**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

------------------------------

**ĐỒ ÁN TỐT NGHIỆP BÁO CÁO ĐỊNH KỲ ĐỢT 1**

***Đề tài*:**

**“XÂY DỰNG ỨNG DỤNG GỌI ĐỒ ĂN ĐỒ ĂN/THỨC UỐNG TRỰC TUYẾN QUA MẠNG INTERNET”**

## Người hướng dẫn : ThS. NGUYỄN ANH HÀO Sinh viên thực hiện : TRẦN NHẬT BẢO

**Mã số sinh viên : N13DCCN012 Lớp : D13CQCP01\_N**

**Khoá : 2013 - 2018**

**Hệ : ĐẠI HỌC CHÍNH QUY**

**TP.HCM, tháng 10 /2017**

**CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**Độc lập- Tự do- Hạnh phúc**

TP. Hồ Chí Minh, ngày tháng 10 năm 2017

## NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC

1. **Tên đề tài: “Xây dựng ứng dụng gọi đồ ăn đồ ăn/thức uống trực tuyến qua mạng Internet”**
2. **Sinh viên: TRẦN NHẬT BẢO Lớp:D13CQCP01\_N**
3. **Giáo viên hướng dẫn: Th.S NGUYỄN ANH HÀO**
4. **Nơi công tác:**

**NỘI DUNG NHẬN XÉT**

1. Đánh giá chung:

…………………………………………………………………………………………

…………………………………………………………………………………………

1. Đánh giá chi tiết:

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

1. Nhận xét về tinh thần, thái độ làm việc:

…………………………………………………………………………………………

…………………………………………………………………………………………

1. Kết luận:

…………………………………………………………………………………………

…………………………………………………………………………………………

1. Điểm hướng dẫn ():

## GIẢNG VIÊN HƯỚNG DẪN

(Ký, ghi rõ họ tên)

Kính thưa quý thầy cô!

Trong quá trình thực hiện đề tài “Xây dựng ứng dụng gọi đồ ăn đồ ăn/thức uống trực tuyến qua mạng Internet”, em đã nhận được rất nhiều sự giúp đỡ , tạo điều kiện của tập thể Ban Giám Hiệu, Khoa Công Nghệ Thông Tin, các thầy cô, cán bộ các phòng ban chức năng Học Viện Công Nghệ Bưu Chính Viễn Thông tại TP.HCM. Em xin bày tỏ lòng cảm ơn chân thành về sự giúp đỡ đó.

Đặc biệt, em xin bày tỏ lòng biết ơn sâu sắc tới ThS.Nguyễn Anh Hào. Thầy đã hướng dẫn tận tình, truyền đạt kiến thức, chỉ bảo em hoàn thành đề tài này.

Sau cùng, em xin kính chúc các quý thầy cô Khoa Công Nghệ Thông Tin và ThS.Nguyễn Anh Hào thật dồi dào sức khỏe, niềm tin để tiếp tục ươm mầm những thế hệ mai sau.

Trân trọng!

TP.HCM, ngày 13 tháng 10 năm 2017

Sinh viên thực hiện

Trần Nhật Bảo

# LỜI MỞ ĐẦU

Công nghệ thông tin là một ngành khoa học đang phát triển mạnh ứng dụng rộng rãi trên nhiều lĩnh vực khác nhau. Cùng với xu hướng phát triển của các phương tiện truyền thông như báo chí, truyền hình, radio… thì việc sử dụng Internet ngày càng phổ biến. Truy cập Internet, chúng ta có được một kho thông tin khổng lồ phục vụ cho mọi nhu cầu, mục đích của con người.

Nhận thức được nhu cầu tìm hiểu thông tin, giải trí của xã hội, là sự ra đời của hàng loạt các website cho các mục đích thương mại, giải trí, tin tức…Mua sắm là một trong những nhu cầu tất yếu của con người, đặc biệt là nhu cầu về ẩm thực. Để đáp ứng được việc cập nhật thông tin, giá thành sản phẩm, mỗi nhà hàng/quán ăn đều có một website riêng để hỗ trợ khách hàng.

Chính vì vậy, đề tài: **“Xây dựng ứng dụng gọi đồ ăn đồ ăn/thức uống trực tuyến qua mạng Internet”** là một website được thiết kế và thực hiện nhằm bắt kịp các công nghệ mới và đáp ứng nhu cầu thực tế đối với người dùng.

Tuy nhiên, vì thời gian và kiến thức còn hạn chế em đã cố gắng rất nhiều để có thể hoàn thành đề tài, nhưng trong quá trình làm bài em không tránh khỏi những thiếu sót, kính mong được lời nhận xét và góp ý của thây cô.

# PHẦN I: LÝ THUYẾT

## Tìm hiểu về quy trình bán thức ăn:

* + Địa điểm tham khảo:
  + Nhà hàng Pasta Paradise – thiên đường Mì Ý, địa chỉ: 224 Nguyễn Thị Minh Khai, Phường 6, Quận 3, Hồ Chí Minh.
  + Nhà hàng mì Ý Double B, địa chỉ: Hẻm 391 Trần Hưng Đạo, Cầu Kho, Quận 1, Hồ Chí Minh
  + Tham khảo từ một số trang web bán hàng của các quán ăn/nhà hàng:
    - <http://www.urbanstation.com.vn/>,
    - <http://dominos.vn/>,
    - <http://www.thecoffeehouse.com/>
  + Sau khi đã tìm hiểu kỹ về quá trình bán thức ăn của các quán kể trên, em nhận ra một website bán thức ăn cần có những yếu tố sau:
* Cần cho khách hàng biết rõ những sản phẩm mình bán.
* Cần cho khách hàng biết về các chương trình khuyến mãi, cái ưu đãi trong ngày.
* Cần cho khách hàng thanh toán trực tuyến.
* Cần hỗ trợ giao hàng tận nơi.
* Đối với các quán ăn, cần cho khách hàng biết rõ tình trạng của quán, quán còn bàn không? Muốn đặt bàn trước thì có thể đặt được không?

## Tìm hiểu về Spring Framework: https://projects.spring.io/spring-framework/

## Giới thiệu Spring Framework

* + - **Spring Framework** là một [bộ khung ứng dụng](https://vi.wikipedia.org/w/index.php?title=B%E1%BB%99_khung_%E1%BB%A9ng_d%E1%BB%A5ng&amp;action=edit&amp;redlink=1) và [bộ chứa](https://en.wikipedia.org/wiki/Servlet_container) [đảo ngược điều khiển](https://en.wikipedia.org/wiki/inversion_of_control) cho [nền tảng Java](https://vi.wikipedia.org/wiki/Java_(c%C3%B4ng_ngh%E1%BB%87)). Chức năng tính của bộ khung này có thể áp dụng cho bất kỳ ứng dụng Java nào, tuy vậy, nhiều bản mở rộng dành cho việc xây dựng ứng dụng nền web cũng được phát triển trên nền tảng [Java EE](https://vi.wikipedia.org/w/index.php?title=Java_EE&amp;action=edit&amp;redlink=1). Tuy bộ khung không ấn định lên một cụ thể nào, Spring Framework đã trở nên phổ biến trong cộng đồng [Java](https://vi.wikipedia.org/wiki/Java_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)) như một phương án song hành, thay thế, hoặc thậm chí bổ sung cho mô hình [Enterprise JavaBeans](https://vi.wikipedia.org/w/index.php?title=Enterprise_JavaBeans&amp;action=edit&amp;redlink=1) (EJB). Đây là bộ khung [mã nguồn mở](https://en.wikipedia.org/wiki/open_source).
    - Spring là một framework giúp các nhà phát triển xây dựng những hệ thống và ứng dụng chạy trên JVM một cách đơn giản, tiện gọn, nhanh chóng và mềm dẻo.

## Đôi nét về lịch sử

### EJB và Spring

* + - * Có thể nói rằng Spring Framework ra đời là nhờ sự thành công của **EJB**. EJB (Enterprise JavaBeans) là một kiến trúc phát triển dành cho các ứng dụng doanh nghiệp đòi hỏi tính mạnh mẽ và khả năng mở rộng cao. EJB là một thành phần quan trọng của nền tảng Java EE. EJB được tạo ra bởi hai gã khổng lồ công nghệ là IBM và Sun vào những năm cuối thập niên 90. Dưới sự phát triển của hai tập đoàn này, EJB đã trở nên nặng nề hơn. Tiêu chuẩn EJB cũng nhanh chóng được thông qua. Sau tất cả, EJB được ca ngợi là giải pháp tốt nhất để xây dựng các ứng dụng phân tán dành cho doanh nghiệp.
      * Nhưng rồi những chỉ trích về chi tiết kỹ thuật của EJB đã tăng nhanh còn hơn sự nổi tiếng của nó. EJB là một framework lớn với độ phức tạp cao. Các nhà phát triền khi làm việc với EJB đã nhận ra rằng nó không hoạt động như những gì nó đã đề ra. EJB chỉ tuyệt vời trên bàn giấy, nhưng ra thực tế đó lại là một chuyện khác.
      * EJB vẫn được tiếp tục phát triển, dễ sử dụng hơn nhiều so với buổi ban đầu. Nhưng những chỉ trích vẫn còn đó, nỗi thất vọng trong cộng đồng Java ngày càng tăng lên, đòi hỏi cần có một giải pháp thay thế.

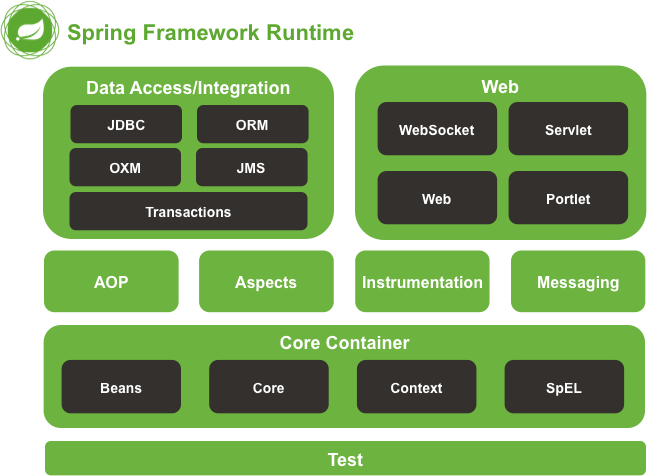
### Sự ra đời của Spring

* + - * Vào năm 2002, Spring Framework phát hành phiên bản đầu tiên bởi Rod Johnson. Việc xây dựng các ứng dụng doanh nghiệp trở nên đơn giản và dễ dàng hơn. Rod Johnson đã cho xuất bản một cuốn sách được phổ biến rộng rãi có nhan đề "J2EE Development without EJB". Điều này đã cho Spring Framework rất nhiều động lực để thay thế EJB.
      * Ngày nay, Spring đã trở thành framework mã nguồn mở phổ biến nhất để xây dựng các ứng dụng doanh nghiệp. Cách tiếp cận thực tế ban đầu của Rod Johnson tiếp tục được phát triển và hướng tới một bộ công cụ hoàn chỉnh dành cho xây dựng các ứng dụng doanh nghiệp. Theo một số nguồn, trên 50% các ứng dụng web Java hiện nay đang sử dụng Spring.
      * Để ngăn chặn sự phức tạp trong phát triển các ứng dụng, Spring Framework thường dựa trên các quan điểm như sau:
      * Đơn giản hóa công việc phát triển thông qua việc sử dụng các đối tượng Java đơn giản hay còn được gọi là POJO (Plain Old Java Object)
      * Nới lỏng ràng buộc giữa các thành phần thông qua việc sử dụng Dependency Injection và viết các interface
      * Tiếp cận lập trình khai báo bằng cách sử dụng các quy tắc (convention) và các khía cạnh (aspect) chung
      * Giảm thiểu các mã nghi thức và soạn sẵn (boilerplate) thông qua việc sử dụng các khuôn mẫu (template) và các khía cạnh

### Spring Core

Spring Core chính là thành phần trung tâm, cốt lõi của Spring Framework. Đây chính là nền tảng để xây dựng nên các thành phần khác.

### Lược đồ dưới đây mô tả mối quan hệ giữa Spring Core với các thành phần khác trong Spring Framework:



### Spring Bean

* + - * Spring Bean là trung tâm của Spring Core và là trái tim của một ứng dụng Spring. Trái ngược với EJB, Spring Framework được thiết kế từ lõi bằng cách sử dụng các POJO hay các Spring Bean. Spring Bean có thể được hiểu là các đối tượng Java đơn giản. Điều này khá tương đồng với nguyên lý thiết kế "Đơn trách nhiệm" (single responsibility) của Robert C Martin trong lập trình hướng đối tượng.

### Dependency Injection (DI)

* + - * Dependency Injection (có thể dịch tiêm các thành phần phụ thuộc) là một sức mạnh nổi bật của Spring Framework. Trong các phiên bản gốc của EJB người ta không thấy sự xuất hiện của Dependency Injection, nhưng có lẽ do nhận thấy sự thành công từ Spring khi sử dụng Dependency Injection, các phiên bản EJB 3.x đã cung cấp các annotation để thực hiện inject các dependency. Dependency Injection là một mẫu thiết kế phần mềm mà các đối tượng phụ thuộc sẽ được inject vào một lớp nào đó. Dependency Injection là một implementation cụ thể của khái niệm Inversion of Control (đảo ngược điều khiển).
      * Ta sẽ có một ví dụ đơn giản về Dependency Injection như sau: Bạn có một web controller có nhiệm vụ lưu thông tin gửi từ form người dùng. Theo nguyên lý Đơn trách nhiệm, bạn không muốn lớp controller tương tác với cơ sở dữ liệu. Thay vào đó, bạn sẽ sử dụng một lớp service để làm công việc này. Như vậy, controller của bạn sẽ chỉ phải xử lý dữ liệu của form (get form data, validate data, ...) rồi gọi một phưong thức của lớp service được inject để lưu dữ liệu. Controller không cần phải quan tâm hay lo lắng về kết nối cơ sở dữ liệu, pooling hay bảng nào sẽ được update. Cũng như service không cần phải biết request có những thông tin gì.

### Spring Context

* + - * Spring Context mang mọi thứ lại với nhau. Spring Context kế thừa các tính năng của Spring Bean và bổ sung các hỗ trợ cho internationalization (ví dụ như các resource bundle), event propagation, resource loading ... Ngoài ra, Spring Context cũng hỗ trợ các tính năng của Java EE như EJB, JMX và truy cập từ xa cơ bản. Interface **ApplicationContext** là tiêu điểm của Spring Context.

### Spring Expression Language (SpEL)

* + - * Spring Expression Language là một ngôn ngữ ngắn gọn giúp cho việc cấu hình Spring Framework trở nên linh hoạt hơn.

## Các dự án trong Spring Framework

* + - Trên thực tế Spring Framework là một tập hợp của nhiều dự án con. Spring Core như chúng ta đã tìm hiểu là nền tảng của các dự án trong Spring Framework. Một một dự án sẽ đảm nhận một chức năng riêng trong việc xây dựng các ứng dụng doanh nghiệp. Chúng ta sẽ cùng điểm qua các dự án lớn trong Spring.
    - Spring MVC
    - Spring MVC được thiết kế dành cho việc xây dựng các ứng dụng nền tảng web. Đây là một dự án chúng ta không thể bỏ qua khi xây dựng các ứng dụng Java web.
    - Spring Data
    - Cung cấp một cách tiếp cận đúng đắn để truy cập dữ liệu từ cơ sở dữ liệu quan hệ, phi quan hệ, map-reduce và thậm chí còn hơn thế nữa.
    - Spring Security
    - Dự án này cung cấp các cơ chế xác thực (authentication) và phân quyền (authorization) cho ứng dụng của bạn.
    - Spring Boot
    - Spring Boot là một framework giúp chúng ta phát triển cũng như chạy ứng dụng một cách nhanh chóng.
    - Spring Batch
    - Dự án này giúp chúng ta dễ dàng tạo các lịch trình (scheduling) và tiến trình (processing) cho các công việc xử lý theo mẻ (batch job).
    - Spring Integration
    - Spring Integration là một implementation của Enterprise Integration Patterns (EIP). Dự án này thiết kế một kiến trúc hướng thông điệp hỗ trợ việc tích hợp các hệ thống bên ngoài.
    - Spring XD
    - Bạn đang băn khoăn về Big Data, tại sao không thử tìm hiểu Spring XD - mục tiêu của dự án này là đơn giản hóa công việc phát triển các ứng dụng Big Data.
    - Spring Social
    - Dự án này sẽ kết nối ứng dụng của bạn với các API bên thứ ba của Facebook, Twitter, Linkedin ...
    - Sau khi tìm hiểu, em quyết định sử dụng **spring boot** để xây dựng ứng dụng.
      * **Spring Boot** là một dự án khá nổi bật trong hệ sinh thái Spring Framework. Nếu như trước đây, công đoạn khởi tạo một dự án Spring khá vất vả từ việc khai báo các dependency trong file pom.xml cho đến cấu hình bằng XML hoặc annotation phức tạp, thì giờ đây với Spring Boot, chúng ta có thể tạo dự án Spring một cách nhanh chóng và cấu hình cũng đơn giản hơn. Dưới đây là một số tính năng nổi bật của Spring Boot:
* Tạo các ứng dụng Spring độc lập
* Nhúng trực tiếp Tomcat, Jetty hoặc Undertow (không cần phải deploy ra file WAR)
* Các starter dependency giúp việc cấu hình Maven đơn giản hơn
* Tự động cấu hình Spring khi cần thiết
* Không sinh code cấu hình và không yêu cầu phải cấu hình bằng XML ...
* Một số lợi ích của Spring Boot:
* Nó rất dễ dàng để phát triển các ứng dụng dựa trên **Spring** với **Java** hoặc **Groovy**.
* Nó giảm thiểu thời gian phát triển và tăng năng xuất
* Nó tránh việc phải viết nhiều mã nguyên mẫu (boilerplate Code), Annotations và các cấu hình XML.
* Nó dễ dàng để bạn tương tác các ứng dụng **Spring Boot** với các hệ sinh thái của **Spring** như **Spring JDBC**, **Spring ORM**, **Spring Data**, **Spring Security** etc.
* Nó đi theo cách tiếp cận “Nguyên tắc cấu hình mặc định” để giảm thiểu thời gian và nỗ lực cần thiết để phát triển ứng dụng.
* Nó cung cấp các Server nhúng (Embedded HTTP servers) như là **Tomcat**, **Jetty** .... để phát triển và test các ứng dụng web nhanh chóng và dễ dàng.
* Nó cung cấp công cụ **CLI** (Command Line Interface) dể phát triển và test các ứng dụng **Spring Boot**(**Java** hoặc **Groovy**) từ các dòng lệnh (command prompt) rất dễ dàng và nhanh chóng.
* Nó cung cấp rất nhiều các plugin để phát triển và test các ứng dụng **Spring Boot** nhanh chóng sử dụng các công cụ **Build** như **Maven** và **Gradle**
* Nó cung cấp nhiều plugin để làm việc với các cơ sở dữ liệu nhúng (embedded database) và các cơ sở dữ liệu lưu trữ trên bộ nhớ (in-memory Databases) một cách dễ dàng.

Trong spring framework, để thực hiện đăng nhập và phân quyền, ta dùng **Spring Security**:

### Spring Security

## Giới thiệu

Spring Security là một dự án nổi bật trong hệ sinh thái Spring. Spring Security cung cấp các dịch vụ bảo mật toàn diện cho các ứng dụng doanh nghiệp có nền tảng Java EE.

Spring Security cung cấp 2 cơ chế cơ bản:

**Authentication** (xác thực): là tiến trình thiết lập một principal. Principal có thể hiểu là một người, hoặc một thiết bị, hoặc một hệ thống nào đó *có thể* thực hiện một hành động trong ứng dụng của bạn.

**Authorization** (phân quyền) hay **Access-control**: là tiến trình quyết định xem một principal *có được phép*thực hiện một hành động trong ứng dụng của bạn hay không. Trước khi diễn tiến tới Authorization, principal cần phải được thiết lập bởi Authentication.

Ta có thể thấy đây là 2 cơ chế khá phổ biến trong các dịch vụ bảo mật, không chỉ riêng Spring Security.

Các thành phần cốt lõi

Security, SecurityContext và Authentication

**SecurityContext** là interface cốt lõi của Spring Security, lưu trữ tất cả các chi tiết liên quan đến bảo mật trong ứng dụng. Khi chúng ta kích hoạt Spring Security trong ứng dụng thì SecurityContext cũng sẽ được kích hoạt theo.

Chúng ta sẽ không truy cập trực tiếp vào SecurityContext, thay vào đó sẽ sử dụng lớp **SecurityContextHolder**. Lớp này lưu trữ security context hiện tại của ứng dụng, bao gồm chi tiết của principal đang tương tác với ứng dụng. Spring Security sẽ dùng một đối tượng **Authentication** để biểu diễn thông tin này. Đoạn code dưới đây sẽ giúp chúng ta lấy được username của principal đã được xác thực (username ở đây ta nên hiểu là username trong cặp username - password mà người dùng nhập vào khi đăng nhập):

Object principal = SecurityContextHolder.getContext().getAuthentication().getPrincipal();

if (principal instanceof UserDetails) {

String username = ((UserDetails) principal).getUsername();

} else {

String username = principal.toString();

}

## Đoạn code này có thể đặt ở bất kỳ đâu trong ứng dụng.

UserDetails và UserDetailsService

Trong đoạn code trên, chúng ta có được một principal từ đối tượng Authentication. Principal đơn giản chỉ là một đối tượng và sẽ được ép kiểu sang **UserDetails**.

UserDetails là một interface cốt lõi của Spring Security. Nó đại diện cho một principal nhưng theo một cách mở rộng và cụ thể hơn. Vậy UserDetails cung cấp cho ta những thông tin gì? UserDetails bao gồm các method sau:

getAuthorities(): trả về danh sách các quyền của người dùng getPassword(): trả về password đã dùng trong qúa trình xác thực getUsername(): trả về username đã dùng trong qúa trình xác thực

isAccountNonExpired(): trả về true nếu tài khoản của người dùng chưa hết hạn isAccountNonLocked(): trả về true nếu người dùng chưa bị khóa isCredentialsNonExpired(): trả về true nếu chứng thực (mật khẩu) của người

dùng chưa hết hạn

isEnabled(): trả về true nếu người dùng đã được kích hoạt

Chúng ta có thể thấy UserDetails mới chỉ cung cấp các phương thức để truy cập các thông tin cơ bản của người dùng. Để mở rộng thêm các thông tin, chúng ta sẽ tạo một lớp CustomUserDetails implements org.springframework.security.userdetails.UserDetails (tên lớp là tùy ý, bạn đặt tên thế nào cũng được).

Câu hỏi tiếp theo đặt ra là ta sẽ tạo implementation của UserDetails ở đâu trong ứng dụng? Câu trả lời là ta sẽ dùng **UserDetailsService**. UserDetailsService là một interface có duy nhất một phương thức:

UserDetails **loadUserByUsername**(String username) throws UsernameNotFoundException;

Tham số truyền vào chỉ gồm có username của người dùng. Ta sẽ tìm kiếm trong CSDL, record thỏa mãn username. Nếu không tìm thấy, ta sẽ ném ra ngoại

lệ UsernameNotFoundException.

Phương thức loadUserByUsername() sẽ trả về một implementation của UserDetails. Implementation ở đây có thể là:

org.springframework.security.core.userdetails.User CustomUserDetails implements

org.springframework.security.userdetails.UserDetailsmà mình đã nói ở trên

Nhiệm vụ của chúng ta là cần phải tạo một lớp UserDetailsServiceImpl implements UserDetailsService.

GrantedAuthority

Ở phần trên, em đã đề cập đến phương thức getAuthorities(). Phưong thức này sẽ trả về một tập hợp các đối tượng **GrantedAuthority**. Một GrantedAuthority là một quyền được ban cho principal. Các quyền đều có tiền tố là **ROLE\_**, ví dụ như ROLE\_ADMIN, ROLE\_MEMBER ...

## HTML, CSS, JAVASCRIPT.

Tổng quan về HTML **Giới thiệu về HTML** Căn bản về HTML:

HyperText Markup Language. Đây là một ngôn ngữ đơn giản được sử dụng trong các tài liệu siêu văn bản. Nó cũng chỉ là các tài liệu văn bản bình thường nhưng có chức các thành phần đặc biệt gọi là các thẻ (hoặc các cặp thẻ) đánh dấu. Dựa theo các cặp thẻ này mà trình duyệt có thể biết được nó phải thực hiện cái gì. Bản chất HTML không phải là một ngôn ngữ lập trình nên các cũng chẳng cần lo lắng đến những thuật toán lằng nhằng. Nó chỉ là một “ngôn ngữ” để đánh dấu văn bản thôi.

Cách sử dụng mẫu biểu trong HTML

Một biểu mẫu (form) trong HTML bao gồm nhiều thành phần khác nhau. Các thành phần có thể là ô văn bản, ô kéo thả, ô danh sách, nút, nút bấm, hay các ô check. Mẫu biểu được bắt đầu bằng thẻ <form> và kết thúc bởi thẻ </form>. Giữa 2 cặp thẻ này, các bạn có thể sử dụng các cặp thẻ HTML khác.

Thẻ form có một số thuộc tính sau: Thuộc tính này có 2 giá trị POST hoặc GET, để xác định dữ liệu gửi lên theo kiểu POST hay GET. Kiểu GET chính là kiểu mà khi nhập dữ liệu lên máy chủ, các dữ liệu này sẽ được hiển thị trên ô Address dưới dạng các cặp tên giá trị. Nhược điểm của kiểu này là toàn bộ URL và xâu tên giá trị kia sẽ bị giới hạn dưới 255 kí tự. Vì vậy để có thể gửi nhiều dữ liệu hơn, người ta sinh ra kiểu POST. Với kiểu này, dữ liệu sẽ không bị giới hạn chiều dài 255 ký tự của chuỗi địa chỉ do không bị gộp vào chuỗi địa chỉ. Kiểu POST cũng thường dùng để truyền các dữ liệu nhạy cảm mà người dùng không muốn hiển thị ở trên ô Address.

* + - 1. Tổng quan về JavaScript

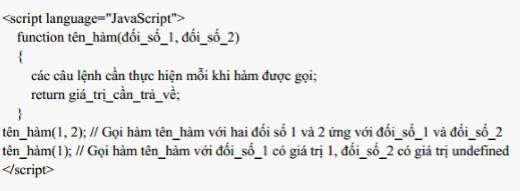
Cùng thời điểm Netscape bắt đầu sử dụng công nghệ Java trên trình duyệt Netscape. LiveScript đã được đổi tên thành JavaScipt để được chú ý hơn bởi ngôn ngữ lập trình Java lúc đó đang được coi là một hiện tượng. JavaScript được bổ sung vào trình duyệt Netscape bắt đầu vào tháng 10 năm 1995. Trên thực tế, hai ngôn ngữ lập trình Java và JavaScript không có

liên quan đến nhau, ngoại trừ cú pháp của cả hai ngôn ngữ cùng được phát triển dựa trên cú phấp của C. JavaScript gồm 2 mảng là client-server thực hiện lệnh trên máy của end-user và web-server.

Sau thành công của JavaScript, Microsoft bắt đầu phát triển Jscript, một ngôn ngữ có cùng ứng dụng và tương thích với JavaScript. Jscript được bổ sung vào trình duyệt Internet Explorer bắt đầu từ Internet Explorer phiên bản 3.0 được phát hành tháng 8 năm 1996.

DOM (Document Object Model), một khái niệm thường được nhắc đến với JavaScript trên thực tế không phải một phần của chuẩn ECMAScript, DOM là một chuẩn riêng biệt có liên quan chặt chẽ với XML.

Hàm là một khối các lệnh với một danh sách một hoặc nhiều đối số (có thể không có đối số) và thường có tên (mặc dù trong JavaScript hàm không nhất thiết phải có tên). Hàm có thể trả lại một giá trị. Cú pháp của hàm như sau:



Trong JavaScript, khi gọi hàm không nhất thiết phải gọi hàm với cùng đối số như khi định nghĩa hàm, nếu đối số ít hơn khi định nghĩa hàm, nhưng đối số không được chuyển cho hàm sẽ mang giá trị underfined. Các kiểu cơ bản được chuyển vào hàm theo giá trị, đối tượng sẽ được chuyển vào hàm theo tham chiếu.

Hàm là đối tượng hạng nhất trong JavaScript. Tất cả các hàm là đối tượng của nguyên mẫu Function. Hàm có thể được tạo ra và dùng trong phép toán gán như bất kỳ một đối tượng nào khác, và cũng có thể được dùng làm đối số cho các hàm khác. Do đó, JavaScript hỗ trợ hàm cấp độ cao.

## Tổng quan về CSS.

1. Khái niệm về CSS

* *CSS là gì?:* CSS: Cascading Style Sheet: Đây là những mẫu để quy định cách thức thể hiện các thẻ HTML.
* *CSS có 3 cách sử dụng:*
  + Sử dụng trực tiếp kèm với các thẻ HTML (Inline Style Sheet)
  + Định nghĩa trong 1 trang web (Internal Style Sheet)
  + Định nghĩa thành 1 file CSS riêng (External Style Sheet)

Style được đưa vào HTML 4.0 để giải quyết một số vấn đề. CSS giúp bạn tiết kiệm được rất nhiều thời gian và công sức cho việc thiết kế web. Bạn có thể định nghĩa nhiều style vào một thẻ HTML Style (mẫu định dạng) giải quyết một số vấn đề chung: Ta biết rằng các thẻ HTML chuẩn được thiết kế để định nghĩa nội dung của một văn bản. Đầu tiên các thẻ HTML hỗ trợ cách viết "Đây là tiêu đề", "Đây là đoạn", "Đây là bảng".... bằng cách sử dụng các thẻ <H1>, <P>, <TABLE>... Cách bố trí văn bản này được qui định bởi trình duyệt web và không có bất cứ một thẻ nào để định dạng văn bản.

Đến các trình duyệt thế hệ sau đặc biệt là Nescape và Internet Explorer tiếp tục đưa thêm vào các thẻ HTML mới cùng các thuộc tính định dạng riêng của mình (như các thẻ <FONT> và thuộc tính Color...). Do đó ngày càng khó để tạo ra được một web site khi mà nội dung của nó bị tách rời khỏi cách bố trí.

Để giải quyết vấn đề này World Wide Web Consortium (W3C) đã tạo ra STYLE để đưa thêm vào trong HTML 4.0. Cả hai trình duyệt lớn là Nescape và Internet Explorer đều hỗ trợ CSS. CSS giúp bạn tiết kiệm được rất nhiều thời gian và công sức cho việc thiết kế web.

Style trong phiên bản HTML 4.0 trở lên qui định cách thức thể hiện các thẻ HTML tương tự như thẻ <FONT> hay thuộc tính COLOR trong HTML 3.2. Style thường được lưu trong các file nằm ngoài trang web. Chúng giúp bạn có thể thay đổi cách thức định dạng và cách bố trí các trang web chỉ bằng cách thay đổi riêng file CSS. Chỉ khi bạn muốn thay đổi lại toàn bộ màu sắc, cách định dạng của các tiêu đề, nội dung bạn mới hiểu rõ tác dụng thực sự của CSS.

CSS cho phép chúng ta điều khiển cách định dạng và cách bố trí của cùng lúc nhiều trang web với chỉ duy nhất 1 lần thay đổi tại một vị trí. Là một người thiết kế web, bạn có thể định nghĩa 1 file CSS cho các thẻ HTML và áp dụng nó vào nhiều trang web mà bạn muốn. Để thay đổi tổng thể các trang web này bạn chỉ đơn giản là thay đổi file CSS và tất cả các trang đã áp dụng sẽ được thay đổi một cách tự động. Bạn có thể định nghĩa nhiều style vào một thẻ HTML.

CSS cho phép bạn đưa các thông tin định nghĩa thẻ thông qua nhiều con đường khác nhau. Style có thể được qui định ở trong chỉ một thẻ HTML, được qui định trong một trang web hoặc ở trong một file CSS bên ngoài. Thứ tự áp dụng các định dạng. Như trên đã nói, ta có thể sử dụng nhiều cách khác nhau để làm CSS. Điều gì sẽ xảy ra nếu bạn áp dụng nhiều cách định dạng cho 1 thẻ HTML? Theo một cách chung nhất ra có thể nói các style của bạn sẽ được "xếp tầng" (cascade). Việc xếp tầng này tuân theo thứ tự sau: (Số 1 là ưu tiên nhất, số 4 là kém ưu tiên nhất).

* 1. Inline Style (Style được qui định trong 1 thẻ HTML cụ thể).
  2. Internal Style (Style được qui định trong phần <HEAD> của 1 trang HTML ).
  3. External Style (style được qui định trong file CSS ngoài).
  4. Browser Default (thiết lập mặc định của trình duyệt). Như vậy ta thấy các thiết lập trong 1 thẻ HTML có mức ưu tiên cao nhất, Những gì được định nghĩa ở đây sẽ bị bỏ qua tất cả các định nghĩa khác (như trong thẻ <HEAD>, File CSS ngoài,...).

1. Cách sử dụng CSS

Khi trình duyệt web đọc trang web của bạn nó sẽ định dạng trang web theo cách CSS đã qui định cho nó.

Chúng ta có tất cả 3 loại CSS dó đó chúng ta cũng sẽ có 3 cách để chèn CSS vào trang web của mình.

* + Với External Style ( Sử dụng file CSS được định nghĩa thành 1 file riêng). Sử dụng External Style là một cách lý tưởng khi ta cần phải định dạng nhiều trang web theo một mẫu thống nhất.
  + Với External Style chúng ta có thể thay đổi dáng vẻ của một trang web chỉ với việc thay đổi duy nhất 1 file. Mỗi trang web sử dụng file CSS ngoài này đều phải sử dụng thẻ <LINK>. Thẻ <LINK> được đặt bên trong thẻ

<HEAD>, với thuộc tính href sẽ trỏ đến file .CSS bên ngoài.

## Mô hình MVC:

* + Mô hình MVC (Model- View- Controller) là một kiến trúc phần mềm hay mô hình thiết kế được sử dụng trong kỹ thuật phần mềm. Nó giúp cho các developer tách ứng dụng của họ ra thành 3 phần khác nhau Model, View và Controller. Mỗi thành phần có một nhiệm vụ riêng biệt và độc lập với các thành phần khác.
  + Mô hình MVC được giới thiệu từ những năm 70 như một phần của Smalltalk, nhưng đối với nền tảng web, thì nó mới được thịnh hành gần đây.
  + Ý tưởng đằng sau MVC là để chia rõ 3 thành phần chính là model (xử lý, truy xuất database), view (giao diện) và controller (điều hướng yêu cầu người dùng).
  + MVC thể hiện tính chuyên nghiệp trong lập trình, phân tích thiết kế. Do được chia thành các thành phần độc lập nên giúp phát triển ứng dụng nhanh, dơn giản, dễ nâng cấp, bảo trì.

1. Google Map API:

**5.1) Google Map API là gì?**

* Google Map là một dịch vụ ứng dụng vào công nghệ bản đồ trực tuyến trên web miễn phí được cung cấp bởi Google, hỗ trợ nhiều dịch vụ khác của Google đặc biệt là dò đường và chỉ đường; hiển thị bản đồ đường sá, các tuyến đường tối ưu cho từng loại phương tiện, cách bắt xe và chuyển tuyến cho các loại phương tiện công cộng (xe bus, xe khách ...), và những địa điểm (kinh doanh, trường học, bệnh viện, cây ATM...) trong khu vực cũng như khắp nơi trên thế giới.
* Map API là gì?

Là một phương thức cho phép 1 website B sử dụng dịch vụ bản đồ của website A (gọi là Map API) và nhúng vào website của mình (site B). Site A ở đây là google map, site B là các website cá nhân hoặc tổ chức muốn sử dụng dịch vụ của google (di chuột, room, đánh dấu trên bản đồ…)

Các ứng dụng xây dựng trên maps được nhúng vào trang web cá nhân thông qua các thẻ javascripts do vậy việc sử dụng API google rất dễ dàng.

Google Map API đã được nâng cấp lên phiên bản v3 không chỉ hỗ trợ cho các máy để bàn truyền thống mà cho cả các thiết bị di động; các ứng dụng nhanh hơn và nhiều hơn .

Các dịch vụ hoàn toàn miễn phí với việc xây dựng một ứng dụng nhỏ. Trả phí nếu đó là việc sử dụng cho mục đích kinh doanh, doanh nghiệp.

**5.2) Một số ứng dụng của Google Map API**

* Đánh dấu các địa điểm trên bản đồ kèm theo thông tin cho địa điểm đó : khu vui chơi giải trí, nhà hàng khách sạn, cây ATM, bệnh viện, trường học,… bất cứ địa điểm nào bạn muốn
* Chỉ dẫn đường đến các địa điểm cần tìm(đường tối ưu và nhiều option khác),chỉ dẫn đường giao thông công cộng, có thể là các địa điểm cung cấp như trên. Ở đây sử dụng các service google cung cấp.
* Khoanh vùng khu vực: các trung tâm kinh tế, khu đô thị, khu ô nhiễm…
* Tình trạng giao thông các khu vực… từ đó đưa ra các giải pháp…