**Command Design Pattern:** The Command Design Pattern is a [behavioral design pattern](https://www.geeksforgeeks.org/behavioral-design-patterns/) that turns a request into a stand-alone object, allowing parameterization of clients with different requests, queuing of requests, and support for undoable operations.

* The Command Pattern encapsulates a request as an object, allowing for the separation of sender and receiver.
* Commands can be parameterized, meaning you can create different commands with different parameters without changing the invoker(responsible for initiating command execution).
* It decouples the sender (client or invoker) from the receiver (object performing the operation), providing flexibility and extensibility.
* The pattern supports undoable(action or a series of actions that can be reversed or undone in a system) operations by storing the state or reverse commands.

For example: let take an remote control which can control various appliances like AC, TV, Bulb etc.

Let’s take remote control for AC.



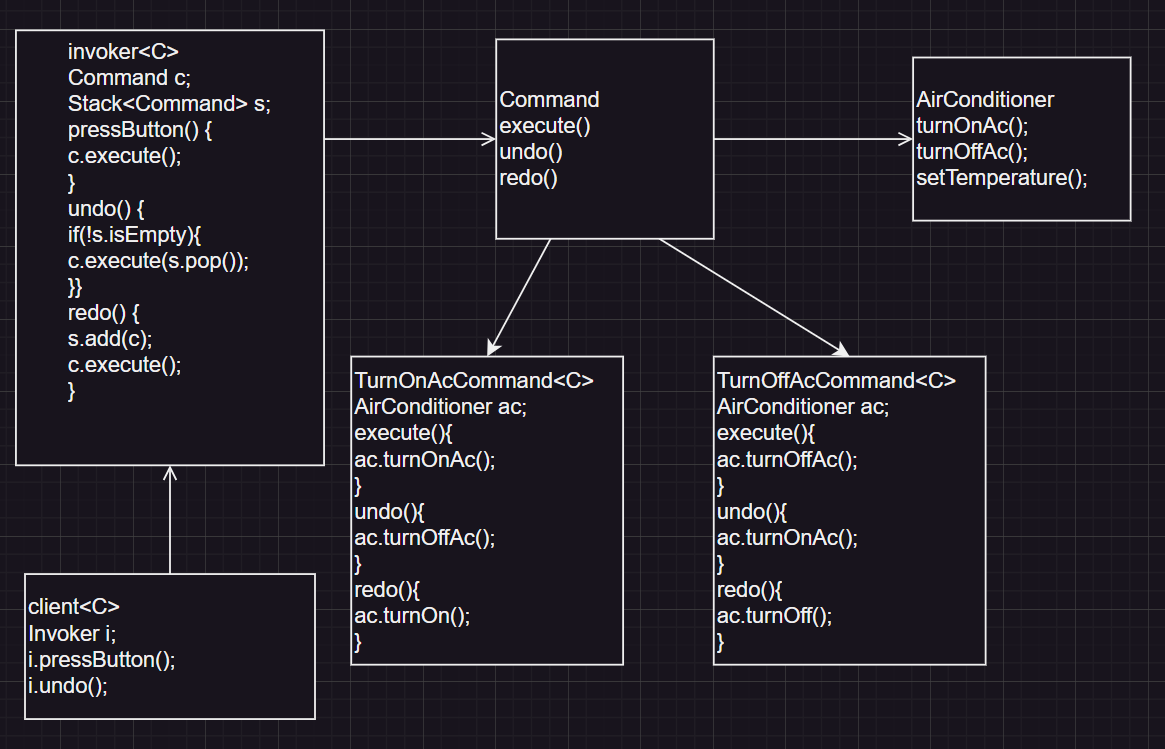
In the above we have Several issues.

* Lack of abstraction: Today process of turning on AC is simple, but in feature if there are more steps involved to turn on AC, then client has to aware of that.
* Undo/Redo Functionality: what if we want to implement undo/redo functionality in the given code. How it will be handled.
* Difficulty in code maintenance: what if in future, we have to support more commands for more devices like bulb & fan.

For this scenario we have to use Command design pattern.

We are going to segregate the problem into three parts:

1. Invoker(Client)
2. Command
3. Reciever



In above diagram we showed that segregated the problem into invoker, command & receiver model.

* So even in feature if the AC turn on steps changes, no issues for the client & client no need to aware.
* Implementing undo & redo operation is easy at invoker, but putting the commands into the stack.
* Even if we want to add more commands we can add with out any code changes, command interface already have default we just need to implement.