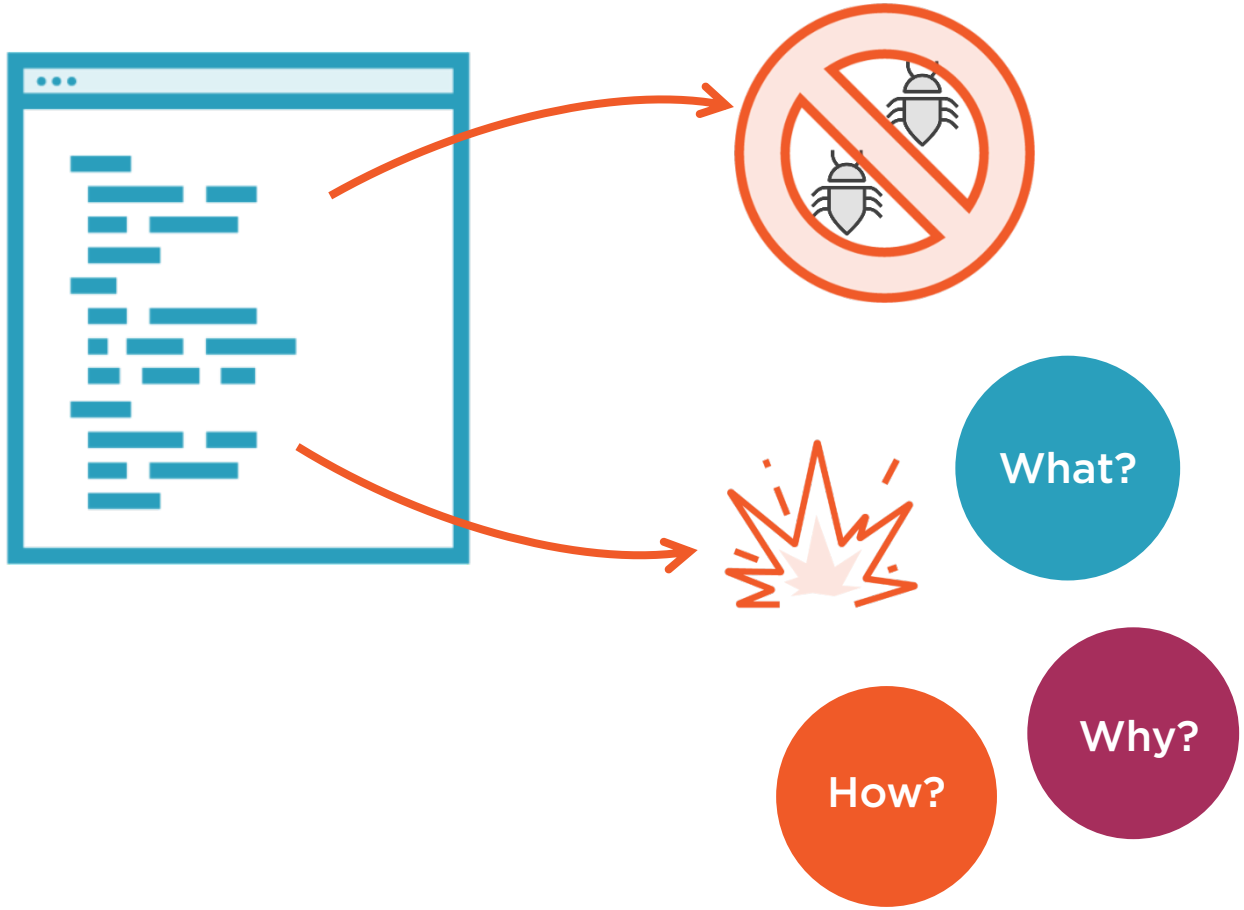
Defensive Coding in Java

WHY DEFENSIVE CODING MATTERS

Andrejs Doronins
TEST AUTOMATION ENGINEER

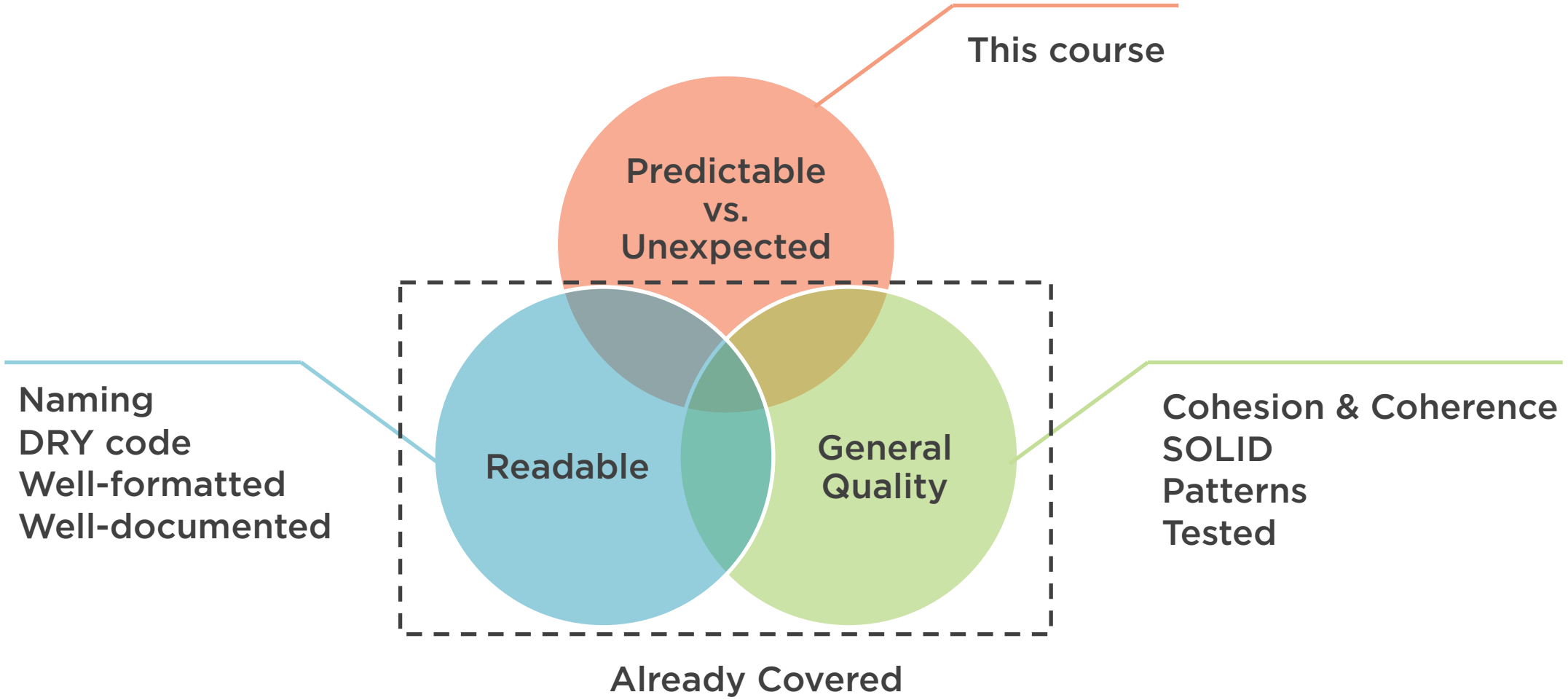

What?
How?
Why?

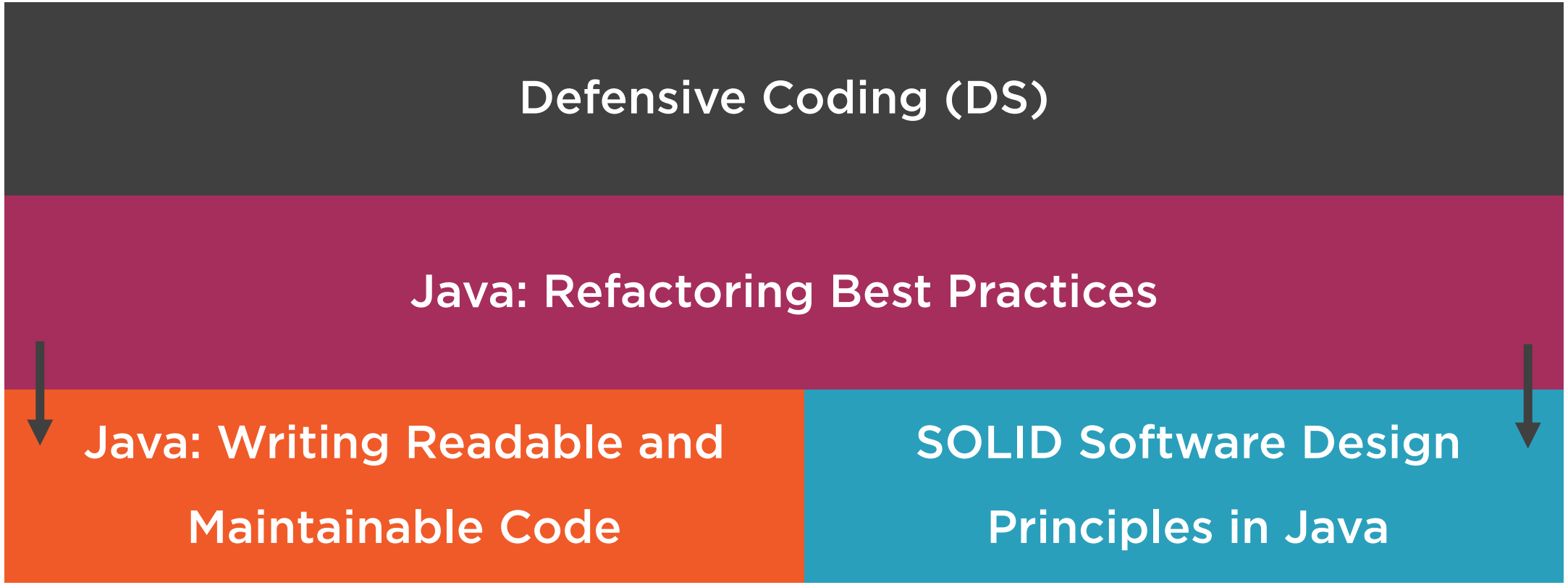
Defensive Coding

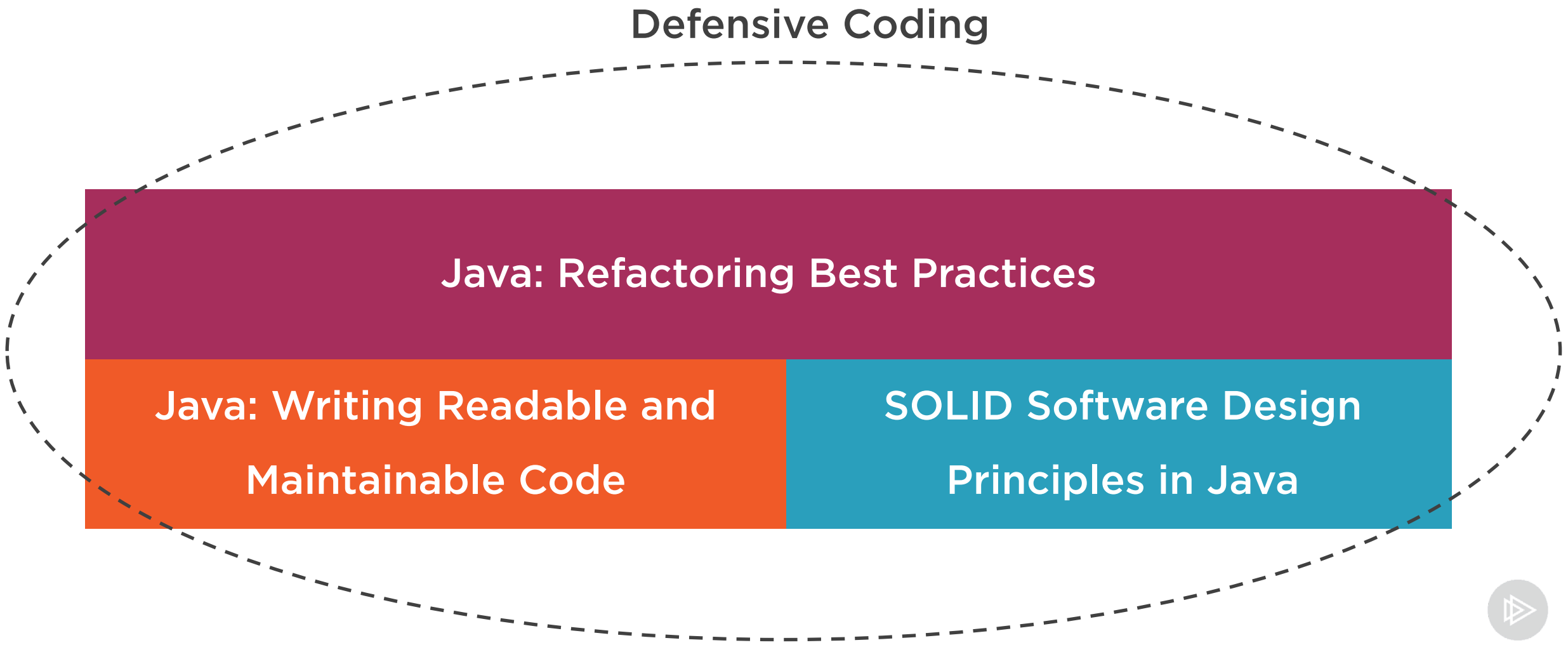
Defensive Coding (Programming)

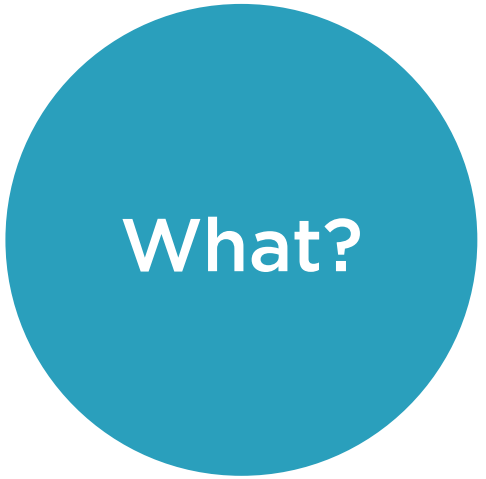
Defensive programming is an approach to improve software and source code, in terms of:

- General quality – reducing the number of software bugs and problems.
- Making the source code comprehensible – the source code should be readable and understandable so it is approved in a code audit.
- Making the software behave in a predictable manner despite unexpected inputs or user actions.

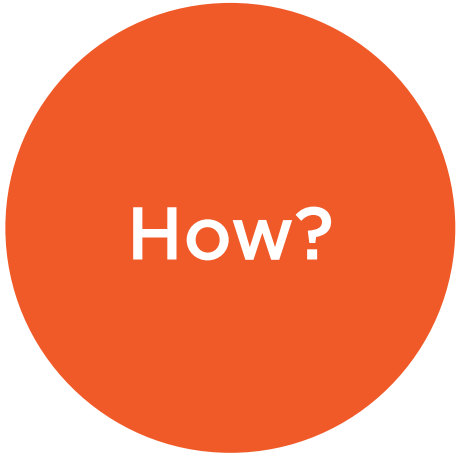

This course
Predictable
vs.
Unexpected
Naming
DRY code
Well-formatted
Well-documented
Readable
General
Quality
Cohesion & Coherence
SOLID
Patterns
Tested
Already Covered


Defensive Coding (DS)
Java: Refactoring Best Practices
Java: Writing Readable and Maintainable Code
SOLID Software Design Principles in Java


Defensive Coding
Java: Refactoring Best Practices
Java: Writing Readable and Maintainable Code
SOLID Software Design Principles in Java

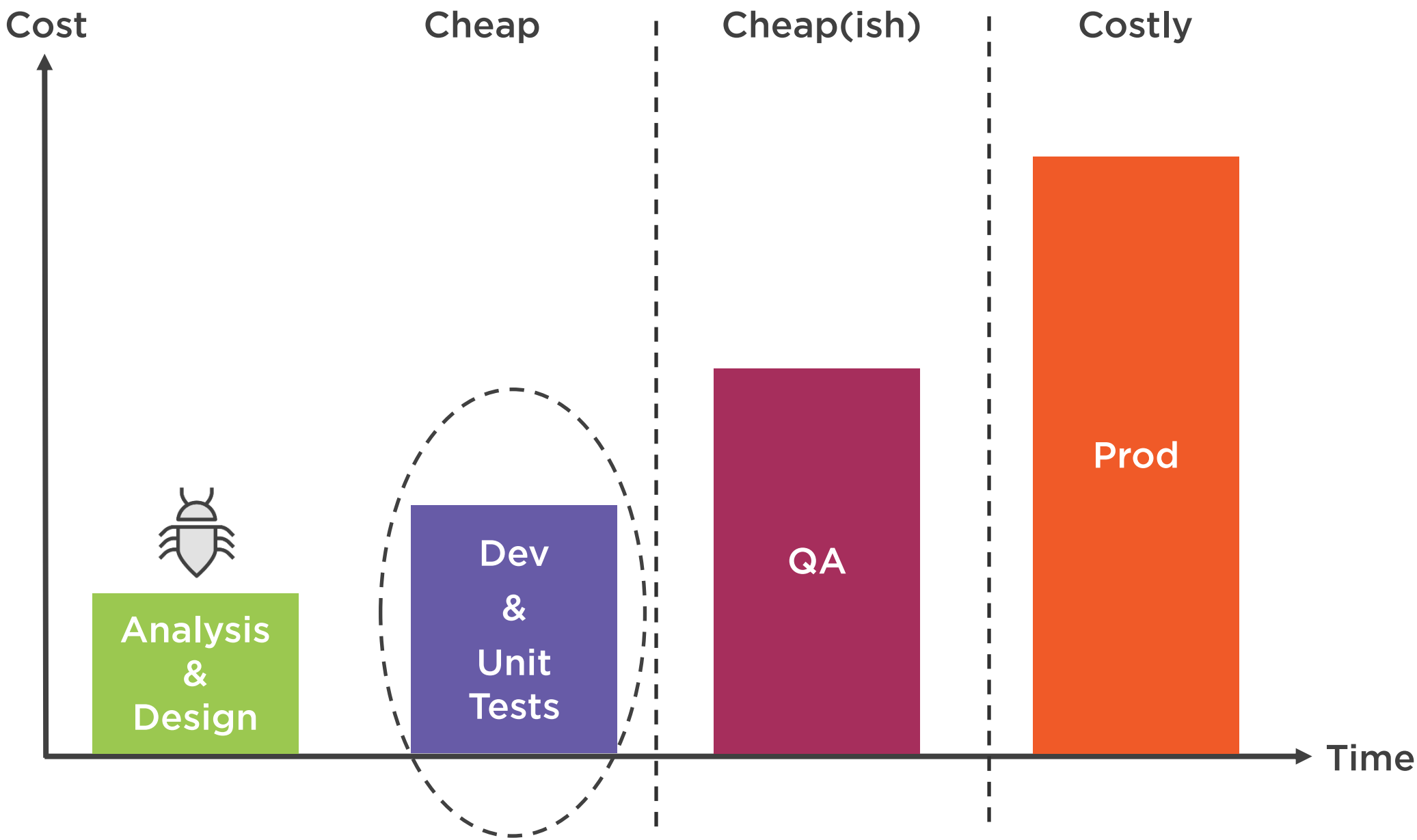


Defensive Coding



“The whole point of defensive programming is guarding against errors you don’t expect. ”

Steve McConnell, Code Complete


Cost
Cheap
Cheap(ish)
Costly
Analysis & Design
Dev & Unit Tests
QA
Prod
Time

The sooner we find a bug,
the better

Defensive coding is all about:
1) Reacting early
2) Preventing

React

Prevent

Prerequisites

Java and OOP

Any IDE

Course Overview

Validating Method Input

Using Frameworks for Validation

Improving Method Return Values

Using Other Defensive Coding Practices