HCI – MVC
Model-View-Controller

Phạm Thị Ngọc Diễm
Khoa CNTT&TT – ĐHCT

2016-2017

Nội dung

- Giới thiệu MVC
- Model
- View
- Controller
- Ví dụ
- MVC mở rộng

Giới thiệu
Mẫu thiết kế

- Định nghĩa (wikipedia): *một mẫu thiết kế (Design Pattern) là một giải pháp tổng thể có thể tái sử dụng cho các vấn đề chung trong thiết kế phần mềm*.
-
- Một mẫu thiết kế :
  - Không phải là một thiết kế hoàn thiện có thể được chuyển đổi trực tiếp thành mã;
  - Là một mô tả hay template mô tả các giải pháp cho các vấn đề về kiến trúc hay thiết kế phần mềm, có thể tái sử dụng trong nhiều tình huống khác nhau.

Giới thiệu
Mẫu thiết kế

- Ví dụ:
  - Các mẫu thiết kế hướng đối tượng :
    - Cho thấy mối quan hệ và sự tương tác giữa các lớp hay các đối tượng,
    - Không cần chỉ rõ các lớp hay đối tượng của từng ứng dụng cụ thể.
  - Các giải thuật không được xem là các mẫu thiết kế, vì
    - Chúng giải quyết các vấn đề về tính toán
    - Không giải quyết các vấn đề về thiết kế.

Giới thiệu - MVC

- MVC là một *mẫu thiết kế* dùng trong thiết kế giao diện người dùng, nó cho phép tách riêng
  - Mô hình (nghiệp vụ logic, truy cập dữ liệu, …)
  - Khung nhìn (giao diện người dùng : trình bày dữ liệu và giao diện nhập dữ liệu cho người sử dụng).
  - => *Thay đổi thành phần này sẽ không ảnh hưởng thành phần kia → thuận lợi cho việc bảo trì*
- Được giới thiệu vào năm 1979 bởi Trygve Reenskaug

Giới thiệu - MVC

- Gắn liền với các khái niệm hướng đối tượng (dùng đầu tiên trong ngôn ngữ Smalltalk)

- Sử dụng mẫu thiết kế này :
  - Có xu hướng tăng số lượng các class cần định nghĩa và;
  - Có thể tăng gánh nặng cho việc thiết kế ứng dụng
  - Nhưng việc tách này đảm bảo dễ dàng bảo trì.

Giới thiệu - MVC

- Ba thành phần chính của mô hình MVC
  - Model : dữ liệu và các xử lý dữ liệu
  - View : biểu diễn dữ liệu
  - Controller : hành vi của ứng dụng

Model

Lõi chức năng của ứng dụng

- Đại diện cho :
  - Dữ liệu của ứng dụng,
  - Các xử lý dữ liệu,
  - Các quy tắc truy xuất và cập nhật các dữ liệu này,
  - Đối tượng mô tả dữ liệu như các Class, hàm xử lý, ...
-
- Model không bao gồm bất kỳ thành phần hiển thị/trình bày dữ liệu

View

Đầu ra của ứng dụng

- Đảm nhận việc hiển thị thông tin, tương tác với người dùng, nơi chứa tất cả các đối tượng GUI như textbox, images.
- Tương tác với Model
- View liên quan chủ yếu đến việc biểu diễn các dữ liệu của Model trên màn hình (hoặc trên các thiết bị đầu ra khác)
- Có thể có nhiều view cho cùng Model
- Không có bất kỳ thành phần xử lý nào khác trong view

Controller

Hành vi và quản lý dữ liệu vào của ứng dụng

- Quản lý tương tác với người dùng
- Dịch các hành động của người dùng thành các hành động trên Model
- Cung cấp view thích hợp đối với các hành động người dùng và các phản ứng của Model
- Controller có thể :
  - Gọi các phương thức của Model để phản ứng với các sự kiện (vd : yêu cầu thêm khách hàng)
  - Làm thay đổi view về khía cạnh hiển thị
  - Khởi tạo một view mới
- Có thể có nhiều Controller cho cùng view nhưng chỉ một Model.

Ưu điểm MVC

- Thể hiện tính chuyên nghiệp trong lập trình, phân tích thiết kế.
- Độc lập giữa 'dữ liệu', 'trình bày' và hành vi
  - Model cung cấp dữ liệu cho view
  - View là các thành phần đồ họa
  - → Đơn giản, dễ nâng cấp, bảo trì...
- Module hóa và có thể tái sử dụng
- Tránh được việc lưu dữ liệu nhiều nơi
- Khả năng Model thông báo cho view những thay đổi (vd thêm 1 dòng vào bảng)

Hạn chế MVC

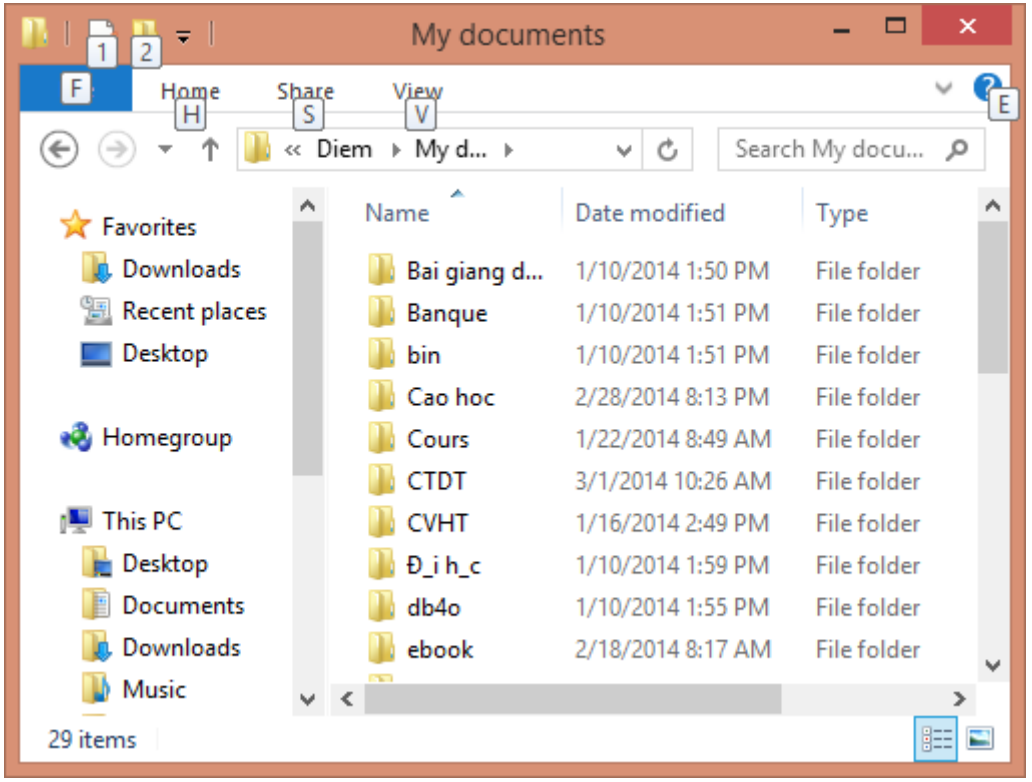
- Ít thích nghi với các ứng dụng web
- Tách riêng các công việc:
  - Tăng độ phức tạp vào lúc cài đặt
  - Tùy tình huống có thể phải *'phân vùng'* những người phát triển (lập trình viên)
  - Controller và view thường vẫn liên kết chặt chẽ với Model
- Đối với dự án nhỏ
  - Gây cồng kềnh,
  - Tốn thời gian trong quá trình phát triển.

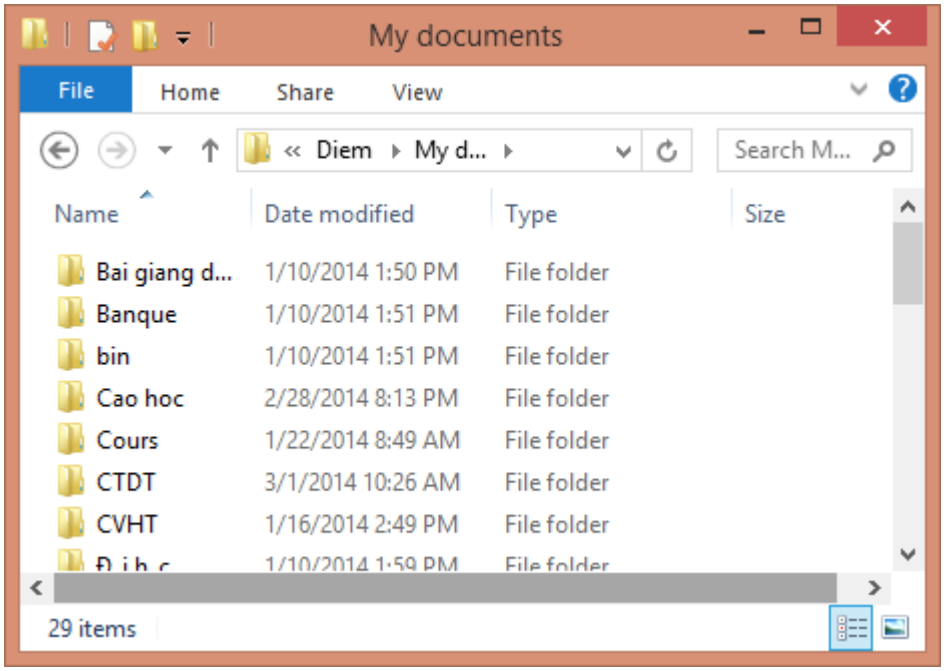
Ví dụ

- Hệ thống tập tin của hệ điều hành
  - View: Nhiều view khác nhau cho bởi HĐH trên hệ thống tập tin của nó
  - Controller: có thể nhận thông qua giao diện người dùng
    - Các sự kiện cập nhật Model (xóa tập tin, thêm thư mục,...) và chuyển chúng đến Model
      - Các view lắng nghe Model sẽ được thông báo các sửa đổi của Model
      - Chúng sẽ truy vấn Model để cập nhật việc hiển thị
    - Sự kiện sửa đổi hiển thị dữ liệu (hiển thị chi tiết danh sách tập tin, thư mục) sẽ được truyền đến view để thay đổi cách hiển thị không cần mà không cần truy vấn lại Model

Ví dụ

- Có thể có nhiều view trên cùng Model





Cài đặt cơ bản MVC trong Swing

- Mapping các class tới các thành phần MVC
  - View là một widget Swing như JFrame, JButton, …
  - Controller là một ActionListener
  - Model là một class Java thông thường (hoặc CSDL)

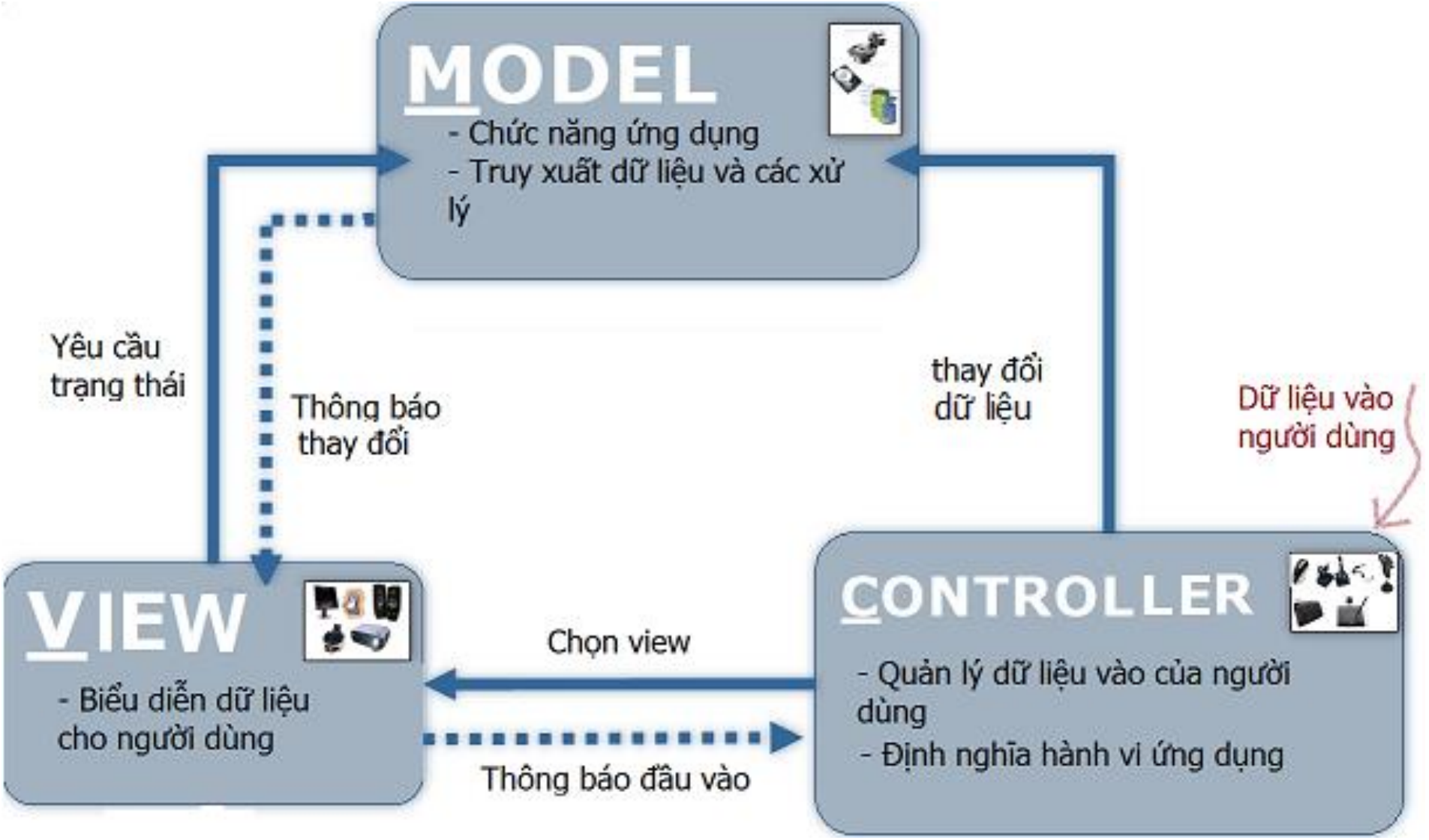
Cơ chế cơ bản MVC

- Cài đặt (Setup)
  - Khởi tạo Model
  - Khởi tạo View
    - Có tham chiếu đến Controller, khởi tạo null
  - Khởi tạo Controller với tham chiếu đến cả hai
    - Controller đăng ký với view, vì vậy lúc này view có một tham chiếu không null đến controller
- Thực thi (Execution)
  - View nhận dạng sự kiện
  - View gọi phương thức thích hợp trong controller
  - Controller truy xuất tới model, có thể cập nhật nó
  - Nếu model được thay đổi, view được cập nhật (nhờ vào controller)

Bổ sung MVC – Các dịch vụ tối thiểu

- Tách riêng lõi chức năng khỏi giao diện là một điều tốt...
- Nhưng lõi chức năng phải cung cấp các dịch vụ tối thiểu để có thể cài đặt đúng giao diện
- **Ví dụ**:
  - *Thông báo*: Khả năng để một module bên ngoài được thông báo khi trạng thái của Model thay đổi
  - *Ngăn ngừa lỗi* : khả năng để biết nếu một lời gọi hàm là hợp lệ không trong một vài ngữ cảnh
  - *Hủy bỏ*: khả năng để quay trở lại các trạng thái trước đó.

MVC – Mở rộng


MODEL
- Chức năng ứng dụng
- Truy xuất dữ liệu và các xử lý
Yêu cầu trạng thái
Thông báo thay đổi
thay đổi dữ liệu
Dữ liệu vào người dùng
VIEW
- Biểu diễn dữ liệu cho người dùng
Chọn view
Thông báo đầu vào
CONTROLLER
- Quản lý dữ liệu vào của người dùng
- Định nghĩa hành vi ứng dụng

MVC – Mở rộng

- Mẫu quan sát (Observer Pattern)
  - Một đối tượng được thông báo về sự thay đổi của các đối tượng khác.
  - Trong MVC mở rộng, view được thông báo về sự thay đổi của Model, view là observer của Model.
- Ứng dụng trong MVC
  - Cập nhật Model không đồng bộ
    - Model thay đổi độc lập với các hành động người dùng
    - View được kết hợp phải được thông báo về các thay đổi để biết rằng nó phải cập nhật
  - Một Model có nhiều view
    - Mỗi view chỉ cho một Model
    - Tất cả các view phải được cập nhật khi Model thay đổi

Cơ chế mở rộng MVC

- Cài đặt (Setup)
  - Khởi tạo Model
    - Có tham chiếu tới View, khởi tạo null
  - Khởi tạo View với tham chiếu tới model
    - View đăng ký với model
  - Khởi tạo Controller với tham chiếu đến cả hai
    - Controller đăng ký với view
- Thực thi (Execution)
  - View nhận dạng sự kiện
  - View gọi phương thức thích hợp trong controller
  - Controller truy xuất tới model, có thể cập nhật nó
  - Nếu model được thay đổi, nó thông báo với tất cả view đã đăng ký với nó
  - View truy vấn mô hình, trả về thông tin mới thích hợp

MVC trong Zend Framework

- Cài đặt (Setup)
  - Khởi tạo Model
    - Có tham chiếu tới View, khởi tạo null
  - Khởi tạo View với tham chiếu tới model
    - View đăng ký với model
  - Khởi tạo Controller với tham chiếu đến cả hai
    - Controller đăng ký với view
- Thực thi (Execution)
  - View nhận dạng sự kiện
  - View gọi phương thức thích hợp trong controller
  - Controller truy xuất tới model, có thể cập nhật nó
  - Nếu model được thay đổi, nó thông báo với tất cả view đã đăng ký với nó
  - View truy vấn mô hình, trả về thông tin mới thích hợp

Tóm tắt

- Mẫu thiết kế MVC cho phép tách riêng phần nghiệp vụ phần mềm khỏi phần giao diện đồ họa.

- Model-View-Controller
  - Model bao gồm dữ liệu (data)
  - View hiển thị model tới người dùng (presentation)
  - Controller sửa đổi model (business logic)