

Reversed Tests Pyramid

Wiktor Żołnowski

@streser

<http://www.agilezkolenia.pl>

<http://codesprinters.com>



Mrs. Perfekt

Can you imagine perfect software?





End to End Tests

Functional/Integration Tests

Unit Tests

**It would be perfect to work with perfect
software every day...**

But...



**Our everyday work looks a little bit
different...**

It's called...



Legacy Code

First of all...

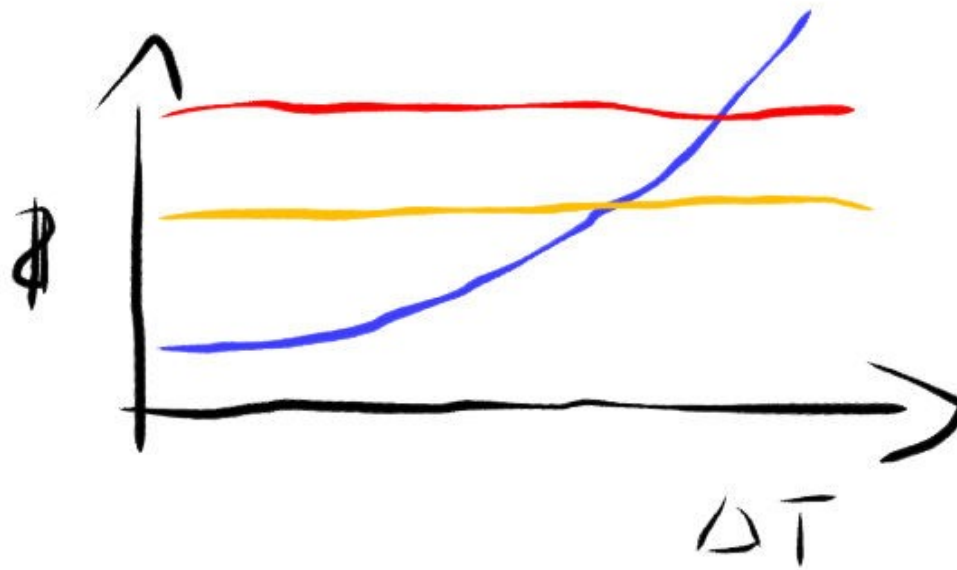
How did we get to this point?



It's all because of the Technical Debt



Is it possible to pay Technical Debt back?

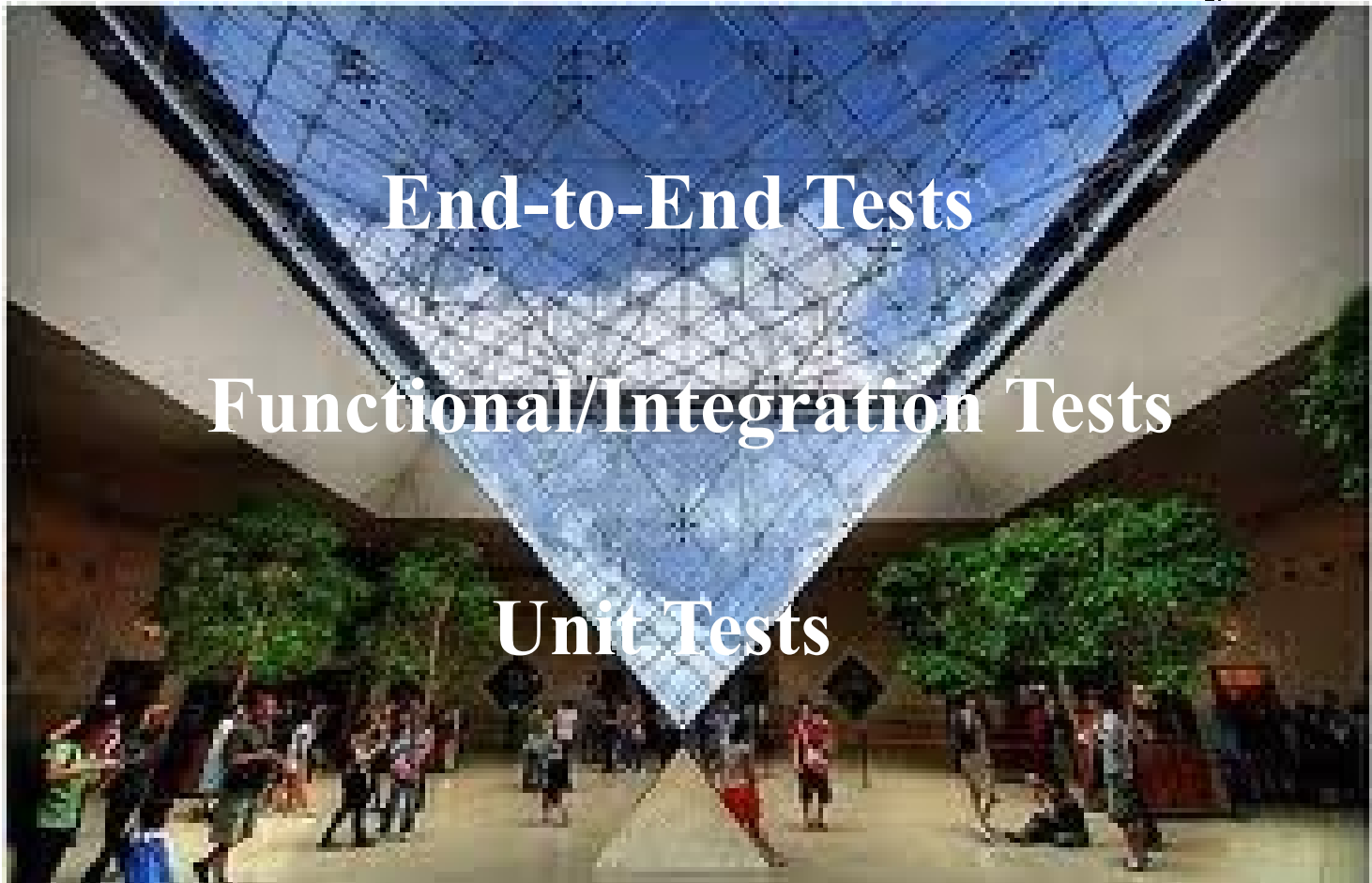


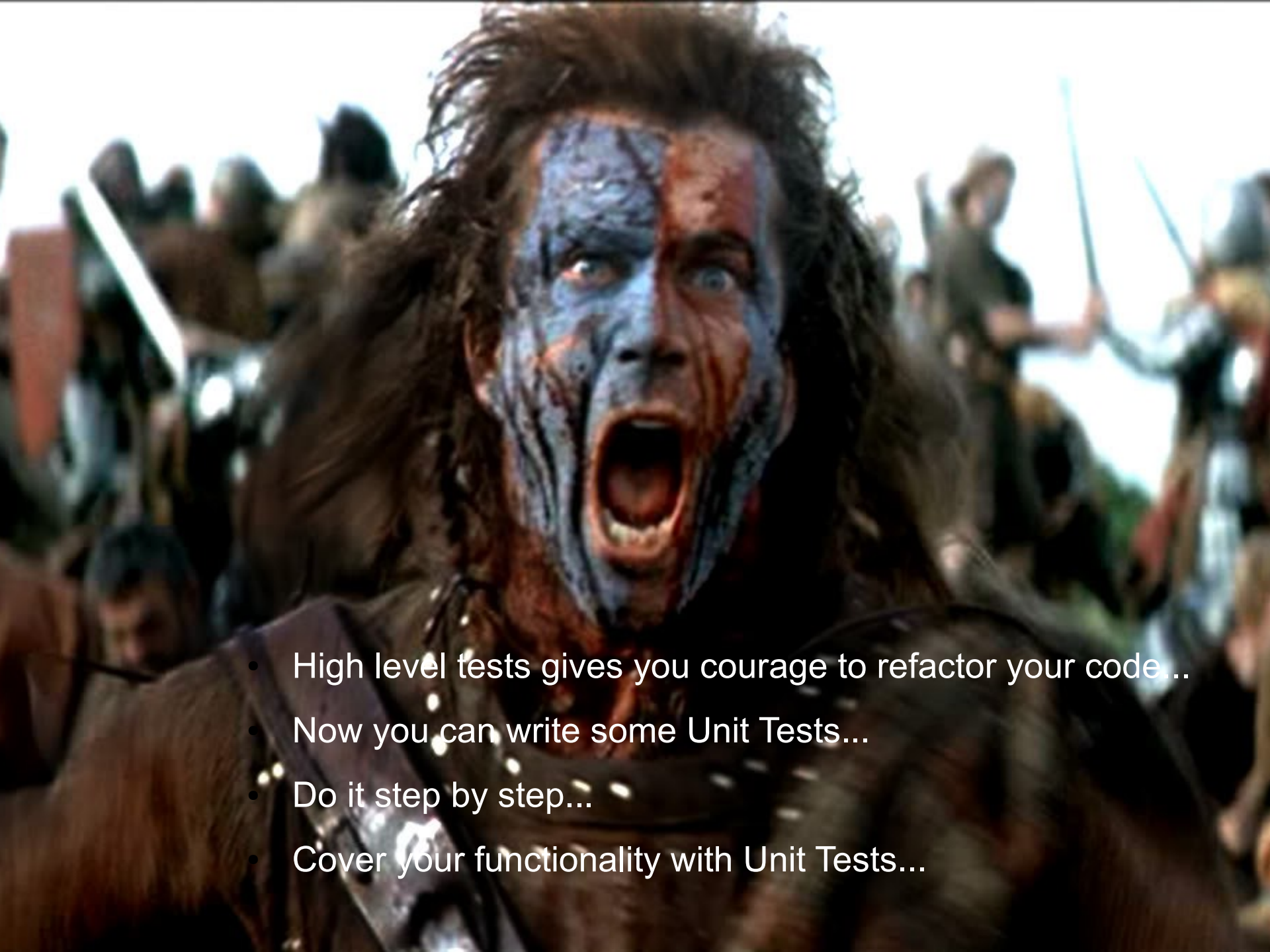
Technical Debt is evil!

- Success in Software Development is something which is not continuous...
- Success is state that you can achieve but also lose very fast if you can't respond to changes fast enough...



Reversed Tests Pyramid





- High level tests gives you courage to refactor your code..
- Now you can write some Unit Tests...
- Do it step by step...
- Cover your functionality with Unit Tests...

But.. There are few reasons why you shouldn't reverse tests pyramid

• End-To-End tests are too long...



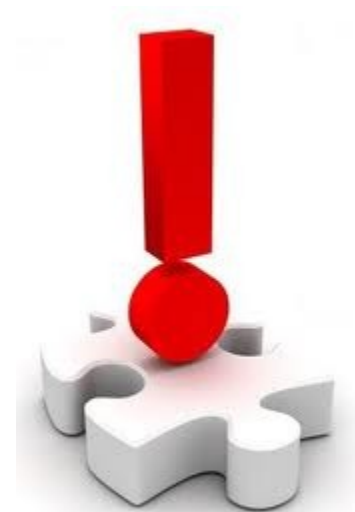
- End-to-end tests are difficult to maintain...
- If we need end-to-end tests we are probably doing something wrong with our architecture...

So it's all about reversing back our tests pyramid.



But...

- Remember that creating reversed tests pyramid and reversing it back will take some time...
- You need to deal with it if you want to pay back your technical debt...



A collection of tools including a wrench, pliers, and several nails on a wooden surface. The wrench is silver and positioned in the upper left. The pliers have red handles and are in the lower right. The nails are scattered across the wooden surface, some forming rectangular shapes. The text "Few final thoughts..." is in the upper right, and "Keep your technical debt as low as possible and try to pay it back every time you can. For example use your slack time for that!" is in the lower left.

Few final thoughts...

Keep your technical debt as low as possible and try to pay it back every time you can. For example use your slack time for that!

Beware of refactoring just for refactoring!



WIKING

DEATH

WANDER

**Resist temptation to re-write from scratch –
history is against you, such projects usually fail.**

Remember to always remove your (duplicated) tests!

The background of the slide features a central black point from which numerous bright green and white light rays radiate outwards, creating a starburst or sunburst effect. The rays vary in thickness and intensity, with some being very bright and others more subtle. The overall color palette is dominated by vibrant greens and bright whites against a dark background.



Software quality in many cases could be understood as ability to introduce changes into software!

Wiktor Żołnowski
www.agilezkolenia.pl

Questions?

