



## DEPARTMENT OF INFORMATION TECHNOLOGY

**COURSE CODE:** DJ19ITC802

**DATE:** 14/02/2024

**COURSE NAME:** Design Patterns Laboratory

**CLASS:** BE - IT

### EXPERIMENT NO. 3

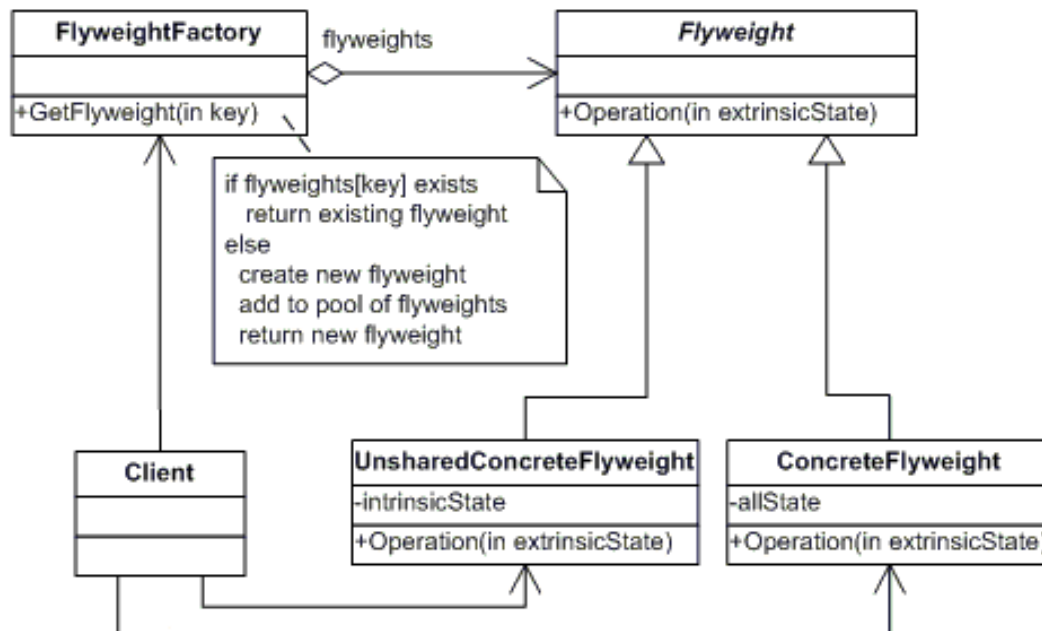
**CO/LO:** Identify and apply the most suitable design pattern to address a given application design problem.

**AIM:** Draw class diagram for Flyweight Pattern and implement the same to demonstrate the working of Counter Strike Game (Instead of creating objects for each player, use Flyweight Pattern to create only 2 objects one for Terrorists class and other for Counter Terrorists class, and reuse them)

#### DESCRIPTION:

Flyweight pattern is one of the structural design patterns as this pattern provides ways to decrease object count thus improving application required object's structure. Flyweight pattern is used when we need to create many similar objects.

#### SOURCE CODE:



**Class Diagram for Flyweight Pattern**



Shri Vile Parle Kelavani Mandal's

**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**

(Autonomous College Affiliated to the University of Mumbai)

NAAC Accredited with "A" Grade (CGPA : 3.18)



```
class Player:
```

```
    def __init__(self, name, weapon, team):
```

```
        self.name = name
```

```
        self.weapon = weapon
```

```
        self.team = team
```

```
    def get_stats(self):
```

```
        return f"{self.name} ({self.weapon}, {self.team})"
```

```
class PlayerFactory:
```

```
    _players = { }
```

```
    def get_player(self, name, weapon, team):
```

```
        key = (weapon, team)
```

```
        if key not in self._players:
```

```
            self._players[key] = Player(name, weapon, team)
```

```
        return self._players[key]
```

```
# Client code
```

```
factory = PlayerFactory()
```

```
player1 = factory.get_player("Player1", "AK47", "Terrorist")
```

```
player2 = factory.get_player("Player2", "M4A1", "Counter Terrorist")
```

```
player3 = factory.get_player("Player3", "AK47", "Terrorist")
```

```
print(player1.get_stats())
```

```
print(player2.get_stats())
```

```
print(player3.get_stats())
```

## OUTPUT

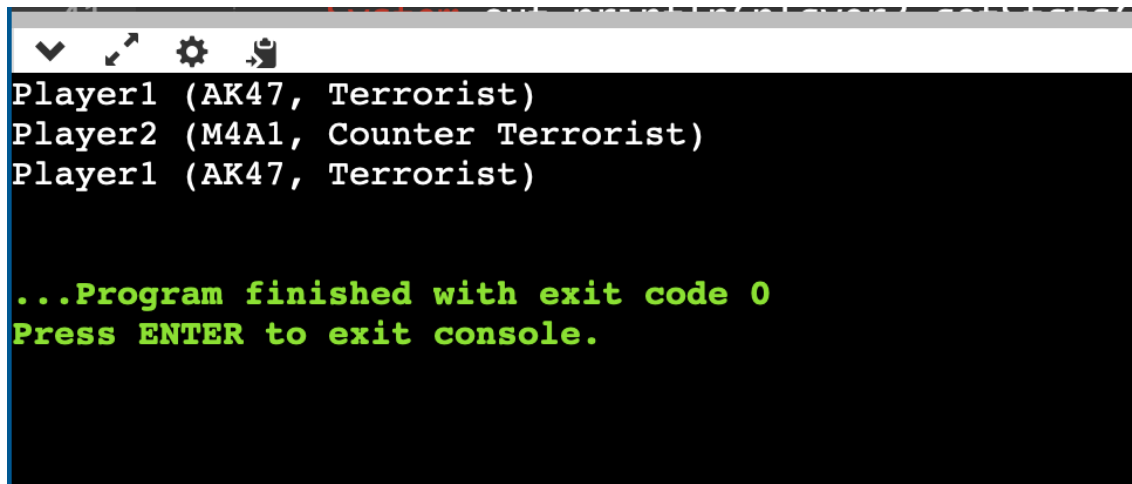


Shri Vile Parle Kelavani Mandal's

**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**

(Autonomous College Affiliated to the University of Mumbai)

NAAC Accredited with "A" Grade (CGPA : 3.18)



```
✓ ↗ ⚙ 🖨  
Player1 (AK47, Terrorist)  
Player2 (M4A1, Counter Terrorist)  
Player1 (AK47, Terrorist)  
  
...Program finished with exit code 0  
Press ENTER to exit console.
```

## CONCLUSION:

We have a Player class that represents a player in the game. We also have a PlayerFactory class that creates instances of Player based on the player's weapon and team. The factory uses the Flyweight Pattern to reuse existing Player objects, if possible, rather than creating new ones. In the client code, we use the PlayerFactory to create three players: two with different weapons and teams, and one with the same weapon and team as the first player. We then print the stats for each player to show that the first and third players are the same object, while the second player is a different object.

## REFERENCES:

[1] <https://www.geeksforgeeks.org/flyweight-design-pattern/>