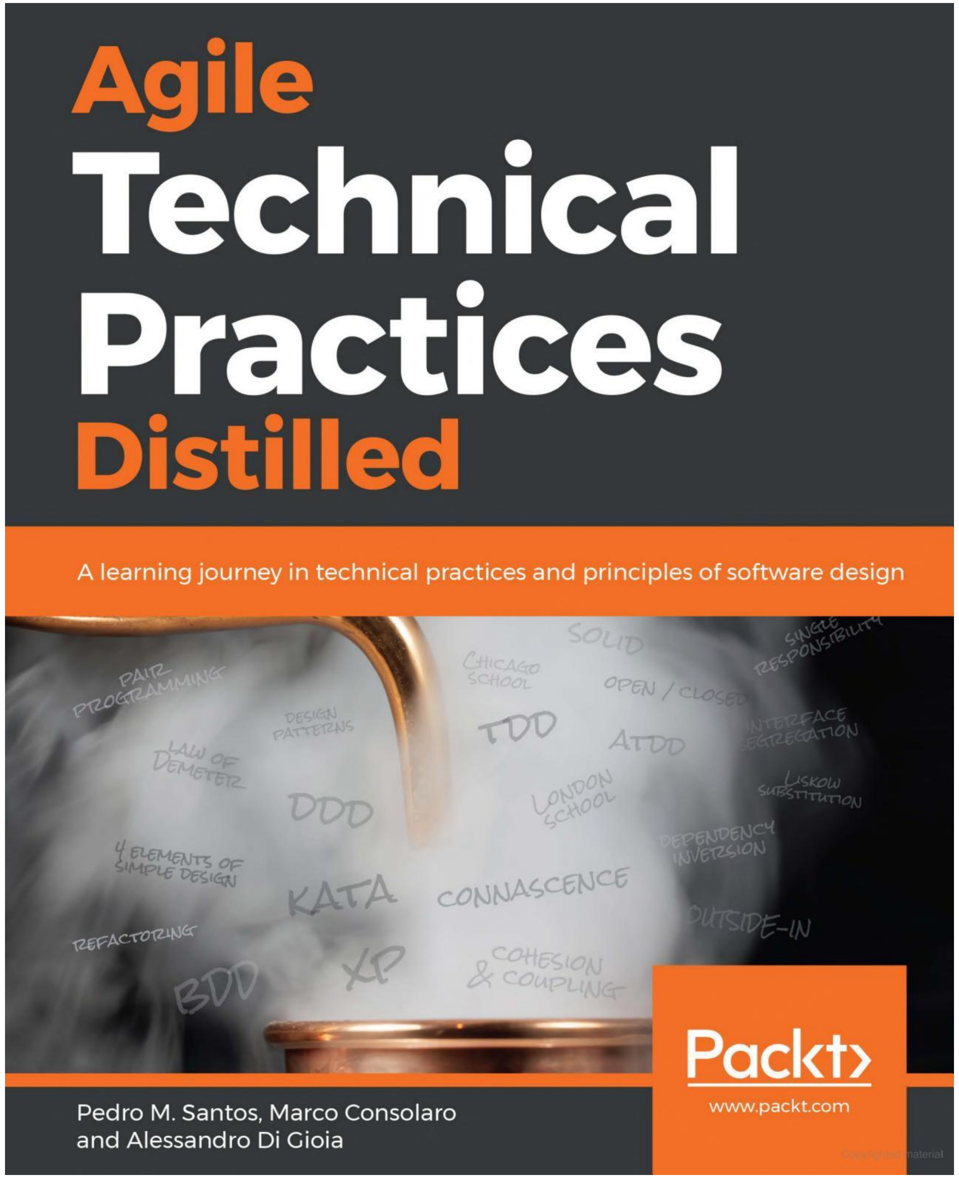
- Draw a dot or write a word on the back of your hand.
- Put a piece of string around your wrist.
- Use it as a reminder to concentrate on a particular concept.
- For example:
  - Writing short methods.
  - Wrapping primitives and strings.
  - Sticking to the single responsibility principle.

Acronyms

- Helps to remember concepts.
- KISS (keep is simple, stupid)
- DRY (don’t repeat yourself)
- YAGNI (ya ain’t gonna need it)
- …

Review

- Re-read the ALCOR academy course slides.
- Read this book.
- Read other books.



Conclusion

- Practice
  - TDD
  - Pair/Mob programming
  - IDE agility
  - Test doubles
- Revision
  - Review course slides
  - Read books

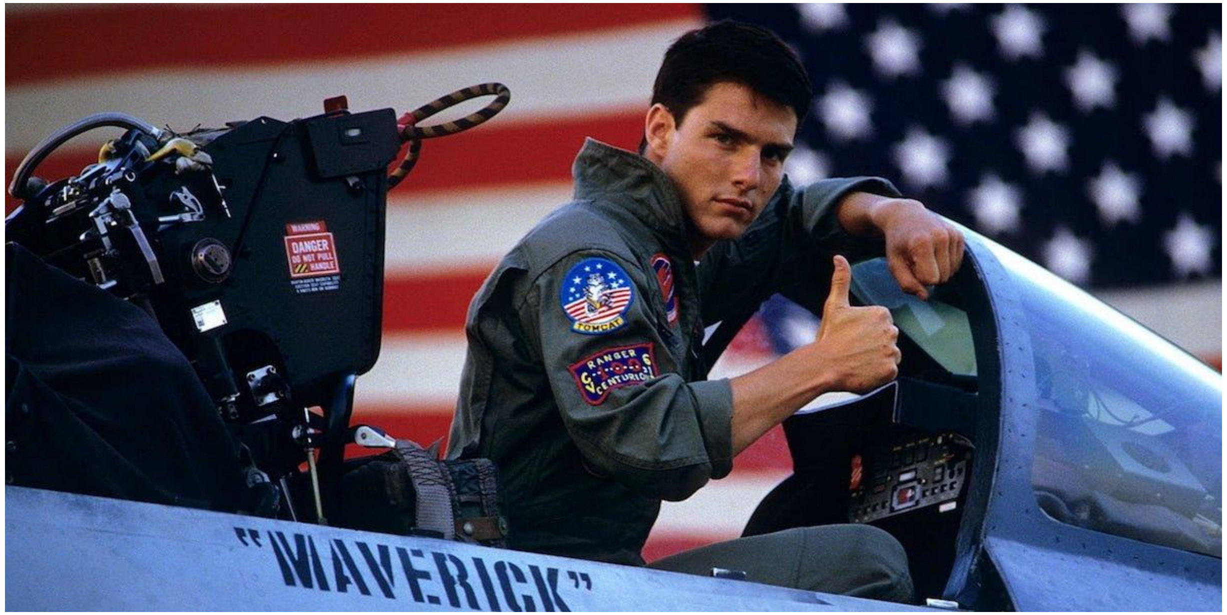
- Reminders
  - Cheat Sheets
  - Tip of the Week
  - Physical reminders
  - Acronyms

Questions or suggestions?

Thank you!

Leanne Matuszyk

leanne.matuszyk@css.ch



References:

- https://arstechnica.com/gaming/2022/05/top-gun-maverick-spoiler-free-review-a-worthy-return-to-the-danger-zone/
- https://nerdist.com/article/familys-sidewalk-monty-pythons-ministry-of-silly-walks/
- https://variety.com/2015/tv/reviews/making-of-the-mob-new-york-review-amc-1201515956/
- https://www.dpreview.com/news/4355334799/two-rio-photographers-captured-almost-exactly-the-same-usain-bolt-photo-and-both-went-viral
- https://www.europosters.ch/posters/hulk-smash-v26164
- https://www.pinterest.ch/pin/598141198957O425/
- https://www.thetimes.co.uk/article/how-true-to-life-is-top-gun-maverick-nwn2rgj3x
- https://co.pinterest.com/pin/583919907907615928/
- https://m.facebook.com/Shrek/photos/more-layers-more-to-love-whats-your-favorite-shrek-quote/10160128402130183/
- https://cardplayerlifestyle.com/3-steps-to-avoid-poker-study-overwhelm/
- https://angelesacademyofmusic.com/perfect-practice-makes-perfect/
- https://www.etsy.com/listing/645629806/adjustable-men-woman-red-string-bracelet
- https://www.military.com/off-duty/2019/07/22/79-cringeworthy-errors-top-gun.html
- https://www.cinemablend.com/news/2496202/wait-why-is-today-top-gun-day