Architectural Pattern

주어진 상황에서 소프트웨어 아키텍처에서 흔히 발생하는 문제에 대한 일반적으로 재사용 가능한 해법.

https://en.wikipedia.org/wiki/Architectural_pattern

Architecture, Modeling, DesignPattern, ArchitecturalPattern

Model-View-Presenter

Presenter가 중개인이자 데이터 가공 역할. View와 Presenter간의 위임으로 View에서 데이터 처리 요청 및 그에 대한 Callback을 받는다. 또한 Model에 접근할 수 있는건 오직 Presenter.

Interface

```
package api;

public interface BaseMVPContract {
    interface View {}

    interface Presenter<V extends BaseMVPContract.View> {
        void setView(V view);
    }
}
```

```
package api.concrete;

import api.BaseMVPContract;

public interface TaskMVPContract {
    interface View extends BaseMVPContract.View {
        void onResponse(String message);
    }

    interface Presenter extends
BaseMVPContract.Presenter<TaskMVPContract.View> {
        void request(String message);
    }
}
```

ConcreteView

```
package view;

import api.concrete.TaskMVPContract;
import presenter.TaskPresenter;

public class TaskView implements TaskMVPContract.View {

    private TaskPresenter presenter;
    public TaskView() {
        presenter = new TaskPresenter();
        presenter.setView(this);
    }
    public void requestData(String message) {
        presenter.request(message);
    }
    @Override
    public void onResponse(String message) {
        System.out.println(message);
    }
}
```

ConcretePresenter

```
package presenter;

import api.concrete.TaskMVPContract;
import api.concrete.TaskMVPContract.View;

public class TaskPresenter implements TaskMVPContract.Presenter {

    private TaskMVPContract.View view;
    @Override
    public void setView(View view) {
        this.view = view;
    }

    @Override
    public void request(String message) {
        StringBuilder sb = new StringBuilder();
        sb.append("+++");
        sb.append(message);
        sb.append("+++");
        view.onResponse(sb.toString());
    }

}
```

Client

만약 GUI 환경이라면 사용자 입력에 대한 동작.

```
import view.TaskView;

public class Client {
    public static void main(String[] args) {
        TaskView v = new TaskView();
        v.requestData("Hello world");
    }
}

/*
+++Hello world+++
*/
```

From:
https://ledyx.xyz/ - **Ledyx WIKI**

Permanent link:
https://ledyx.xyz/doku.php?id=architectural_pattern:architectural_pattern&rev=1509434898

Last update: **2026/10/24 07:03**