

A person wearing large headphones is looking intently at a laptop screen. Their hands are clasped together near their chin, suggesting deep concentration or contemplation. The scene is dimly lit, with the primary light source coming from the laptop screen, which is partially visible in the lower right. The background is blurred, showing what appears to be a modern office or studio environment with some equipment and lights.

COMPONENTS

Java Clean code

Outline

Lesson 11

Components

Lesson 12.

Architecture

Lesson 13.

Test Driven Development

*Any fool can write code that
a computer can understand.
Good programmers write code that
humans can understand.*

Martin Fowler



Components

- Jar, DLL: dynamic link libraries
- Loose coupling
- Consists of Classes
- Independent deployment & development



Release Reuse Equivalency Principle

- Client code change
- Manageable Components
- Components need to be big





The Common closure principle

- Same reason to change
- Isolate components
- Loose coupling

The Common reuse principle



Similar to the Interface Segregation Principle



Group together classes



Separate classes.



Component coupling

- The Acyclic Dependencies Principle
- The Stable Dependencies Principle
- The Stable Abstraction Principle



The Acyclic Dependencies principle

- Downward dependencies
- It worked yesterday
- Weekly build



The Stable Dependencies principle

- Components that change should not be dependent on difficult to change components
- Components that are heavily dependent should be very stable
- Components that rely on many other components are highly volatile



The Stable Abstraction principle

- Components should be as abstract as it is stable.
- Dependencies run in the direction of abstraction
- Stable == interfaces and abstract classes, Instable == concrete



Components Summary

- Classes that change for the same reason should be in the same component.
- Classes that change for different reason should be in separate Use names that are
- The dependencies should be directed downwards not in circle
- A component should only depend on components that are more stable than he is.

Course Progress

Lesson 11

Components

Lesson 12

Architecture

Lesson 13

Test Driven Development

A sepia-toned photograph of a person clapping their hands. In the foreground, a wooden desk holds an open notebook with a smartphone resting on it. A laptop is partially visible in the background. A dark grey rectangular box with a thin white border is overlaid on the left side of the image, containing the text 'THANK YOU!'.

THANK YOU!