OOP

Blueprint of an entity defines what properties & behaviours this entity posses. We call this blueprint as "Class".

Using the blueprint we define real life entities.

For example → With a Movie blueprint (class) we define Doctor Strange as a real life movie. These real life entities are called "Objects"

Eg Class → Product

↓

Object → Iphone X

→ Samsung S10

→ Macbook air

⋮

OOP brings some features along with it.

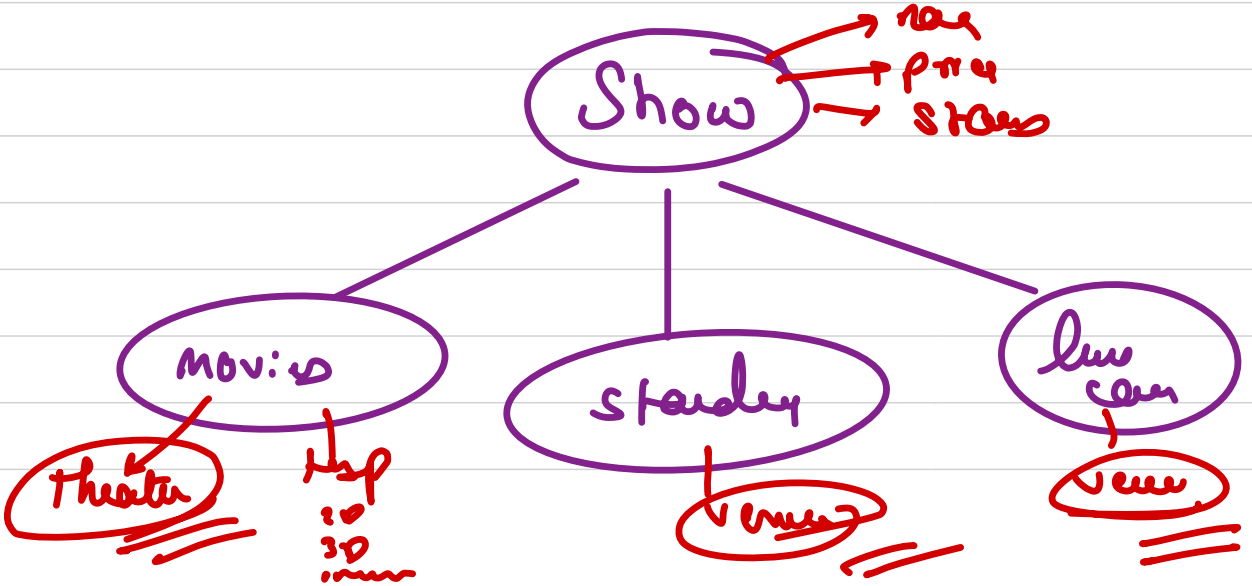
1) Encapsulation
2) Abstraction
3) Inheritance
4) Polymorphism

Inheritance → Book My Show

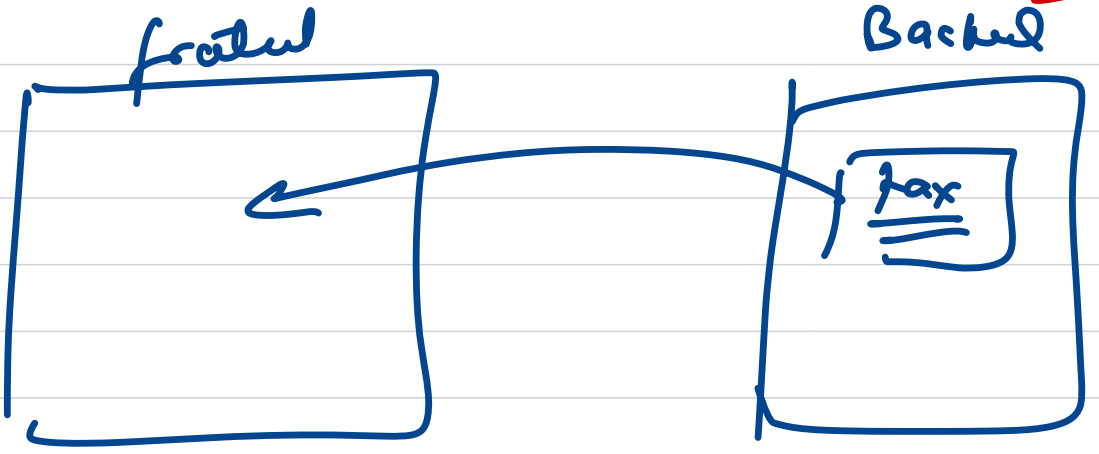
↳ movies

↳ standup

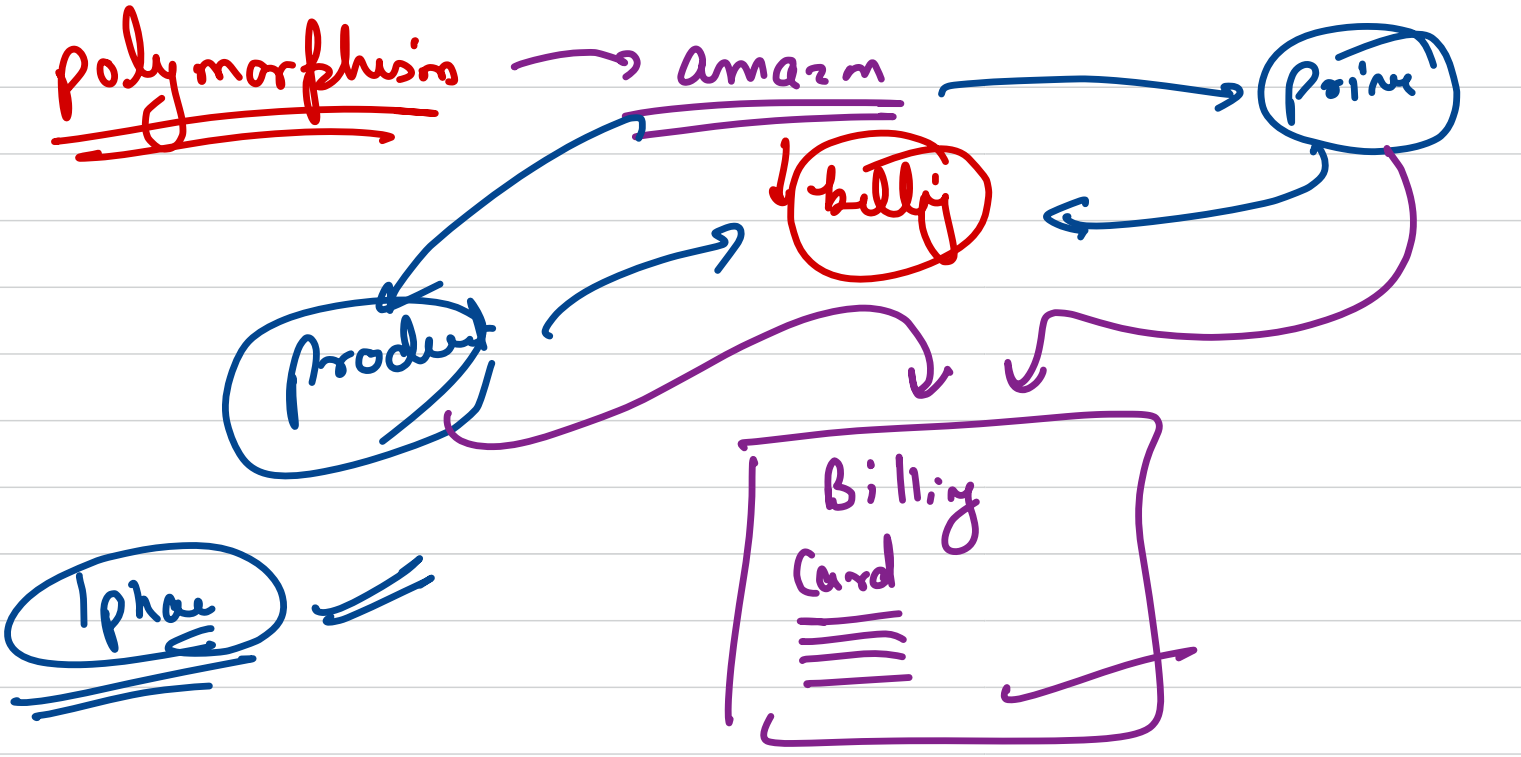
↳ live concert



abstraction → movie → price, tax, value

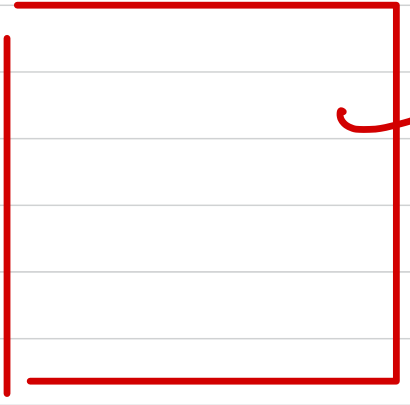


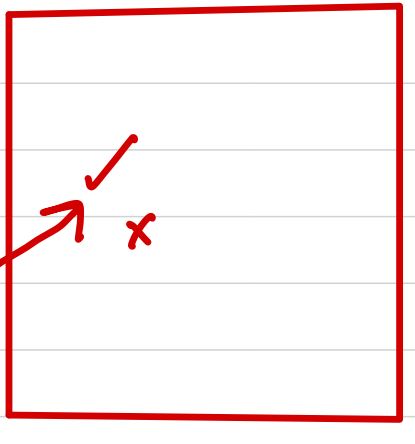
→ showing only essential details & hiding all the non-essentials.



→ perform one operation in multiple forms

Encapsulation →



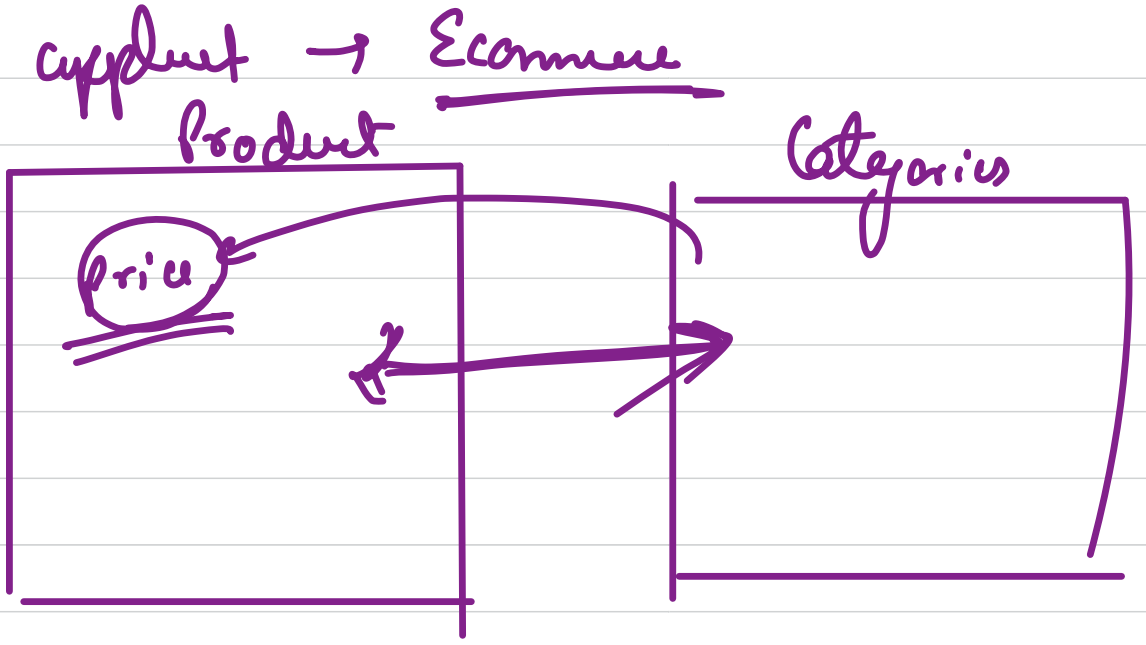


code

single binding utility

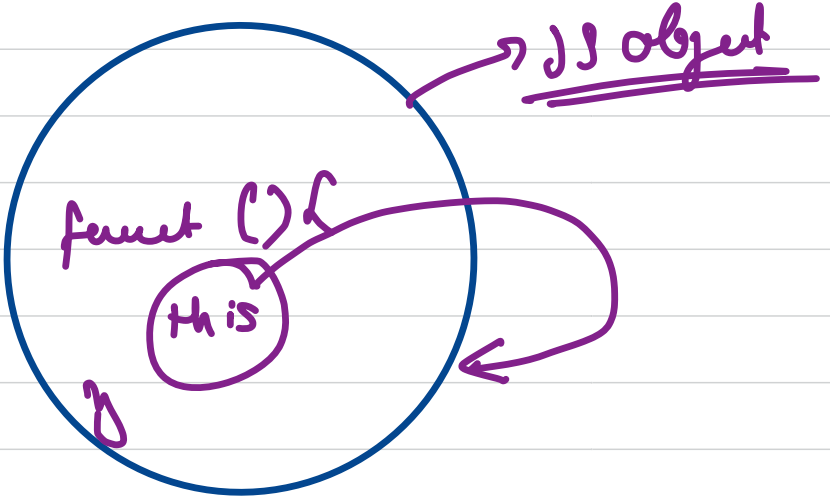
Example

cypdust → Ecommerce



this keyword → "this" keyword can be used w.r.t classes and functions both.

A function's this keyword references the execution context for the function call. This content is determined entirely by how function was called



if we call this from a function, which object (execution context) is alloy the function will be returned by this.

new keyword → When we use new keyword with a function it binds the calling context of the function with an empty object.

→ 1) the new keyword creates a brand new empty object.

2) * Link the object to another object

3) Call func¹ with this property set to the new object created in step 1!

4) If the function does not return an object, it assumes we have to return the value of this.

Note → Inside arrow function this is treated as normal variable and resolved by lexical scope.

That means if you use new keyword with arrow fn, you will get an error.

constructor f^n can create objects