**Câu 1: phân biệt jsp với servlet**

***Về servlet***

* Cần nắm được định nghĩa nhưng ngắn gọn là
* nó là nơi xử lý các request đến từ người dung và trả về kết quả xử lý trong đó có thể là html luôn
* nó là thuần code java, phun html bằng các câu lệnh out.print

***Về JSP***

* nó đơn giản hóa đi, cho mình viết code html
* nó sẽ generate ra out.print cho mình như vậy tiện cho việc layout, thiết kế html

**Câu 2: phân biệt web tĩnh và web động**

* web tĩnh được viết bằng ngôn ngữ HTML, Javascript, CSS..etc…

Với web tĩnh, khi server nhận request từ trang web và server sẽ trả về respone (ko làm thêm 1 tiến trình nào nữa.)

* Nội dung vẫn như vậy chỉ khi được thay đổi bơi 1 ai đó.
* web động được viết bởi JAVA, ASP.NET.. Nội dung trang khác nhau với mỗi khách hàng xem web.

|  |  |  |
| --- | --- | --- |
| **SL.NO** | **STATIC WEB PAGE** | **DYNAMIC WEB PAGE** |
| 1. | In static web pages, Pages will remain same until someone changes it manually. | In dynamic web pages, Content of pages are different for different visitors. |
| 2. | Static Web Pages are simple in terms of complexity. | Dynamic web pages are complicated. |
| 3. | In static web pages, Information are change rarely. | In dynamic web page, Information are change frequently. |
| 4. | Static Web Page takes less time for loading than dynamic web page. | Dynamic web page takes more time for loading. |
| 5. | In Static Web Pages, database is not used. | In dynamic web pages, database is used. |
| 6. | Static web pages are written in languages such as: HTML, JavaScript, CSS, etc. | Dynamic web pages are written in languages such as: CGI, AJAX, ASP, ASP.NET, etc. |

* Nhiều thời gian để load trang web hơn.
* Được dùng khi nội dung thay đổi thường xuyên như chứng khoán hoặc thời tiết

**Difference between Static and Dynamic Web Pages:**

**Câu 3: Các loại scope trong servlet**

* **Scope là định nghĩa về thời gian mà dữ liệu của bạn có tồn tại cho người dùng trong ứng dụng web.**
* Có 4 loại
* **page**: biến sẽ chỉ tồn tại trong một trang jsp, hay trong một trang servlet
* **request**: Request scope bắt đầu ngay khi một HTTP request được gửi tới server và kết thúc khi server trả về một HTTP response
* Các tham số/thuộc tính trong một Request scope có thể được truy cập từ các servlet hoặc jsp cùng phục vụ trong 1 request. Ví dụ bạn gọi 1 servlet/jsp sau đó các servlet/jsp này lại gọi các servlet/jsp khác rồi mới trả về reponse.
* **session**: biến được để trong session có thể được sử dụng mọi nơi, mọi trang cho đến khi người dùng tắt cái trang web, trình duyệt của mình hay hết thời gian session thì mới biến mất nên nó hay dc dung để lưu trạng thái log in của người dung
* Các giá trị bên trong session scope có thể được truy cập từ nhiều request khác nhau của cùng 1 client
* Một tính năng đáng chú ý của trình duyệt web (browser) là  session được chia sẻ giữa các tab khác nhau. Bạn có thể gửi request từ 1 tab khác mà không cần login lại. Ví dụ bạn login gmail ở 1 tab, sau đó bạn sang tab khác vẫn có thể tải mail, gửi mail mà không cần phải login lại. Do đó, hãy luôn mở một cửa sổ trình duyệt mới khi bạn muốn thực hiện các giao dịch bảo mật.
* **application:** nó tồn tại khi server vẫn đang chạy cho đến khi stop server bật lại nó mới mất, nên hay dùng để đếm số lượng user, Tổng truy cập

**Câu 4: Java bean**

* JavaBean nói một cách đơn giản chỉ là một class thông thường trong đó các thuộc tính được lưu trữ và truy xuất bằng các phương thức getter và setter.
* JSP truy cập Java Bean qua các tag action và nhận kết quả trả về mà không cần biết cấu trúc của Java Bean và cách thức xử lý của nó.
* java bean là một object như model có đặc điểm : một class muốn là java bean thì ko có hàm khởi tạo các thuooc tính phải private và truy suất thông qua getter/setter

**Câu 5: Các cách truyền dữ liệu giữa các page ( Nghĩa là xử lý hay tính toán ở page này truyền dữ liệu sang trang jsp hay sevlet khác bằng cách nào)**

* Có nhiều cách
* cách thông dụng là request set parameter: cách này nó chỉ gửi đến một trang duy nhất (nó tương ứng vs scope page context bên trên)
* Cách set biến 3 loại scope: application, session, request



**Câu 6: Nắm về mô hình MVC**

* Mô hình MVC thực chất là nó phân tách cấu trúc project web ra làm 3 phần
* model là các class đối tượng, mục đích các class này là để lưu trữ dữ liệu tạm thời, là đối tượng trung gian để load dữ liệu từ db lên hay lưu dữ liệu xuống. Thông qua các model sẽ dễ dàng làm việc hơn. Nói chung là class nào d ùng để chứa thoing tin hoặc chứa các phương thức hỗ trợ gọi là model
* Ví dụ bài tập funix: quản lý các bài post thì mình cần tạo một class post để lưu các thông tin của một bài post cần tạo class user để lưu các thông tin của một user => các class đó gọi là model
* View: các code nào để tạo ra giao diện gọi là view (kể cả web app hay desktop app)
* Controller: để điều khiển, điều hướng, kiểm soát, tính toán nhẹ làm công việc tiếp nhận yêu cầu gọi xử lý rồi trả lời cho view. Vai trò trung gian điều phối giữa model và view. Gọi các hàm xử lý của model (có thể xử lý nhỏ một ít) và gửi lại cho view
* Áp dụng trong môn web java thì Jsp các page hiển thị chính là view, các class là model, Sevlet làm controller
* Jsp có làm controller dc ko? 🡪 có thể làm controller dc luôn vì nó thật ra vẫn là servlet, vẫn code điều hướng xử lý dc
* Tương tự sevlet ng ta cũng cos thể làm view dc luôn nhưng ít ai làm thế vì ng ta sẽ làm theo chuyên biệt : Jsp chuyên hiển thị và Sevlet chuyên điều hướng để dễ quản lý , dễ bảo trì
* Phân biệt mô hình MVC1 v à MVC2
* Mô hình mvc1: Trang jsp vừa làm view vừa làm controller luôn, Model thì vẫn riêng
* Mô hình mvc2 là tách biệt rõ ra ba phần JSP là view, Servlet làm controller, Các class làm model

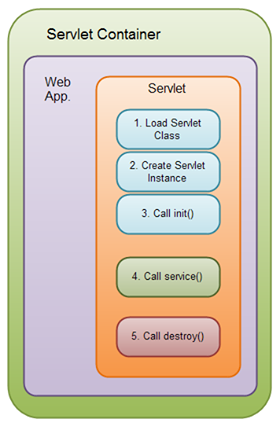
**Câu 7: file web xml cấu hình được những gì**

* file web.xml có thê cấu hình mapping (mapping sevlet) nghĩa là nếu muốn đặt tên khác cho đường dẫn truy cập trang sevlet thì chỉ cần đặt tên khác và mapping tới cái servlet muốn chay tới
* công dụng tiếp theo của web.xml là để set file chạy đầu tiên khi run project web

**Câu 8: cấu trúc thư mục web**

* web page: chứa html css js image, (jsp nằm ở web page luôn)
* source package: chứa các file java, sevlet
* library: chứa các thư viện jdk, tomcat

**Câu 9: tìm hiểu về vòng đời của servlet (servlet lifecycle)**



* Khi có HTTP requests từ người dùng tới server, thì các yêu cầu này sẽ được ủy nhiệm cho Servlet Container
* Ngay khi nhận được yêu cầu đầu tiên của người dùng thì các bưới 1, 2, 3 sẽ được thực thi và chỉ thực thi một lần duy nhất
* Sau đó, Servlet Container sẽ gọi phương thức service() ở bước 4 để xử lý các yêu cầu của người dùng. Phương thức service() sẽ được thực thi nhiều lần
* Mỗi khi nhận một yêu cầu cho một Servlet, thì Servlet Container tạo một Thread mới và triệu hồi service(). Phương thức service() kiểm tra kiểu yêu cầu HTTP (Kiểu GET, POST, PUT, DELET, .v.v.) và gọi các phương thức doGet, doPost, doPut, doDelete .v.v. tương ứng một cách thích hợp.

**Câu 10: Cách truyền gửi dữ liệu từ form xuống controller**

* Mình dùng form và lấy theo name của các input
* Dưới controller thì request.getparameter

**Câu 11: các element sử dụng để viết code java trong jsp**

* Có 5 loại
* jsp expression: Để hiển thị giá trị của biến hay biểu thức tính toán. Ký hiệu là <%=
* jsp scriptlet: Để viết code java vào trang jsp. Ký hiệu là <%
* jsp declaration: Để khai báo thuộc tính hay định nghĩa phương thức. Ký hiệu là <%!
* jsp directive: để cung cấp các chỉ lệnh tới container mà liên quan tới JSP page hiện tại . Ký hiệu <%@
* jsp comment Để comment code. <%—

**Câu 12: nắm một số object có sẵn trong jsp**

* Out, request, response, config, application, session…..

**Câu 13: Jstl để làm gì**

* Để tạo sự đồng bộ trong trang jsp, tất cả đều là các thẻ dom ko có chèn code vào. Khi mình chèn code java vào làm nó ko thoings nhất, toàn vẹn
* nó giúp viết code ngắn, rõ ràng hơn, giúp các chương trình layout html ko bị lỗi những chỗ có mã scriptlet
* VD:



Đây là cách kiểm tra user đăng nhập có tên là jheaton hay ko. Thay vì gọi hàm getName thì chỉ việc “.name” như một thuộc tính

MVC và mô hình 3 lớp: <https://techblog.vn/phan-biet-mo-hinh-3-layer-va-mvc>

Web server là một loại [máy chủ](https://www.codehub.vn/May-chu-Server-La-Gi) (hay server) được dùng để xử lý các truy cập được gửi từ máy khách thông qua giao thức HTTP. Các truy cập HTTP này thường được gửi từ các chương trình duyệt web trên máy tính cá nhân.

Như chúng ta đã biết web server dựa trên giao thức http là một giao thức stateless. Tức là mỗi một lần có một user và trang web của chúng ta sẽ tương ứng với một **cặp** : request và response.

**Servlet là gi? =>** Để biết được nên response gì thì chúng ta cần biết thông tin của request.  
Trong java để mô tả **cặp** thông tin này thì chúng ta được cung cấp một class gọi là **Servlet**.

Jsp và html

## **JSP là gì**

JSP là viết tắt của **Java Server Pages** . Các tệp này có phần mở rộng .jsp. Ưu điểm chính của JSP là lập trình viên có thể chèn mã Java bên trong HTML. Có các thẻ JSP để chèn mã Java. Lập trình viên có thể viết thẻ <% để bắt đầu mã Java và viết thẻ%> ở cuối mã Java. Hơn nữa, có các thẻ JSP khác nhau để thực hiện các nhiệm vụ khác nhau. Có các thẻ để chia sẻ dữ liệu giữa các yêu cầu và trang, vượt qua sự kiểm soát giữa các trang và để lấy dữ liệu từ cơ sở dữ liệu, v.v.

JSP được xây dựng dựa trên công nghệ dựa trên [Java có](https://pediaa.com/what-is-the-difference-between-jsp-and-servlet/#Servlet) tên là [Java Servlet](https://pediaa.com/what-is-the-difference-between-jsp-and-servlet/#Servlet) . Thông thường, JSP làm việc với Servlets để xử lý logic nghiệp vụ phức tạp trong các ứng dụng Enterprise. JSP có quyền truy cập vào nhiều API Java như JDBC, JNDI và EJB. Nhìn chung, JSP giúp tạo các trang web động. Nó giúp hoàn thành các nhiệm vụ như thu thập dữ liệu người dùng từ các trang web và thể hiện dữ liệu từ cơ sở dữ liệu, v.v.

## **HTML là gì**

HTML là một ngôn ngữ đánh dấu giúp xây dựng cấu trúc của trang web. Vì nó là một ngôn ngữ đánh dấu, nó bao gồm nhiều thẻ. Có các thẻ để hiển thị văn bản, bảng, danh sách theo thứ tự, danh sách không có thứ tự, v.v ... Một trang HTML bao gồm hai phần chính: phần đầu và phần thân. Siêu dữ liệu (dữ liệu mô tả trang) nằm trong phần đầu trong khi phần thân bao gồm tất cả các thẻ cần thiết để thể hiện nội dung hiển thị của trang web.

Có nhiều phiên bản HTML khác nhau. Phiên bản mới nhất là [HTML 5](https://pediaa.com/what-is-the-difference-between-html-and-html5/#HTML5) . Nó có các tính năng nâng cao hơn như Geo-location, âm thanh gốc và hỗ trợ video, Canvas, ổ cắm web, v.v. Thông thường, HTML là một ngôn ngữ dễ học và dễ sử dụng. Một lập trình viên có thể tạo một tệp HTML bằng trình soạn thảo văn bản đơn giản và thực hiện nó bằng trình duyệt web.

## **Mối quan hệ giữa JSP và HTML**

* JSP cho phép chèn mã Java bên trong tệp HTML.

## **Sự khác biệt giữa JSP và HTML**

### **Định nghĩa**

JSP là một công nghệ giúp các nhà phát triển phần mềm tạo các trang web được tạo động dựa trên HTML. HTML là ngôn ngữ đánh dấu tiêu chuẩn để tạo các trang web và ứng dụng web. Vì vậy, đây là sự khác biệt chính giữa JSP và HTML.

### **Mẫu dài**

JSP là viết tắt của Java Server Pages trong khi HTML là viết tắt của Hyper Text Markup Language.

### **Sử dụng**

Hơn nữa, một sự khác biệt khác giữa JSP và HTML là JSP giúp phát triển các ứng dụng web động trong khi HTML giúp tạo cấu trúc của các trang web.

### **Phần kết luận**

Sự khác biệt chính giữa JSP và HTML là JSP là một công nghệ để tạo các ứng dụng web động trong khi HTML là ngôn ngữ đánh dấu tiêu chuẩn để tạo cấu trúc của các trang web. Tóm lại, tệp JSP là một tệp HTML có mã Java.