* design patterns ⇒ reusable solutions to common problems in software design
// make sure nothing left to take away — DRY (do not repeat yourself)

entity ⇒ class or object

Creational → deal with entity creation e.g Singleton and Factory design patterns
Structural → identify a simple way to realize realitonships between entities. eg Adapter design patter
Behavioural → identify common communication patterns between entities e.g Observer design pattern

* Delegation and Extensions

MVC MVVM
↓
as Architectural design pattern

SOLID principles

Single Responsibility Principle ⇒ Entities should have one responsibility

Open-Closed Principle ⇒ Entities should be open to extension but closed to modification

Liskov Substitution Principle
if class A is a subtype of class B, then we should be able to replace B with A without disrupting the behaviour of our program

Interface Segregation
Entities should not be forced to use interfaces they don't use. Large interfaces should be split into smaller ones

Dependency Inversion
High level modules should not depend on low level modules, both should depend on abstractions.

42-2 01:36