Design Pattern: Command

MPCS 51050

April 24, 2018

Intent

Intent

- An object-oriented callback
- Encapsulate a request as an object
- Promote "invocation of a method on an object" to full object status
- Issuing requests without knowing operation details

UML

UML

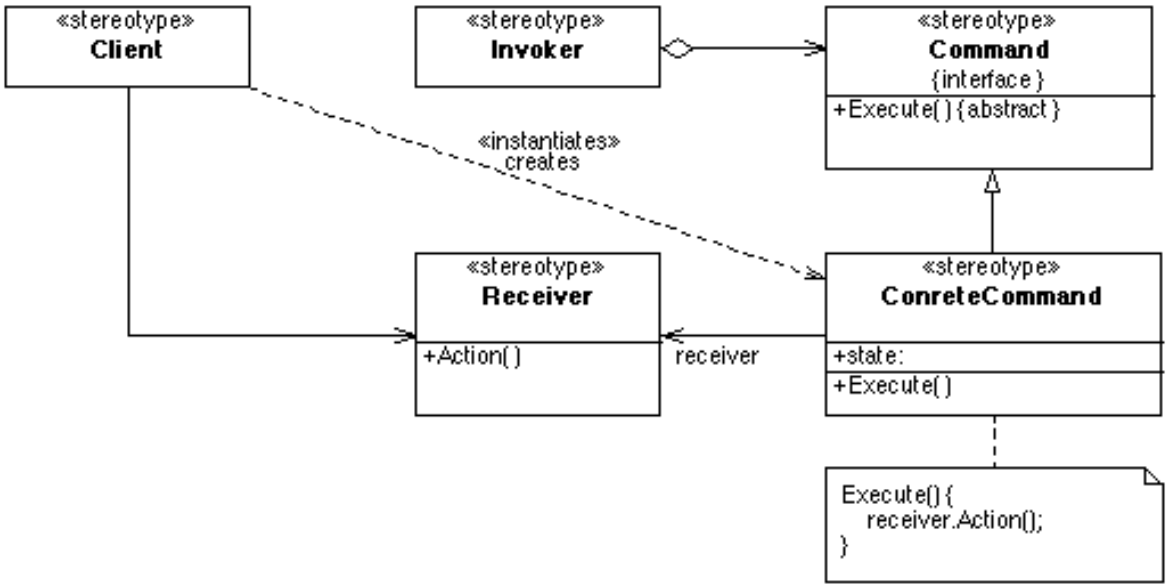


Figure: Command Pattern UML

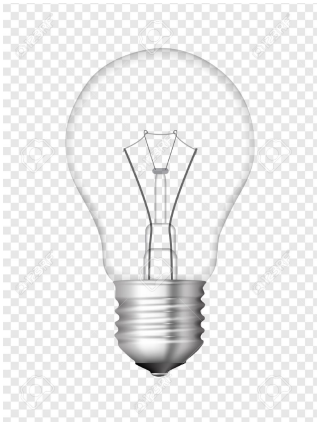
Features

Features

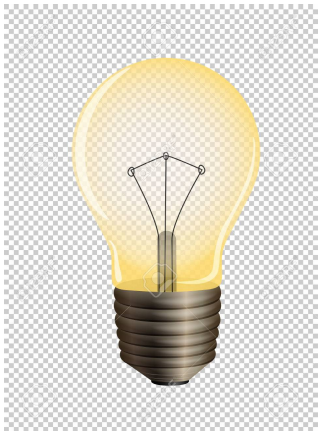
- Command decouples the object that invokes the operation from the one that knows how to perform it.
- All clients of Command objects treat each object as a "black box".
- Sequences of Command objects can be assembled into composite (or macro) commands.

Code Example

Code Example



(a)



(b)

Figure:

Code Example

Code Example

```
abstract class Light {
  boolean status;
  abstract public void turnOn();
  abstract public void turnOff();
  abstract public void display();
}

class BinaryLight extends Light {..}
class Bulb extends Light {..}

public class PressSwitch {
  public static void main(String[] args) {
    Light lamp = new Bulb();
    ..
  }
}
```

Code Example

Code Example

```
abstract class Light {
  turnOn(); turnOff(); display();
}

abstract class Command {
  Light light;
  abstract public void execute();
}

class LightCommandFlipUp extends Command {
  public void execute() {
    this.light.turnOn();
    this.light.display();
    sleep(100);
  }}

class Switch {
  List<Command> log;
  public void action(Command cmd) {
    this.log.add(cmd);
    cmd.execute();
  }}

public class PressSwitch {
  public static void main(String[] args) {
    Light lamp = new Bulb();
    Command switchUp = new LightCommandFlipUp(lamp);
    Switch  mySwitch = new Switch();
  }}
```

Thanks!