

# Phân tích và thiết kế hệ thống


Giảng viên: Nguyễn Bá Ngọc

Hà Nội-2022

# Nội dung

1. Tổng quan về thiết kế giao diện
2. Ngôn ngữ mô hình hóa luồng tương tác
3. Ví dụ tổng hợp

# Nội dung

- 
1. Tổng quan về thiết kế giao diện
  2. Ngôn ngữ mô hình hóa luồng tương tác
  3. Ví dụ tổng hợp

# Phân loại giao diện

*Giao diện là phương tiện để tương tác với hệ thống, theo đối tượng sử dụng các giao diện được phân loại thành:*

- Giao diện hệ thống: Hỗ trợ tương tác máy-máy, phục vụ tác nhân là hệ thống khác.
- Giao diện người dùng: Hỗ trợ tương tác người-máy, phục vụ tác nhân là người.
  - Đồng thời cũng như diện mạo hệ thống đối với người dùng - người dùng chỉ nhìn thấy các giao diện chứ không nhìn thấy các xử lý bên trong - cách tiếp cận lấy người dùng làm trọng tâm (user-centered)..

*Chúng ta sẽ tập trung vào giao diện người dùng*

# Các thành phần hỗ trợ tương tác người-máy

## *Tương tác người-máy / Human-Computer Interaction (HCI)*

Thiết bị: Màn hình thường, màn hình cảm ứng, bàn phím thực, bàn phím ảo, chuột, webcam, mic, v.v..

Các giao diện người dùng: Cửa sổ, biểu mẫu, hộp thoại, hình ảnh, nút bấm, danh sách, bảng, biểu đồ, v.v..



# Các nguyên tắc thiết kế giao diện

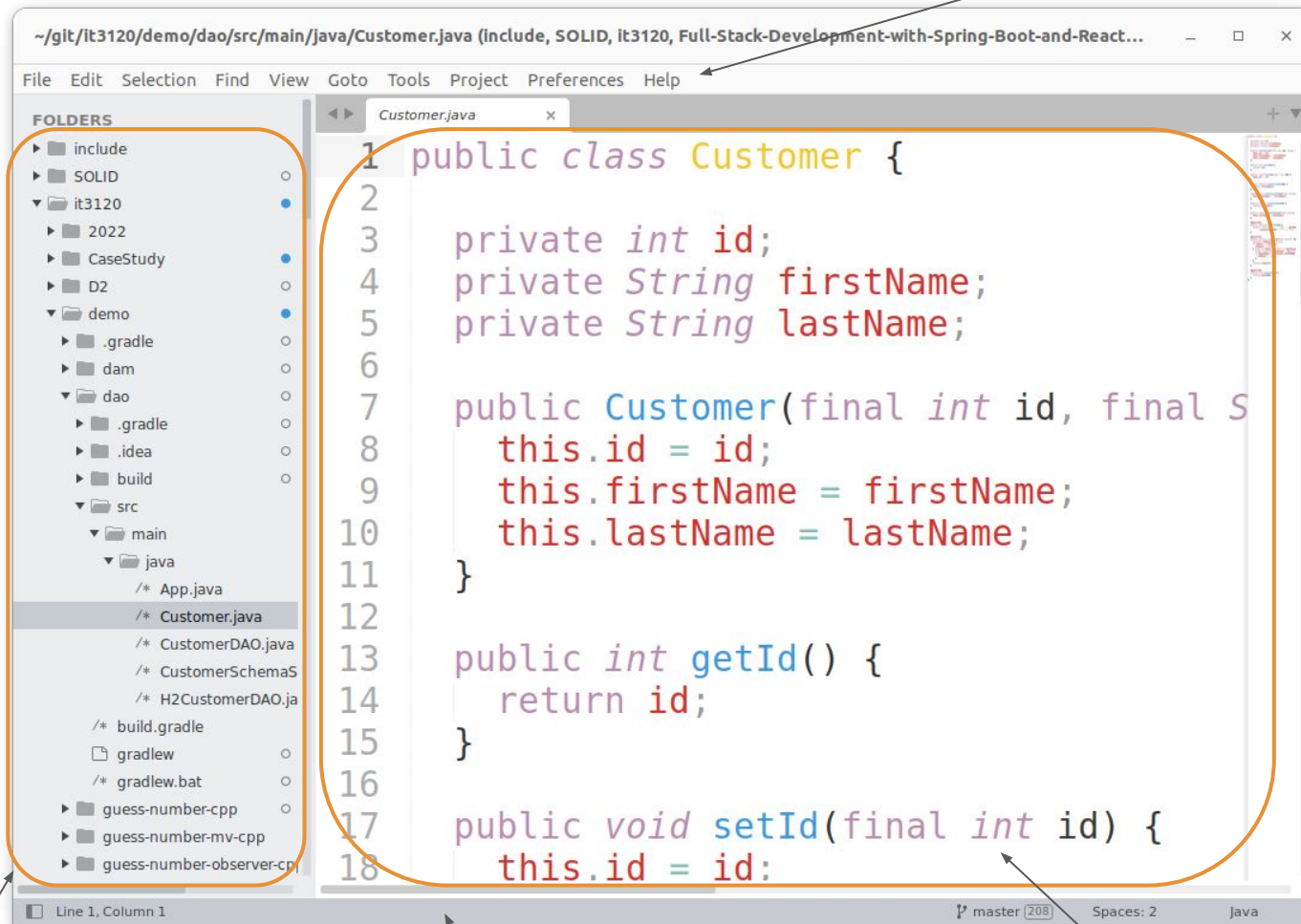
- **Bố cục** - Giao diện nên là 1 chuỗi các phân vùng màn hình ổn định cho các mục đích.
- **Nhận biết nội dung** - Người dùng nên được báo cáo về tính năng đang sử dụng và thông tin đang được hiển thị.
- **Thẩm mỹ** - Giao diện phải đẹp đối với người dùng, hữu ích và lôi cuốn sử dụng.
- **Trải nghiệm người dùng** - Giao diện phải dễ sử dụng, tự nhiên, dễ học, tạo cảm giác tích cực cho người dùng.
- **Nhất quán** - Các thành phần giao diện trong các ngữ cảnh khác nhau phải thống nhất - quan trọng để dễ sử dụng.
- **Tiết kiệm công sức** - Giao diện phải đơn giản, giúp người dùng hoàn thành công việc nhanh chóng, hạn chế lỗi.

# Bố cục

- Bố cục/Layout: Cách bố trí các thành phần trên màn hình
- Các thành phần được gom nhóm trong các vùng.
- Các vùng có thể tiếp tục được gom thành vùng lớn hơn
- Các vùng được bố trí theo trật tự tạo thành luồng liên mạch, tự nhiên.
  - Trật tự tự nhiên có thể thay đổi theo đối tượng người dùng, phụ thuộc vào các yếu tố văn hóa;
  - Chúng ta đọc từ trái sang phải, từ trên xuống dưới,
  - ... tuy nhiên cũng có những nơi đọc từ phải sang trái.

# Ví dụ 1. Bố cục giao diện

Các danh mục chức năng



Cây thư mục

Thanh trạng thái

Vùng soạn thảo

# Nhận biết nội dung

Giúp người dùng xác định các thông tin đang xử lý

- Sử dụng tiêu đề cửa sổ, các nhãn vùng, nhãn tên trường dữ liệu v.v..
- Trực quan tách biệt các vùng
- Sử dụng các danh mục (menu) ngữ cảnh
- V.v...

# Ví dụ 2. Nhận biết nội dung

*Các đường dẫn giúp xác định tệp hiện hành*

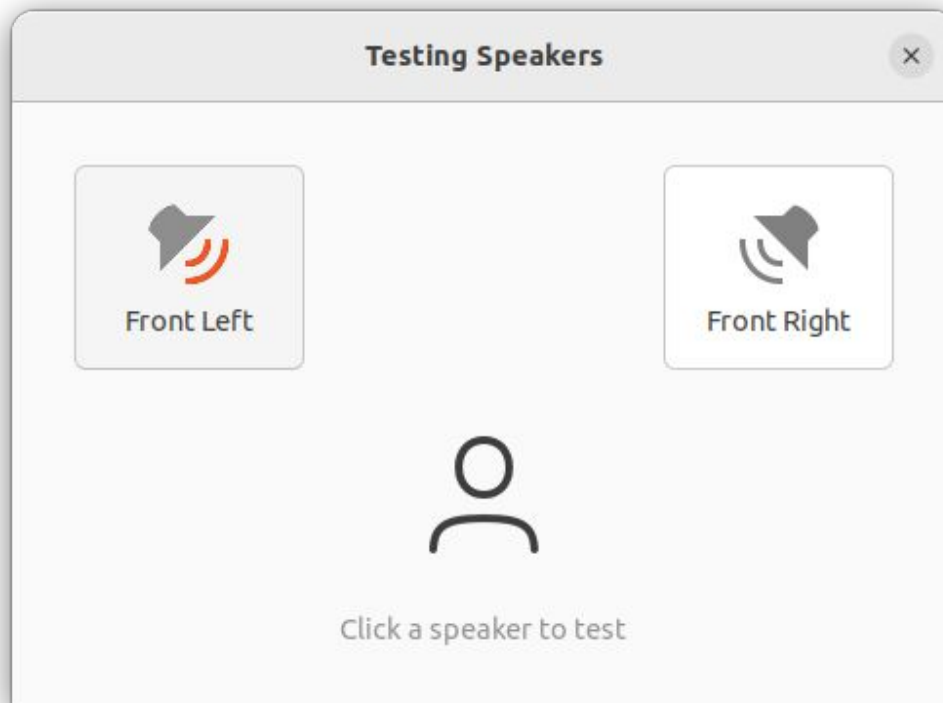
Sơ đồ, chỉ số dòng hỗ trợ định vị nội dung

```
~/git/it3120/demo/dao/src/main/java/Customer.java (include, SOLID, it3120, Full-Stack-Development-with-Spring-Boot-and-React... - □ ×  
File Edit Selection Find View Goto Tools Project Preferences Help  
Customer.java — dam/.../java × Customer.java — dao/.../java ×  
1 public class Customer {  
2  
3 private int id;  
4 private String firstName;  
5 private String lastName;  
6  
7 public Customer(final int id, final S  
8 this.id = id;  
9 this.firstName = firstName;  
10 this.lastName = lastName;  
11 }  
12  
13 public int getId() {  
14 return id;  
15 }  
16  
17 public void setId(final int id) {  
18 this.id = id;
```

The screenshot shows an IDE window with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with a file 'Customer.java' selected. The code editor shows the content of 'Customer.java' with line numbers 1 through 18. Annotations include arrows pointing from the Vietnamese text to the file explorer, line numbers, and the code content.

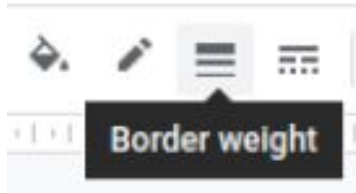
# Mô tả chức năng

- Sử dụng hình ảnh, biểu tượng, sơ đồ v.v.. giúp người dùng phán đoán chức năng và hành vi hệ thống
- Hiện thị và phản hồi trực quan thao tác người dùng
  - Cho biết đối tượng giống thứ gì và làm gì?



# Mô tả chức năng<sub>(2)</sub>

- Gợi ý chức năng
  - Cung cấp thêm thông tin về chức năng cho người dùng
  - Kích hoạt: Khi giữ chuột trên nút bấm, lần đầu mở cửa sổ mới, v.v...



# Tính thẩm mỹ

- Các giao diện phải hữu dụng, tạo động lực, đồng thời phải có tính thẩm mỹ
- Thiết kế thường càng đơn giản càng tốt
- Các khoảng trắng có vai trò quan trọng để phân tách các phần nội dung
- Mật độ thông tin phù hợp với kỹ năng người dùng
  - Mật độ thấp (< 50%) đối với người dùng mới.
  - Mật độ cao (> 50%) đối với người dùng chuyên nghiệp.
- Phong cách và khả năng đọc văn bản: Kích thước, kiểu chữ với nét hoa mỹ (serif) hoặc đơn giản (sans serif), cách sử dụng chữ hoa, v.v..
- Màu sắc và hoa văn.
- v.v..

# Độ phức tạp và hiệu quả sử dụng

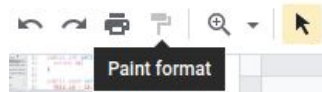
- Độ phức tạp:
  - Giao diện với ít thông tin thường đơn giản, dễ sử dụng đối với người dùng ít kinh nghiệm
    - Rất quan trọng đối với hệ thống có nhiều người dùng;
    - ... nhưng có thể không hiệu quả đối với người dùng chuyên nghiệp.
- Hiệu quả sử dụng:
  - Giao diện với nhiều thông tin có thể hiệu quả hơn đối với người dùng chuyên nghiệp (có thể yêu cầu đào tạo)
    - Rất quan trọng trong các hệ thống chuyên dụng;
    - ... nhưng có thể phức tạp đối với người dùng ít kinh nghiệm;
- Quan hệ bù trừ: Người thiết kế có thể phải lựa chọn ưu tiên người dùng mới hoặc người dùng chuyên nghiệp
  - ... hoặc hỗ trợ đồng thời các giao diện cơ bản & nâng cao.
  - Sử dụng những thiết kế tương tự làm giảm độ phức tạp.

# Tính nhất quán

- Phạm vi
  - Trong 1 chương trình
  - Trong 1 nhóm chương trình
  - Giữa các phiên bản của 1 chương trình theo thời gian
  - Giữa các phiên bản của 1 chương trình trên nhiều nền tảng
- Đặc biệt quan trọng để giảm thời gian tìm hiểu
  - Giúp người dùng phán đoán hành vi
  - Áp dụng kỹ năng học được ở 1 nơi cho những nơi khác
- Các yếu tố cơ bản
  - Những Lô-gic hoạt động chung
  - Điều khiển điều hướng
  - Thuật ngữ
  - Biểu tượng

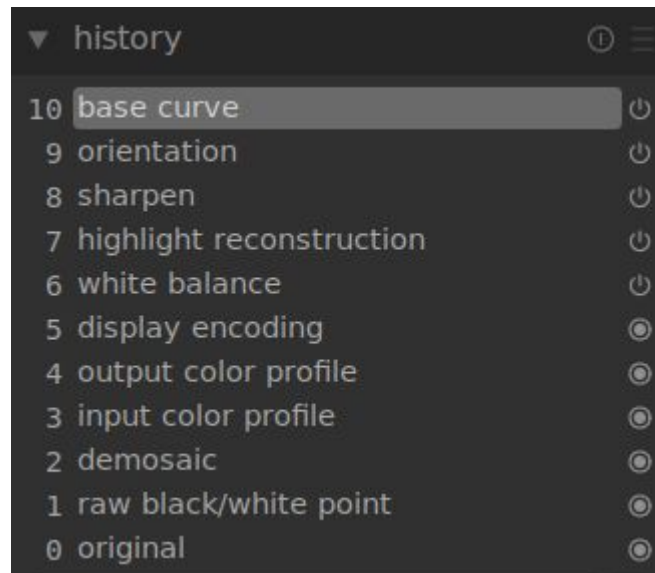
# Hạn chế lỗi thao tác

- Có thể vô hiệu các chức năng khi các điều kiện chưa được đáp ứng



*Cần đánh dấu trước khi có thể sao chép định dạng*

- Bảo vệ kết quả
  - Ghi nhớ các thay đổi
  - Hoàn tác / Khôi phục trạng thái trước thao tác lỗi



# Tối ưu thao tác người dùng

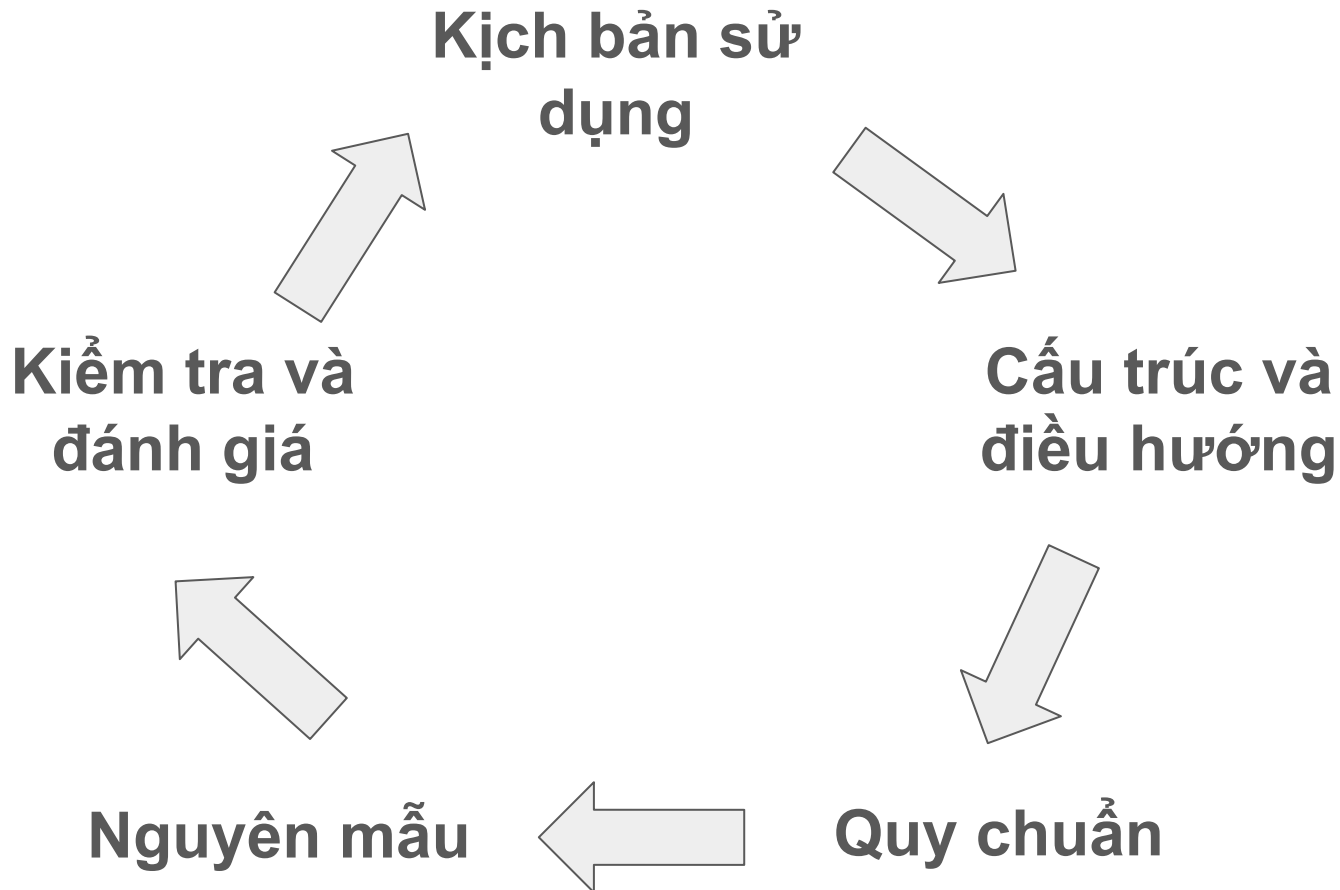
- Các giao diện cần được thiết kế để tối ưu thao tác của người dùng
  - Giúp người dùng hoàn thành công việc nhanh nhất, với số lượng thao tác tối thiểu.
  - Ở 1 góc độ nhất định có thể đánh giá độ phức tạp của thao tác người dùng thông qua số lần kích chuột
- Ví dụ quy tắc 3 lần kích chuột
  - Từ màn hình chính của hệ thống người dùng có thể tiếp cận bất kỳ mục nào với không quá 3 lần kích chuột



# Tiến trình phát triển giao diện người dùng

- Các nguyên tắc quan trọng: *Dẫn dắt bởi ca sử dụng, lặp và tăng dần, tập trung vào người dùng sớm và xuyên suốt dự án, kiểm thử thiết kế để đảm bảo nhu cầu sử dụng.*
- Cơ sở: Ca sử dụng, sơ đồ tuần tự và sơ đồ giao tiếp.
  - *Tập trung vào người dùng và công việc của họ, cái họ cần cho công việc.*
- Các bước thực hiện:
  - Biên soạn các kịch bản sử dụng với các bước tương tác chi tiết, có thể sử dụng các thành phần giao diện thực tế.
    - *(Có thể phát hiện thêm các yêu cầu mới).*
  - Thiết kế cấu trúc và điều hướng;
  - Biên soạn quy chuẩn;
  - Tạo nguyên mẫu.
  - Kiểm tra, đánh giá.

# Thiết kế GD theo nguyên tắc lặp và tăng dần



# Biên soạn kịch bản sử dụng

- Kịch bản sử dụng mô tả các bước được thực hiện bởi người dùng để hoàn thành 1 nhiệm vụ / 1 phần công việc của họ
- Mỗi kịch bản sử dụng tương ứng với 1 luồng sự kiện trong 1 ca sử dụng thiết yếu.
  - Tuy nhiên không phải là bản sao 1-1
  - Được mô tả với các chi tiết về các công cụ tác nghiệp.
- Cần lưu ý các kịch bản được sử dụng nhiều nhất / phổ biến nhất để ưu tiên tạo những giao diện phù hợp, dễ dùng cho những tình huống đó.

# Các ca sử dụng thực tế

- Phụ thuộc vào triển khai: Mô tả chi tiết cách sử dụng hệ thống sau khi được triển khai
- Các ca sử dụng thiết yếu được phát triển thành thực tế bằng cách mô tả chi tiết giao diện người dùng thực tế
- Đối với ứng dụng đa nền tảng, ví dụ, máy bàn, máy tính bảng, và điện thoại thông minh, ca sử dụng thực tế cần được phát triển cho từng nền tảng mà ca sử dụng được triển khai.

# Thiết kế cấu trúc và điều hướng

- Xác định các thành phần giao diện và lô-gic điều hướng
  - Các thành phần điều khiển điều hướng giúp người dùng di chuyển giữa các giao diện trong hệ thống
  - Các tương tác để đáp ứng nhu cầu sử dụng
- Đồng thời thông báo về kết quả thực hiện các hành động: Thành công, thất bại, đang được thực hiện, v.v..
- Có thể biểu diễn bằng IFML và các ca sử dụng thực tế.
  - IFML - **I**nteraction **F**low **M**odeling **L**anguage (Ngôn ngữ mô hình hóa luồng tương tác - thêm 1 quy chuẩn từ OMG).
- Nguyên lý cơ bản:
  - Sử dụng 1 trật tự ổn định cho các thành phần (ví dụ, Tập > Tạo mới; Mở vs. Tập > Mở; Tạo mới)


# Các hình thức điều khiển điều hướng

- Các nút bấm
- Các liên kết
- Danh mục / Menus
  - Người dùng được cung cấp 1 danh sách lựa chọn
  - Có nhiều hình thức (thanh menu, menu ngữ cảnh, v.v..)
- Vuốt và chạm (đối với màn hình cảm ứng)
- Kéo và thả
- Ngôn ngữ
  - Hình thức - Người dùng sử dụng hệ lệnh của hệ thống
  - Tự nhiên - Hệ thống trình diễn ngôn ngữ tự nhiên
- Điều khiển bằng giọng nói
- v.v..

# Các thông báo

- Thông báo lỗi - Cung cấp thông tin về lỗi phát sinh
  - Ví dụ, Có trường dữ liệu bắt buộc bị để trống
- Thông báo xác nhận - Hạn chế lỗi / Yêu cầu xác nhận đối với các thao tác quan trọng (và không thể hoàn tác)
  - Ví dụ, Bạn có chắc chắn muốn xóa hay không?
- Thông báo kết quả thành công
  - Ví dụ, đơn hàng đã được tạo
- Thông báo trạng thái
  - Ví dụ, đã hoàn thành 90% công việc
- Các thông điệp trợ giúp - Cung cấp thông tin bổ xung để hỗ trợ người dùng thực hiện công việc.
- V.V...

# Biên soạn quy chuẩn giao diện

- Quy chuẩn giao diện bao gồm các thành phần giao diện cơ bản được sử dụng thống nhất trong toàn hệ thống
- Các thành phần tiêu biểu thường được chuẩn hóa:
  - Khuôn mẫu giao diện: Bố cục tổng quan, Gam màu, lô-gic sắp xếp các cửa sổ, v.v..
  - Biểu diễn đối tượng:
    - Tên của các thứ trong giao diện
    - Thuật ngữ cho các hành động
    - Các biểu tượng giao diện, ví dụ biểu tượng trên các nút bấm, v.v.
- Các hình tượng tương tác có thể gợi ý hành vi
  - Cho biết đối tượng giống với thứ gì trong thế giới thực từ đó có thể phán đoán chức năng của đối tượng.
  - Ví dụ, giỏ hàng trong hệ thống bán hàng (giống giỏ hàng trong siêu thị) được sử dụng để lưu sản phẩm được chọn.

# Các hình tượng trong tương tác người-máy

- Hình tượng thao tác trực tiếp
  - Hình tượng trong đó các đối tượng được hiển thị (hình ảnh / biểu tượng) giống đối tượng vật lý.
  - Sử dụng giống như sử dụng các đối tượng vật lý.
- Hình tượng bàn làm việc
  - Màn hình được bố cục thành các vùng, với 1 không gian làm việc rộng ở chính giữa và tập hợp các công cụ được bố trí trên biên.
- Hình tượng tài liệu
  - Hình tượng trong đó dữ liệu được biểu diễn trực quan như các trang giấy hoặc biểu mẫu
- Hình tượng hội thoại
  - Hình tượng trong đó người dùng và máy tính tương tác giống như đang thảo luận thông qua các giao diện.

# Ví dụ 3. Hình tượng thao tác trực tiếp

- Giỏ hàng lưu các sản phẩm được chọn



- Kéo tệp vào (biểu tượng) thùng tái chế để xóa



- Các công cụ vẽ



- V.V....

# Ví dụ 4. Hình tượng bàn làm việc & tài liệu

Activities Google Chrome 24 Thg 6 11:51 en

IT3120 - Google Drive PattersonSuperstore\_vi - 20212-ThietKe-p2 - Google Docs

docs.google.com/document/d/1RbmDWaGKv0d1HqsVN06CL8GFZF9KNJ8qUjpmF796dzc/edit#

Search the menus (Alt+ /) 75% Normal text Times New... 12

2 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18

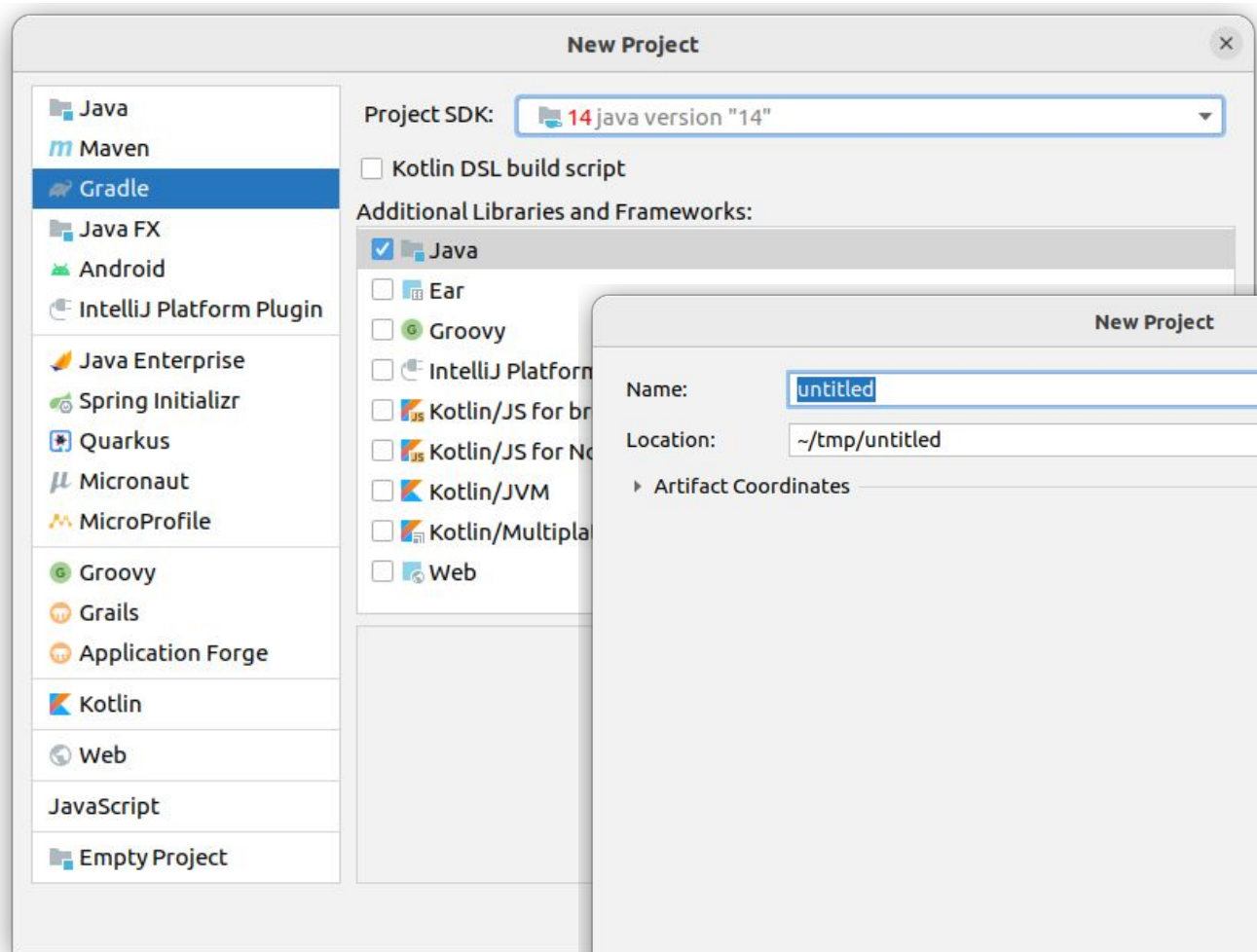
**Hình 9-G. Ảnh xạ lớp lĩnh vực sang bản ORDBMS đã cập nhật**

Thứ 2, cô chỉ ra rằng, với thêm 1 chút công sức, tất cả các bảng ORDBMS đã thiết kế có thể được ánh xạ sang 1 tập các bảng RDBMS. Thực tế là, tập các bảng RDBMS đã được sử dụng trong hệ thống phòng khám sức khỏe hiện có. Tuy nhiên sau khi thảo luận kỹ với Ruby và Ben, Jo đồng ý vì cả Hệ thống bán thuốc theo đơn và Hệ thống phân phối dịch vụ phòng khám sức khỏe tích hợp đều là các sản phẩm chiến lược hướng tới nâng cao lợi thế của Patterson trong thị trường di động, tích nhắc ảnh xạ ORDBMS và sử dụng hệ thống mới như sự khởi đầu để chuyển đổi hệ thống cũ từ RDBMS sang ORDBMS là hợp lý. Hơn nữa, Jo đồng ý để đội của cô chuyển đổi các bảng trong RDBMS cũ sang các bảng ORDBMS và chuyển đổi hệ thống cũ để khai thác lợi thế của thay đổi này. Nhưng tạm thời, cô ấy đề nghị nhóm CSDL của cô ấy trước tiên cần tạo 1 tập các bảng ORDBMS tạm thời có thể ánh xạ tới

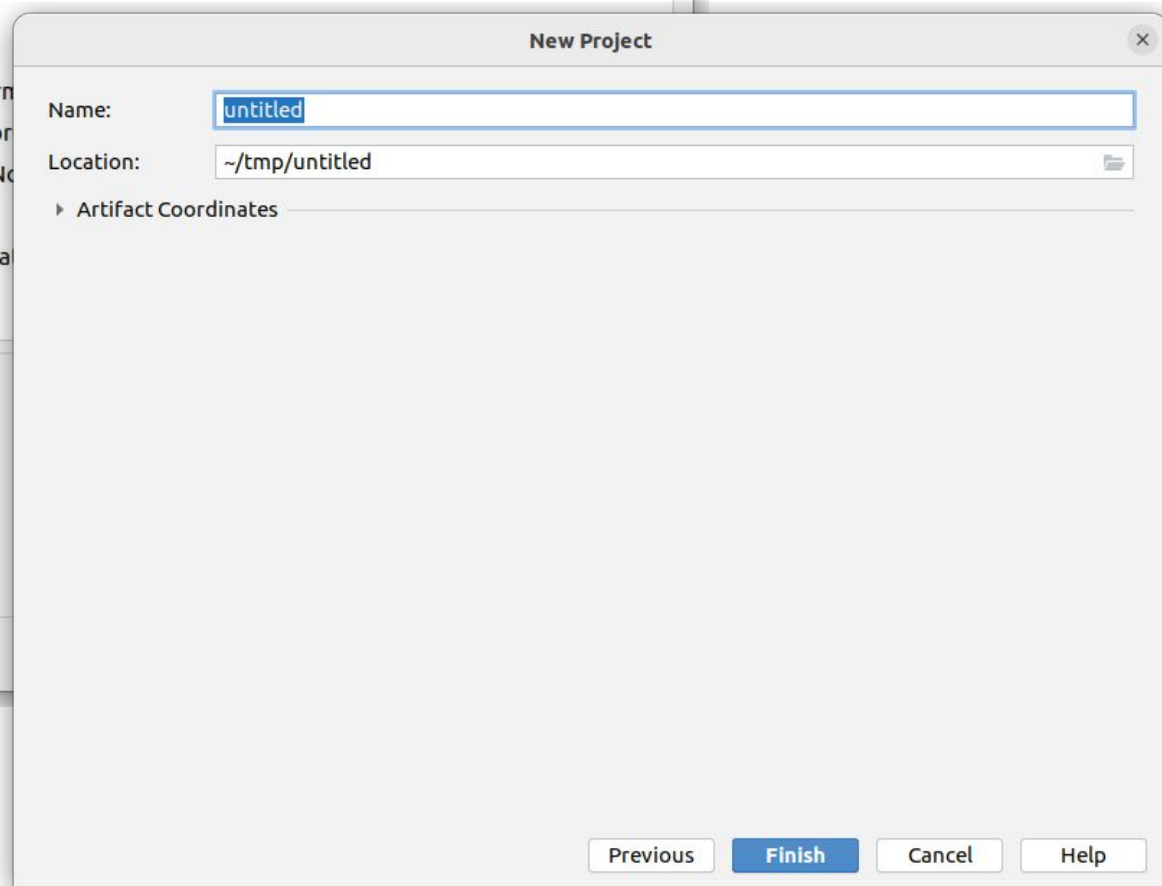
70

# Ví dụ 5. Hội thoại

Bạn muốn tạo loại dự án nào?



Bạn muốn lưu các tệp ở đâu?



*Các hộp thoại tạo dự án trong IntelliJ IDEA*

# Tạo nguyên mẫu giao diện

- Các giao diện có thể được:
  - Vẽ / phác thảo bằng các công cụ vẽ thông thường.
  - Tạo bằng các phần mềm chuyên dụng cho phép giả lập các thao tác chuyển giao diện.
  - Tạo bằng ngôn ngữ lập trình như trong pha thực thi:
    - Tuy nhiên chỉ giả lập các xử lý thực tế:
    - Ví dụ tạo nguyên mẫu trang Web với HTML, CSS, Javascript, và dữ liệu mô phỏng
    - => Gần nhất với giao diện khi triển khai thực tế
- Tương thích và nhất quán với các mô hình cấu trúc và điều hướng.
- Các giao diện có thể được tập hợp và bố trí theo cấu trúc điều hướng tạo thành các bảng phân cảnh / Storyboard
  - Hữu ích - Giúp người dùng hình dung về hoạt động của hệ thống thực tế (trong tương lai).

# Thiết kế nhập

- Các biểu mẫu được sử dụng để nhập dữ liệu
- Dữ liệu có thể:
  - Có cấu trúc: Ngày, sản phẩm, v.v...
  - Phi cấu trúc: Bình luận, mô tả.
- Các nguyên lý cơ bản
  - Nhận dữ liệu từ nguồn (ví dụ Mã vạch vs. RFID);
  - Xử lý trực tuyến vs. xử lý theo gói;
  - Tối ưu số thao tác nhập (sử dụng giá trị mặc định cho những giá trị được dùng thường xuyên).

# Các loại đầu vào

- Các hộp nhập dữ liệu:
  - Hộp nhập văn bản
  - Hộp nhập số, có thể hỗ trợ dạng tự động
    - Ví dụ: Nhập số điện thoại
  - Hộp mật khẩu ẩn các ký tự với các dấu \* và không cho cắt dán hoặc sao chép
  - V.V..
- Các hộp lựa chọn
  - Hộp đánh dấu / Check box, có thể lựa chọn nhiều mục
  - Hộp chọn loại trừ / Radio box - chọn 1 mục trong nhóm.
  - Hộp danh sách - Biểu diễn 1 danh sách lựa chọn
- Thanh kéo / Slider - Có thể di chuyển con trỏ trên 1 thang tương ứng với 1 khoảng giá trị.
- V.V..

# Kiểm tra đầu vào

- Dữ liệu cần được kiểm tra trước khi được đưa vào hệ thống để đảm bảo tính đúng đắn.
- Chỉ tiếp nhận dữ liệu hợp lệ.
- Các kiểm tra tiêu biểu:
  - Định dạng
  - Khoảng giá trị
  - Chữ số kiểm tra - Giúp giảm lỗi nhập số
    - Ví dụ chữ số cuối trong mã vạch
  - Tính nhất quán với các trường dữ liệu liên quan
  - Các ràng buộc trong CSDL

# Thiết kế xuất

- Các báo cáo được tạo từ dữ liệu được xuất bởi hệ thống
- Các nguyên lý cơ bản:
  - Mục đích và tần suất sử dụng ảnh hưởng đến bố cục của báo cáo
  - Quản lý lượng thông tin trong 1 báo cáo - Chỉ cung cấp những gì cần thiết và đặt những thông tin quan trọng nhất ở vị trí đầu tiên.
  - Giảm thiểu độ lệch, đặc biệt trong thiết kế đồ họa (biểu đồ).

# Các loại đầu ra

- Báo cáo chi tiết - Người dùng cần thông tin đầy đủ
- Báo cáo ngắn gọn - Các chi tiết được tổng hợp (ví dụ, tổng, trung bình)
- Báo cáo ngoại lệ
- Tài liệu xoay vòng - Đầu ra quay lại thành đầu vào
- Biểu đồ - Để thuận tiện so sánh trực quan
- Có thể xuất báo cáo trên thiết bị điện tử (xem trên màn hình) hoặc bản cứng (in trên giấy)

# Kiểm tra giao diện

- Lý tưởng, đánh giá các giao diện khi hệ thống đang được thiết kế - Trước khi bắt đầu xây dựng nó
  - Giúp phát hiện và giải quyết sớm các vấn đề
  - Người dùng có thể muốn thay đổi giao diện sau khi nhìn thấy nó.
- Huy động tối đa các thành viên tham gia

# Các phương pháp đánh giá giao diện

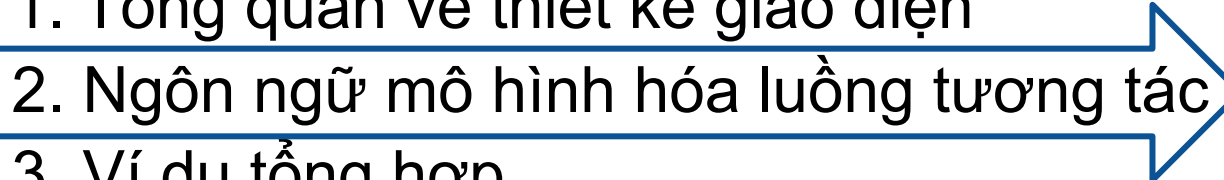
- Theo kinh nghiệm - Đối chiếu thiết kế với các nguyên lý và quy tắc đã biết
- Thuyết trình & đánh giá - Nhóm thiết kế trình diễn nguyên mẫu với người dùng và giải thích lô-gic hoạt động
- Tương tác - Người dùng thử sử dụng nguyên mẫu và thảo luận trực tiếp với các thành viên đội dự án.
- Kiểm thử kín - Người dùng thử sử dụng nguyên mẫu ngôn ngữ trong phạm vi phòng thí nghiệm.

# Các yêu cầu phi chức năng

*Thiết kế giao diện có thể bị ảnh hưởng bởi các yêu cầu phi chức năng (được xác định từ giai đoạn đầu tiên), ví dụ:*

- Các yêu cầu vận hành
  - Triển khai hệ thống theo hình thức ứng dụng Web
- Độ tin cậy
  - Có thể khôi phục hoạt động của hệ thống trong tình huống sự cố trong giới hạn 5 p
- Hiệu năng
  - Cập nhật trạng thái theo thời gian thực với độ trễ không quá 1s
- Các yêu cầu bảo mật
  - Sử dụng kết nối HTTPS

# Nội dung

1. Tổng quan về thiết kế giao diện
  2. Ngôn ngữ mô hình hóa luồng tương tác
  3. Ví dụ tổng hợp
- 

# Các thành phần mô hình

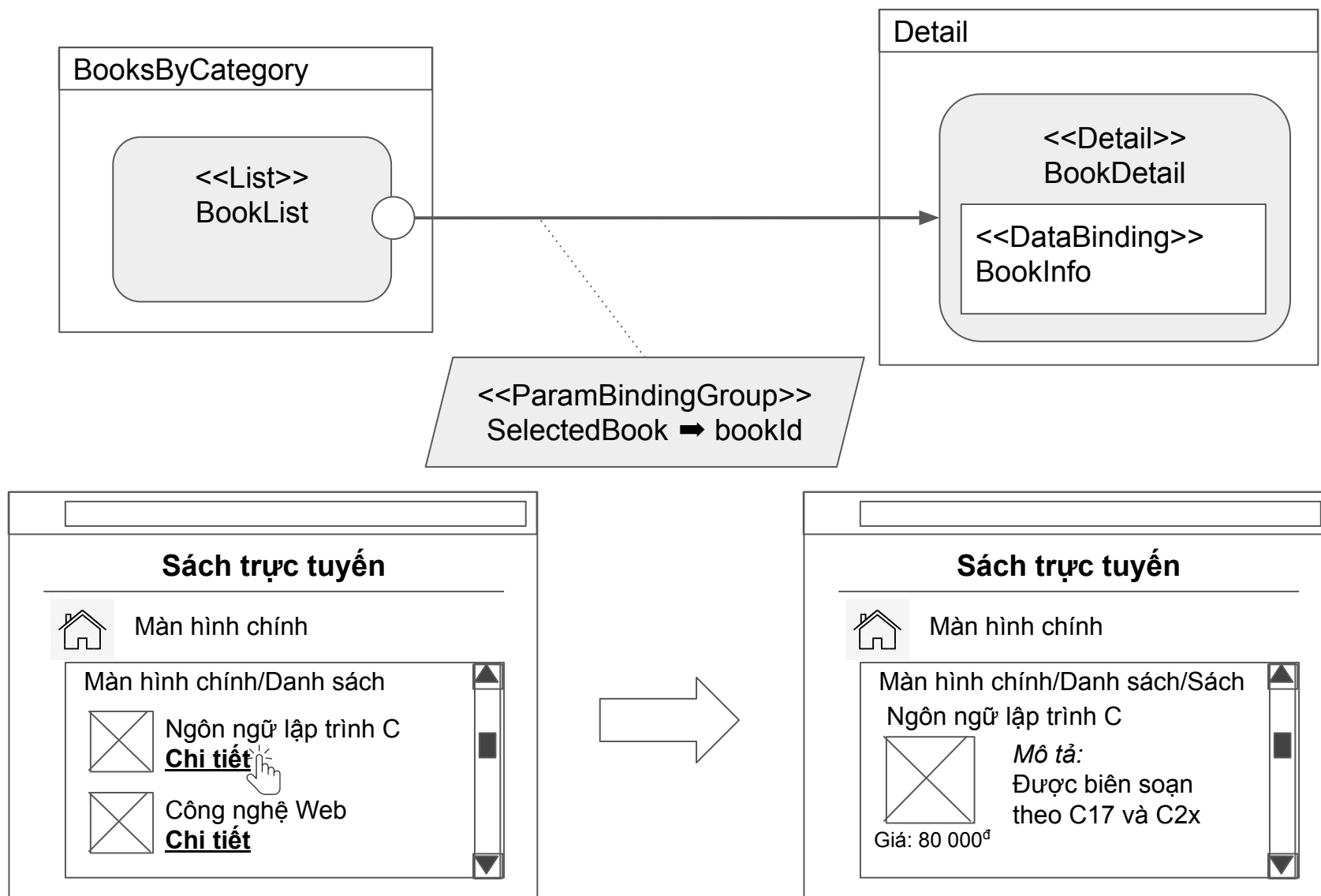
*Sơ đồ luồng tương tác (trong Ngôn ngữ mô hình hóa luồng tương tác - **Interaction Flow Modeling Language - IFML**) biểu diễn các khía cạnh cơ bản của giao diện người dùng*

- **Bố cục:** Chia màn hình giao diện thành các phân vùng, các phân vùng có thể lồng nhau, có thể được thiết lập trạng thái hiển thị, trạng thái khả dụng.
  - Trong IFML phân vùng được biểu diễn bằng thành phần khung/chứa đựng - **ViewContainer**.
- **Nội dung:** Các thành phần nội dung trong phân vùng
  - Thành phần nội dung được biểu diễn bằng thành phần hiển thị - **ViewComponent**.

# Các thành phần mô hình<sub>(2)</sub>

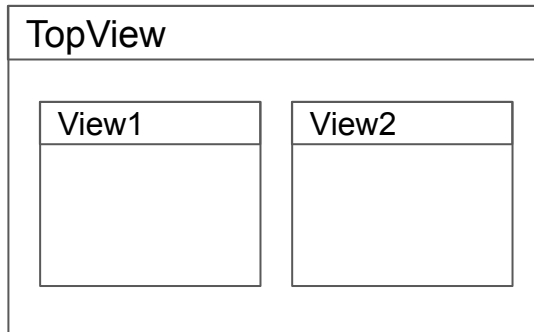
- Sự kiện: Các diễn biến ảnh hưởng tới trạng thái giao diện người dùng - Events - Có thể được kích hoạt bởi tương tác người dùng, tự động theo trạng thái hệ thống hoặc bởi hệ thống ngoại.
- Điều hướng: Biểu diễn hệ quả của 1 sự kiện trên giao diện người dùng như cập nhật và hiển thị nội dung, kích hoạt hành động, hoặc sự kết hợp của nhiều hiệu ứng.
- Truyền tham số: Xác định mối quan hệ đầu ra-đầu vào giữa các thành phần giao diện, phân vùng giao diện, và các hành động.

# Ví dụ 6. Sơ đồ luồng tương tác

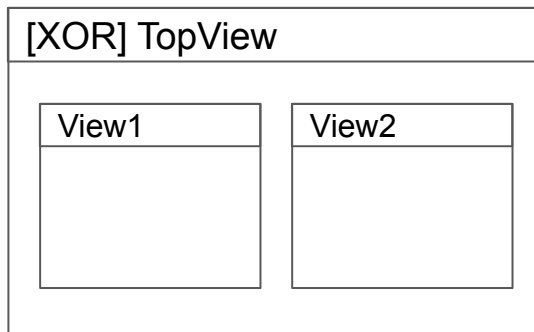


# Các phân vùng lồng nhau

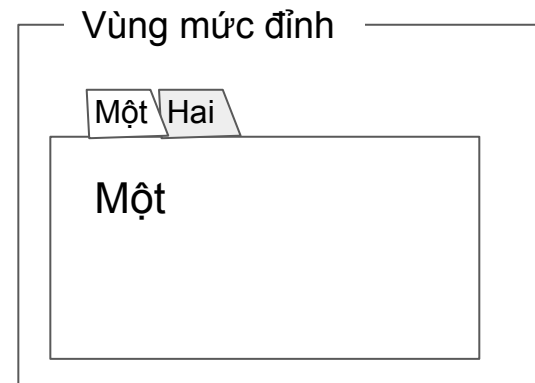
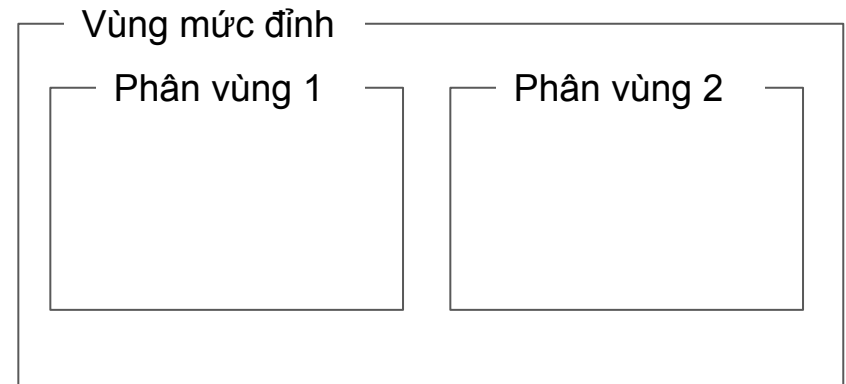
Các phân vùng con cùng nằm trong 1 phân vùng có thể được hiển thị đồng thời hoặc loại trừ lẫn nhau.



Đồng thời

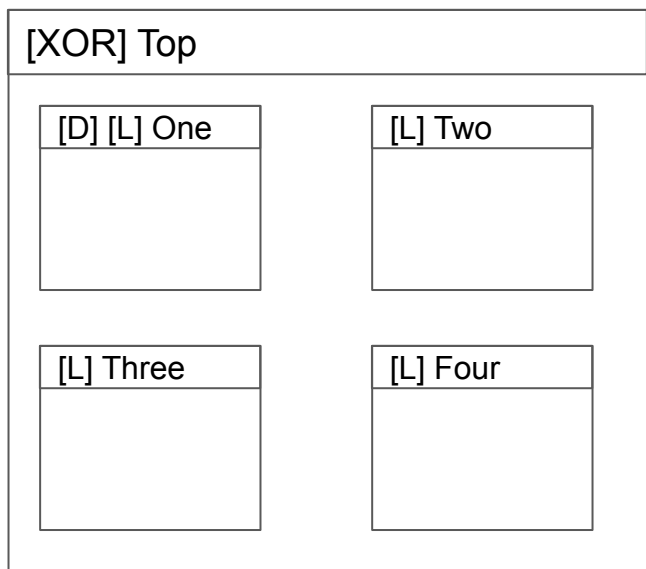


Loại trừ

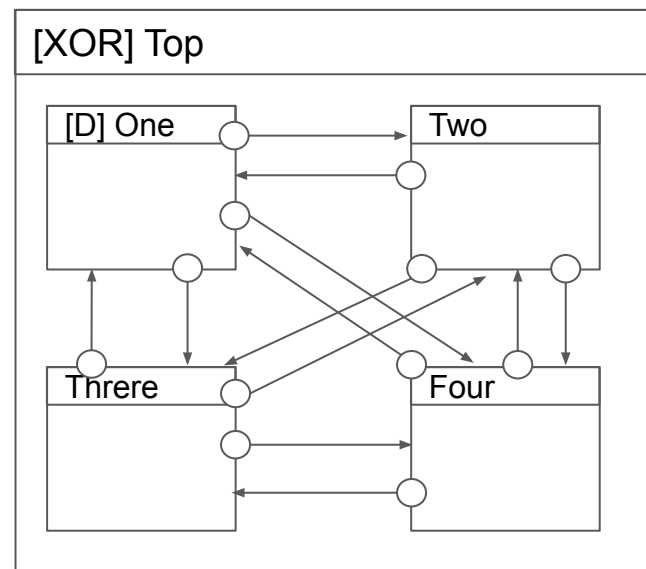


# Các thuộc tính của phân vùng

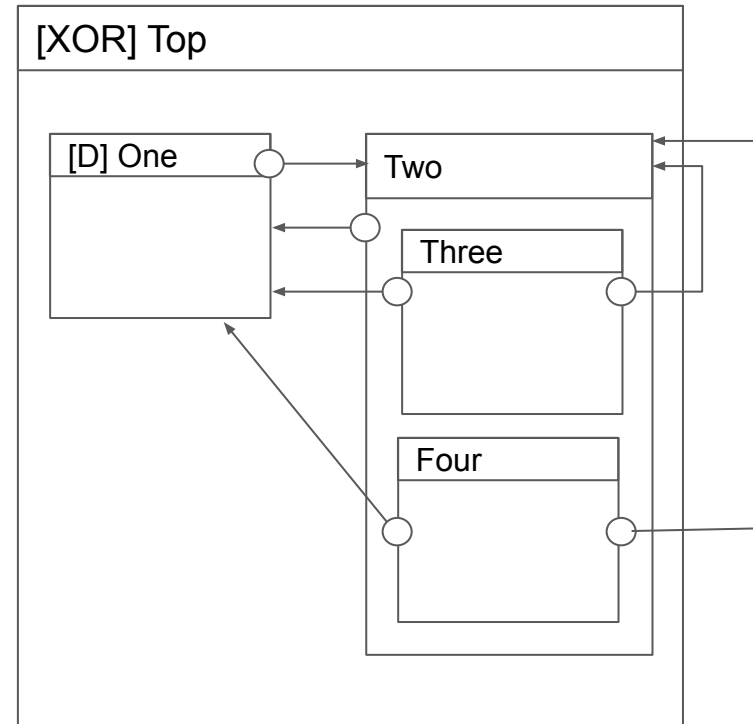
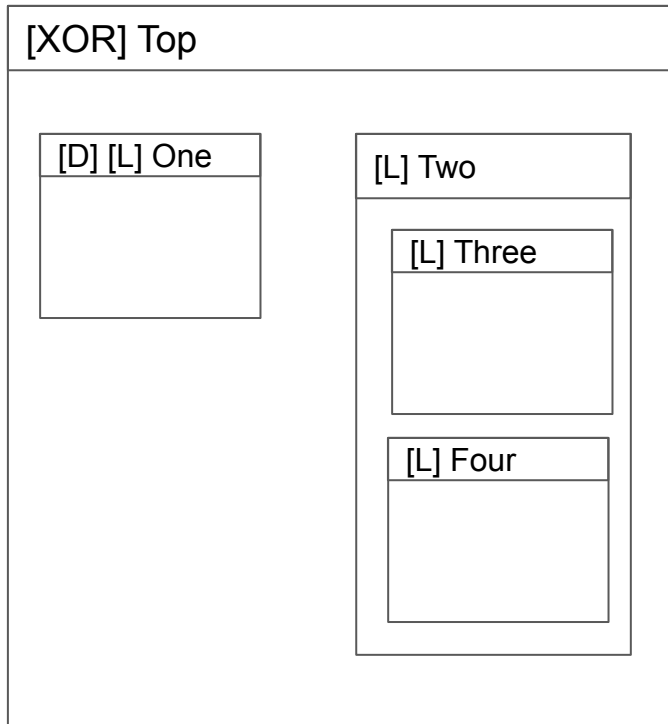
- Phân vùng mặc định - Được chọn để hiển thị nếu không có quy ước cụ thể khác khi phân vùng chứa nó được kích hoạt. Được ký hiệu bằng ký tự D.
- Phân vùng đánh dấu - Có thể hướng tới từ bất kỳ phân vùng nào nằm trong phân vùng chứa nó. Được ký hiệu bằng ký tự L.



< tương đương >



# Các thuộc tính của phân vùng<sub>(2)</sub>



# Cửa sổ

Cửa sổ là 1 loại thành phần khung đặc biệt, có thể được thiết lập chế độ khóa (Modal) hoặc nói lỏng (Modeless).

- Một cửa sổ khóa được mở sẽ vô hiệu khả năng tương tác với các cửa sổ ở chế độ nền cho tới khi được đóng lại.
- Một cửa sổ nói lỏng được mở vẫn cho phép tương tác với các thành phần giao diện khác.

<<Window>> Step1

<<Modal>> Confirm

<<Modeless>> AnotherWindow

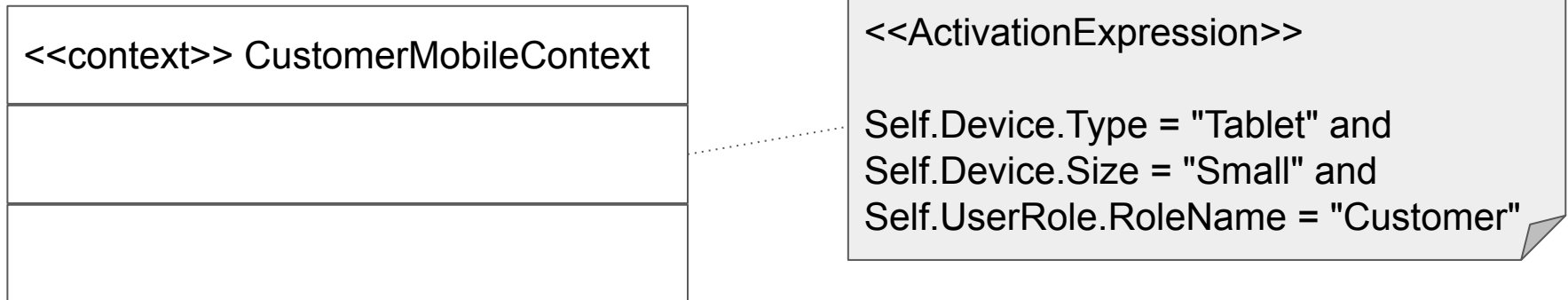
# Ngữ cảnh và chiều ngữ cảnh

- Ngữ cảnh (Context) mô tả các đặc điểm trạng thái hệ thống ở thời gian thực thi và xác định cách giao diện người dùng thích nghi với các đặc điểm đó. Chiều ngữ cảnh (ContextDimension) là 1 thành phần của ngữ cảnh.
- Các mở rộng của chiều ngữ cảnh trong IFML
  - Vai trò người dùng (UserRole) - Bao gồm các thuộc tính mà hồ sơ người dùng phải đáp ứng để kích hoạt ngữ cảnh.
  - Thiết bị (Device) - Biểu diễn các đặc điểm của thiết bị.
  - Vị trí (Position) - Biểu diễn thông tin vị trí và định hướng của thiết bị được sử dụng bởi chương trình ứng dụng.

# Biểu thức kích hoạt

Biểu thức kích hoạt (ActivationExpression) xác định điều kiện Boolean để kích hoạt ngữ cảnh (hoặc thành phần IFML khác)

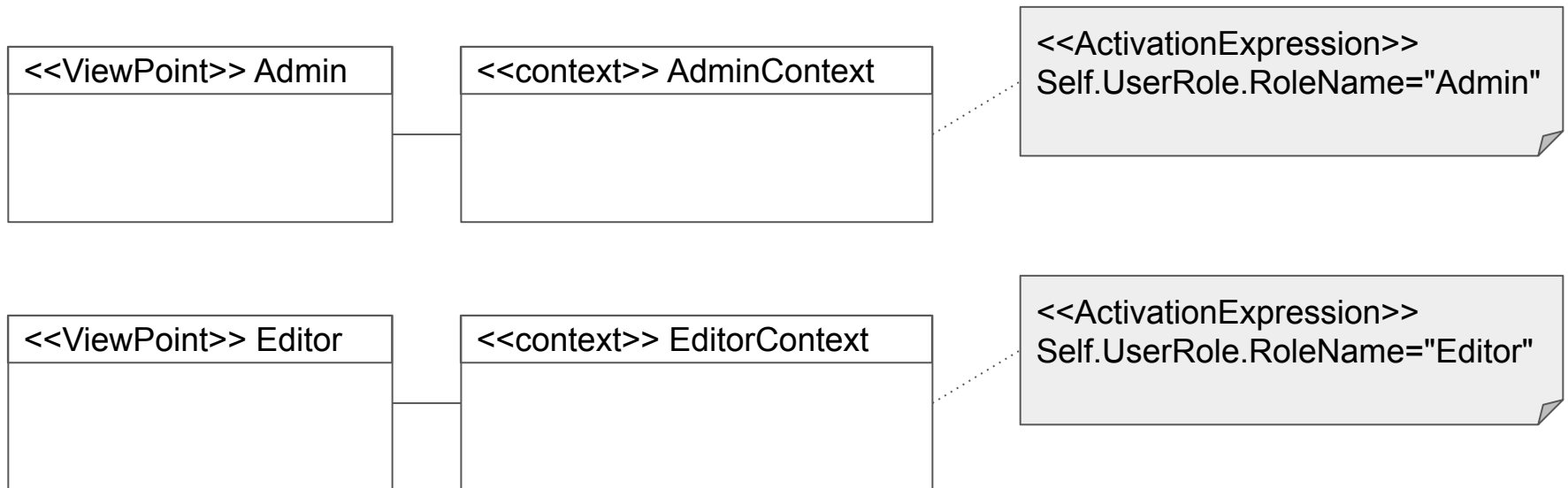
- Kích hoạt nếu đúng và vô hiệu nếu sai.



*Biểu thức kích hoạt xác định các điều kiện để kích hoạt ngữ cảnh Người dùng thiết bị di động*

# Điểm nhìn

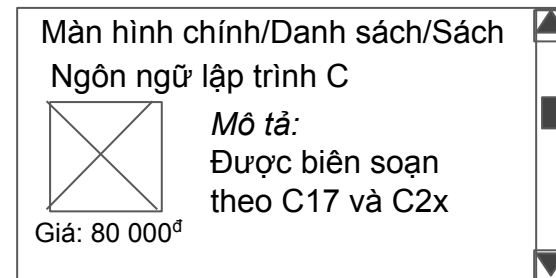
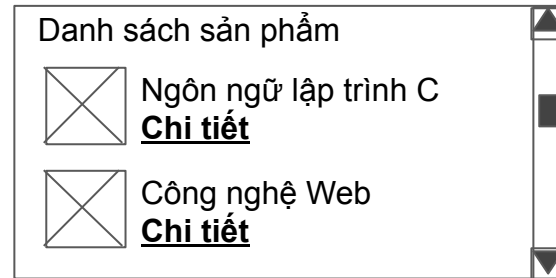
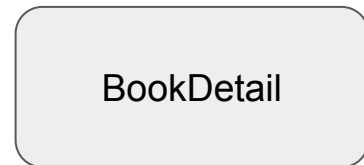
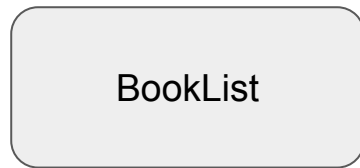
Mỗi điểm nhìn (ViewPoint) là 1 đặc tả toàn bộ mô hình giao diện người dùng có hiệu lực khi 1 ngữ cảnh cụ thể được bật.



# Thành phần hiển thị

- Thành phần hiển thị (ViewComponent) - Thành phần có thể được nhìn thấy trong giao diện người dùng được sử dụng để xuất dữ liệu hoặc nhận dữ liệu từ người dùng.
- Mỗi thành phần hiển thị có thể được hợp thành từ nhiều bộ phận được gọi là phần tử hiển thị ViewComponentPart
  - Phần tử hiển thị (ViewComponentPart) có thể chỉ tồn tại trong thành phần hiển thị/không tồn tại bên ngoài phạm vi thành phần hiển thị.
- Sự kiện và điều hướng: Các thành phần hiển thị có thể tiếp nhận tương tác bằng cách gắn kết với các sự kiện.

# Ví dụ 7. Các thành phần hiển thị



# Nạp nội dung và dữ liệu

- Thành phần hiển thị xuất nội dung được gắn với nó lên màn hình giao diện.
- Nạp nội dung/ContentBinding: Xác định nguồn nội dung của thành phần hiển thị bằng URI.
- Nạp dữ liệu/DataBinding: Dữ liệu bên trong thành phần hiển thị được xác định bởi kiểu dữ liệu, điều kiện lọc và các thuộc tính được sử dụng để hiển thị.
- Hành vi động/DynamicBehavior: Nhận dữ liệu từ cổng dịch vụ

WebView

```
<<ContentBinding>>  
http://mysite.com/index.html
```

MessageList

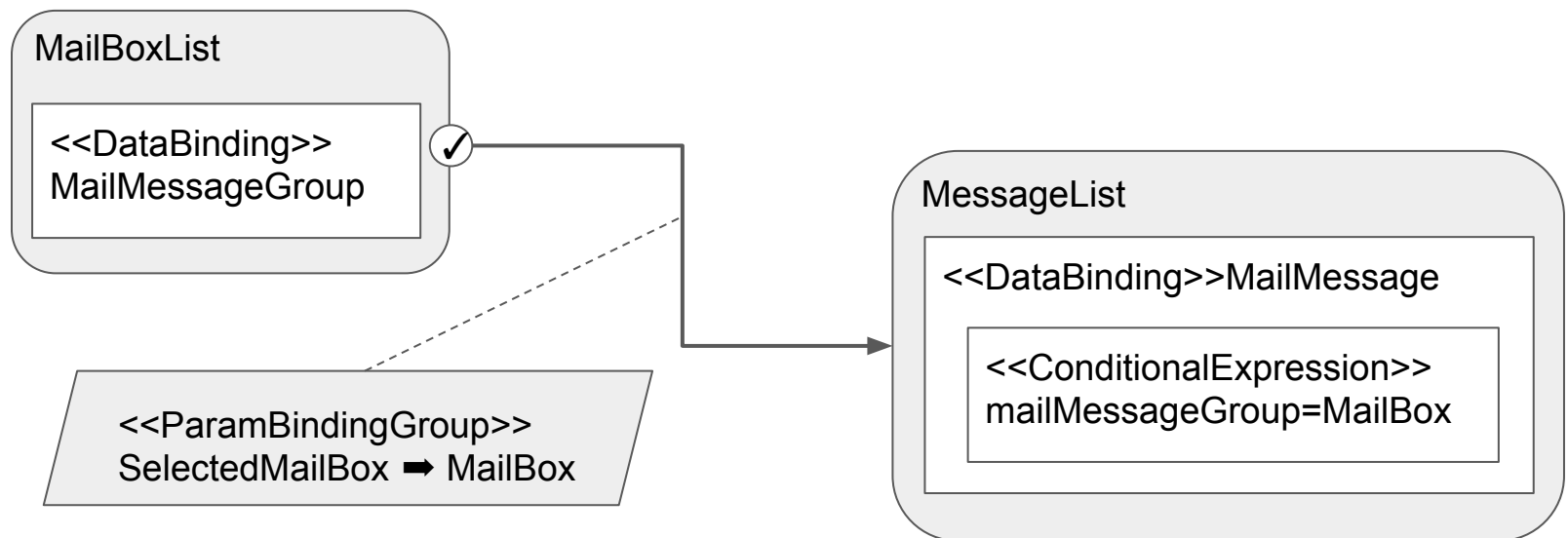
```
<<DataBinding>>  
MailMessage
```

TweetList

```
<<DynamicBehavior>>  
TwitterAPI.Search(query, resNum)
```

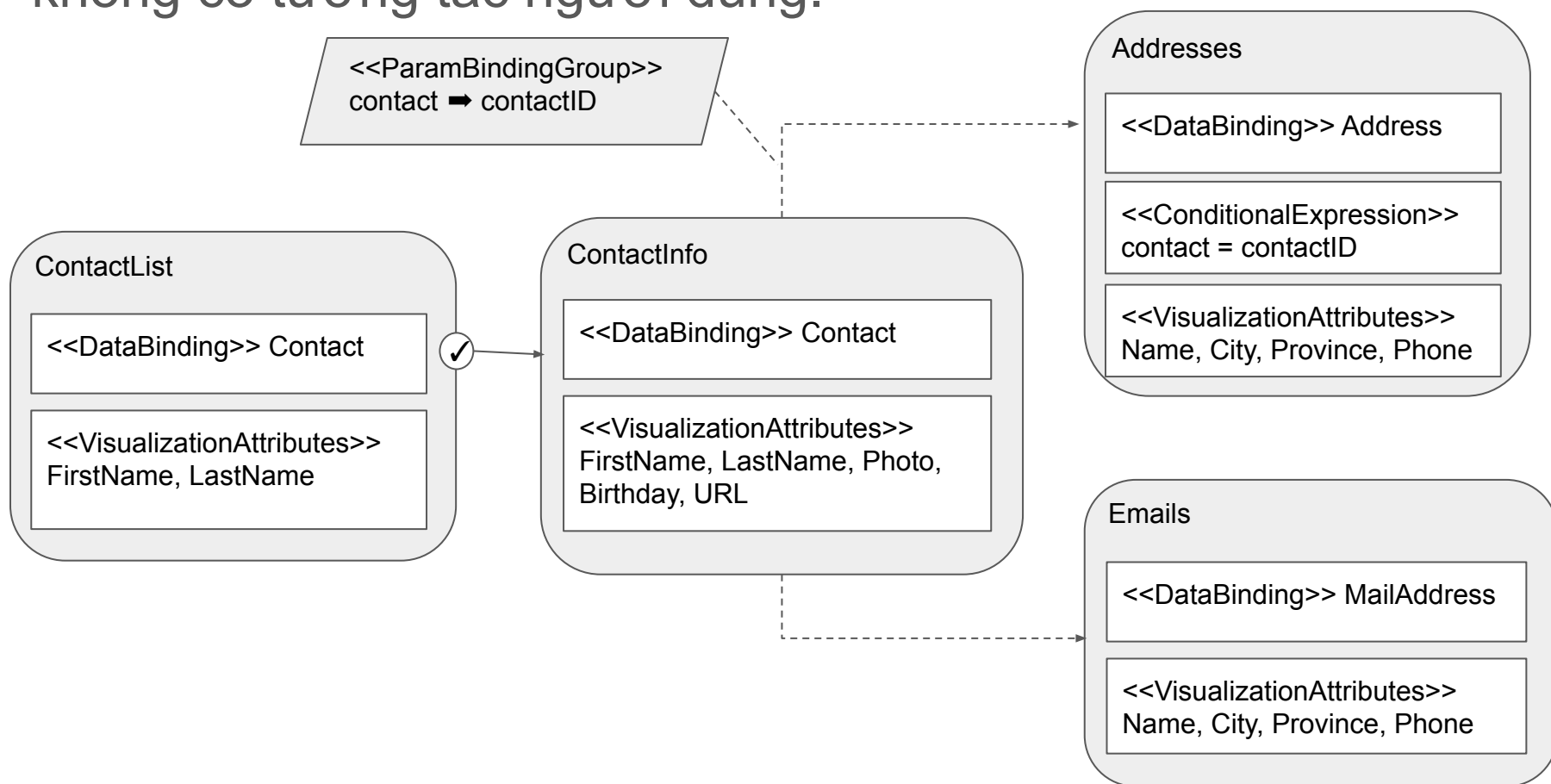
# Truyền tham số

- Kết nối tham số đầu ra của 1 thành phần hiển thị với tham số đầu vào của 1 thành phần hiển thị khác.
- Trong trường hợp truyền đồng thời nhiều tham số các tham số được gom thành nhóm - Truyền nhóm tham số.



# Luồng dữ liệu

Luồng dữ liệu là luồng tương tác cho biết các tham số được truyền từ 1 thành phần nguồn tới 1 thành phần đích mà không có tương tác người dùng.



# Mở rộng IFML

- Các thành phần mở rộng được ký hiệu bởi nhãn loài và có thể chứa các phần tử bổ xung.
- IFML cho phép mở rộng thành phần hiển thị cơ bản (ViewComponent) với các đặc tả của người dùng.
- Ví dụ các mở rộng của thành phần hiển thị:

<<List>> ListName

<<Details>> DetailsName

<<Form>> FormName

<<SimpleField>> Field1: type1

<<SelectionField>> Sel1

# Các thành phần mở rộng để xuất dữ liệu

- Danh sách/List: Mở rộng của ViewComponent để hiển thị 1 danh sách các đối tượng. Nếu danh sách được gắn với 1 sự kiện thì bất kỳ đối tượng nào trong nó cũng có thể được sử dụng để kích hoạt sự kiện và truyền như đối số.
- Danh sách đa lựa chọn/MultiChoiceList: Hỗ trợ nhiều sự kiện (checking, unchecking, submit) và cho phép chọn nhiều phần tử.
- Chi tiết/Detail: Được sử dụng để hiển thị các thuộc tính của đối tượng. Đối tượng trong nó có thể được sử dụng để kích hoạt sự kiện và truyền đi như đối số.
- Sự kiện chọn/SelectEvent: Hỗ trợ lựa chọn 1 hoặc nhiều phần tử từ 1 tập hợp. Khi được kích hoạt nó khiến các giá trị được lựa chọn được truyền đi như đối số.

# Biểu mẫu

- Biểu mẫu/Form là 1 thành phần hiển thị có thể bao gồm nhiều trường dữ liệu.
- Trường/Field là bộ phận của biểu mẫu được xác định bởi kiểu dữ liệu của giá trị mà nó tiếp nhận hoặc hiển thị cho người dùng.
- Trường đơn giản/SimpleField Là 1 loại trường tiếp nhận giá trị có định kiểu. Giá trị của nó là tham số đầu ra và có thể được truyền tới thành phần khác hoặc hành động.
- Trường lựa chọn/SelectionField là 1 loại trường cho phép chọn 1 hoặc nhiều giá trị từ 1 tập đã được định nghĩa.
- Sự kiện gửi/SubmitEvent là 1 loại sự kiện biểu diễn việc gửi 1 hoặc nhiều giá trị. Nó kích hoạt truyền tham số từ thành phần chứa sự kiện tới đích của luồng điều hướng.

# Ví dụ 8. Biểu mẫu

<<Form>>  
BookEntry

Thông tin về sách

Tên: Ngôn ngữ lập trình C

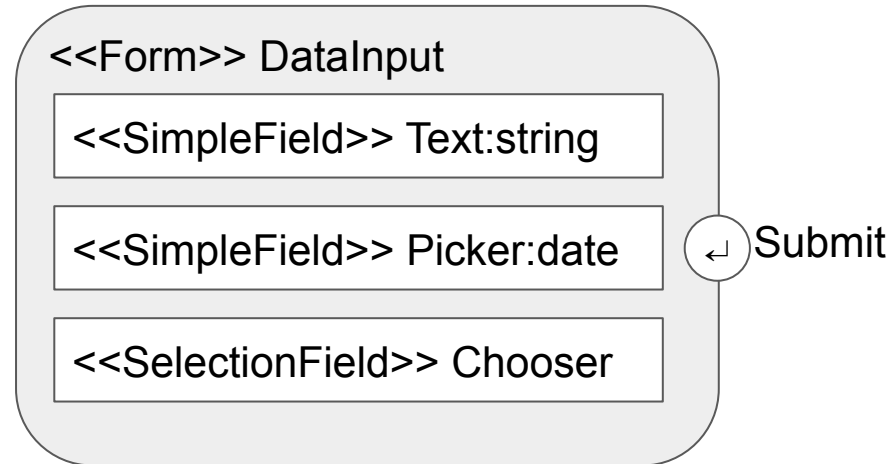
Giá: 80000 Đơn vị: đ ▼

Mô tả:

Được biên soạn theo C17 và C2x

Cập nhật

## Ví dụ 8. Biểu mẫu<sub>(2)</sub>



Nhập dữ liệu

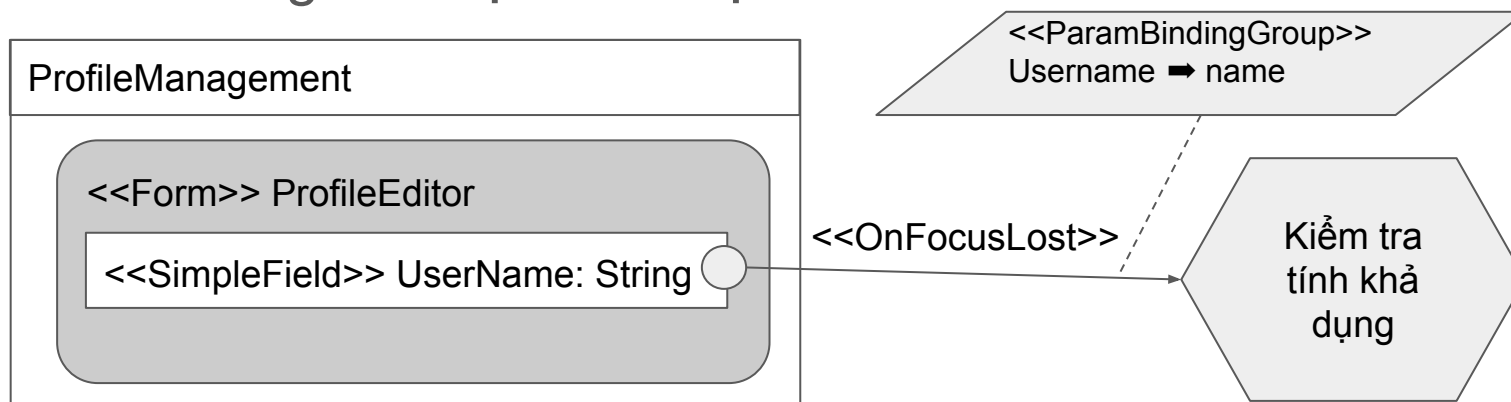
Văn bản:

Chọn ngày:

Lựa chọn:

# Các mở rộng cho giao diện cửa sổ

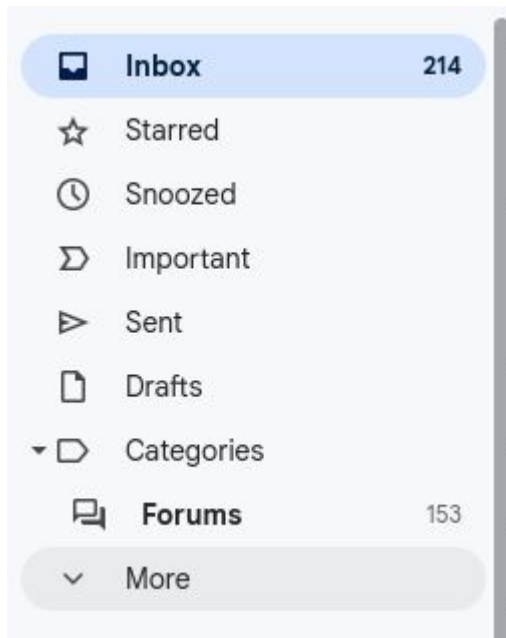
- Mở rộng sự kiện:
  - Khi mất tiêu điểm/OnFocusLost: Sự kiện được kích hoạt khi người dùng rời trường dữ liệu. Có thể được gắn với trường đơn giản hoặc toàn bộ biểu mẫu.



- Bắt đầu kéo/OnDragStart - Bắt đầu kéo. Sự kiện đích được xác định bởi `OnDropEvent`.
- Thả/OnDrop - Được kích hoạt khi kết thúc tương tác thả.

# Các mở rộng cho giao diện cửa sổ<sub>(2)</sub>

- Các thành phần hiển thị:
  - Cây/Tree: Được sử dụng để hiển thị dữ liệu phân cấp, ví dụ cây danh mục email.
  - Bảng/Table: Hiển thị dữ liệu theo dạng bảng và hỗ trợ người dùng soạn thảo.



Danh mục sản phẩm				
Mã	Tên	Đơn giá	Đơn vị	
T110	Tua vít	10000	cái	<a href="#">Xóa</a>
K221	Kìm mỏ nhọn	15000	cái	<a href="#">Xóa</a>
D335	Dây thép	30000	kg	<a href="#">Xóa</a>
O668	Ốc vít	50000	kg	<a href="#">Xóa</a>

# Các mở rộng cho giao diện Web

- Phân vùng: Các mở rộng của ViewContainer
  - Trang - page - 1 mở rộng của thành phần khung biểu diễn 1 trang Web có địa chỉ URL riêng.
  - Khu vực - Area - 1 mở rộng của thành phần khung loại trừ biểu diễn 1 nhóm trang hoặc các khu vực khác, được gom nhóm theo mục đích ứng dụng cụ thể.
  - Miền - SiteView - 1 mở rộng của thành phần khung loại trừ biểu diễn các khu vực của ứng dụng Web và các trang được gom lại theo mục đích ứng dụng cụ thể, thường để đáp ứng nhu cầu của 1 tác nhân.

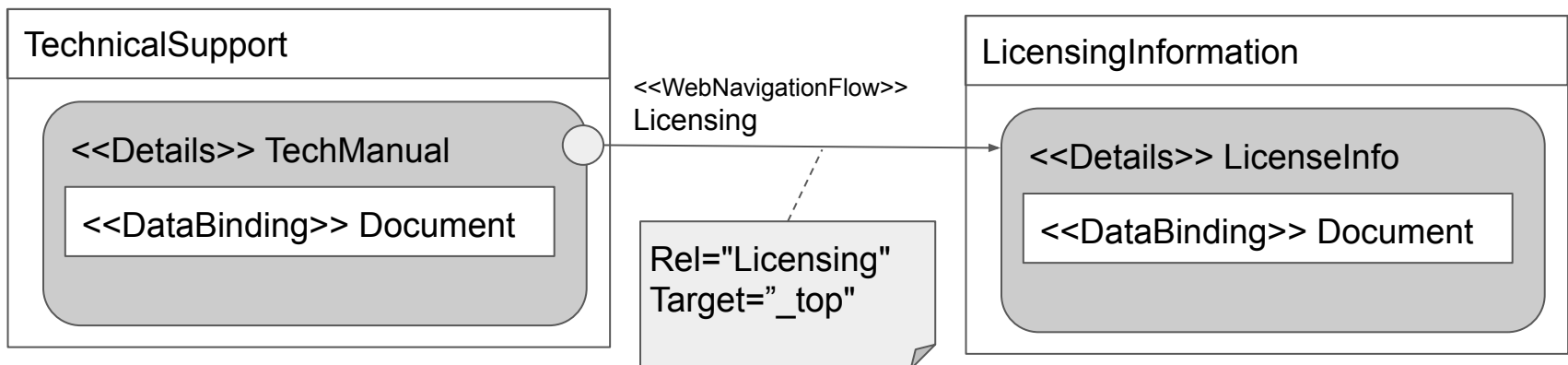
# Các mở rộng cho giao diện Web<sub>(2)</sub>

Miền/Khu vực/Trang được bổ xung thêm các thuộc tính mới cần thiết cho môi trường Web:

- Nhãn URL/URL label: Chuỗi cho biết địa chỉ cố định của miền/trang/khu vực. Đối với các trang động, URL thường bao gồm các tham số để tính nội dung của nó.
- Bảo mật/Security: Nếu thuộc tính có giá trị là "được bảo mật" thì tất cả các trang của khu vực được truy cập bằng giao thức HTTPS.
- Bảo vệ/Protection: Nếu thuộc tính có giá trị là "được bảo vệ" thì tất cả trang trong khu vực hoặc miền, hoặc 1 trang riêng lẻ được giới hạn truy cập.

# Các mở rộng cho giao diện Web<sub>(3)</sub>

- Liên kết/Link: Luồng điều hướng Web (WebNavigationFlow) là 1 mở rộng của luồng điều hướng (NavigationFlow) với các thuộc tính bổ sung:
  - Quan hệ/Rel: Xác định mối quan hệ giữa tài liệu hiện thời và tài liệu được liên kết; giá trị của nó được mã hóa bằng HTML.
  - Đích/Target: Xác định đích mở tài liệu, thường là 1 cửa sổ trình duyệt; có thể là cùng cửa sổ trình duyệt với tài liệu hiện tại hoặc 1 cửa sổ mới.



# Các mở rộng cho giao diện Web<sub>(4)</sub>

- Thành phần hiển thị:
  - Danh sách sắp xếp động/DynamicSortedList: Là mở rộng của thành phần danh sách (List) cho phép người dùng sắp xếp dữ liệu bằng các thuộc tính trực quan.
  - Danh sách cuộn/ScrollableList: Là mở rộng của danh sách cho phép gom các phần tử thành khối (phân trang) và cho phép di chuyển đến các khối.
  - Danh sách lồng nhau/NestedList: Là mở rộng của danh sách với thành phần có thể là danh sách.


# Mở rộng cho giao diện di động

- Ngữ cảnh:
  - Thiết bị/Device có các chiều sau:
    - Kích thước đường chéo/DiagonalSize: Kích thước vật lý theo đường chéo của màn hình.
    - Phân khúc kích thước/SizeCategory: Các kích thước có thể được gom thành các lớp như (Nhỏ, Bình thường, Lớn, rất lớn).
    - Phân khúc mật độ/DensityCategory: Các giá trị mật độ được gom thành các lớp (Thấp, Trung bình, Cao, Rất cao).
    - Kích thước theo điểm ảnh/PixelSize: Kích thước thực tế theo chiều ngang và chiều dọc của màn hình được đo bằng điểm.
    - Mật độ: Số lượng điểm ảnh trên đơn vị diện tích được đo bằng dpi (dots per inch).
    - Kết nối mạng/Network connectivity: Có thể được sử dụng để tùy chỉnh chất lượng của nội dung trên giao diện dựa trên tốc độ kết nối mạng.
    - V.V..

# Mở rộng cho giao diện di động<sub>(2)</sub>

- Phân vùng:
  - Nhãn dòng <<system>>: Thành phần khung được gán nhãn <<system>> biểu diễn 1 vùng cố định của giao diện, được quản lý bởi hệ điều hành hoặc nền tảng giao diện khác.
- Thành phần hiển thị và sự kiện/Component and event:
  - Nhãn dòng <<system>>: Thành phần hiển thị được gán nhãn <<system>> nhấn mạnh giao diện sử dụng các thành phần của hệ thống.
  - <<nhấn>> và <<chạm>>: <<press>> and <<touch>>.

# Nội dung

1. Tổng quan về thiết kế giao diện
  2. Ngôn ngữ mô hình hóa luồng tương tác
  3. Ví dụ tổng hợp
- 

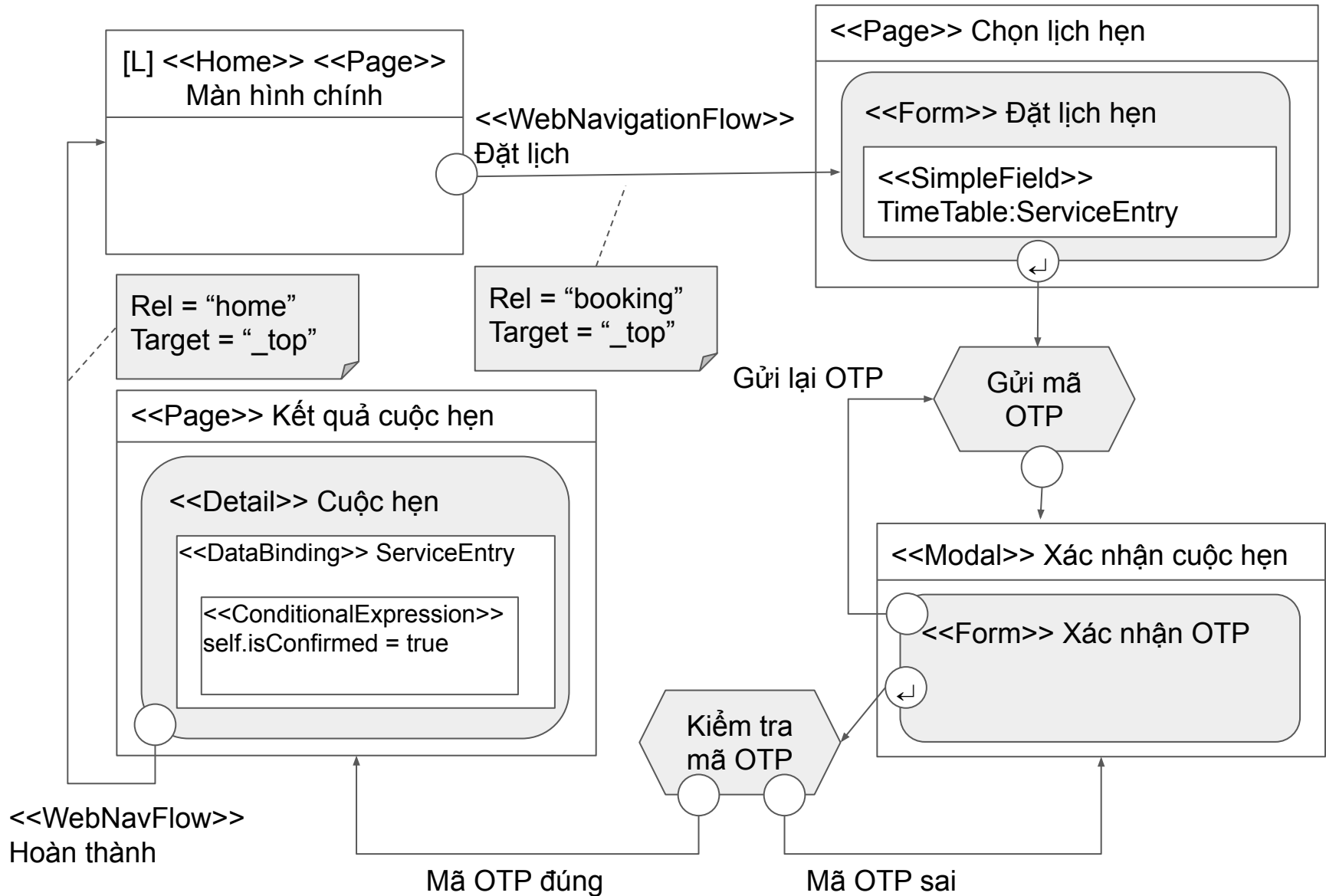
Ví dụ thiết kế giao diện Web  
cho kịch bản Đặt lịch hẹn dịch vụ Spa

# Kịch bản sử dụng

## **Kịch bản Khách hàng đặt lịch hẹn dịch vụ Spa**

1. Khách hàng yêu cầu Chọn lịch hẹn
2. Hệ thống hiển thị lịch làm việc
3. Khách hàng lựa chọn thời gian phù hợp
4. Hệ thống gửi mã xác nhận tới khách hàng
5. Khách hàng nhập mã xác nhận
6. Hoàn thành đặt lịch nếu mã xác nhận hợp lệ.
7. Nếu mã xác nhận không hợp lệ
  - Khách hàng có thể nhập lại
  - hoặc yêu cầu gửi lại mã xác nhận  
(*rồi quay lại bước 5*).
  - hoặc kết thúc không đặt lịch hẹn.

# Mô hình hóa luồng tương tác



# Quy chuẩn giao diện

- Các nút bấm
  - Gửi yêu cầu - Đặt lịch hẹn theo thông tin đã chọn
  - Xác nhận - Chứng minh người dùng thực đang gửi yêu cầu
  - Hoàn thành - Đóng cửa sổ đặt lịch hẹn và quay về màn hình chính
- Các danh mục mức đỉnh
  - Đặt lịch hẹn - Bắt đầu thực hiện kịch bản Đặt Lịch hẹn
- Các liên kết
  - Gửi lại mã - Gửi lại mã OTP để định danh người dùng
- Hộp nhập mã
  - 1 dòng, độ dài tối đa 6 ký tự
  - Hiện ký tự
  - Cỡ chữ 18pt

# Quy chuẩn giao diện<sub>(2)</sub>

- Thời gian biểu:
  - Hiển thị theo ngày, khoảng cách 30 phút
  - Dòng là thời gian, cột là dịch vụ
  - Để trống ô có thể chọn, ô đã có lịch hẹn được tô màu đỏ, ô được chọn được tô màu xanh.

Ngày **10/04/2023**

	Gội đầu		Chăm sóc da mặt
	Ghế 1	Ghế 2	
8h30			
9h00			
11h00			
11h30			
13h30			
14h00			

Khách chọn  
chăm sóc da  
mặt lúc 8h30

# Quy chuẩn giao diện<sub>(3)</sub>

- Chi tiết cuộc hẹn:
  - Hiện thị thông tin cuộc hẹn đã được đăng ký thành công  
**Đặt lịch hẹn thành công!**


**Tên dịch vụ:** Chăm sóc da mặt

**Thời gian:** 9h00

**Ngày:** 10/04/2023

# Nguyên mẫu giao diện: Trang chính

[Đặt lịch hẹn](#)

Người dùng

<<Bài viết về thẩm mỹ viện>>

<< Thông tin liên hệ>>

# Nguyên mẫu giao diện: Đặt lịch hẹn

Màn hình chính/Đặt lịch hẹn

 Người dùng

Ngày **10/04/2023**

	Gội đầu		Chăm sóc da mặt
	Ghế 1	Ghế 2	
8h30			
9h00			
11h00			
11h30			
13h30			
14h00			

Gửi yêu cầu

# Nguyên mẫu giao diện: Xác nhận lịch hẹn

Màn hình chính/Đặt lịch hẹn/Xác nhận

Người dùng

Mã OTP đã nhận:

123567

(Chưa nhận được mã)  
Gửi lại

Xác nhận

<< Thông tin liên hệ>>

# Nguyên mẫu giao diện: Trạng thái đặt lịch

Màn hình chính/Đặt lịch hẹn/Kết quả

Người dùng

**Đặt lịch hẹn thành công!**  
**Tên dịch vụ:** Chăm sóc da mặt  
**Thời gian:** 9h00  
**Ngày:** 10/04/2023

Hoàn thành

<< Thông tin liên hệ>>

# Bảng phân cảnh Đặt lịch hẹn

