



TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI  
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

# Nhập môn Công nghệ phần mềm Introduction to Software Engineering (IT3180)

## **Lab Guide 4**

### Bộ thẻ JSTL của JSP

Thông tin GV

# Nội dung

- Giới thiệu JSP scripting elements
- Giới thiệu JSTL (JSP Standard Tag Library)



TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI  
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

# Trang JSP

# Trang JSP là gì?

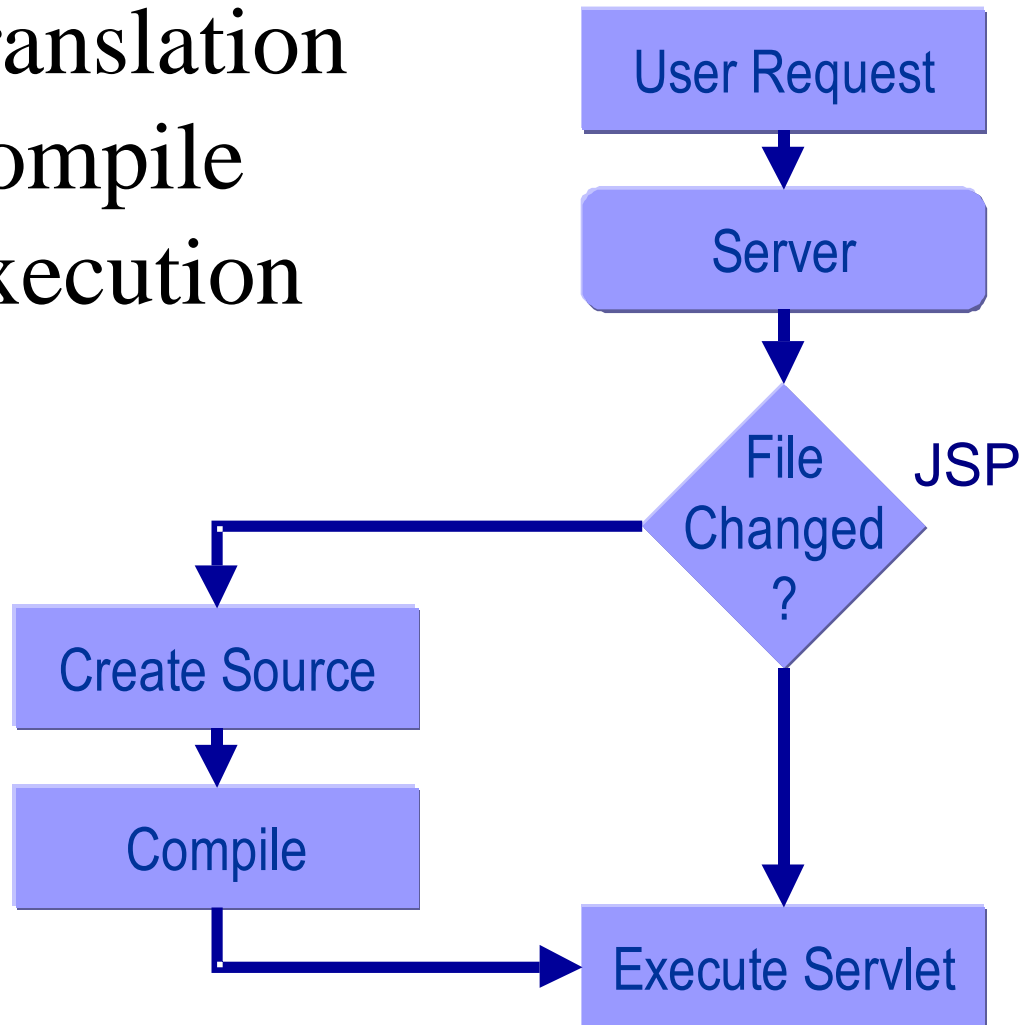
- Là **1 tài liệu text** có thể trả về cả static và dynamic content cho trình duyệt
- Static content và dynamic content có thể được ghép lẫn với nhau
- Static content
  - HTML, XML, Text
- Dynamic content
  - Mã Java
  - Các thuộc tính hiển thị của JavaBeans
  - Các thẻ Custom tags

```
<html>
<body>
  Hello World!
<br>
  Current time is <%= new java.util.Date() %>
</body>
</html>
```

(Blue: static, Red: Dynamic contents)

# JSP làm việc như thế nào?

- Translation
- Compile
- Execution



## 5. Kỹ thuật sinh nội dung động với công nghệ JSP

- Gọi mã Java trực tiếp trong JSP
- Gọi mã Java gián tiếp trong JSP
- Sử dụng JavaBeans
- Tự phát triển và sử dụng các **custom tags**
- Sử dụng **3rd-party custom tags** hoặc **JSTL** (JSP Standard Tag Library)
- Sử dụng mẫu thiết kế MVC
- Sử dụng Model2 frameworks đã được chuẩn hóa, kiểm chứng



TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI  
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

# JSP Scripting Elements

# JSP Scripting Elements

- Cho phép chèn mã nguồn Java vào servlet được sinh tương ứng cho trang JSP
- Nên **GIẢM THIỂU** sử dụng JSP scripting elements trong trang JSP nếu có thể
- Có 3 dạng:
  - Expressions: `<%= Expressions %>`
  - Scriptlets: `<% Code %>`
  - Declarations: `<%! Declarations %>`



# Expressions

- Trong giai đoạn thực thi trang JSP:
  - Expression được tính giá trị và chuyển thành một String
  - String sau đó được chèn trực tiếp vào output stream của Servlet tương ứng
  - Kết quả tương đương với lệnh `out.println(expression)`
  - Có thể sử dụng các biến định nghĩa trước đó (implicit objects) trong 1 expression
- Cú pháp
  - `<%= Expression %>` hoặc
  - `<jsp:expression>Expression</jsp:expression>`
  - **KHÔNG** được dùng dấu ; trong các expressions

# Ví dụ: Expressions

- **Hiển thị thời gian hiện tại sử dụng lớp Date**
  - Current time: `<%= new java.util.Date() %>`
- **Hiển thị 1 số ngẫu nhiên sử dụng lớp Math**
  - Random number: `<%= Math.random() %>`
- **Sử dụng các **implicit objects****
  - Hostname: `<%= request.getRemoteHost() %>`
  - Parameter: `<%= request.  
getParameter("yourParameter") %>`
  - Server: `<%= application.getServerInfo() %>`
  - Session ID: `<%= session.getId() %>`

# Scriptlets

- Sử dụng để chèn mã Java bất kỳ vào phương thức `jspService()` của Servlet tương ứng
- Có thể làm được các việc mà expressions không thể
  - Thiết lập **response headers** và **status codes**
  - Ghi log cho server
  - Cập nhật CSDL
  - Thực thi mã nguồn có điều khiển lặp, rẽ nhánh
- Có thể sử dụng các biến đã định nghĩa (implicit objects)
- Cú pháp:
  - `<% Java code %>` hoặc
  - `<jsp:scriptlet> Java code</jsp:scriptlet>`

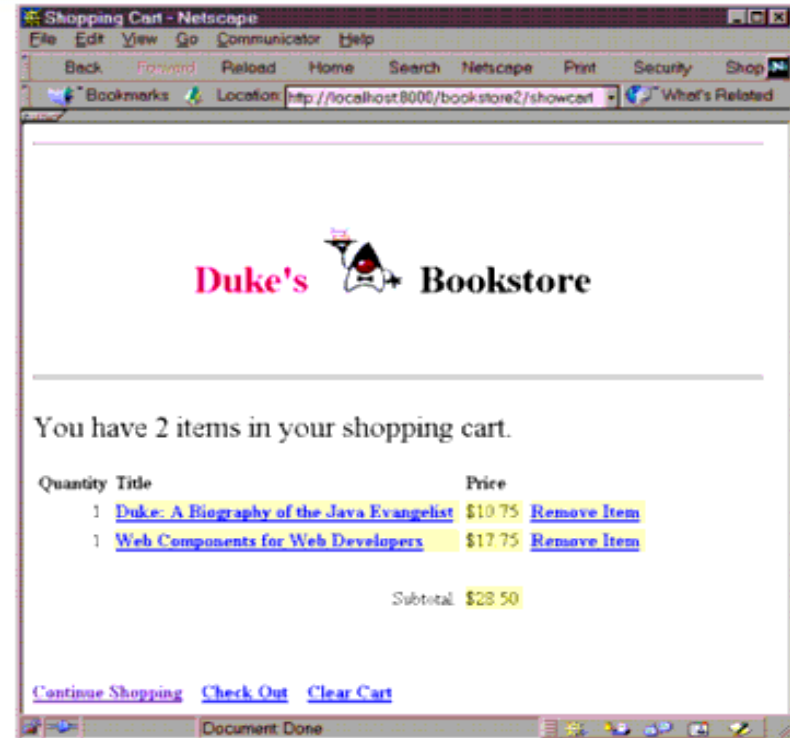
# Ví dụ: Scriptlet với điều khiển lặp

```
<%  
    Iterator i = cart.getItems().iterator();  
    while (i.hasNext()) {  
        ShoppingCartItem item =  
            (ShoppingCartItem)i.next();  
        BookDetails bd = (BookDetails)item.getItem();  
    %>
```

```
    <tr>  
    <td align="right" bgcolor="#ffffff">  
        <%=item.getQuantity()%>  
    </td>  
    <td bgcolor="#ffffaa">  
        <strong><a href="  
        <%=request.getContextPath()%>/bookdetails?bookId=  
        <%=bd.getBookId()%>"><%=bd.getTitle()%></a></strong>  
    </td>
```

...

```
<%  
    // End of while  
    }  
    %>
```



# Declarations

- Được sử dụng để định nghĩa các biến hoặc phương thức (sẽ được chèn vào trong lớp Servlet tương ứng)
  - Nằm ngoài phương thức `_jspService()`
  - Declarations không được truy cập các Implicit objects (VD: đối tượng `HttpSession`)
- Thường được sử dụng với Expressions hoặc Scriptlets
- Để thực hiện thao tác khởi tạo và dọn dẹp trong trang JSP, sử dụng declarations để override phương thức `jspInit()` và `jspDestroy()`
- Cú pháp:
  - `<%! Mã nguồn khai báo biến hoặc phương thức %>`
  - `<jsp:declaration> Mã nguồn khai báo biến hoặc phương thức </jsp:declaration>`

# Ví dụ

**<H1>Some heading</H1>**

**<%!**

```
private String randomHeading() {  
    return("<H2>" + Math.random() + "</H2>");  
}
```

**%>**

**<%= randomHeading() %>**

# Tại sao nên sử dụng cú pháp XML?

- **Hỗ trợ từ JSP 1.2**
- **Ví dụ**
  - `<jsp:expression>Expression</jsp:expression>`
  - `<jsp:scriptlet> Java code</jsp:scriptlet>`
  - `<jsp:declaration> declaration code</jsp:declaration>`
- **Lợi ích**
  - XML validation (qua XML schema)
  - Có sẵn nhiều công cụ XML
    - editor
    - transformer
    - Java APIs

# Directives

- **Directives là các thông điệp (messages) chuyển đến JSP container, hướng dẫn cách biên dịch trang JSP**
- **Không sinh ra output**
- **Cú pháp**
  - **`<%@ directive {attr=value}* %>`**



# 3 loại Directives

- **page**: Các thuộc tính của trang JSP
  - `<%@ page import="java.util.*" %>`
- **include**: Được sử dụng để đính kèm text/mã nguồn khi dịch trang JSP thành Servlet
  - `<%@ include file="header.html" %>`
- **Taglib**: Chỉ ra 1 thư viện thẻ mà JSP container cần phải **interpret**
  - `<%@ taglib uri="mytags" prefix="codecamp" %>`

# Page Directives

- **Đưa thêm các thông tin vào Servlet biên dịch từ trang JSP tương ứng**
- **Cho phép điều khiển**
  - **Import các class nào**
    - `<%@ page import="java.util.*" %>`
  - **Loại MIME sinh ra là gì**
    - `<%@ page contentType="MIME-Type" %>`
  - **Xử lý đa luồng như thế nào**
    - `<%@ page isThreadSafe="true" %>` `<%!--Default --%>`
    - `<%@ page isThreadSafe="false" %>`
  - **Xử lý các lỗi không mong đợi như thế nào**
    - `<%@ page errorPage="errorpage.jsp" %>`



TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI  
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

# Custom tags

# Custom tags là gì?

- Là các phần tử JSP do User tự định nghĩa (ngược với các thẻ chuẩn tắc: **standard tags**)
- Đóng gói các tác vụ phải thực hiện nhiều lần
- Lấy từ thư viện thẻ (**tag library**) tự định nghĩa

# Tại sao cần custom tags?

- **Tách biệt phần presentation với phần business logic (và các chức năng khác)**
  - Thiết kế trang: thực hiện phần presentation
  - Business logic developers: tạo các custom tags
- **Đóng gói phần business logic**
  - Tái sử dụng và bảo trì dễ dàng
- **Đơn giản cho người thiết kế trang**
  - Không cần biết Java, sử dụng cú pháp đơn giản

# Khai báo một thư viện thẻ

- Phải có **taglib directive** trước khi sử dụng các thẻ
- Cú pháp
  - `<%@taglib prefix="myprefix" uri="myuri"%>`
  - prefix: tên của thư viện thẻ (tùy chọn tên)
  - uri: xác định duy nhất **tag library descriptor** (TLD), trực tiếp hoặc gián tiếp

# Cú pháp Custom Tag trong trang JSP

**<prefix:tag attr1="value" ... attrN="value" />**

- **Hoặc**

**<prefix:tag attr1="value" ... attrN="value" >**

**body**

**</prefix:tag>**

- **prefix**: tên của thư viện thẻ
- **tag**: định danh của thẻ (trong thư viện thẻ tương ứng)
- **attr1 ... attrN**: các thuộc tính của thẻ

# Ví dụ cú pháp JSP 1.2 với Scriptlets

```
<%-- Output Shopping Cart --%>
<%@ page import="com.acme.util.*" %>
<%@ taglib prefix="util" uri="http://mytaglib" %>
```

```
<html>
  <body>
    <util:getShoppingCart var="cart" />
    <table>
      <% for( int i = 0; i < cart.size(); i++ ) {
        CartItem item=(CartItem)cart.get(i);
      %>
        <tr>
          <td><%= item.getName() %></td>
          <td><%= item.getPrice() %></td>
        </tr>
      <% } %>
    </table>
  </body>
</html>
```



# Ví dụ cú pháp JSP 2.0, không cần Scriptlets

```
<%-- Output Shopping Cart --%>
<%@ taglib prefix="util" uri="http://mytaglib" %>
<%@ taglib prefix="c"
    uri="http://java.sun.com/jsp/jstl/core" %>
<html>
  <body>
    <util:getShoppingCart var="cart" />
    <table>
      <c:forEach var="item" values="{cart}">
        <tr>
          <td>{item.name}</td>
          <td>{item.price}</td>
        </tr>
      </c:forEach>
    </table>
  </body>
</html>
```

# JSTL là gì?

- Tập chuẩn các thư viện thẻ
- Đóng gói các chức năng chung trong rất nhiều ứng dụng JSP
  - Điều kiện và lặp
  - XML
  - Truy cập database
  - internationalized formatting
- Vẫn đang được tiếp tục phát triển

# Tại sao cần JSTL?

- LTV không cần phải tự viết các thẻ
- LTV học và sử dụng tập chuẩn các thư viện thẻ được cung cấp bởi Java EE platforms
- Các nhà cung cấp đã tối ưu hóa việc cài đặt
- Đảm bảo tính Portability cho ứng dụng

# Các thư viện thẻ JSTL

- **Core (prefix: c)**
  - Variable support, Flow control, URL management
- **XML (prefix: x)**
  - Core, Flow control, Transformation
- **Internationalization (i18n) (prefix: fmt)**
  - Locale, Message formatting, Number and date formatting
- **Database (prefix: sql)**
  - SQL query and update
- **Functions (prefix: fn)**
  - Collection length, String manipulation

# Khai báo các thư viện thẻ JSTL

- **Core**
  - `<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>`
- **XML**
  - `<%@ taglib prefix="x" uri="http://java.sun.com/jsp/jstl/xml"%>`
- **Internationalization (i18n)**
  - `<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>`
- **Database (SQL)**
  - `<%@ taglib prefix="sql" uri="http://java.sun.com/jsp/jstl/sql"%>`
- **Functions**
  - `<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions" %>`



TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI  
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

# Các thẻ cơ bản (core tags)

# Core Tags Types (1)

- **Hỗ trợ về biến (Variable)**

- **<c:set>**
- **<c:remove>**

- **Điều kiện**

- **<c:if>**
- **<c:choose>**
  - **<c:when>**
  - **<c:otherwise>**

- **Lặp**

- **<c:forEach>**
- **<c:forTokens>**

# Core Tags Types (2)

- **Quản lý URL**
  - **<c:import>**
    - <c:param>
  - **<c:redirect>**
    - <c:param>
  - **<c:url>**
    - <c:param>
- **Mục đích chung khác**
  - **<c:out>**
  - **<c:catch>**



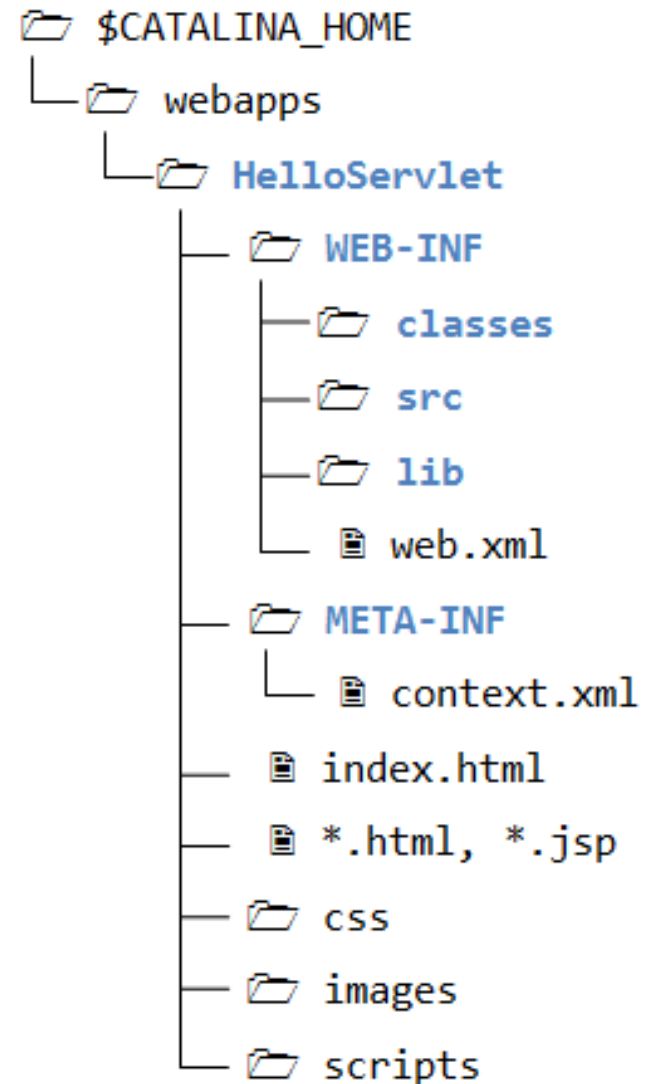


TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI  
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

# Development Tree Structure

# Development Tree Structure

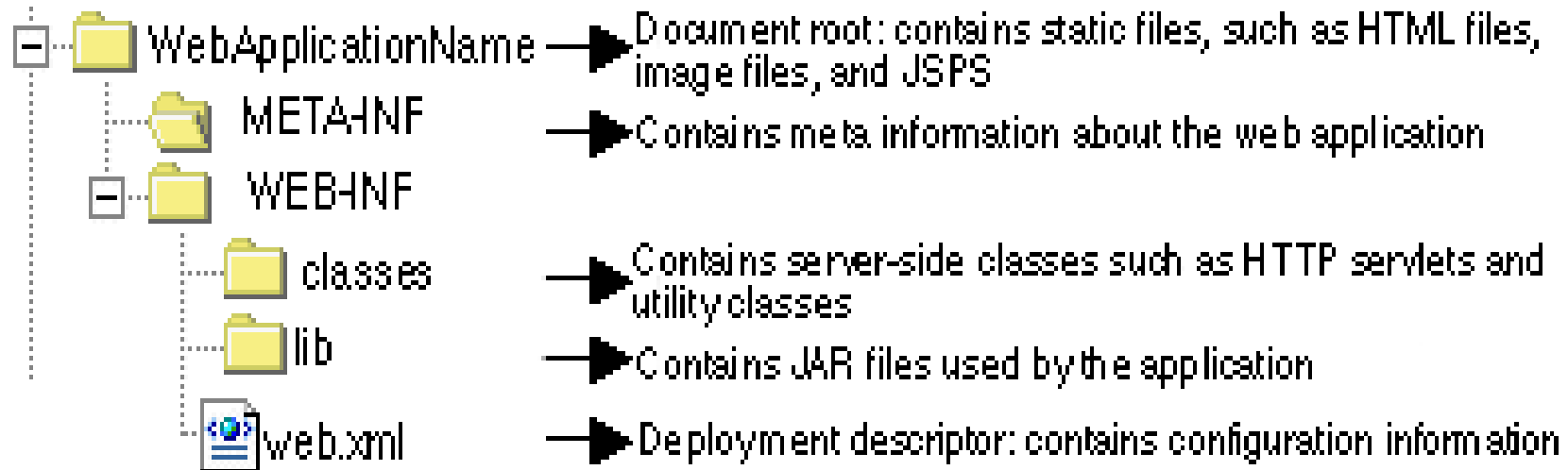
- Tạo cấu trúc thư mục (Development Tree Structure)
  - Cần tách biệt mã nguồn với các file biên dịch
  - Thư mục gốc bao gồm:
    - **src**: Mã nguồn Java cho các servlets và các JavaBeans
    - **web**: các trang JSP, HTML, images, CSS, javascript
    - **web.xml**



# Document Root & Context

- **Document Root** của ứng dụng Web
  - Thư mục mức 1 trong file WAR
  - Chứa các trang JSP pages, thư viện, lớp phía client, tài nguyên Web tĩnh
  - Chứa cả thư mục WEB-INF
- Một **context** là 1 định danh (name) được map với **document root** của 1 ứng dụng Web
  - /hello1 là context cho ví dụ hello1
  - Phân biệt các ứng dụng Web trong **MỘT** Web container
  - Được chỉ định như 1 phần của URL

# Cấu trúc thư mục của 1 file \*.WAR



# Thư mục WEB-INF

- Là thư mục con của **Document root**
- Chứa:
  - web.xml : **Web application deployment descriptor**
  - Các file **JSP tag library descriptor**
  - Classes : là thư mục chứa các lớp phía server: servlets, lớp tiện ích, các JavaBeans
  - lib : là thư mục chứa các file thư viện JAR (tag libraries, các thư viện tiện ích được gọi bởi các lớp phía server)

# HTTP request URL & Web component URL (alias) & Context

- **Request URL: yêu cầu truy cập tới 1 tài nguyên Web**
  - `http://[host]:[port]/[request path]?[query string]`
  - [request path] bao gồm **context** + URN của component
  - `http://localhost:8080/hello1/greeting?username=Minh`
- **Một số ký hiệu đường dẫn đặc biệt**

Ký hiệu	Ý nghĩa
/	Thư mục gốc của website
./	Thư mục hiện tại của trang web đang sử dụng (mặc định)
../	Quay lên thư mục cha

# Ví dụ

[-] root # 127.0.0.1/demo

file A

[-] Thu mục 1

file B

Thu mục 1\_1

file C

[-] Thu mục 1\_2

file D

Thu mục 1\_2\_1

file E

Thu mục 2

file F

**file C cần tạo liên kết đến file D:**

**<a href="URL">liên kết đến D</a>**

**URL =**

**[/demo/Thu mục 1/Thu mục 1\\_2/file D.htm](/demo/Thu mục 1/Thu mục 1_2/file D.htm)**

**[../../Thu mục 1\\_2/file D.htm]( ../../Thu mục 1_2/file D.htm)**

**[../Thu mục 1\\_2/file D.htm]( ../Thu mục 1_2/file D.htm)**

**[http://127.0.0.1/demo/Thu mục 1/  
Thu mục 1\\_2/file D.htm](http://127.0.0.1/demo/Thu mục 1/Thu mục 1_2/file D.htm)**