**MÔ HÌNH MVC (MODEL - VIEW - CONTROLLER)**

**I. Giới thiệu mô hình MVC**

**MVC** là viết tắt của **Model - View - Controller**, một **mô hình kiến trúc phần mềm** được sử dụng phổ biến trong quá trình phát triển các ứng dụng có giao diện người dùng. Mục đích chính của mô hình MVC là **tách riêng ba thành phần chính** của ứng dụng: **xử lý dữ liệu (Model), giao diện người dùng (View), và điều khiển luồng xử lý (Controller)**.

**II. Các đặc điểm nổi bật của mô hình MVC**

1. **Phân tách rõ ràng trách nhiệm**:
   * Mỗi thành phần có nhiệm vụ riêng biệt, giúp mã nguồn dễ bảo trì và mở rộng.
2. **Tái sử dụng cao**:
   * View có thể thay đổi mà không ảnh hưởng đến Model.
   * Model có thể tái sử dụng trong các View khác nhau.
3. **Dễ kiểm thử (test)**:
   * Từng phần có thể kiểm thử độc lập nhờ tách biệt chức năng.
4. **Hỗ trợ phát triển đồng thời**:
   * Frontend và backend có thể phát triển song song mà ít bị phụ thuộc.
5. **Phù hợp với ứng dụng lớn**:
   * Mô hình này là nền tảng cho nhiều framework mạnh mẽ như Spring MVC, Laravel, ASP.NET MVC, Django...

**III. Cấu trúc mô hình MVC**

**1. Model (M) – Mô hình dữ liệu**

* Là phần **chứa dữ liệu** và **logic nghiệp vụ** của ứng dụng.
* Quản lý dữ liệu, kết nối với cơ sở dữ liệu, thực hiện các thao tác thêm, sửa, xóa, tìm kiếm...
* Không liên quan đến giao diện hoặc điều khiển.

**Ví dụ:**

java

Sao chépChỉnh sửa

public class SinhVien {

private String maSV;

private String hoTen;

private float diemTB;

// Getter, Setter, và các phương thức xử lý dữ liệu

}

**2. View (V) – Giao diện hiển thị**

* Là phần **hiển thị dữ liệu** cho người dùng.
* Không chứa logic xử lý hay dữ liệu – chỉ trình bày thông tin được cung cấp bởi Model thông qua Controller.
* Thường là các thành phần HTML, JSP, XML, GUI...

**Ví dụ:**

html

Sao chépChỉnh sửa

<h1>Danh sách sinh viên</h1>

<table>

<tr><td>Họ tên:</td><td>${sinhvien.hoTen}</td></tr>

</table>

**3. Controller (C) – Bộ điều khiển**

* Là **cầu nối giữa View và Model**.
* Nhận sự kiện từ người dùng (click, nhập liệu...), gọi các phương thức xử lý trong Model, sau đó cập nhật View.
* Điều phối luồng hoạt động của toàn bộ ứng dụng.

**Ví dụ:**

java

Sao chépChỉnh sửa

public class SinhVienController {

public void luuSinhVien(HttpServletRequest req) {

String ten = req.getParameter("ten");

SinhVien sv = new SinhVien();

sv.setHoTen(ten);

sinhVienDAO.luu(sv);

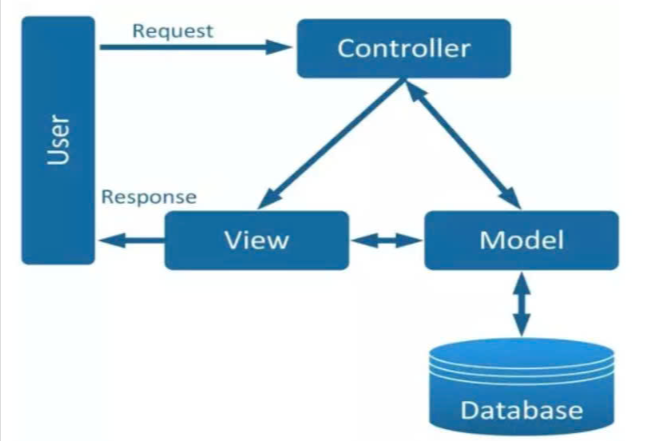
}

}

**IV. Nguyên lý hoạt động của mô hình MVC**

1. Người dùng tương tác với **giao diện View** (ví dụ: nhập thông tin, bấm nút).
2. **View** gửi các sự kiện (yêu cầu) đến **Controller**.
3. **Controller** xử lý sự kiện, thực hiện các thao tác với **Model**.
4. **Model** xử lý dữ liệu (đọc/ghi từ cơ sở dữ liệu, xử lý logic) và trả về kết quả.
5. **Controller** nhận dữ liệu từ Model và chuyển lại cho **View** để hiển thị kết quả lên giao diện.

**V. Sơ đồ minh họa mô hình MVC**



**VI. Ưu điểm của mô hình MVC**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Ưu điểm | Mô tả | | Tổ chức mã nguồn rõ ràng | Dễ đọc, dễ hiểu, tách biệt từng vai trò. | | Dễ bảo trì và mở rộng | Chỉnh sửa một phần không ảnh hưởng đến phần khác. | | Tái sử dụng cao | Có thể tái sử dụng View hoặc Model cho các mục đích khác nhau. | | Hỗ trợ làm việc nhóm | Backend và frontend có thể phát triển song song. | | Dễ kiểm thử | Mỗi phần có thể test riêng biệt. | |

**VII. Nhược điểm của mô hình MVC**

|  |  |
| --- | --- |
| Nhược điểm | Mô tả |
| Cấu trúc ban đầu phức tạp | Không phù hợp với ứng dụng nhỏ hoặc đơn giản. |
| Tốn thời gian phát triển ban đầu | Do phải tạo nhiều lớp (Model, View, Controller). |
| Dễ gây rối khi không tổ chức tốt | Nếu không tuân thủ đúng nguyên tắc, mã nguồn sẽ lộn xộn. |

**VIII. Ứng dụng của mô hình MVC trong thực tế**

Mô hình MVC là nền tảng của rất nhiều framework hiện đại như:

|  |  |
| --- | --- |
| Ngôn ngữ | Framework MVC phổ biến |
| Java | Spring MVC, JSF |
| PHP | Laravel, CodeIgniter |
| Python | Django |
| JavaScript | Angular (MVVM), React (một phần), Vue |
| C# (.NET) | ASP.NET MVC |

**IX. Ví dụ thực tế: Trang quản lý sinh viên**

* **Model**: SinhVien.java chứa thông tin sinh viên.
* **View**: sinhvien.jsp hiển thị danh sách sinh viên.
* **Controller**: SinhVienController.java xử lý thêm, sửa, xóa sinh viên

**X. Kết luận**

Mô hình MVC là một kiến trúc thiết kế mạnh mẽ, giúp ứng dụng có tổ chức rõ ràng, dễ phát triển và mở rộng. Việc hiểu và áp dụng tốt mô hình MVC sẽ giúp lập trình viên xây dựng được những ứng dụng phần mềm có chất lượng cao, dễ dàng bảo trì trong dài hạn.