Template Method in Java

Template Method is a behavioral design pattern that allows you to define a skeleton of an algorithm in a base class and let subclasses override the steps without changing the overall algorithm’s structure.

Learn more about Template Method

Complexity:

Popularity:

Usage examples: The Template Method pattern is quite common in Java frameworks. Developers often use it to provide framework users with a simple means of extending standard functionality using inheritance.

Here are some examples of Template Methods in core Java libraries:

- All non-abstract methods of `java.io.InputStream`, `java.io.OutputStream`, `java.io.Reader` and `java.io.Writer`.
- All non-abstract methods of `java.util.AbstractList`, `java.util.AbstractSet` and `java.util.AbstractMap`.
- In `javax.servlet.http.HttpServlet` class, all the `doXXX()` methods send the HTTP 405 “Method Not Allowed” error by default. However, you can override any of those methods to send a different response.

Identification: Template Method can be recognized if you see a method in base class that calls a bunch of other methods that are either abstract or empty.

Overriding standard steps of an algorithm

In this example, the Template Method pattern defines an algorithm of working with a social network. Subclasses that match a particular social network, implement these steps according to the API provided by the social network.

networks

networks/Network.java: Base social network class

```java
package refactoring_guru.template_method.example.networks;

/**
 * Base class of social network.
 */
public abstract class Network {
    String userName;
    String password;

    Network() {}

    /**
     * Publish the data to whatever network.
     */
    public boolean post(String message) {
        // Authenticate before posting. Every network uses a different
        // authentication method.
        if (logIn(this.userName, this.password)) {
            // Send the post data.
            boolean result =  sendData(message.getBytes());
            logOut();
            return result;
        }
        return false;
    }

    abstract boolean logIn(String userName, String password);
    abstract boolean sendData(byte[] data);
    abstract void logOut();
}
```

networks/Facebook.java: Concrete social network

```java
package refactoring_guru.template_method.example.networks;

/**
 * Class of social network
 */
public class Facebook extends Network {
    public Facebook(String userName, String password) {
        this.userName = userName;
        this.password = password;
    }

    public boolean logIn(String userName, String password) {
        System.out.println("\nChecking user's parameters");
        System.out.println("Name: " + this.userName);
        System.out.print("Password: ");
        for (int i = 0; i < this.password.length(); i++) {
            System.out.print("*");
        }
        simulateNetworkLatency();
        System.out.println("\n\nLogIn success on Facebook");
        return true;
    }

    public boolean sendData(byte[] data) {
        boolean messagePosted = true;
        if (messagePosted) {
            System.out.println("Message: '" + new String(data) + "' was posted on Facebook");
            return true;
        } else {
            return false;
        }
    }

    public void logOut() {
        System.out.println("User: '" + userName + "' was logged out from Facebook");
    }

    private void simulateNetworkLatency() {
        try {
            int i = 0;
            System.out.println();
            while (i < 10) {
                System.out.print(".");
                Thread.sleep(500);
                i++;
            }
        } catch (InterruptedException ex) {
            ex.printStackTrace();
        }
    }
}
```

networks/Twitter.java: One more social network

```java
package refactoring_guru.template_method.example.networks;

/**
 * Class of social network
 */
public class Twitter extends Network {

    public Twitter(String userName, String password) {
        this.userName = userName;
        this.password = password;
    }

    public boolean logIn(String userName, String password) {
        System.out.println("\nChecking user's parameters");
        System.out.println("Name: " + this.userName);
        System.out.print("Password: ");
        for (int i = 0; i < this.password.length(); i++) {
            System.out.print("*");
        }
        simulateNetworkLatency();
        System.out.println("\n\nLogIn success on Twitter");
        return true;
    }

    public boolean sendData(byte[] data) {
        boolean messagePosted = true;
        if (messagePosted) {
            System.out.println("Message: '" + new String(data) + "' was posted on Twitter");
            return true;
        } else {
            return false;
        }
    }

    public void logOut() {
        System.out.println("User: '" + userName + "' was logged out from Twitter");
    }

    private void simulateNetworkLatency() {
        try {
            int i = 0;
            System.out.println();
            while (i < 10) {
                System.out.print(".");
                Thread.sleep(500);
                i++;
            }
        } catch (InterruptedException ex) {
            ex.printStackTrace();
        }
    }
}
```

Demo.java: Client code

```java
package refactoring_guru.template_method.example;

import refactoring_guru.template_method.example.networks.Facebook;
import refactoring_guru.template_method.example.networks.Network;
import refactoring_guru.template_method.example.networks.Twitter;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

/**
 * Demo class. Everything comes together here.
 */
public class Demo {
    public static void main(String[] args) throws IOException {
        BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
        Network network = null;
        System.out.print("Input user name: ");
        String userName = reader.readLine();
        System.out.print("Input password: ");
        String password = reader.readLine();

        // Enter the message.
        System.out.print("Input message: ");
        String message = reader.readLine();

        System.out.println("\nChoose social network for posting message.\n" +
                "1 - Facebook\n" +
                "2 - Twitter");
        int choice = Integer.parseInt(reader.readLine());

        // Create proper network object and send the message.
        if (choice == 1) {
            network = new Facebook(userName, password);
        } else if (choice == 2) {
            network = new Twitter(userName, password);
        }
        network.post(message);
    }
}
```

OutputDemo.txt: Execution result

```
Input user name: Jhonatan
Input password: qswe
Input message: Hello, World!

Choose social network for posting message.
1 - Facebook
2 - Twitter
2

Checking user's parameters
Name: Jhonatan
Password: ****
..........

LogIn success on Twitter
Message: 'Hello, World!' was posted on Twitter
User: 'Jhonatan' was logged out from Twitter
```