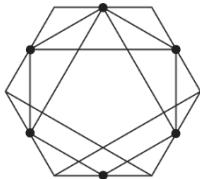
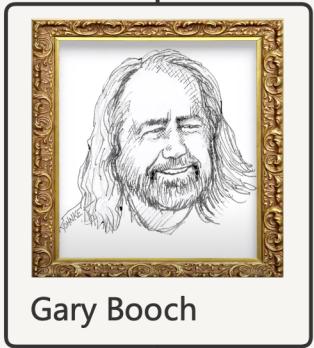


DEPENDENCY INVERSION PRINCIPLE



Clean Code

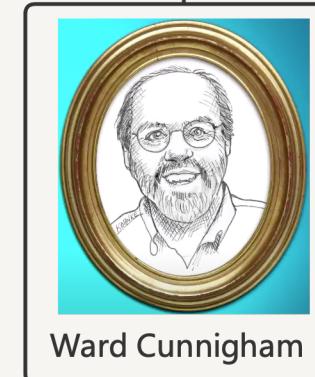


Simple & Direct

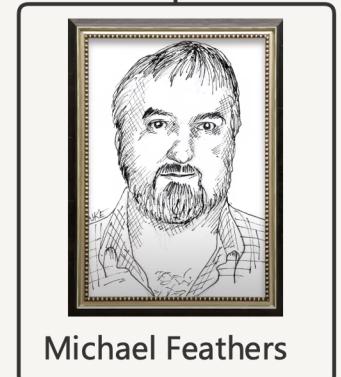
Reads as well-written prose



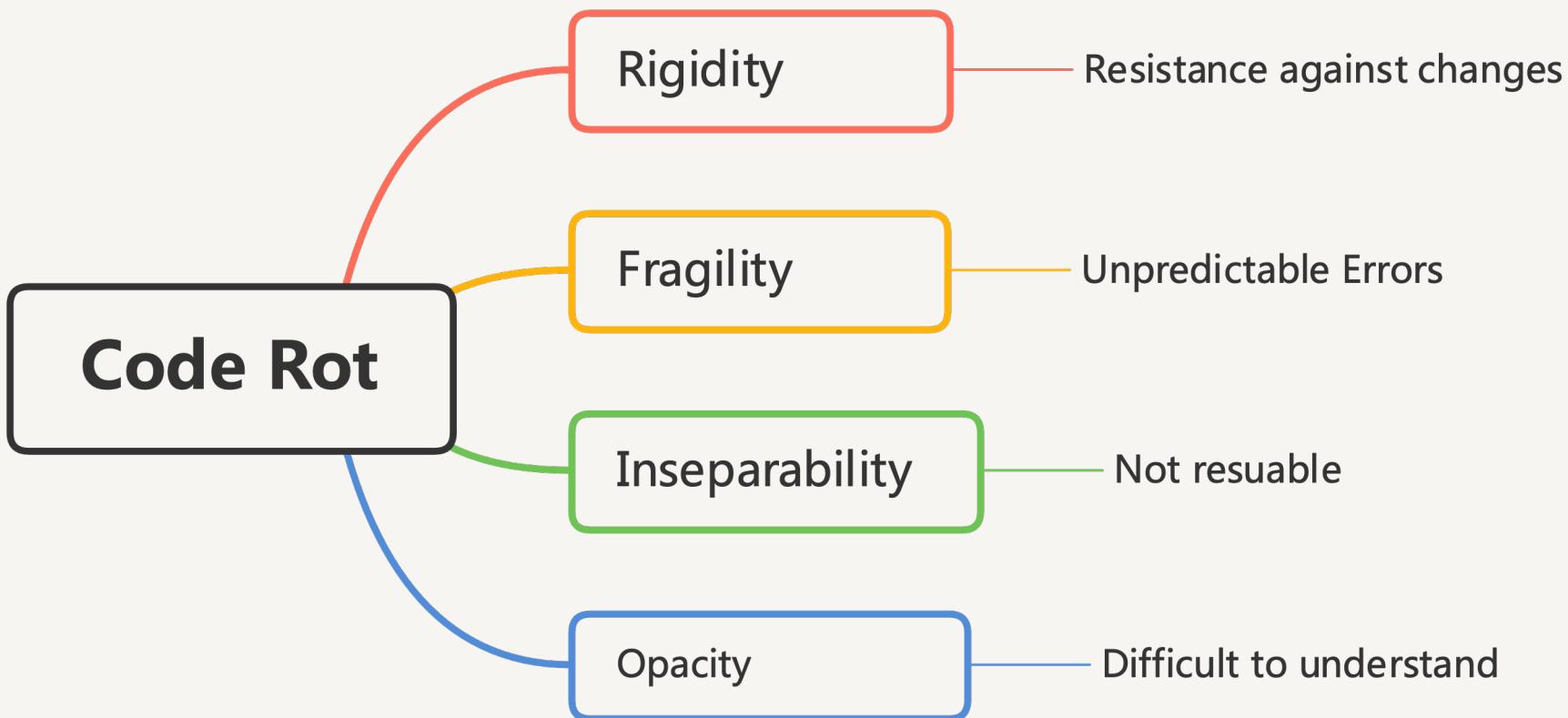
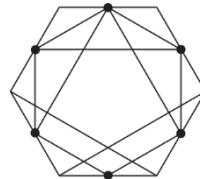
Elegant & Efficient. Should do one thing

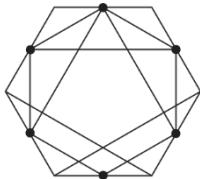


Every Routine is pretty
much what you expected



Written by someone who cares





SOLID

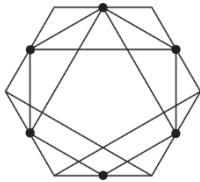
Single Responsibility Principle

Open Close Principle

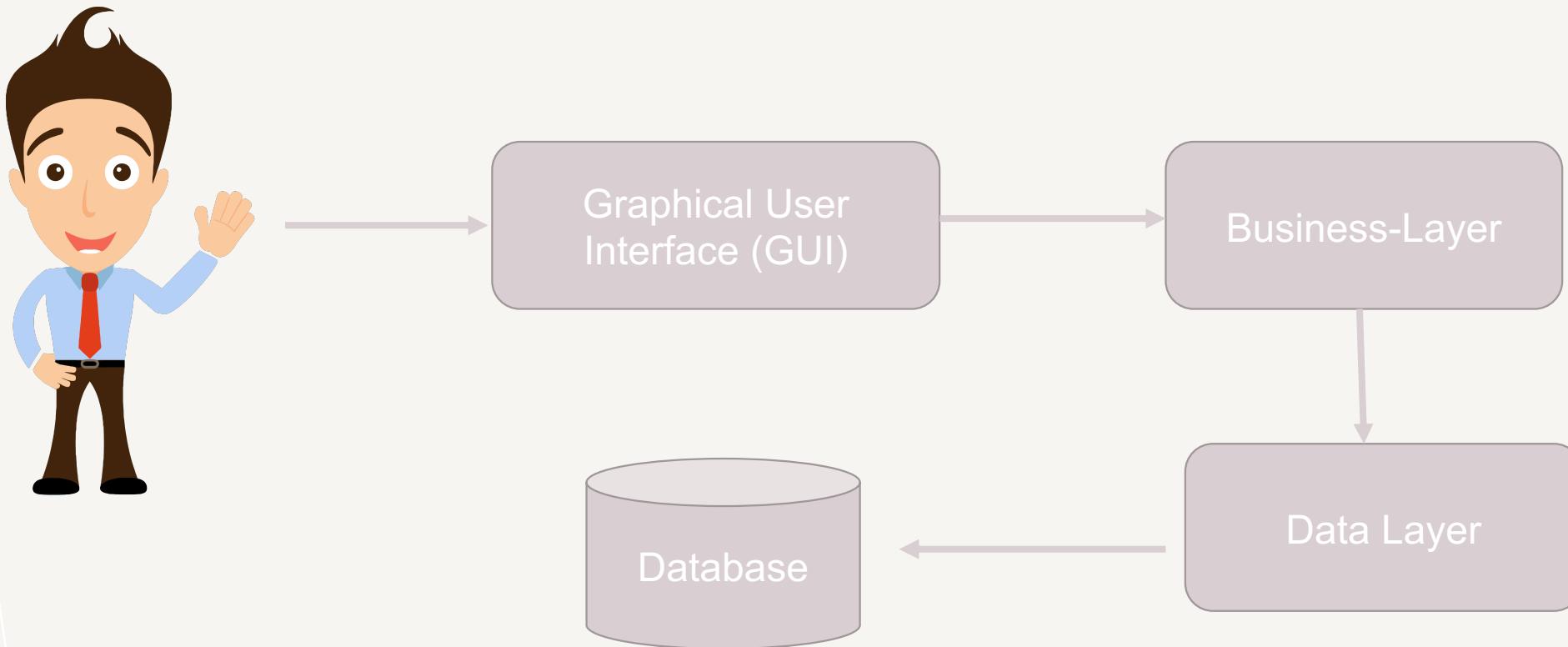
Liskov Substitution Principle

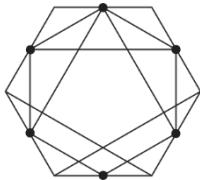
Interface Segregation Principle

Dependency Inversion Principle



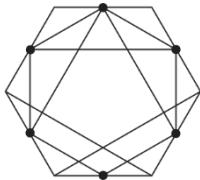
CLASSIC PROGRAM





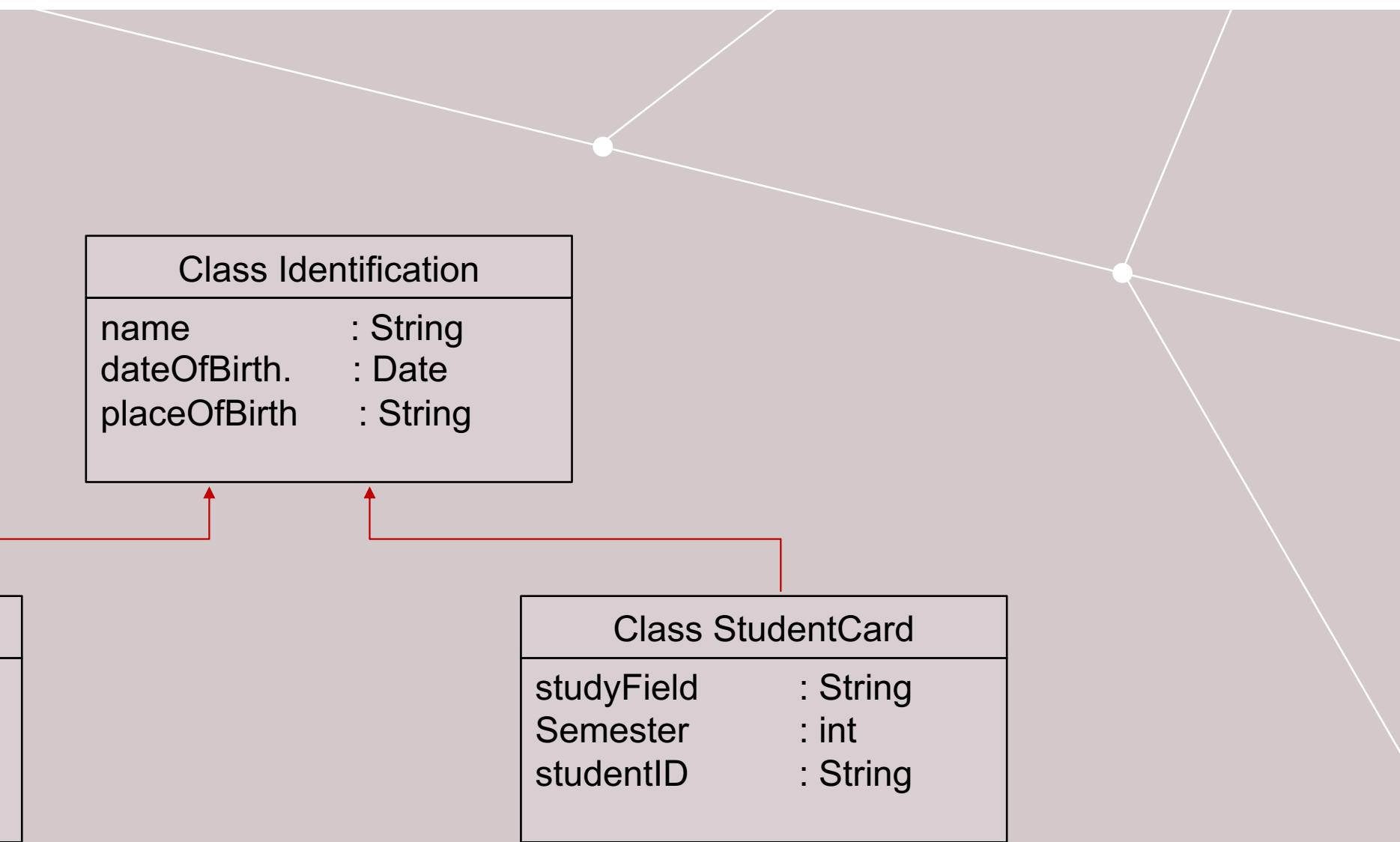
STUDENT

- Name
- Date of Birth
- Place of Birth
- Study
- Semester
- Student ID



CITIZEN

- Name
- Date of Birth
- Place of Birth
- Eye Color
- height
- Adminstration

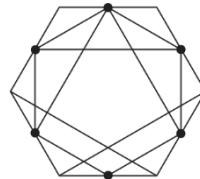


INHERITANCE

Class IdentificationCard	
eyeColoe	: String
Height	: double
adminstration	: String

Class Identification	
name	: String
dateOfBirth.	: Date
placeOfBirth	: String

Class StudentCard	
studyField	: String
Semester	: int
studentID	: String



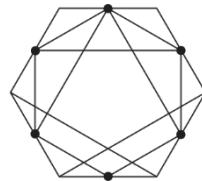
WAS IST OOP WIRKLICH?



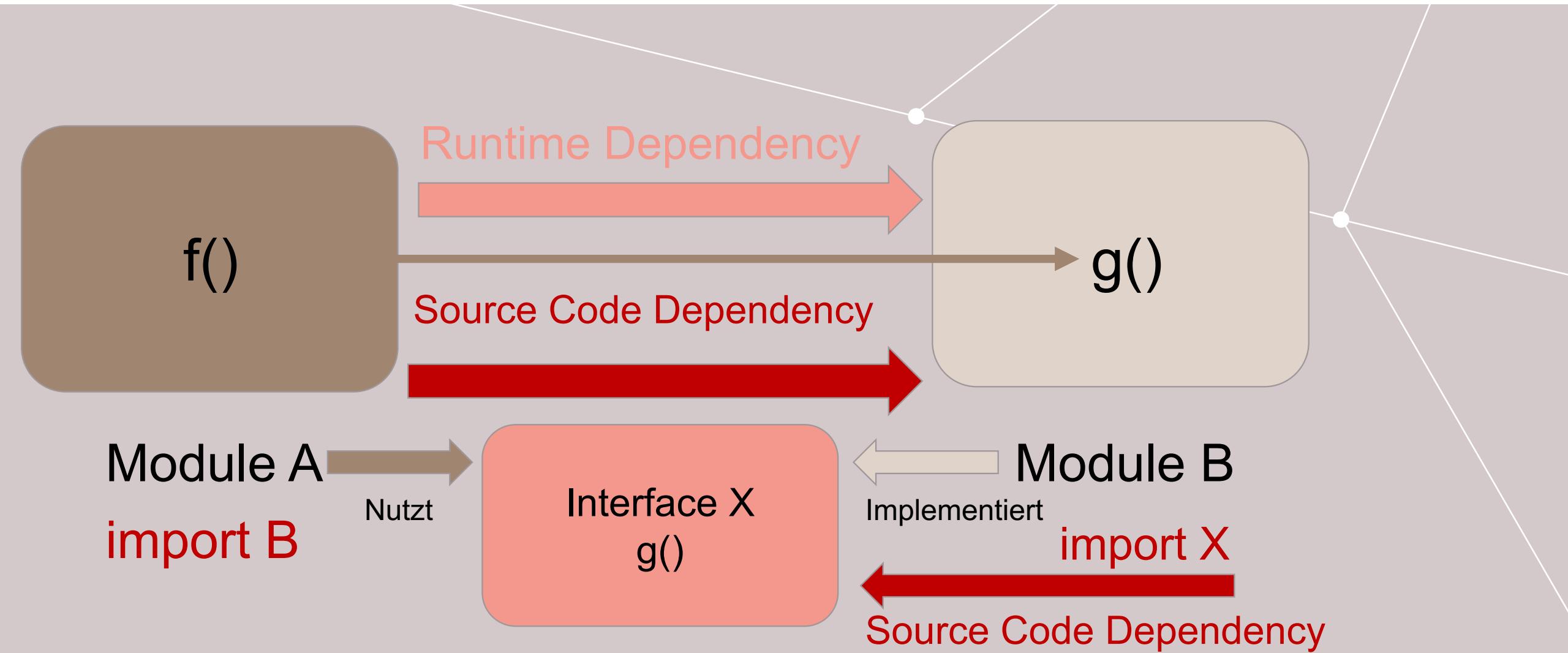
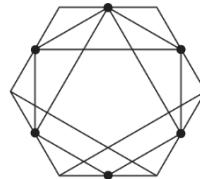
Module A

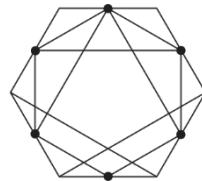
`import B`

Module B

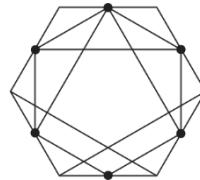


DEPENDENCY?

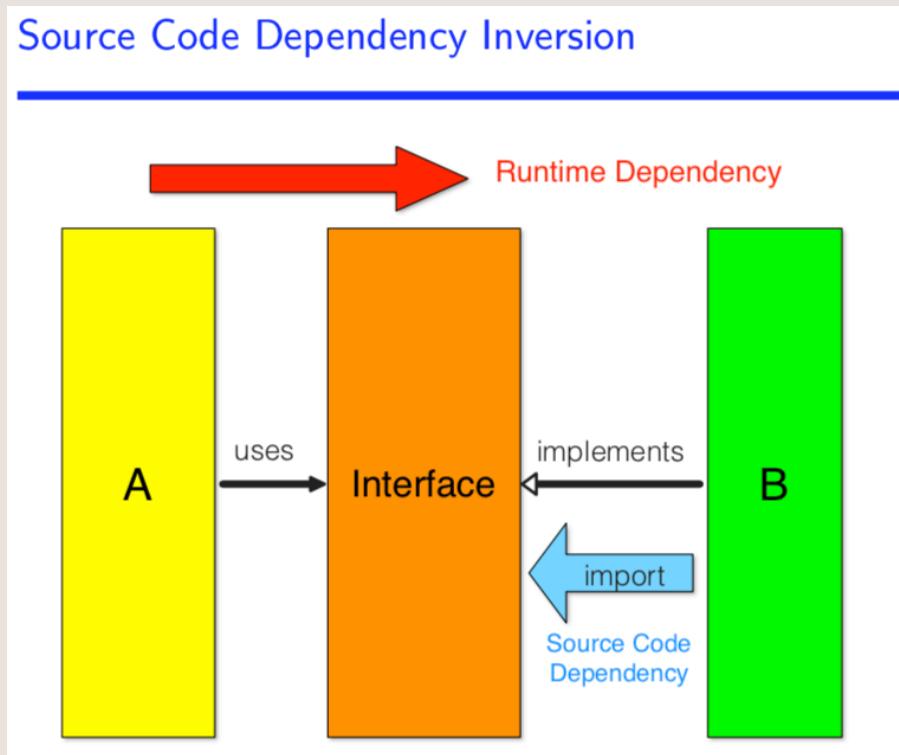




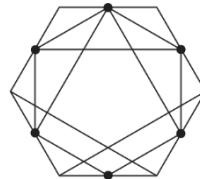
CLEAN ARCHITEKTUR



HEXAGONALE ARCHITECTURE?



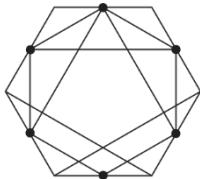
DEPENDENCY INVERSION PRINCIPLE



DEPENDENCY INVERSION PRINCIPLE



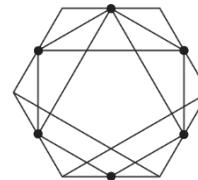
- The core functionality of a system
- does not depend on its environment
- **Concrete artifacts depend on abstractions** (not *vice versa*)
- **Unstable artifacts depend on stable artifacts** (not *vice versa*)
- **Outer layers** of the architecture **depend on inner layers** (not *vice versa*)

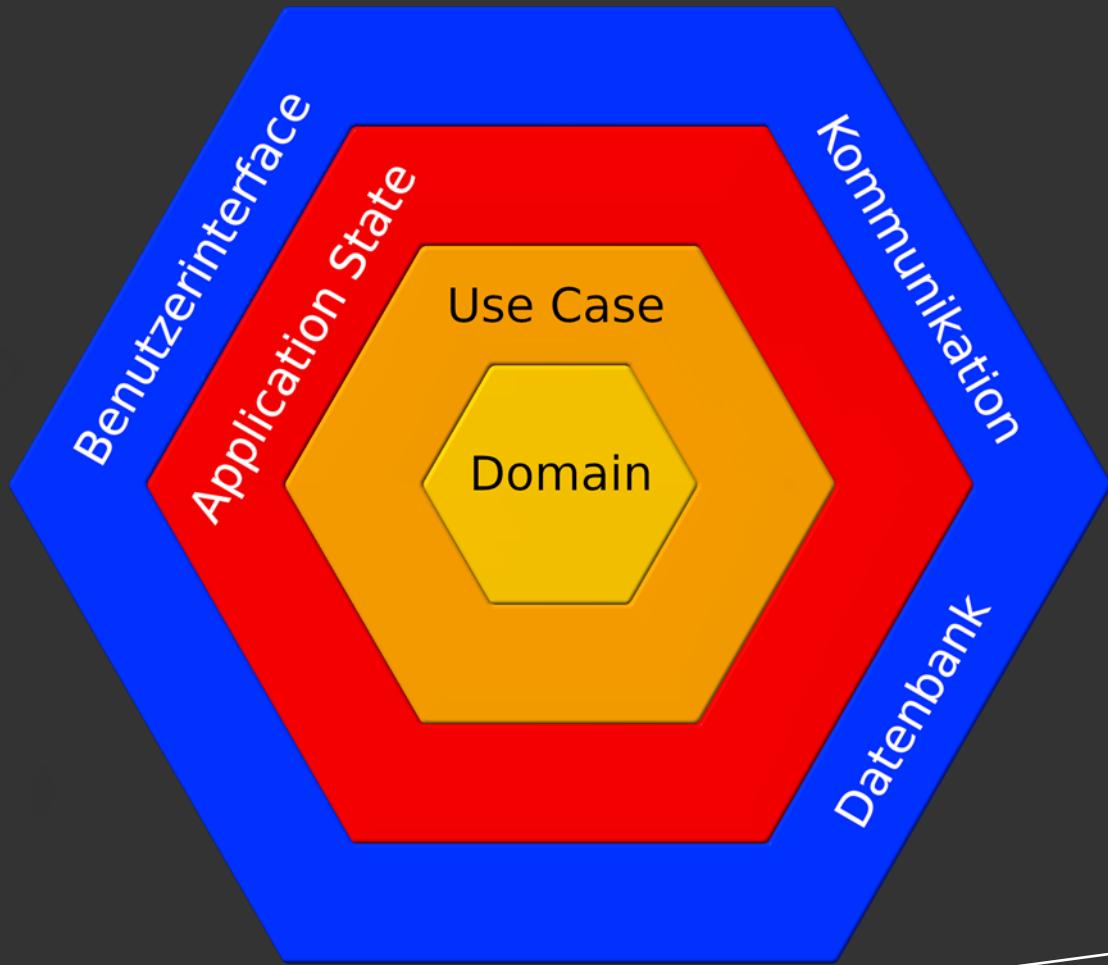
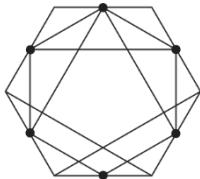


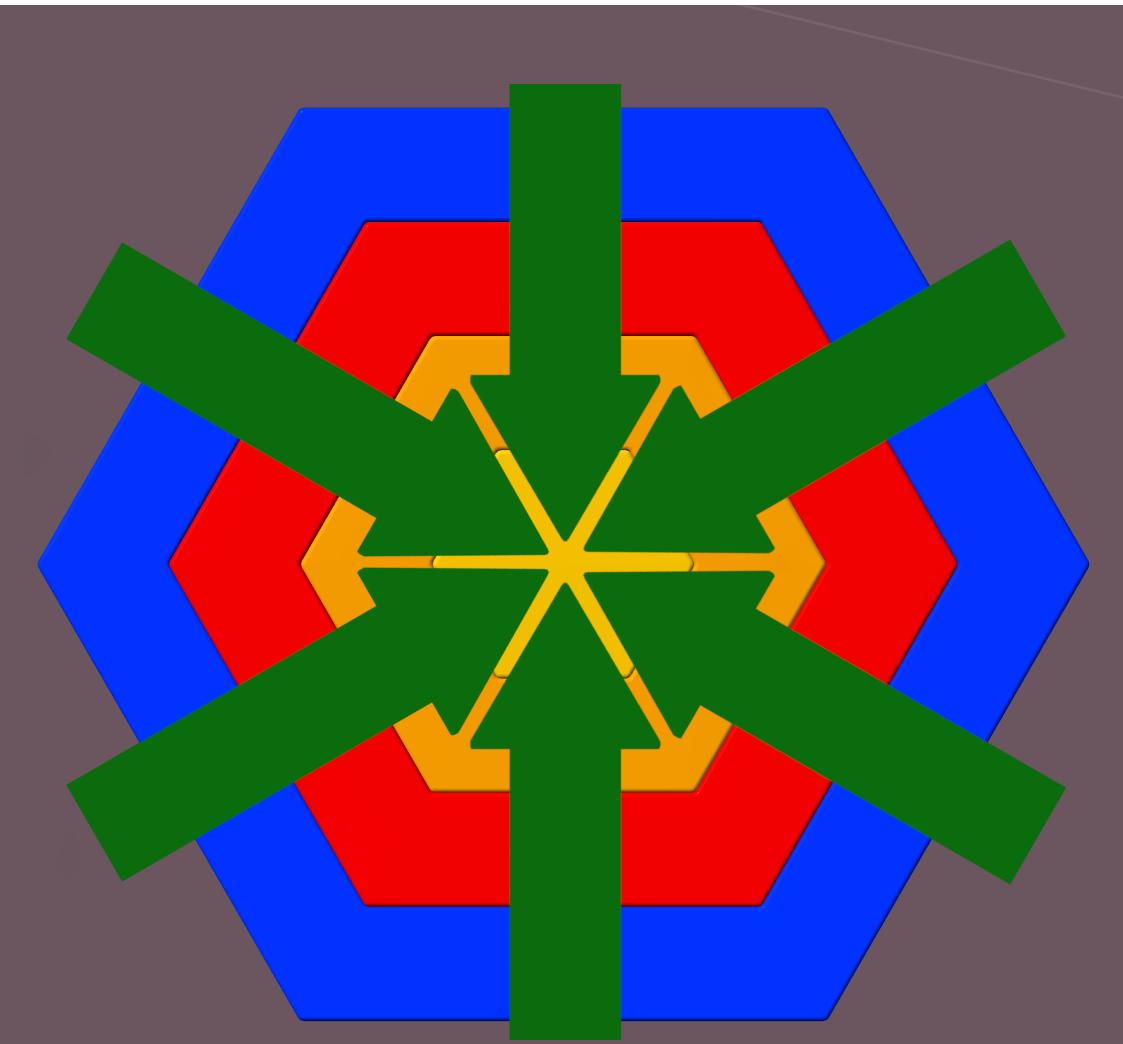
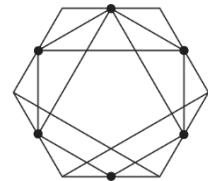
DEPENDENCY INVERSION PRINCIPLE



- Classes/Modules depend on abstractions (e.g., interfaces)
- not on other classes/modules
- Dependency inversion achieves this by introducing interfaces
- that “reverse the dependencies”
- **Outer layers** of the architecture **depend on inner layers** (not *vice versa*)



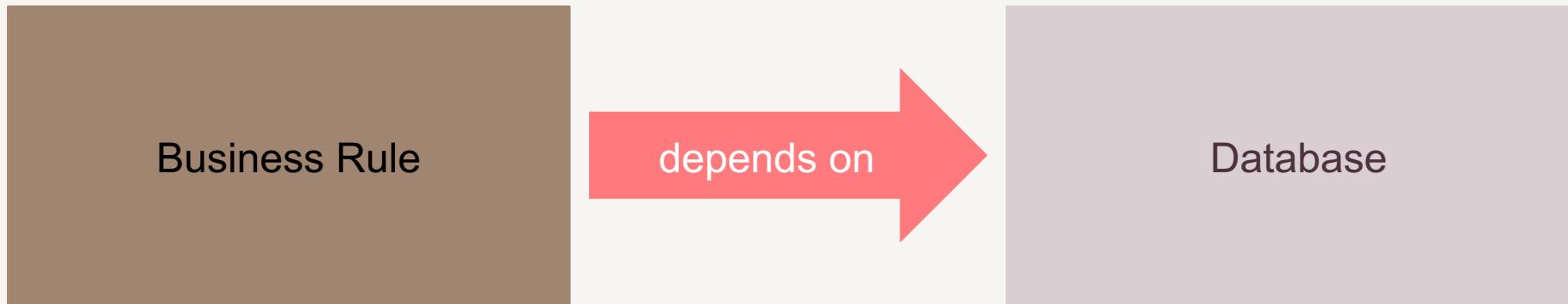




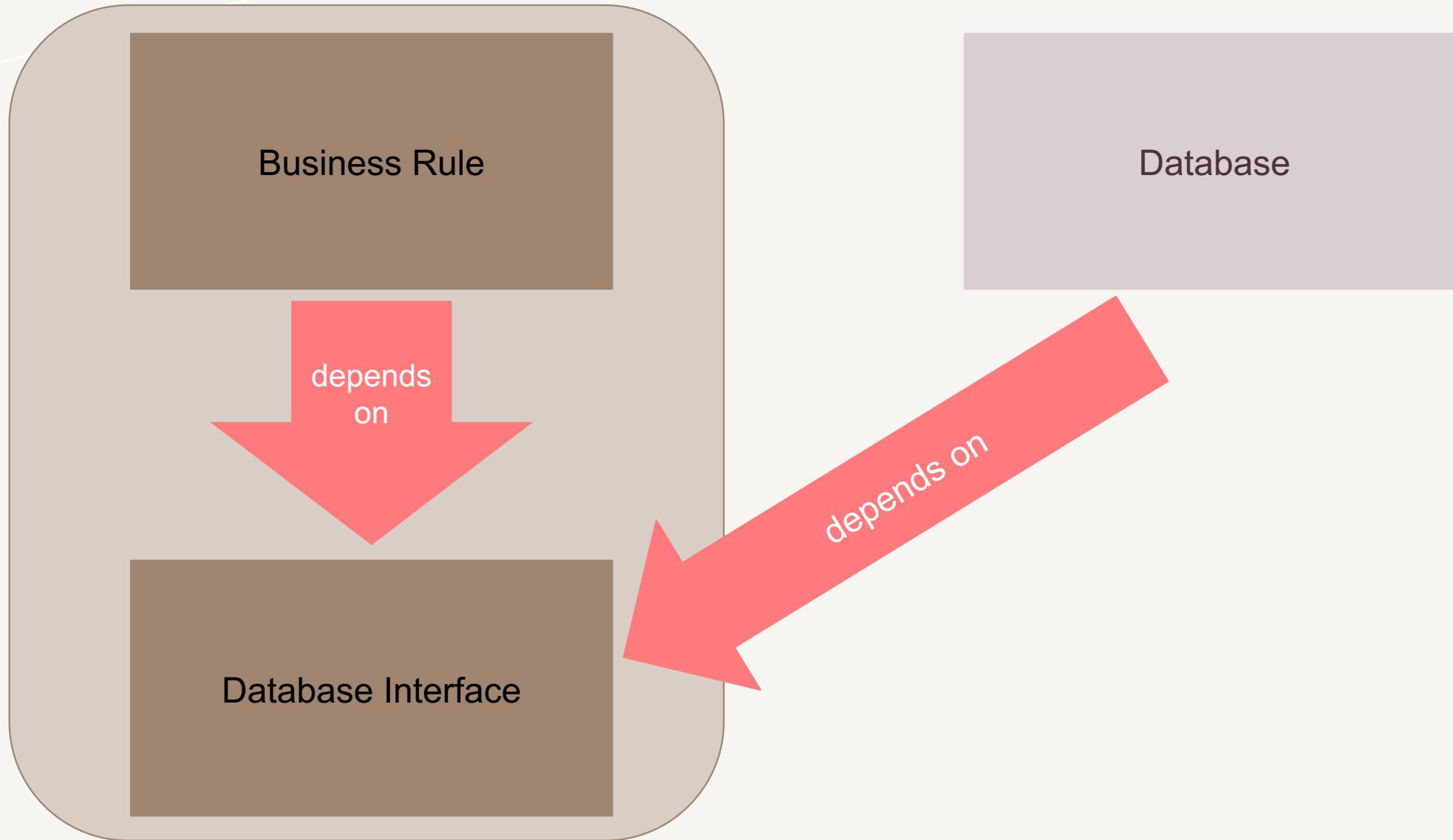
Dependencies



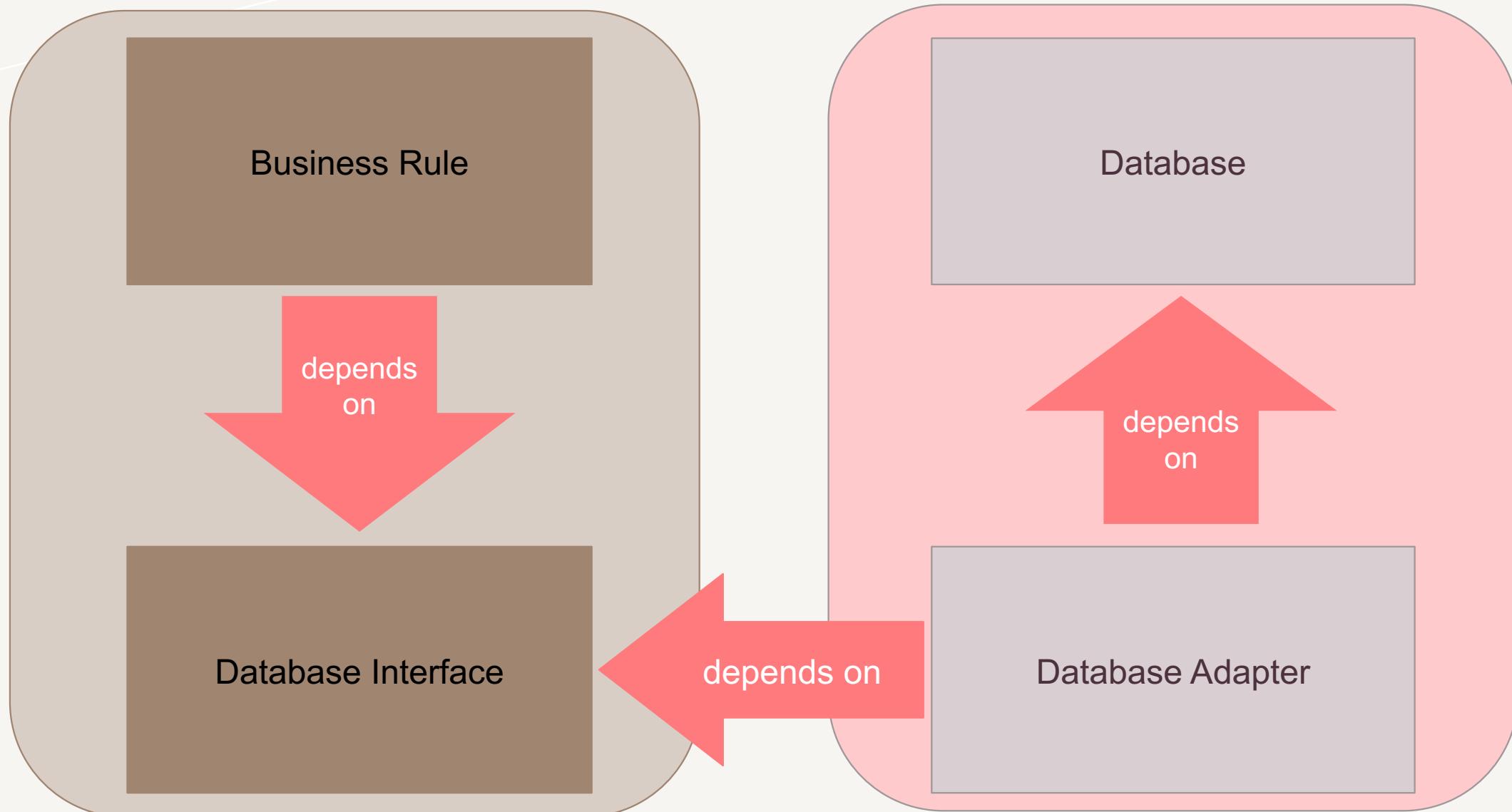
WRONG DEPENDENCY



Dependency Inversion (STEP 1)



Dependency Inversion (STEP 2)



VIELEN DANK FÜR IHRE AUFMERKSAMKEIT.