ĐẠI HỌC CÔNG NGHIỆP TP.HCM
Khoa Công nghệ Thông tin

THIẾT KẾ THÀNH PHẦN TRÌNH BÀY

(Designing Presentation Components)

TS. Huỳnh Hữu Nghĩa

huynhhuunghia@iuh.edu.vn

Nội dung:

- Khái quát
- Bước 1: Hiểu các yêu cầu UI
- Bước 2: Xác định kiểu UI được yêu cầu
- Bước 3: Chọn lựa công nghệ UI
- Bước 4: Thiết kế thành phần trình bày
- Bước 5: Xác định các yêu cầu liên kết
- Bước 6: Xác định chiến lược xử lý lỗi
- Bước 7: Xác định chiến lược xác thực đầu vào

Khái quát

- Thiết kế các thành phần *giao diện người dùng* (UI) và các thành phần logic trình bày là một phần của lớp trình bày.

- Đầu tiên nên hiểu những yêu cầu đối với UI và có thể lựa chọn công nghệ phù hợp, sau đó quyết định cách liên kết logic trình bày và dữ liệu với các điều khiển UI.

- Hiểu những yêu cầu xử lý lỗi và xác nhận bên trong UI.

Bước 1: Hiểu các yêu cầu UI

- Hiểu yêu cầu UI là chìa khóa ra quyết định loại UI, công nghệ và điều khiển để triển khai UI.

- Xác định người dùng ứng dụng, hiểu mục tiêu và công việc mà người dùng muốn trong ứng dụng.

- Có thể quyết định nghiên cứu thực tế để giúp hiểu môi trường tương tác ứng dụng.

Bước 1: Hiểu các yêu cầu UI

- Xem xét kiểu dữ liệu, định dạng trình bày cho dữ liệu như: *kiểu ngày*, *thời gian* và *tiền tệ*.

- Xác định các yêu cầu cá nhân hóa của ứng dụng như cho người dùng thay đổi layout hay styles ở thời gian chạy.

- Xác định cách người dùng tìm kiếm thông tin nhanh và dễ dàng thông qua điều khiển, chức năng tìm kiếm, sitemaps, và các chức năng khác phù hợp, ...

Bước 2: Xác định loại UI yêu cầu

- Dựa trên yêu cầu UI, quyết định loại UI cho ứng dụng. Có một số loại UI khác nhau, mỗi loại có điểm mạnh và điểm yếu.

- Nên tìm những yêu cầu UI có thể đáp ứng nhiều loại UI. Cũng có thể không có loại UI duy nhất nào đáp ứng hoàn toàn các yêu cầu UI.

Bước 2: Xác định loại UI yêu cầu

- **Mobile applications** có thể được phát triển như các ứng dụng *thin client* hay *rich client*. Ứng dụng di động *rich client* có thể hỗ trợ kịch bản ngắt kết nối hay đôi khi được kết nối. Còn *thin client* hay Web chỉ hỗ trợ kịch bản kết nối. Nguồn tài nguyên thiết bị cũng là một ràng buộc khi thiết ứng dụng di động.

Bước 2: Xác định loại UI yêu cầu

- **Rich client applications** là những ứng dụng độc lập hay nối mạng với giao diện người dùng đồ họa, hiển thị dữ liệu bởi một loạt các điều khiển, và được triển khai trên máy tính bàn hoặc laptop.

- **Rich Internet applications (RIAs)** là ứng dụng Web với giao diện đồ họa phong phú chạy trên trình duyệt. RIAs thường sử dụng cho kịch bản kết nối.

Bước 2: Xác định loại UI yêu cầu

- **Web applications** hỗ trợ kịch bản kết nối và chạy trên nhiều trình duyệt khác nhau. Web UI là một lựa chọn tốt khi UI phải dựa trên các tiêu chuẩn, và truy cập trên phạm vi rộng các thiết bị và nhiều nền tảng, rất phù hợp với ứng dụng có nội dung tìm kiếm bởi công cụ tìm kiếm Web.

- **Console-based applications** cung cấp môt trải nghiệm người dùng chỉ văn bản thay đổi, và thường chạy trong cửa sổ lệnh. Phù hợp nhất cho công việc phát triển và quản trị.

Bước 3: Chọn công nghệ UI

- Sau khi xác định kiểu UI, tiếp theo là lựa chọn công nghệ phù hợp. Việc lựa chọn công nghệ phụ thuộc vào kiểu UI đã chọn:

Bước 3: Chọn công nghệ UI

- Giao diện người dùng **Mobile client:**
  - ✓ Microsoft .NET Compact Framework.
  - ✓ ASP.NET for Mobile.
  - ✓ Silverlight for Mobile.

Bước 3: Chọn công nghệ UI

- Giao diện người dùng **Rich client:**
  - ✓ Windows Presentation Foundation (WPF).
  - ✓ Windows Forms.
  - ✓ Windows Forms with WPF User Controls.
  - ✓ WPF with Windows Forms User Controls.
  - ✓ XAML Browser Application (XBAP) using WPF.

Bước 3: Chọn công nghệ UI

- Giao diện người dùng **Rich Internet application:**
  - ✓ Silverlight.
  - ✓ Silverlight with AJAX.

- Giao diện người dùng **Console-based:**
  - ✓ Console Applications.
  - ✓ Power Shell Commandlets.

Bước 3: Chọn công nghệ UI

- Giao diện người dùng **Web application:**
  - ✓ ASP.NET Web Forms.
  - ✓ ASP.NET Web Forms with AJAX.
  - ✓ ASP.NET Web Form with Silverlight Controls.
  - ✓ ASP.NET MVC.
  - ✓ ASP.NET Dynamic Data.

Bước 4: Thiết kế thành phần trình bày

- Các thành phần trình bày:
  - ✓ User Interface Components.
  - ✓ Presentation Logic Components.
  - ✓ Presentation Model Components.

User Interface Components

- Thành phần UI là những yếu tố trực quan hiển thị thông tin đến và nhận đầu vào người dùng (chế độ Views).
- Một số hướng dẫn:
  - ✓ *Chia các trang hay cửa sổ thành những kiểm soát người dùng riêng biệt nhằm giảm thiểu độ phức tạp hay cho phép tái sử dụng*
  - ✓ *Tránh phân cấp kế thừa các kiểm soát và trang người dùng để có thể tái sử dụng mã bằng cách tạo ra các thành phần logic trình bày có thể tái sử dụng.*
  - ✓ *Tránh tạo ra các kiểm soát tùy chỉnh không cần thiết đối với hiển thị hay thu thập dữ liệu chuyên dụng*.

Presentation Logic Components

- Xử lý những khía cạnh không hiển thị của UI bao gồm *xác nhận, phản hồi các hoạt động người dùng, giao tiếp giữa các thành phần UI, và phối hợp tương tác người dùng*.

- Không phải lúc nào cũng cần thiết; chỉ khi nào thực hiện xử lý đáng kể trong lớp trình bày hoặc cải thiện cơ hội việc kiểm tra đơn vị logic trình bày.

Presentation Logic Components

- Hướng dẫn thiết kế:
  - *Nếu UI yêu cầu xử lý phức tạp hay phải giao tiếp với các lớp khác.*
  - *Khi cần lưu trữ trạng thái liên quan đến UI.*
  - *Sử dụng logic trình bày giúp ứng dụng phục hồi từ một thất bại hay lỗi để đảm bảo sau khi khôi phục giao diện người dùng ở trạng thái nhất quán.*
  - *Khi UI yêu cầu hỗ trợ quy trình công việc phức tạp.*

Presentation Model Components

- Biểu diễn dữ liệu từ lớp nghiệp vụ theo một định dạng có thể tiêu dùng cho UI và thành phần logic trình bày.
- Nếu mô hình được gói gọn trong logic nghiệp vụ, nó thường được gọi là thực thể trình bày.
- Mô hình có thể tổng hợp dữ liệu từ nhiều nguồn, chuyển đổi dữ liệu cho UI để hiển thị dễ dàng, hiện thực logic xác thực, và trình bày logic nghiệp vụ và trạng thái bên trong lớp trình bày.
- Các mẫu thường sử dụng như MVP hay MVC.

Presentation Model Components

- Hướng dẫn khi thiết kế:
  - *Nếu dữ liệu hay định dạng được hiển thị là dành riêng cho lớp trình bày, đang sử dụng mẫu MVP hay MVC.*
  - *Khi muốn kiểm soát ràng buộc dữ liệu liên quan đến kiểm soát UI.*
  - *Khi thực hiện xác thực dữ liệu trong lớp trình bày.*
  - *Khi có yêu cầu tuần tự hoá cho dữ liệu sẽ chuyển đến các thành phần mô hình trình bày nếu dữ liệu này được chuyển thông qua mạng hay lưu trữ trên đĩa ở client.*

Presentation Model Components

- Cần chọn một kiểu dữ liệu phù hợp cho thành phần mô hình trình bày và các thực thể trình bày *(do yêu cầu ứng dụng)*

- Đầu tiên, chọn một định dạng dữ liệu cho lớp trình bày.
- Tiếp theo, quyết định cách biểu diễn dữ liệu trong UI.

- Một số định dạng phổ biến để trình bày dữ liệu như: Custom class, Array và Collection, DataSet và DataTable, Typed DataSet, XML, và DataReader.

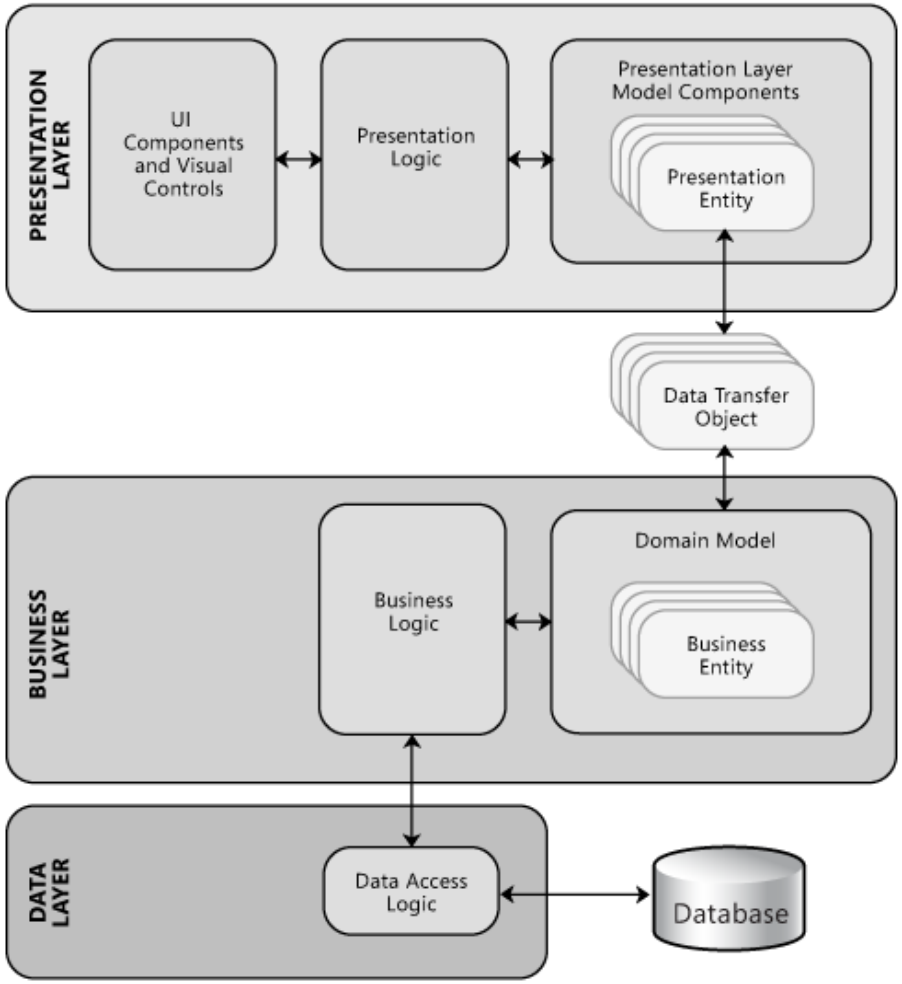
Presentation Entities

- Thành phần mô hình trình bày nên đóng gói cả dữ liệu từ lớp nghiệp vụ, và logic và hành vi nghiệp vụ → *đảm bảo tính hợp lý và nhất quán dữ liệu trong lớp trình bày và giúp cải thiện trải nghiệm người dùng.*

- Thành phần mô hình trình bày có thể là những thực thể nghiệp vụ từ lớp nghiệp vụ, được tiêu thụ trực tiếp bởi lớp trình bày hoặc được thiết kế đặc biệt để hỗ trợ lớp trình bày.

- Các thành phần như vậy được gọi là *thực thể trình bày (Presentation Entity)*.

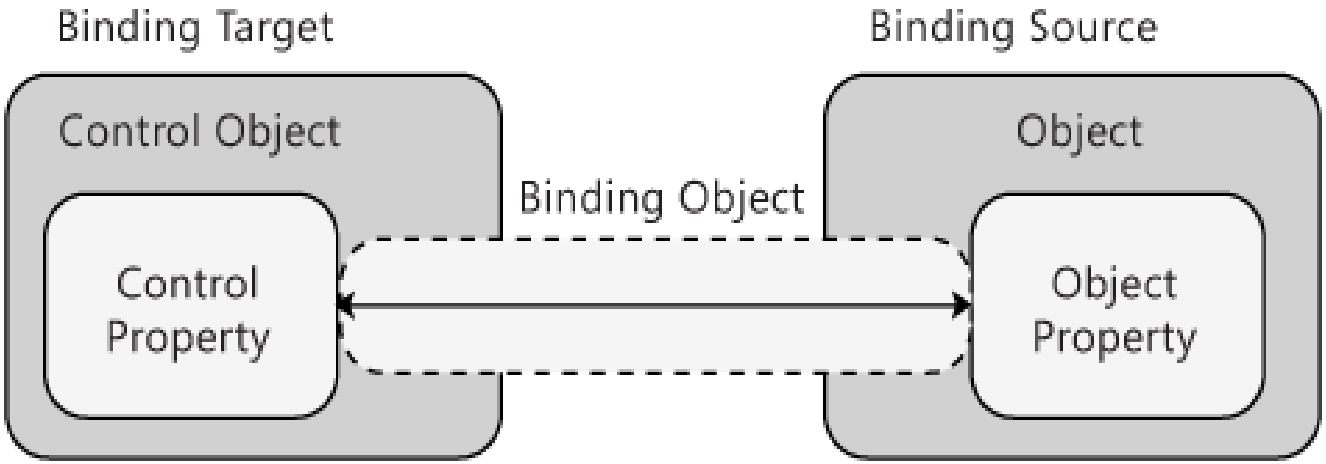
Presentation Entities



Thực thể trình bày
có thể lưu trữ dữ liệu
theo một định dạng
dễ sử dụng cho
thành phần logic
trình bày và UI.

Bước 5: Xác định yêu cầu liên kết

- Liên kết dữ liệu là cách liên kết giữa các kiểm soát trên giao diện người dùng và các thành phần dữ liệu hay logic ứng dụng cho phép hiển thị và tương tác với CSDL cũng như các cấu trúc khác (arrays và collection).



Bước 5: Xác định yêu cầu liên kết

- Có 2 loại liên kết phổ biến:
  - ✓ Liên kết 1 chiều.
  - ✓ Liên kết 2 chiều.

- Các công nghệ hỗ trợ liên kết 2 chiều như: IBindingList (Windows Forms), INotifyPropertyChanged (WPF) và ASP.NET

Bước 6: Xác định chiến lược xử lý lỗi

- Các lựa chọn khi thiết kế chiến lược xử lý lỗi:
  - ✓ Thiết kế một chiến lược xử lý ngoại lệ tập trung.
  - ✓ Các ngoại lệ đăng nhập.
  - ✓ Hiển thị các thông điệp thân thiện người dùng.
  - ✓ Cho phép thử lại.
  - ✓ Hiển thị thông điệp chung chung.

Bước 7: Xác định chiến lược xác thực

- Chiến lược xác thực đầu vào hiệu quả sẽ giúp lọc dữ liệu độc hại và không mong muốn để bảo vệ ứng dụng từ các lỗ hổng.

- Tất cả dữ liệu đầu vào phải được xác thực.

- Các kỹ thuật xác thực phổ biến nhất gồm:
  - ✓ Accept known good
  - ✓ Reject known bad
  - ✓ Sanitize


Q&A