

# Team #3

## Design Pattern **Proxy**

- **Bill Capps**
- **Doug Hoskisson**
- **Rakan Alanazi**

# Proxy Pattern

Provide a surrogate or placeholder for another object to control access to it.

Proxy design pattern common uses are to control access or to provide a wrapper implementation for better performance.

# Pros and Cons

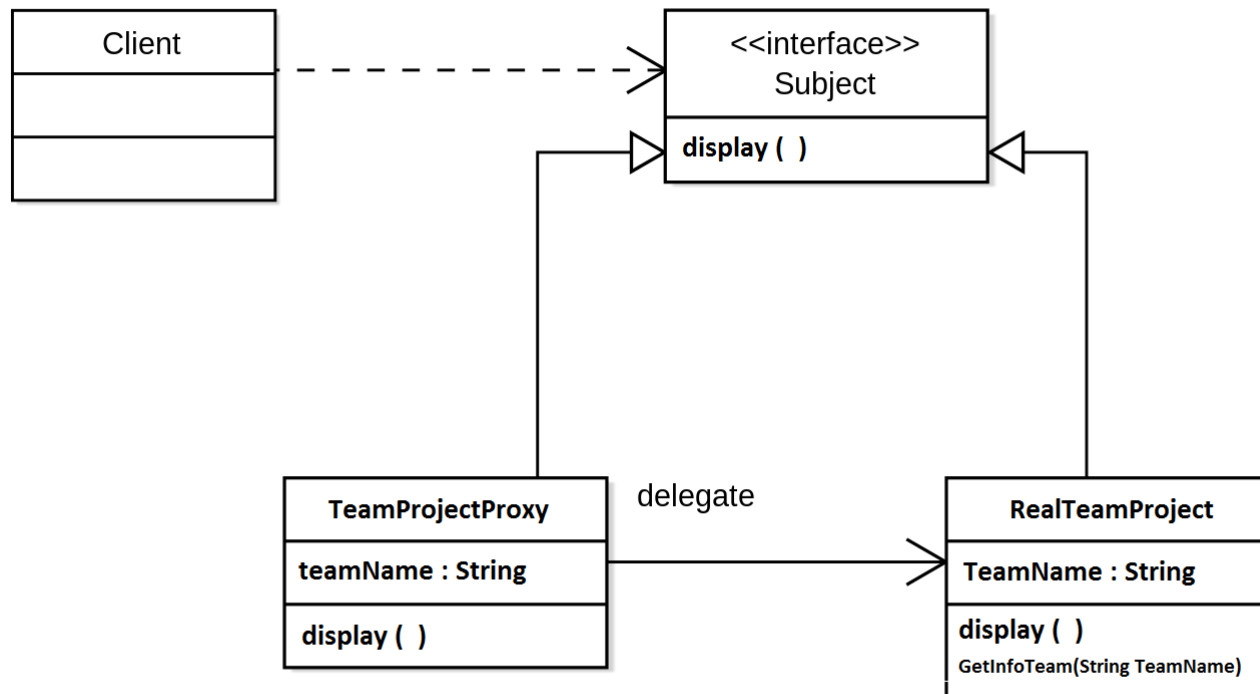
- **Pros**

- Provide security.
- Avoids duplication of objects which can save on the amount of memory used.

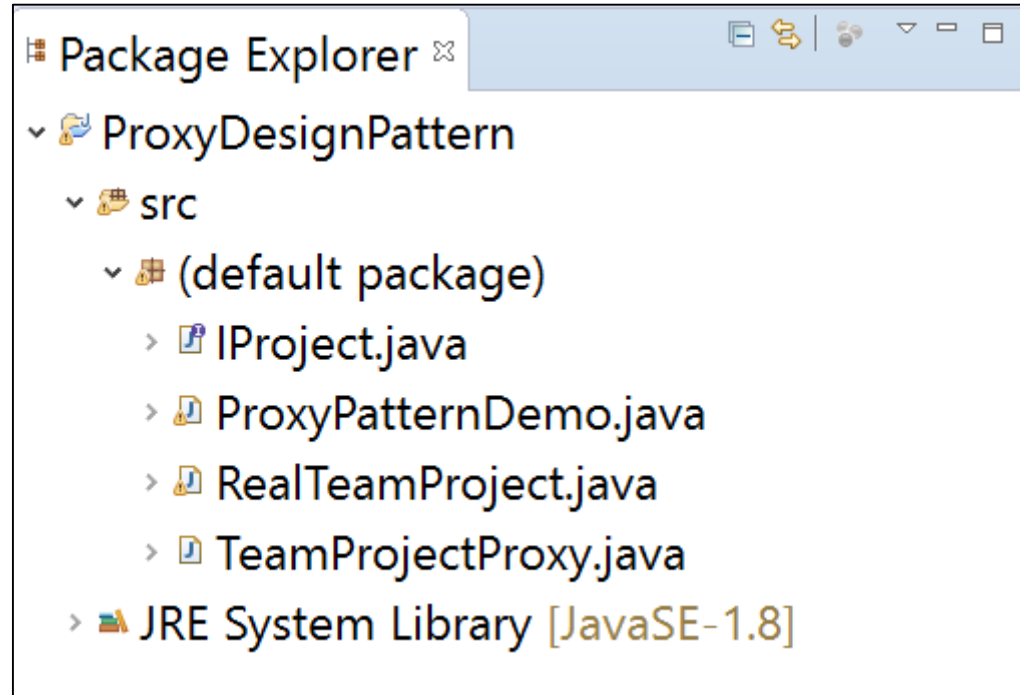
- **Cons**

- Adding a level of indirection while accessing real subject.
- Client can not know that the real subject it is accessing now is the same as pervious one .

# UML Diagram



# Implementation



# Subject << interface>>

```
1  
2 public interface IProject {  
3  
4     public void display();  
5  
6  
7 }
```

# Proxy Class

```
1
2 public class TeamProjectProxy implements IProject {
3     private RealTeamProject realTeamProject;
4     private String teamName;
5
6     public TeamProjectProxy(String username) {
7         this.teamName = username;
8     }
9
10    public void display() {
11
12        realTeamProject = new RealTeamProject(teamName);
13        realTeamProject.display();
14    }
15 }
```

# Real Subject Class

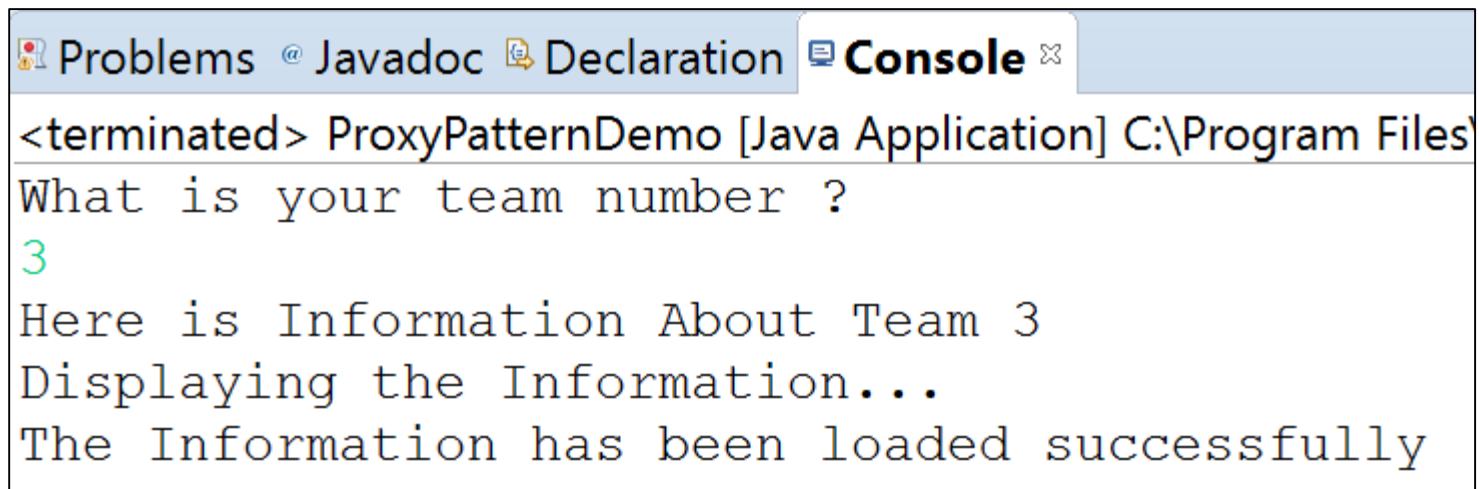
```
1
2
3 public class RealTeamProject implements IProject{
4
5     private String TeamName;
6
7     public RealTeamProject(String TeamName){
8         this.TeamName = TeamName;
9         GetInfoTeam(TeamName);
10    }
11
12    public void display() {
13        System.out.println("Displaying the Information...");
14    }
15
16    private void GetInfoTeam(String TeamName){
17        System.out.println("Here is Information About Team " + TeamName);
18    }
19 }
```



# Demo Class

```
1 import java.util.Scanner;
2
3 public class ProxyPatternDemo {
4
5
6     public static void main(String[] args) {
7         Scanner scanner = new Scanner(System.in);
8         System.out.println("What is your team number ? ");
9         String input = scanner.nextLine();
10
11         IProject teamInfo = new TeamProjectProxy(input);
12
13         teamInfo.display();
14         System.out.println("Information has been loaded successfully");
15
16     }
17 }
```

# The Output



The screenshot shows an IDE window with four tabs: 'Problems', 'Javadoc', 'Declaration', and 'Console'. The 'Console' tab is active, displaying the output of a Java application named 'ProxyPatternDemo'. The output text is as follows:

```
<terminated> ProxyPatternDemo [Java Application] C:\Program Files\
What is your team number ?
3
Here is Information About Team 3
Displaying the Information...
The Information has been loaded successfully
```