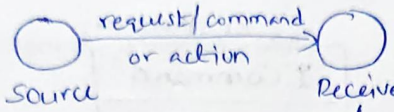
Command Design Pattern :

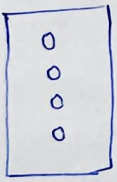


↳ run(), talk(), walk()

Consider we have two objects source and receiver. Source requests or commands receiver to run any of its methods.

But, Command design pattern says that assume the request/command itself an object.

Ex: Smart Home Automation

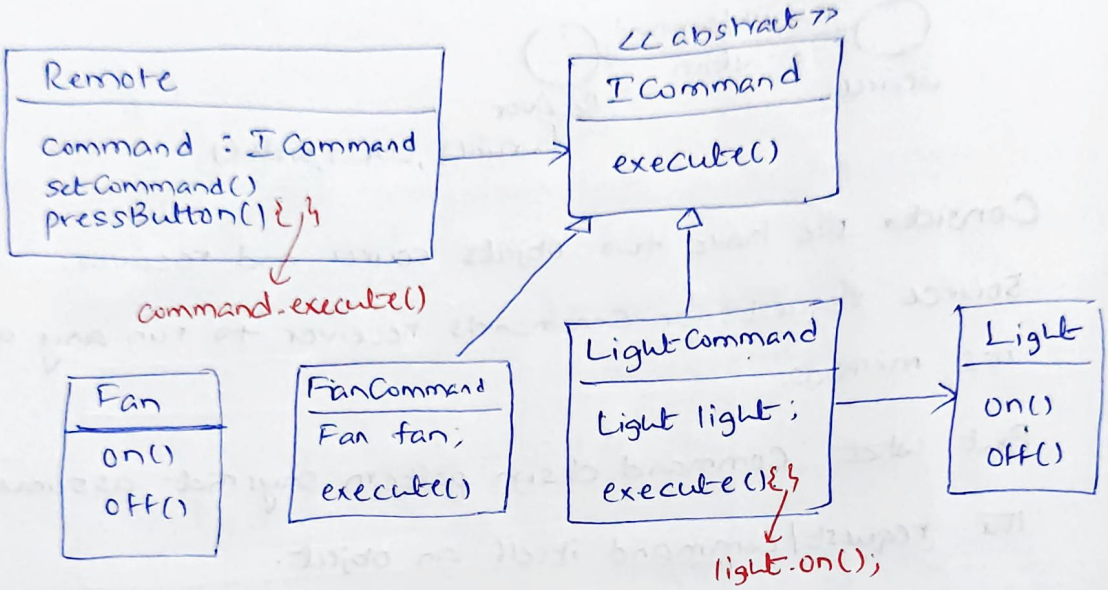


We have a remote with which we control devices like Fan, AC, ~~Radio~~ Light etc. using the buttons. (Those devices are smart devices).

```
Class Light {
    on();
    off();
}
class Remote {
    Light light;
    pressRemoteButton(){
        light.on();
    }
}
```

Here, this remote is tightly coupled with Light device. If we want to add a new device like smart water heater, we go to Remote class and do changes. So it clearly breaks Open-Close principle.

let there is only one button in remote.



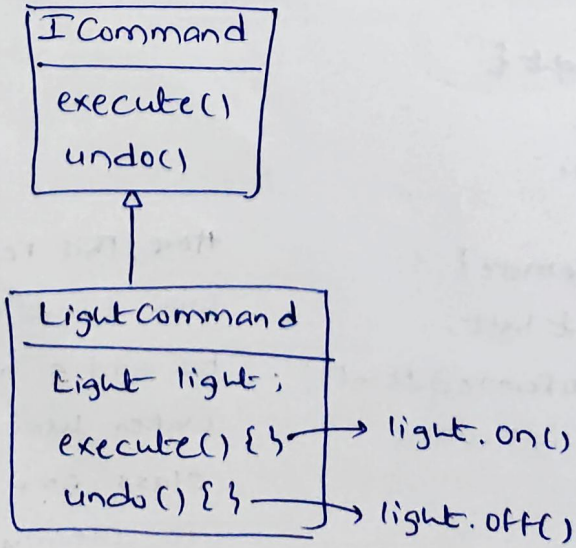
~~for~~ for multiple buttons in remote.

① maintain a list of commands in Remote class.

② pass index of command to pressButton method.

UNDO command :

use undo() method in command interface.



Real-world Example

① Text editor, Photoshop → undo features

② Keyboard shortcuts

Standard Definition :

It encapsulate a Request as an Object, thereby letting you parameterize clients with diff requests, queue or log request and support undoable actions/operations