

# Proxy Pattern

Simply, proxy means an object representing another object.

According to GoF, a Proxy Pattern "**provides the control for accessing the original object**".

So, we can perform many operations like hiding the information of original object, on demand loading etc.

Proxy pattern is also known as **Surrogate or Placeholder**.

RMI API uses proxy design pattern. Stub and Skeleton are two proxy objects used in RMI.

## Advantage of Proxy Pattern

- It provides the protection to the original object from the outside world.

## Usage of Proxy Pattern:

It is used:

- It can be used in **Virtual Proxy** scenario---Consider a situation where there is multiple database call to extract huge size image. Since this is an expensive operation so here we can use the proxy pattern which would create multiple proxies and point to the huge size memory consuming object for further processing. The real object gets created only when a client first requests/accesses the object and after that we can just refer to the proxy to reuse the object. This avoids duplication of the object and hence saving memory.
- It can be used in **Protective Proxy** scenario---It acts as an authorization layer to verify that whether the actual user has access the appropriate content or not. For example, a proxy server which provides restriction on internet access in office. Only the websites and contents which are valid will be allowed and the remaining ones will be blocked.
- It can be used in **Remote Proxy** scenario---A remote proxy can be thought about the stub in the RPC call. The remote proxy provides a local representation of the object which is present in the different address location. Another example can be providing interface for remote resources such as web service or REST resources.
- It can be used in **Smart Proxy** scenario---A smart proxy provides additional layer of security by interposing specific actions when the object is accessed. For example, to check whether the real object is locked or not before accessing it so that no other objects can change it.

```

public void grantInternetAccess() {
    System.out.println("Internet Access granted for employee: " + employeeName);
}
}

```

### Step 3

Create a **ProxyInternetAccess** class that will implement **OfficeInternetAccess** interface for providing the object of **RealInternetAccess** class.

File: *ProxyInternetAccess.java*

```

public class ProxyInternetAccess implements OfficeInternetAccess {
    private String employeeName;
    private RealInternetAccess realaccess;
    public ProxyInternetAccess(String employeeName) {
        this.employeeName = employeeName;
    }
    @Override
    public void grantInternetAccess()
    {
        if (getRole(employeeName) > 4)
        {
            realaccess = new RealInternetAccess(employeeName);
            realaccess.grantInternetAccess();
        }
        else
        {
            System.out.println("No Internet access granted. Your job level is below 5");
        }
    }
    public int getRole(String emplName) {
        // Check role from the database based on Name and designation
        // return job level or job designation.
        return 9;
    }
}

```

### Step 4

Now, Create a **ProxyPatternClient** class that can access the internet actually.

File: *ProxyPatternClient.java*

```

public class ProxyPatternClient {
    public static void main(String[] args)
    {
        OfficeInternetAccess access = new ProxyInternetAccess("Ashwani Rajput");
        access.grantInternetAccess();
    }
}

```

[download this example](#)

### Output

```
No Internet access granted. Your job level is below 5
```













← prev

next →






## Help Others, Please Share







## Learn Latest Tutorials

 Arduino tutorial Arduino	 Digital Electronics tutorial Digital E.	 Google Adwords tutorial Adwords	 MySQL tutorial MySQL
 Python tutorial Python	 Smartsheet Smartsheet	 affiliate marketing Affiliate M.	 Software Testing Tutorial Testing
 Proc*C Proc*C	 social media marketing SMM	 GDB GDB	 Fuzzy Logic Fuzzy Logic








## Preparation

 Aptitude Aptitude	 Logical Reasoning Reasoning	 Verbal Ability Verbal A.	 Interview Questions Interview
 Company Interview Questions Company			


## Trending Technologies

 Artificial Intelligence Tutorial AI	 AWS Tutorial AWS	 Selenium tutorial Selenium	 Cloud tutorial Cloud
---	---	---	---



 Hadoop tutorial Hadoop	 ReactJS Tutorial ReactJS	 Data Science Tutorial D. Science	 Angular 7 Tutorial Angular 7
 Blockchain Tutorial Blockchain	 Git Tutorial Git	 Machine Learning Tutorial ML	 DevOps Tutorial DevOps

## B.Tech / MCA

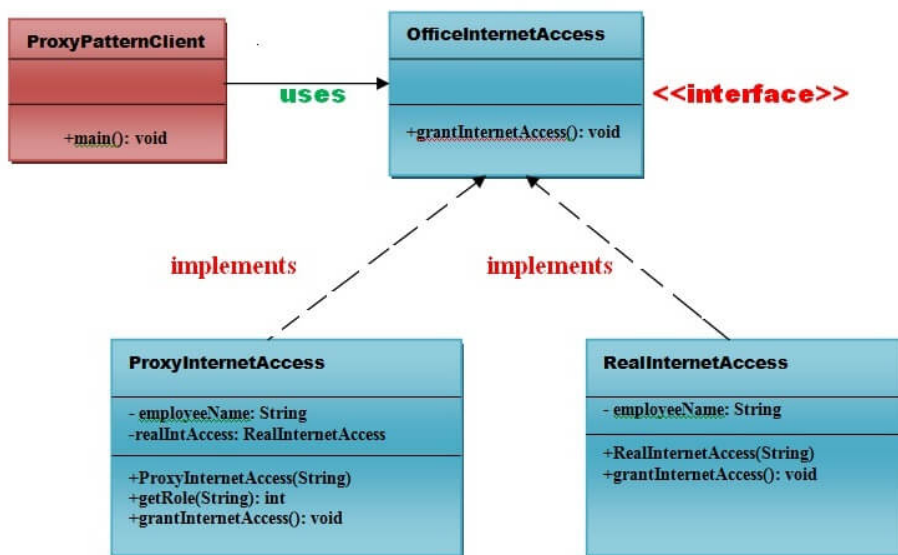
 DBMS tutorial DBMS	 Data Structures tutorial DS	 DAA tutorial DAA	 Operating System tutorial OS
 Computer Network tutorial C. Network	 Compiler Design tutorial Compiler D.	 Computer Organization and Architecture COA	 Discrete Mathematics Tutorial D. Math.
 Ethical Hacking Tutorial E. Hacking	 Computer Graphics Tutorial C. Graphics	 Software Engineering Tutorial Software E.	 html tutorial Web Tech.
 Cyber Security tutorial Cyber Sec.	 Automata Tutorial Automata	 C Language tutorial C	 C++ tutorial C++
 Java tutorial Java	 .Net Framework tutorial .Net	 Python tutorial Python	 List of Programs Programs
 Control Systems tutorial Control S.	 Data Mining Tutorial Data Mining		



## Example of Proxy Pattern

Let's understand the example of proxy design pattern by the above UML diagram.

UML for Proxy Pattern:



Implementation of above UML:

### Step 1

Create an **OfficeInternetAccess** interface.

```
public interface OfficeInternetAccess {
    public void grantInternetAccess();
}
```

### Step 2

Create a **RealInternetAccess** class that will implement **OfficeInternetAccess** interface for granting the permission to the specific employee.

File: *RealInternetAccess.java*

```
public class RealInternetAccess implements OfficeInternetAccess {
    private String employeeName;
    public RealInternetAccess(String empName) {
        this.employeeName = empName;
    }
    @Override
```