MỤC LỤC

MỤC LỤC **3**
BÀI 1. TỔNG QUAN VỀ WEB, INTERNET VÀ HTML CƠ BẢN **3**
1.1. Khái niệm về Internet, Web và HTML 3
1.1.1. Internet 3
1.1.2. Website 3
1.1.3. HTML 3
1.2. Công cụ hỗ trợ Dreamweaver 4
1.2.1. Giới thiệu về phần mềm Adobe Dreamweaver 4
1.2.2. Hướng dẫn cài đặt 4
1.3. Cấu trúc trang HTML 8
1.3.1. Cấu trúc thẻ đánh dấu trong HTML 8
1.3.2. Cấu trúc file HTML 9
1.4. Các thẻ HTML cơ bản 10
1.4.1. Các thẻ tiêu đề (Headings) 10
1.4.2. Thẻ trung hòa 11
1.4.3. Thẻ định dạng văn bản, kiểu chữ 12
1.4.4. Thẻ đánh dấu siêu liên kết 14
2.1. Câu hỏi ôn tập 16
2.2. Bài tập 17
BÀI 2. CHÈN DỮ LIỆU ĐA PHƯƠNG TIỆN 19
1.1. Chèn dữ liệu hình ảnh 19
1.1.1. Cú pháp và các thuộc tính 19
1.1.2. Đường dẫn của ảnh 19
1.1.3. Căn chỉnh vị trí của ảnh 21
1.1.4. Thuộc tính Alt 22
1.1.5. Độ rộng và chiều cao của ảnh 22
1.1.6. Khoảng cách xung quanh ảnh 23

1.2. Chèn dữ liệu âm thanh 24
1.3. Chèn dữ liệu video 25
1.4. Chèn Flash và các thành phần Multimedia khác 26
2.1. Câu hỏi ôn tập 27
2.2. Bài tập 27
BÀI 3. LÀM VIỆC VỚI TABLE VÀ LIST 28
1.1. Làm việc với table 28
1.1.1. Những thẻ cơ bản của table 28
1.1.2. Thẻ được sử dụng để đánh dấu bảng 28
1.1.3. Các thuộc tính của thẻ tr và td 31
1.2. Làm việc với list 34
1.2.1. Thẻ UL 34
1.2.2. Thẻ OL 35
1.2.3. Thẻ DL 36
2.1. Câu hỏi ôn tập 37
2.2. Bài tập 37
BÀI 4. LÀM VIỆC VỚI FORM 39
1.1. Thẻ form 39
1.1.1. Cấu trúc Form 39
1.1.2. Sự khác nhau giữa method post và get 40
1.2. Các thẻ input 40
1.2.1. Thẻ input text và password 41
1.2.2. Thẻ input checkbox 41
1.2.3. Thẻ input radio 43
1.2.4. Thẻ input file 44
1.2.5. Input button, submit và reset 44
1.3. Danh sách với thẻ select 45
1.4. Vùng nhập văn bản với textarea 46
1.5. Các thuộc tính đặc biệt trong form 47

2.1. Câu hỏi ôn tập 48
2.2. Bài tập 48
BÀI 5. HTML5 50
1.1. Tổng quan về HTML5 50
1.1.1. Lịch sử phát triển 50
1.1.2. Ưu điểm của HTML5 50
1.1.3. HTML5 DOCTYPE 50
1.1.4. Những tính năng mới của HTML5 51
1.1.5. Các trình duyệt hỗ trợ HTML5 51
1.2. Các thẻ mới trong HTML5 51
1.2.1. Những cấu trúc thẻ mới 52
1.2.2. Các thẻ media mới 52
1.2.3. Thẻ <canvas> 53
1.2.4. Các phần tử form mới 53
1.2.5. Các thẻ được gỡ bỏ 53
1.3. Thẻ Canvas và hiển thị các đối tượng đồ họa 53
1.3.1. Khái niệm 53
1.3.2. Cách tạo canvas 53
1.3.3. Sử dụng javascript vẽ nội dung canvas 54
1.3.4. Canvas paths 55
1.3.5. Canvas Text 56
1.3.6. Canvas Gradient 57
1.3.7. Canvas Images 58
1.4. Form và các thành phần của form trong HTML5 59
1.4.1. Các thẻ mới của form 60
1.4.2. Các thuộc tính mới của form 62
1.4.3. HTML5 input type 72
HTML5 input type 72
1.5. Trình diễn đa phương tiện với HTML5 73

1.5.1. HTML5 Video 73
1.5.2. HTML5 Audio 75
1.6. Lưu trữ dữ liệu tại trình duyệt 76
1.6.1. Các trình duyệt hỗ trợ 76
1.6.2. Các phương thức web storage của HTML5 77
2.1. Câu hỏi ôn tập 78
2.2. Bài tập 78
BÀI 6. NGÔN NGỮ CSS 80
1.1. Tổng quan về CSS 80
1.1.1. CSS là gì? 80
1.1.2. Cú pháp và quy ước viết CSS 80
1.1.3. Cách sử dụng CSS 81
1.1.4. Đơn vị tính 87
1.2. Selector trong CSS 88
1.2.1. Bộ chọn toàn cục (Universal Selector) 88
1.2.2. Bộ chọn phân cấp (Descendant Selector) 90
1.2.3. Bộ chọn ID 91
1.2.4. Bộ chọn Class 91
1.2.5. Bộ chọn Type Selector 92
1.2.6. Bộ chọn Pseudo (Classes Selector) 93
1.2.7. Bộ chọn liền kề (Adjacent Sibling Selector) 94
1.2.8. Bộ chọn Child Selector 95
1.2.9. Bộ chọn theo thuộc tính (Attribute Selector) 96
1.3. Các thuộc tính cơ bản của CSS 97
1.3.1. Nhóm các thuộc tính background 97
1.3.2. Nhóm các thuộc tính border 99
1.3.3. Box Model, Margin và Padding 99
1.3.4. Nhóm các thuộc tình font 103
1.3.5. Width và Heigh 105

1.3.6. Nhóm các thuộc tính text 106
1.3.7. Float và Clear 108
1.3.8. Position 109
2.1. Câu hỏi ôn tập 111
2.2. Bài tập 111
BÀI 7. NGÔN NGỮ JAVASCRIPT 113
1.1. Tổng quan về Javascript 113
1.1.1. Javascript là gì? 113
1.1.2. Cách sử dụng javascript 113
1.1.3. Biến và cách khai báo javascript 114
1.1.4. Function và các khai báo 115
1.1.5. Toán tử trong javascript 115
1.2. Phát biểu có điều kiện 117
1.2.1. Câu lệnh if - else 117
1.2.2. Câu lệnh while 118
1.2.3. Câu lệnh for 118
1.2.4. Câu lệnh switch – case 119
1.2.5. Phát biểu break và continue 120
1.3. Xử lý form với javascript 121
1.3.1. Javascript và thẻ Text, Password 121
1.3.2. Javascript và thẻ Radio, Checkbox 122
1.3.3. Javascript với thẻ <textarea> 123
1.3.4. Javascript với thẻ Checkbox cùng tên 126
1.4. Các đối tượng trong javascript 129
1.4.1. Giới thiệu về lập trình hướng đối tượng 129
1.4.2. Đối tượng String 130
1.4.3. Đối tượng Array 134
1.4.4. Đối tượng Date 136
1.4.5. Đối tượng Math 138

1.4.6. Đối tượng Window 138
1.4.7. Đối tượng Navigator 139
1.4.8. Đối tượng Location 140
1.4.9. Đối tượng Document 142
1.4.10. Đối tượng Cookie 145
2.1. Câu hỏi ôn tập 147
2.2. Bài tập 148
BÀI 8. JQUERY VÀ CÁC HIỆU ỨNG ĐỘNG 149
1.1. Tổng quan về jQuery 149
1.1.1. jQuery là gì? 149
1.1.2. Cách sử dụng jQuery 151
1.2. Cú pháp cơ bản của jQuery 152
1.3. Truy xuất các phần tử và CSS qua jQuery Selector 152
1.3.1. Cú pháp jQuery selector 153
1.3.2. jQuery CSS selector 153
1.3.3. jQuery Attribute selector 154
1.3.4. Các selector riêng của jQuery 154
1.4. Các sự kiện với jQuery 163
1.4.1. Bộ quản lý sự kiện 164
1.4.2. Một số sự kiện thường sử dụng trong jQuery 164
1.4.3. Gắn kết Event Handler sử dụng phương thức .bind() 166
1.5. Các hiệu ứng với jQuery 174
1.5.1. Thay đổi Inline CSS 174
1.5.2. Một số hiệu ứng sử dụng jQuery thường gặp 175
2.1. Câu hỏi ôn tập 177
2.2. Bài tập 178
BÀI 9. PHOTOSHOP TRONG THIẾT KẾ WEB 180
1.1. Làm quen với photoshop 180
1.1.1. Giới thiệu chung photoshop 180

1.1.2. Một số thao tác cơ bản 181
1.1.3. Toolbox & Panel 182
1.1.4. Thao tác với vùng chọn 187
1.1.5. Thao tác với layer 196
1.1.6. Công cụ cắt xén ảnh Crop 198
1.2. Thiết kế giao diện website 199
1.2.1. Phác thảo ý tưởng thiết kế website 199
1.2.2. Thiết kế giao diện website 203
1.3. Chuyển đổi từ PSD sang HTML 203
2.1. Câu hỏi ôn tập 204
2.2. Bài tập 205

BÀI 1. TỔNG QUAN VỀ WEB, INTERNET VÀ HTML CƠ BẢN

A. NỘI DUNG

1.1. Khái niệm về Internet, Web và HTML

1.1.1. Internet

Internet là một tập hợp của các máy tính được liên kết nối lại với nhau thông qua hệ thống dây cáp mạng và đường điện thoại trên toàn thế giới với mục đích trao đổi,chia sẻ dữ liệu và thông tin. Bất cứ người nào trên hệ thống cũng có thể tiếp cận và đi vào xem thông tin từ bất cứ một máy tính nào trên hệ thống này hay hệ thống khác.

Trước đây mạng Internet được sử dụng chủ yếu ở các tổ chức chính phủ và trong các trường học. Ngày nay mạng Internet đã được sử dụng bởi hàng tỷ người bao gồm cả cá nhân các doanh nghiệp lớn, nhỏ, các trường học và tất nhiên là nhà nước và các tổ chức chính phủ. Phần chủ yếu nhất của mạng Internet là World Wide Web.

Mạng Internet là của chung điều đó có nghĩa là không ai thực sự sở hữu nó với tư cách cá nhân. Mỗi phần nhỏ của mạng được quản lý bởi các tổ chức khác nhau nhưng không ai không một thực thể nào cũng như không một trung tâm máy tính nào nắm quyền điều khiển mạng. Mỗi phần của mạng được liên kết với nhau theo một cách thức nhằm tạo nên một mạng toàn cầu.

1.1.2. Website

Website còn gọi là trang web, trang mạng, là một tập hợp các trang web bao gồm văn bản, hình ảnh, video, flash vv, thường chỉ nằm trong một tên miền (domain name) hoặc tên miền phụ (subdomain). Trang web được lưu trữ (web hosting) trên máy chủ web (server web) có thể truy cập thông qua Internet.

1.1.3. HTML

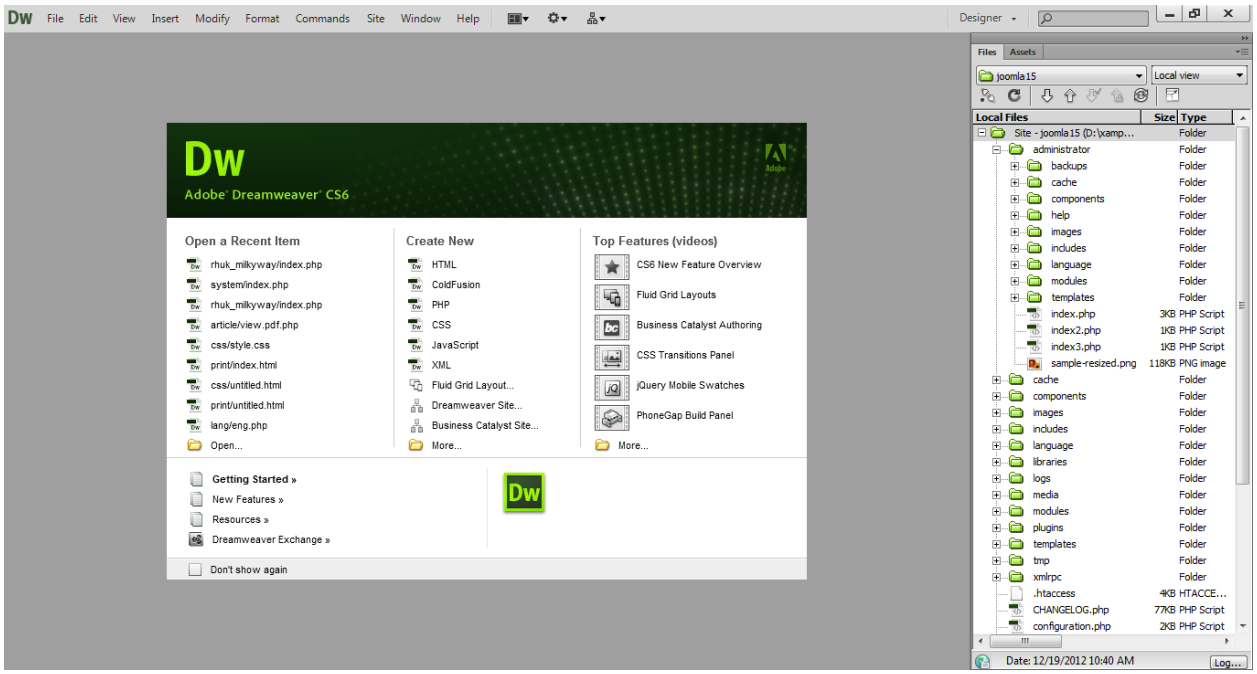
HTML viết tắt cho *Hyper Text Markup Language* (*Ngôn ngữ đánh dấu siêu văn bản)* là một ngôn ngữ đánh dấu được thiết kế ra để tạo nên các trang web, nghĩa là các mẫu thông tin được trình bày trên World Wide Web. HTML giờ đây đã trở thành một chuẩn Internet do tổ chức World Wide Web Consortium *(W3C)* sáng lập và duy trì. Phiên bản hiện này của nó là HTML5.

HTML có tác dụng giúp cho trình duyệt web có thể biết cách thể hiện một trang web như thế nào từ văn bản, hình ảnh cho đến các siêu liêu kết. HTML không có gì khác ngoài văn bản được đánh dấu theo các thẻ *(tag)* và thuộc tính của chúng *(attribute)*.

1.2. Công cụ hỗ trợ Dreamweaver

1.2.1. Giới thiệu về phần mềm Adobe Dreamweaver

Adobe Dreamweaver là một trình soạn thảo trực quan chuyên nghiệp được dùng để xây dựng và quản lý các website. Nó cho phép người dùng tạo nên những trang web với nhiều tính năng mà dường như *"không cần"* viết mã HTML. Dreamweaver được thiết kế để trở thành một chương trình thiết kế dạng "WYSIWYG" *(What You See Is What You Get - Những gì bạn thấy chính là những gì bạn có được).*



Hình 1.1. Adobe Dreamweaver CS6

Phiên bản mới nhất của Adobe Dreamweaver là phiên bản CS6. Nó là một công cụ xử lý dành cho những người thiết kế web, viết code và những nhà phát triển ứng dụng ở mọi cấp độ. Chức năng code được nâng cao tạo cho nó một sự mạnh mẽ khi điều hướng những trang web phức tạp ở thời điểm thiết kế. Những công cụ bố trí được cải thiện làm cho công việc dàn trang được tiến hành từ ý tưởng tổng hợp đến sự đồng ý của khách hàng. Những sáng kiến thông qua bản thử Dreamweaver có thể giúp những đội hoặc những người phát triển web cá nhân tiến đến mức độ tiếp theo như nhau khi thực hiện cũng như về mặc chức năng.

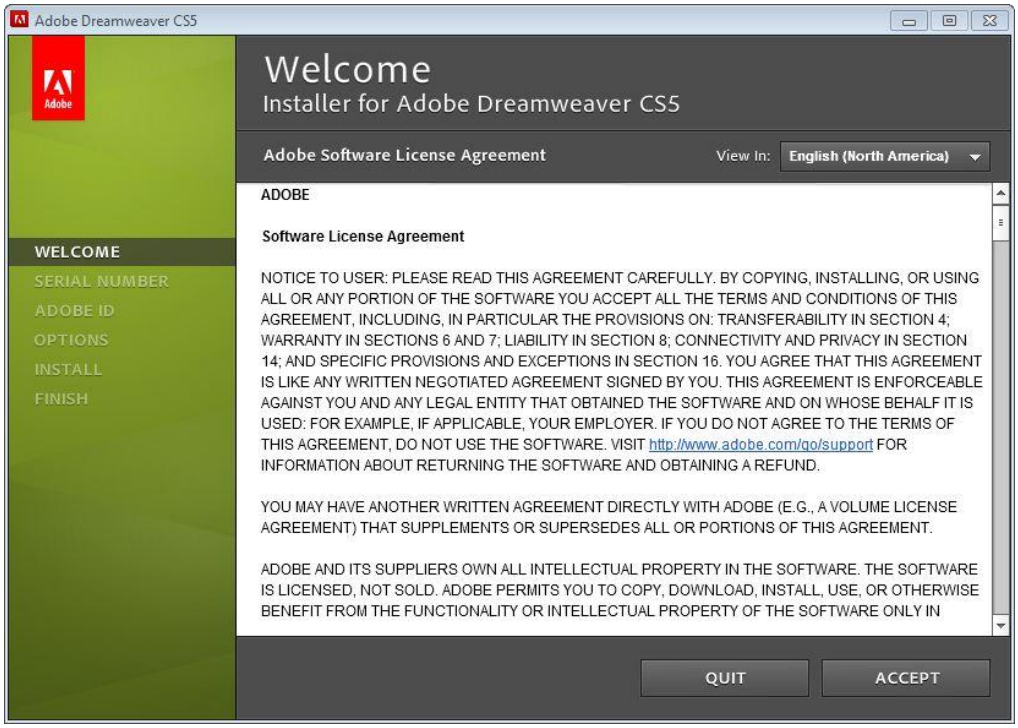
1.2.2. Hướng dẫn cài đặt

- Download phần mềm Dreamweaver tại website của Adobe.

Các bước cài đặt và chạy phần mềm Dreamweaver, thực hiện các bước sau:

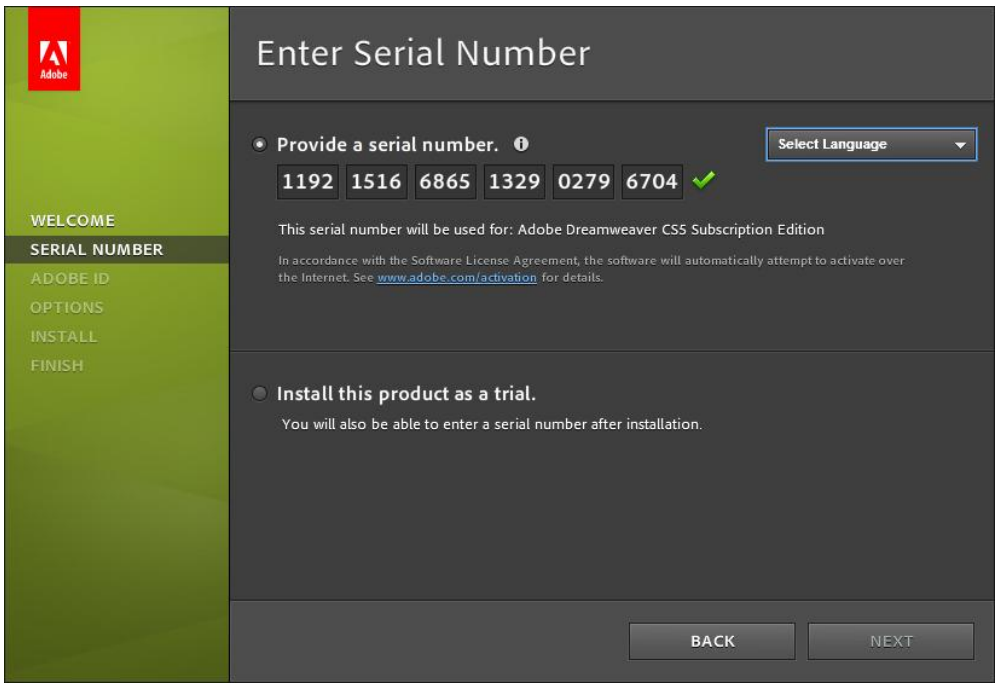
- Bước 1: Mở Window Explorer tìm và chạy tệp tin cài đặt Set-up.exe.

- Bước 2: Sau khi chạy tệp tin cài đặt, chương trình sẽ hiển thị như hình trên. Bạn chọn **Accept** để chương trình tiếp tục cài đặt.



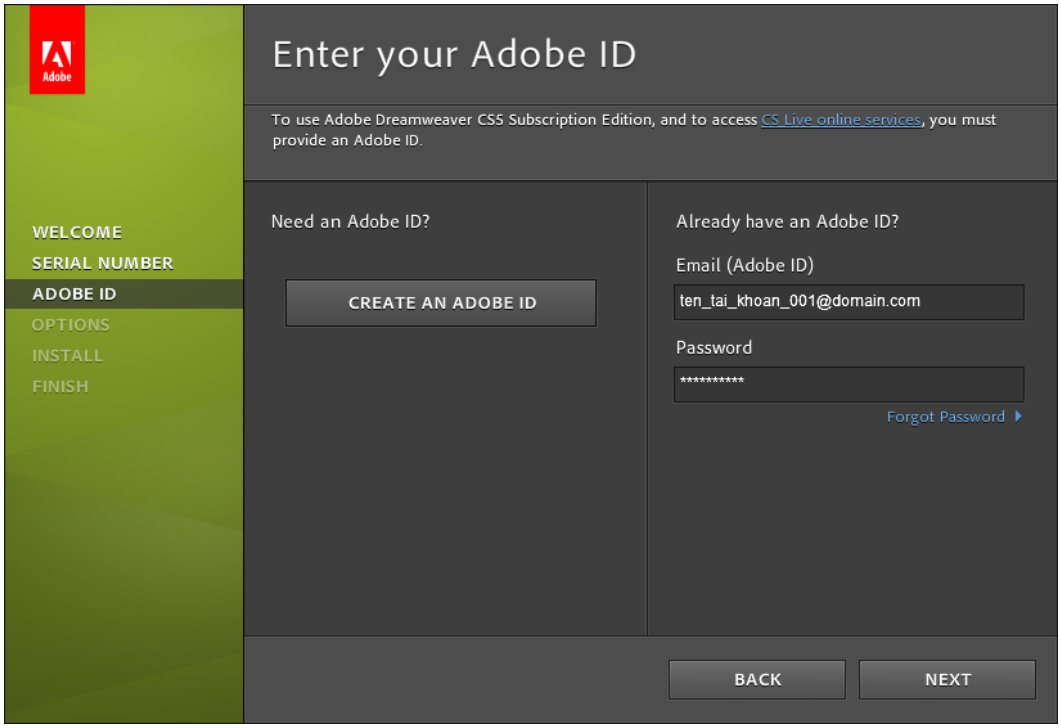
Hình 1.2. Giao diện cài đặt chương trình Adobe Dreamweaver

- Bước 3: Nhập **serial number** và lựa chọn ngôn ngữ trong ô **Select Language.** Sau khi nhập xong, nút **Next** sẽ bật cho phép bạn click để tiếp tục quá trình cài đặt.



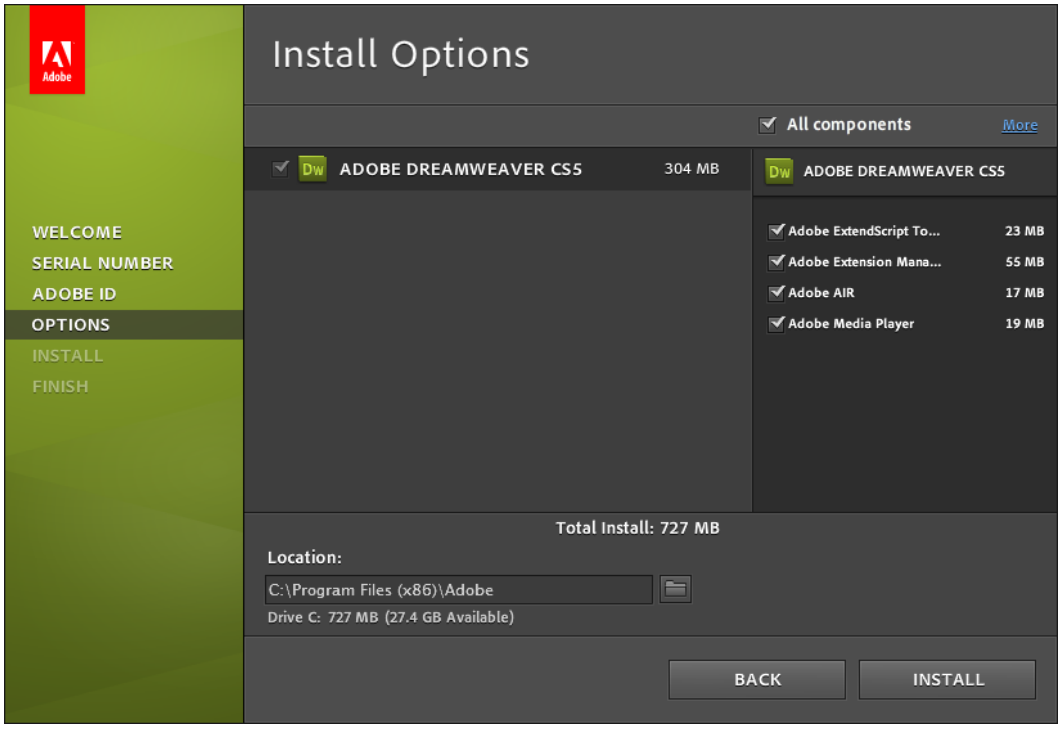
Hình 1. 3. Nhập serial number và chọn ngôn ngữ

- Bước 4: Bạn nhập thông tin tài khoản Adobe bao gồm **email** và **password**, sau đó lựa chọn **next** để tiếp tục quá trình cài đặt.



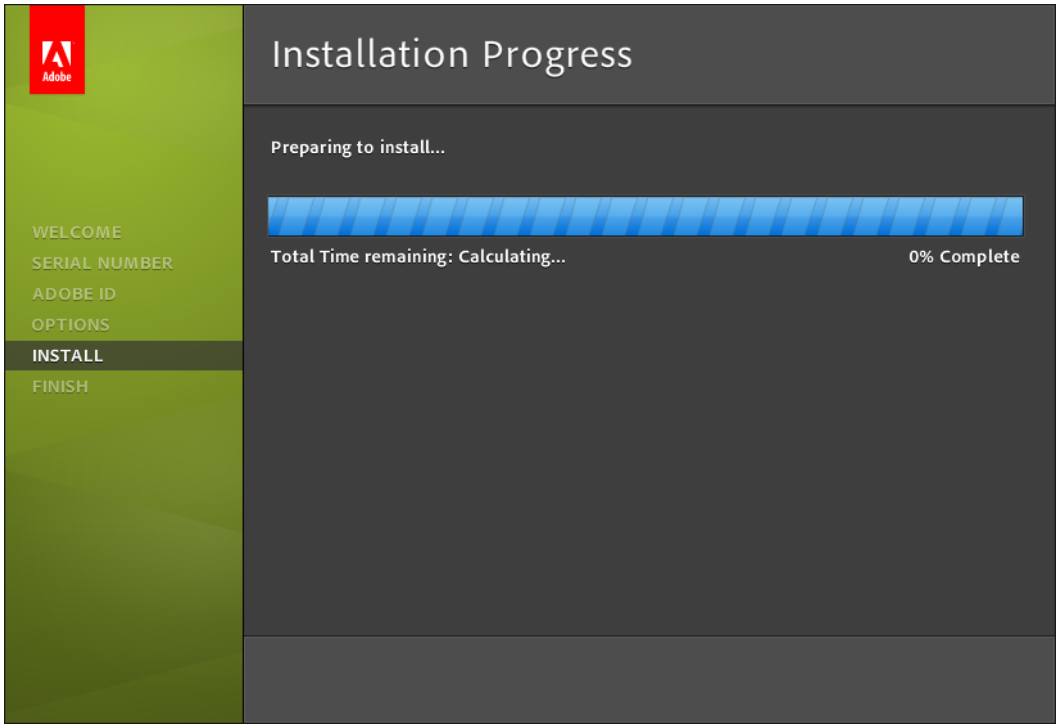
Hình 1.4. Nhập thông tin tài khoản Adobe

- Bước 5: Bạn thực hiện lựa chọn các thành phần cài đặt. Bước này không bắt buộc bạn phải check hết tất cả các thành phần. Tuy nhiên, nếu không hiểu rõ tác dụng của cac thành phần, bạn nên để mặc định rồi click chọn **Next**.



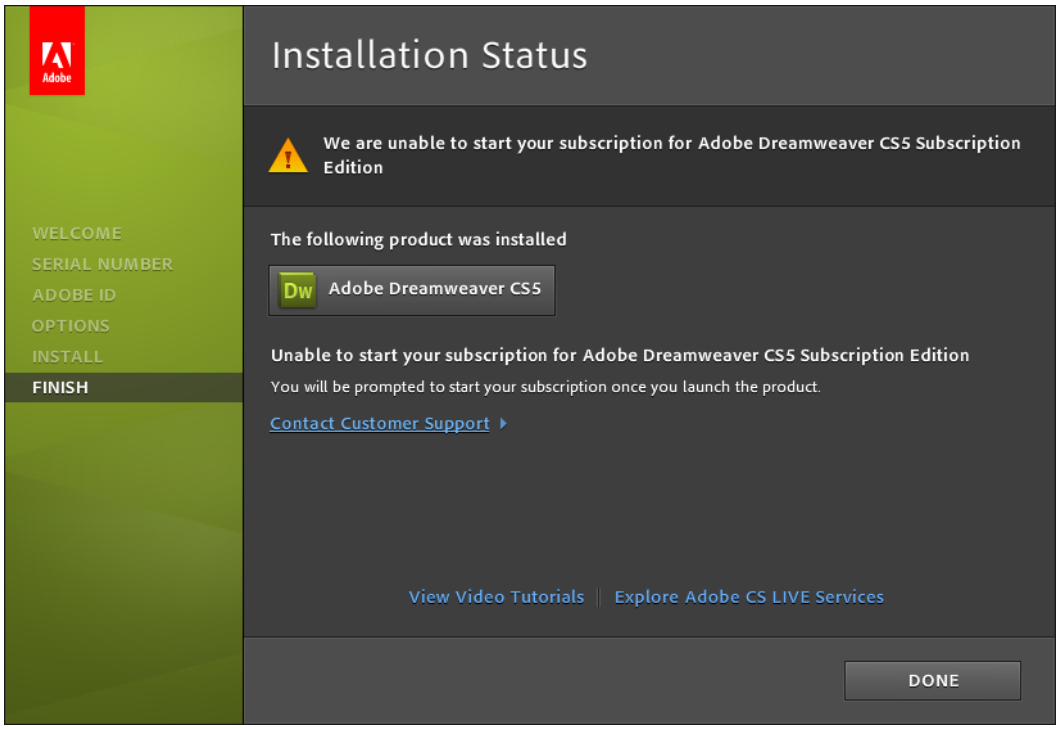
Hình 1.5. Lựa chọn các thành phần cài đặt

- Bước 6: Quá trình cài đặt diễn ra tự động. Công việc của bạn đơn giản là chờ đợi cho đến khi quá trình cài đặt diễn ra thành công.



Hình 1.6. Quá trình cài đặt diễn ra

- Bước 7: Bạn click chọn DONE để kết thúc quá trình cài đặt.



Hình 1.7. Màn hình kết thúc quá trình cài đặt

1.3. Cấu trúc trang HTML

1.3.1. Cấu trúc thẻ đánh dấu trong HTML

Một tài liệu HTML bao gồm các thành phần được đánh dấu bởi các thẻ. Dựa vào các thẻ đó trình duyệt web sẽ biết cách hiển thị nội dung của tài liệu HMTL như thế nào.

Một tag có 3 thành phần chính: start tag *(thẻ mở)*, end tag *(thẻ đóng)* và attribute *(thuộc tính)*. Thuộc tính của thẻ được đặt trong thẻ mở, giữa thẻ mở và thẻ đóng là nội dung tương ứng với thẻ đó.

Cấu trúc thông thường của một thẻ là:

```
<tag attribute1 = "value" attribute2 ="value">Content</tag>
```

Nếu một thẻ không có nội dung sẽ được gọi là thẻ rỗng và khi đó chúng ta có thể không cần dùng đến thẻ đóng mà kết thúc thẻ mở với dấu "/":

```
<tag attribute1 = "value" attribute2 = "value />
```

Ví dụ với đoạn mã HTML sau:

```
<h3>Tieu de - Heading 3 - Bao gom the dong, the mo va noi dung</h3>
<p textalign = "right">The p voi thuoc tinh text align va noi dung<br/>
Cung co the chua nhung the khac ben trong noi dung.</p>
```

Kết quả sẽ cho như hình 1.1:

Hình 1.8. Cấu trúc của file HTML

1.3.2. Cấu trúc file HTML

Cấu trúc một file HTML như sau:

```
<html>
        <head>
                <title>Tên trang</title>
        </head>
        <body>
        <!-- Văn bản và các thẻ HTML -->
        </body>
</html>
```

File HTML bao giờ cũng bắt đầu bằng cặp thẻ `<html></html>`. Cặp thẻ này báo cho trình duyệt Web biết rằng nó đang đọc một tập tin có chứa các mã HTML. Bên trong cặp thẻ `<html></html>` là các cặp thẻ `<head></head>` và `<body></body>` Trong đó:

Bên trong cặp thẻ `<head><head>` có thể chứa:

- Cặp thẻ `<title> văn bản </title>` đánh dấu tiêu đề cho trang web.

- Các thẻ `<meta/>` quy định một số thông tin cho trang web ví dụ quy định trang web sử dụng tiếng việt (Unicode).

- Các thẻ `<link/>` để liên kết với các file khác. Phổ biến nhất là file CSS

- Các cặp thể `<script>Mã lập trình</script>` để chứ các đoạn mã lập trình như Vbscript, Javascript sẽ được đề cập tới trong những bài sau.

Bên trong thẻ `<body></body>` Sẽ chứa các đoạn văn bản và các thẻ đánh dấu toàn bộ nội dung của một trang web và cũng có thể chứa những đoạn mã lập trình với cặp thẻ `<script></script>`

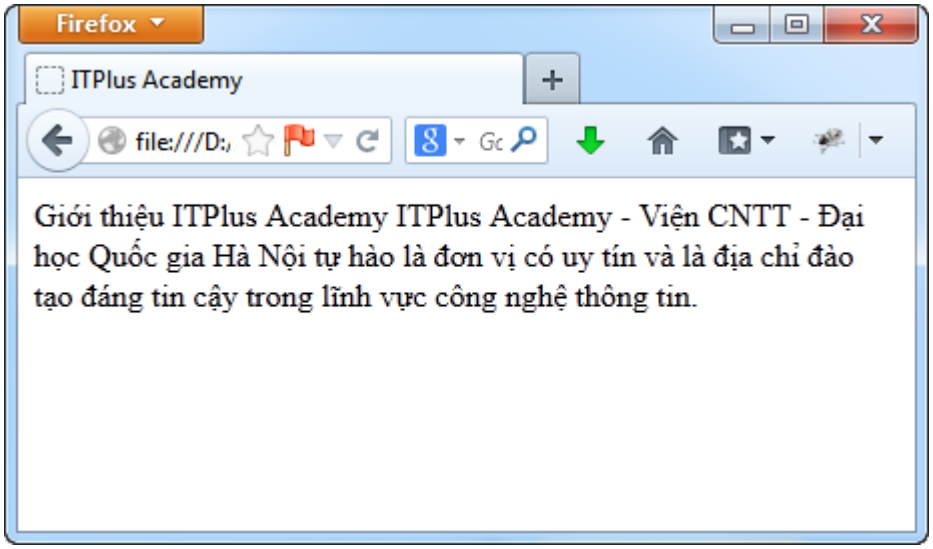
Ngoài ra để ghi chú thích, tiện cho việc xem tag HTML hoặc cập nhật một trang Web, ta có thể sử dụng thẻ ghi chú trong HTML `<!-- Nội dung ghi chú -->`. Cặp thẻ này chỉ có tác dụng cho người lập trình đánh dấu các đoạn mã trong file HTML, trình duyệt sẽ không hiện thị nội dung có trong thẻ này.

Chú ý: Một tài liệu HTML sẽ được bỏ qua các dấu xuống dòng.

Ví dụ:

```
<html>
    <head>
        <meta charset="utf-8">
        <title>ITPlus Academy</title>
    </head>
    <body>
        Giới thiệu ITPlus Academy
        ITPlus Academy - Viện CNTT -
        Đại học Quốc gia Hà Nội tự hào là đơn vị có uy tín
        và là địa chỉ đào tạo đáng tin cậy
        trong lĩnh vực công nghệ thông tin.
    </body>
</html>
```

Thì trên thực tế ta sẽ nhận được là:



Hình 1.9. Kết quả mô tả thẻ body

Một file HTML có thể được lưu với phần mở rộng là *.html* hoặc *.htm* đều được.

1.4. Các thẻ HTML cơ bản

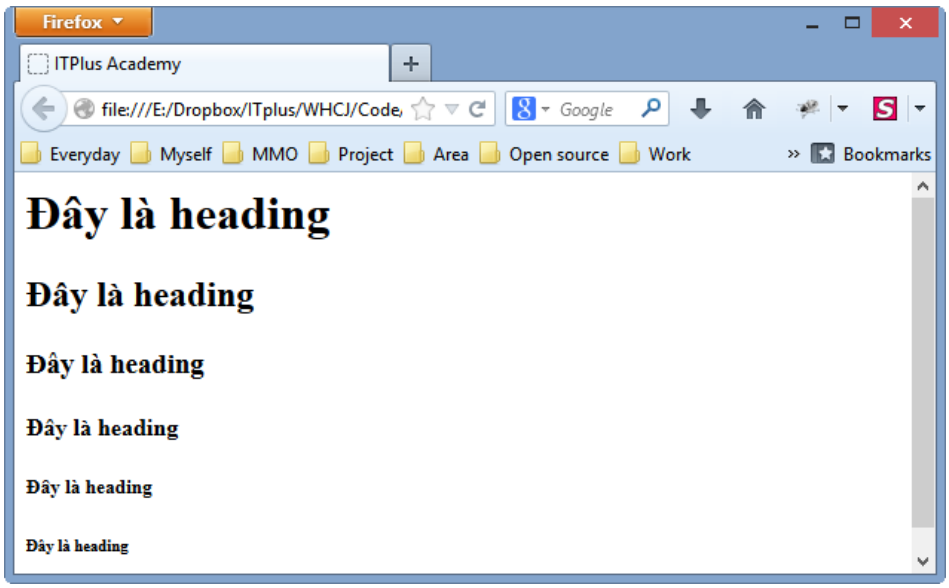
1.4.1. Các thẻ tiêu đề (Headings)

Bao gồm các thẻ từ `<h1>` đến `<h6>` thường được dùng để thể hiện cho tiêu đề bài viết, bản tin, các mục nhấn mạnh. Font chữ (size) của nội dụng trong các thẻ giảm từ `<h1>` đến `<h6>` (`<h1>` lớn nhất và `<h6>` nhỏ nhất). HTML sẽ tự động thêm một dòng trắng trước và sau mỗi heading. Việc sử dụng hợp lý các thẻ Heading là một trong những yếu tố khá quan trọng trong việc tạo cho công cụ tìm kiếm đánh giá tốt khi tìm kiếm trên website của bạn.

Ví dụ:

```
<h1>Đây là heading</h1>
<h2>Đây là heading</h2>
<h3>Đây là heading</h3>
<h4>Đây là heading</h4>
<h5>Đây là heading</h5>
<h6>Đây là heading</h6>
```

Kết quả nhận được sẽ là:



Hình 1.10. Kết quả mô tả các thẻ Heading

1.4.2. Thẻ trung hòa

*a. Nhóm phần tử với thẻ *

Thẻ `<span>` trong HTML thật ra là một thẻ trung hòa, nó không thêm hay bớt bất cứ một thứ gì vào một tài liệu HTML cả. Nhưng chính nhờ tính chất trung hòa này mà nó lại là một công cụ đánh dấu tuyệt vời để qua đó chúng ta có thể viết CSS định dạng *(sẽ được đề cập trong những bài sau của giáo trình)* cho các phần tử mong muốn.

Ví dụ: Đoạn HTML sau trích dẫn câu nói của chủ tịch Hồ Chí Minh:

```
<p>Không có gì quý hơn độc lập, tự do</p>
```

Yêu cầu ở đây là chúng ta hãy dùng CSS tô đậm 2 từ *độc lập, tự do*. Để giải quyết vấn đề này, chúng ta sẽ thêm thẻ `<span>` vào đoạn HTML như sau:

```
<p>Không có gì quý hơn <span  class="nhanmanh">độc lập, tự do</span)</p>
```

Và bây giờ chúng ta có thể viết CSS cho yêu cầu trên:

```
.nhanmanh{font-weight:bold}
```

*b. Nhóm phần tử với thẻ *

Cũng như `<span>`, `<div>` cũng là một thẻ trung hòa và được thêm vào tài liệu HTML với mục đính nhóm các phần tử lại cho mục đích định dạng bằng CSS. Tuy nhiên, điểm khác biệt là `<span>` dùng để nhóm một khối phần tử trong khi đó `<div>` có thể nhóm một hoặc nhiều khối phần tử.

Ví dụ sau đây về việc liệt kê danh sách tỉnh, thành phố sao cho danh sách tỉnh có màu đỏ, danh sách thành phố có màu xanh:

```
<p>Danh Sách Các Tỉnh, Thành Phố Việt Nam:</p>
<ul>
<div id="tp">
<li>Hà Nội</li>
<li>TP. Hồ Chí Minh</li>
<li>Đà Nẵng</li>
</div>
<div id="tinh">
<li>Thừa Thiên Huế</li>
<li>Khánh Hòa</li>
<li>Quãng Ninh</li>
<li>Tiền Giang</li>
</div>
</ul>
```

Và đoạn CSS cho mục đích này sẽ là:

```
#tp{color:#FF0000;}
#tinh{color:#0000FF;}
```

c. Ưu điểm khi sử dụng các thẻ trung hòa

- *Tăng tốc độ website*: Thời gian load một website sẽ nhanh hơn. Dung lượng của một trang web sẽ nhẹ hơn 50% so với cách làm cổ điển. Chưa kể, cùng 1 trang nhưng nếu dùng table thì code HTML sẽ lớn hơn dùng div. Code CSS có dùng div có thể sẽ lớn hơn, nhưng CSS được load 1 lần, sau đó được cache trên client.

- *Thay đổi website nhanh hơn*: Thông qua một thao tác về thiết kế, thay đổi CSS, thì hàng ngàn trang web sử dụng nó sẽ được thay đổi theo.

- *Khả năng phát triển*: Sử dụng CSS hiệu quả sẽ tương tự như sử dụng cấu trúc nội dung với HTML. Với cấu trúc HTML, chúng ta chỉ giữ lại các tag cho đoạn văn, các tag tiêu đề (H1 -> H6), bảng dành cho một số ít bảng biểu đặc biệt sẽ tăng khả năng đáp ứng lượng khách viếng thăm mà không cần xây dựng một phiên bản độc lập để giảm tải.

- *Dễ kiểm soát thông qua vị trí các thành phần trên web*: Việc đánh ID trên từng thẻ DIV, sẽ dễ dàng hơn cho bạn khi cần kiểm soát và hỗ trợ tốt cho Ajax.

- *Các trang web tách biệt phần thiết kế và nội dung*: Giữ lại thiết kế trong một file liên kết, bạn có thể tiết kiệm nhiều thời gian khi cần thêm bớt các thành phần của một trang web.

- *Cải thiện vị trí trong các công cụ tìm kiếm như google*: Tuy nhiên việc này còn phụ thuộc rất nhiều vào thuật toán của các công cụ tìm kiếm.

1.4.3. Thẻ định dạng văn bản, kiểu chữ

a. Thẻ định dạng văn bản

Chúng ta luôn thấy được các bài viết trên các website được chia thành nhiều đoạn văn có định dạng giống như một văn bản word. Với HTML, chúng ta có thẻ dùng cặp thẻ `<p></p>` để phân chia nội văn bản thành từng đoạn như vậy.

Các thuộc tính của thẻ `<p>`**:**

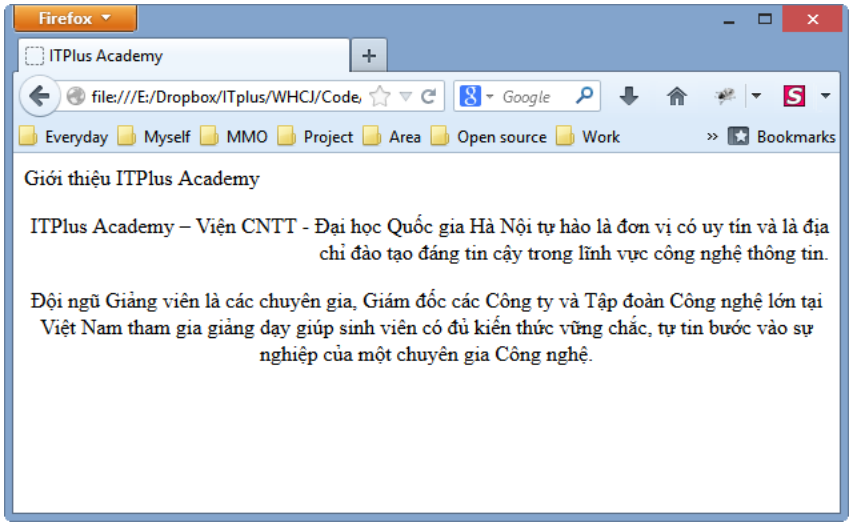
STT	Tên	Giá trị	Mô tả
1	`Align`	`left, right, center, justify`	Căn lề văn bản giống như trên tài liệu word.
2	`Dir`	`ltr, rtl`	Hướng của văn bản từ trái qua phải hay từ phải qua trái

Bảng 1.1. Các thuộc tính của thẻ p

Ví dụ chúng ta có một file HTML như sau:

```
<html>
    <head>
        <meta charset="utf-8">
        <title>ITPlus Academy</title>
    </head>
    <body>
        <p>Giới thiệu ITPlus Academy</p>
        <p align="right">ITPlus Academy-Viện CNTT-Đại học Quốc gia Hà Nội
        tự hào là đơn vị có uy tín và là địa chỉ đào tạo đáng tin cậy trong
        lĩnh vực công nghệ thông tin.</p>
        <p align="center">Đội ngũ Giảng viên là các chuyên gia,
        Giám đốc các Công ty và Tập đoàn Công nghệ lớn tại Việt Nam
        tham gia giảng dạy giúp sinh viên có đủ kiến thức vững chắc,
        tự tin bước vào sự nghiệp của một chuyên gia Công nghệ. </p>
    </body>
</html>
```

Kết quả thu được như sau:



Hình 1.11. Kết quả mô tả thẻ p và các thuộc tính định dạng văn bản

b. Các thẻ định dạng kiểu chữ

Khi trình bày một trang Web, có lúc cần nhấn mạnh, hay để tạo sự khác biệt, bạn cần phải hiển thị văn bản ở các dạng đậm, nghiêng, gạch chân... HTML có các tag định dạng kiểu chữ để giúp bạn.

STT	HTML	Kết quả
1	`<u>Đây là tag gạch chân</u>`	<u>Đây là tag gạch chân</u>

2	`<i>Đây là tag nghiêng</i>`	*Đây là tag nghiêng*
3	`<tt>Đây là tag đánh máy</tt>`	Đây là tag đánh máy
4	`<b>Đây là tag đậm</b>`	**T là tag đậm**
5	`<em>Đây là tag em</em>`	*Đây là tag em*
6	`<sub>Định dạng chữ ở dưới</sub>`	Định dạng chữ ở dưới
7	`<sup>Định dạng chữ ở trên</sup>`	Định dạng chữ ở trên

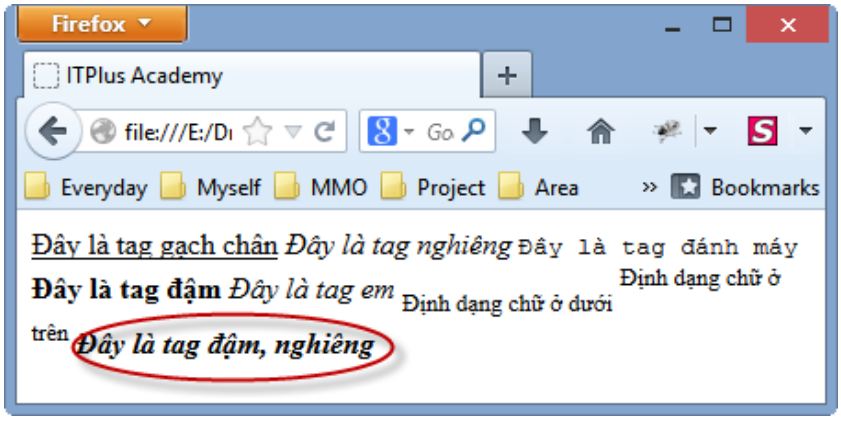
Bảng 1. 2. Các thẻ định dạng font chữ

Bạn có thể kết hợp các tag này với nhau, miễn là chúng được lồng vào nhau một cách chính xác.

Ví dụ:

```
<u>Đây là tag gạch chân</u>
<i>Đây là tag nghiêng</i>
<tt>Đây là tag đánh máy</tt>
<b>Đây là tag đậm</b>
<em>Đây là tag em</em>
<sub>Định dạng chữ ở dưới</sub>
<sup>Định dạng chữ ở trên</sup>
<b><i>Đây là tag đậm, nghiêng</i></b>
```

Kết quả thu được như sau:



Hình 1.12. Kết quả mô tả về các thẻ định dạng kiểu chữ

1.4.4. Thẻ đánh dấu siêu liên kết

Khi lướt qua các website trên internet chúng ta luôn nhận ra rằng các liên kết từ website này tới website khác, từ bài viết này từ bài viết khác được bố trí ở mọi nên trên trang web. Bạn chỉ cần click vào nó để chuyển sang một trang web. Các siêu liên kết còn giúp chúng tra chuyển tới vị trí của một thành phần nào đó trên trang web một cách nhanh chóng. Ta thường gặp trường hợp này ở nhưng trang web có nôi dụng dài và sẽ có một siêu liên kết cho bạn nhấn và để quay lên đầu trang.

a. Cú pháp

Để đánh dấu siêu liên kết các bạn sẽ dùng thẻ `<a>`. Cú pháp của nó là:

```
<a href= "liên kêt" target= "giá trị" title= "text" >Văn bản hiển thị< /a>
```

Trong đó:

- *href*: thuộc tính bắt buộc là một liên kết tới file hoặc một URL

- *target*: cách thức mở trang mới. Có nhiều giá trị cho thuộc tính này nhưng trên thực thế chỉ có một giá trị có tác dụng trực quan nhất đó là "_blank". Đối với các trình duyệt cũ như Internet Explorer 6 sẽ mở liên kết tại một cửa sổ mới, còn các trình duyệt hỗ trợ tab hiện nay sẽ mở ra một tab mới. Và nếu bạn để trống hoặc không có target trình duyệt sẽ chuyển liên kết ngay tại tab hoặc cửa sổ hiện hành.

- *title*: text hiển thị khi di chuột qua linh liên kết

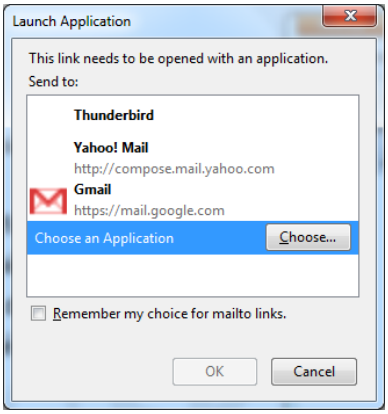
b. Liên kết gửi mail

Ngoài liên tới tới một trang web khác hay một file nào đó. Thẻ `<a>` còn giúp chúng ta liên kết tới một số ứng dụng như *mail* hay *Yahoo Message* như các bạn thường thấy tại phần hỗ trợ online của các trang web hiện nay.

Để liên kết mới một ứng dụng mail với một địa chỉ mail nào đó bạn có thể dùng: `href= "mailto:diachimail".` Khi bạn click vào liên kết này một cửa sổ sẽ hiện lên hỏi bạn sẽ chọn ứng dụng nào để gửi mail. Ví dụ như:

```
<a href= "mailto:info@itplus.edu.vn" >Gửi mail ITPLUS Academy</a>
```

Khi bạn click vào cửa sổ chọn ứng dụng sẽ hiện lên.



Hình 1.13. Cửa sổ chọn ứng dụng của trình duyệt Firefox

Khi đó bạn chọn ứng dụng mà bạn muốn địa chỉ mail sẽ tự động điền cho bạn.

Hoặc bạn có thể link nó tới ứng dụng *Yahoo Messenger* hoặc *Skype* với giá trị thuộc tính lần lượt là: `ymsr:yahoo_account, skype:skype_account`. Khi bạn click vào cửa sổ chọn ứng dụng cũng sẽ mở và bạn chỉ cần nhấn nút OK để mở ứng dụng, bạn có thể sẽ phải đăng nhập nhưng cửa sổ chat với tài khoản được cung cấp vẫn sẽ hiển thị.

c. Liên kết sử dụng ảnh

Hơn nữa chúng ta không chỉ có thể tạo liên kết bằng văn bản mà chúng ta có thể tạo bằng hình ảnh nữa. Chỉ cần chèn thẻ `<img/>` *(Sẽ được học trong những bài sau của giáo trình)* và cặp thẻ `<a></a>` là chúng ta đã có một liên kết bằng hình ảnh.

Ví dụ:

```
<a href= "http://itplus.edu.vn" title= "ITPLUS Academy">
     <img src= "logo.jpg" alt= "ITPLUS Academy logo"/>
</a>
```

d. Liên kết trong nội bộ trang

Để chuyển từ phần này sang phần khác trên cùng 1 trang web ví dụ từ phần trên của trang web ta có thể chuyển xuống phần cuối nếu ta sử dụng thẻ a kết hợp với thuộc tính ID của phần tử HTML (id sẽ được giới thiệu kỹ hơn vào phần cuối của chương)

Ví dụ ta có thẻ `<div id= "top">...</table>` đặt ở trên cùng của trang web. Và sau rất nhiều nội dung khác, ta đặt một thẻ `<a href= "#top">Lên đầu trang</a>` ở dưới cùng thì khi ta click và siêu liên kết này URL sẽ xuất hiện thêm `"#top"` ở sau cùng. Và trình duyệt sẽ chuyển khung nhìn tới chính cái bảng có `id = "top"` mà không hề load lại trang web.

Như vậy chúng ta có thể kết hợp thẻ `<img/>` và các thẻ đánh dấu văn bản khác để tạo ra nhưng liên kết linh hoạt và đẹp mắt.

B. ÔN TẬP

2.1. Câu hỏi ôn tập

Câu 1. HTML là viết tắt của cụm từ nào sau đây?

A. Hyperlinks and Text Markup Language

B. Home Tool Markup Language

C. Hyper Text Markup Language

Câu 2. Chuẩn thiết kế web (Web standards) do ai tạo ra?

A. Google

B. Mozilla

C. The World Wide Web Consortium

D. Microsoft

Câu 3. Lựa chọn thẻ tiêu đề (HTML heading) lớn nhất?

A. <h1>

B. <h6>

C. <head>

D. <heading>

Câu 4. Để thêm màu nền trong HTML ta làm như thế nào?

A. <background>yellow</background>

B. <body style="background-color:yellow;">

C. <body background="yellow">

Câu 5: Thẻ HTML nào là thẻ liên kết trong các trường hợp sau?

A. <a url="http://www.itplus.edu.vn"> itplus.edu.vn </a>

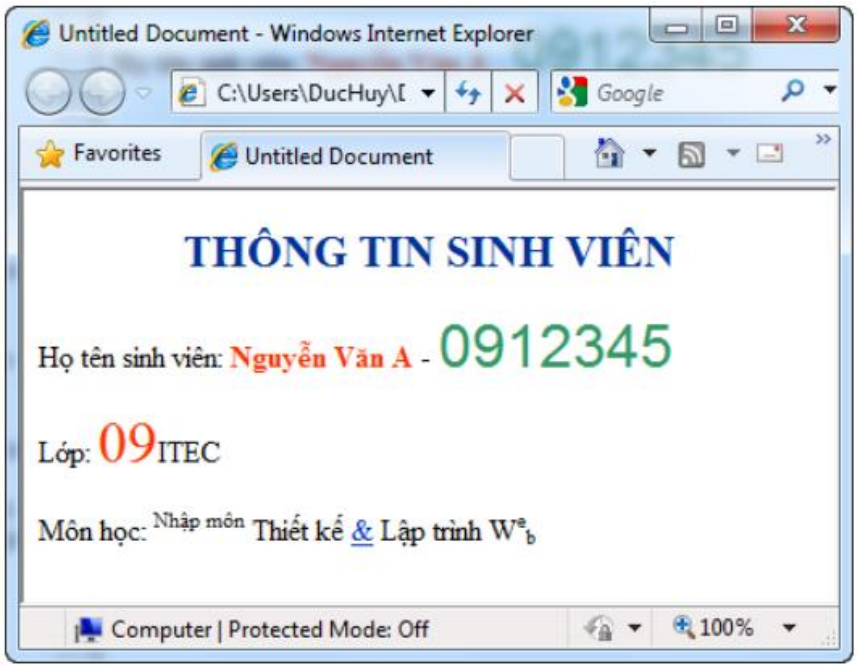
B. <a href="http://www. itplus.edu.vn "> itplus.edu.vn </a>

C. <a name="http://www. itplus.edu.vn "> itplus.edu.vn </a>

D. <a>http://www. itplus.edu.vn</a>

2.2. Bài tập

Bài 1. Tạo file **bai1.1.html** với nội dung như sau:



Bài 2. Tạo file bai1.2.html với nội dung như sau:

Tho ve me hien - Windows Internet Explorer
C:\Users\DucHuy\
Google
Favorites
Tho ve me hien
MẸ !
Mẹ là cả một trời thương
Mẹ là cả một thiên đường trần gian
Công ơn cha mẹ tựa biển trời
Làm sao báo hiếu hỡi người ơi?
Nếu chưa báo hiếu đừng bất hiếu
Bất hiếu làm ta khổ muôn đời.
Một đời vốn liếng mẹ trao
Mẹ cho tất cả mẹ nào giữ riêng
Mẹ hiền như một bà tiên
Mẹ theo con suốt hành trình con đi
Computer | Protected Mode: Off
100%

BÀI 2. CHÈN DỮ LIỆU ĐA PHƯƠNG TIỆN

Để một trang web trở nên sinh động và hấp dẫn hơn thì chỉ có văn bản thông thường thôi chưa đủ mà cần phải có sự đóng góp của các dữ liệu đã phương tiện như hình ảnh, âm thanh, video... Bài này sẽ giới thiệu các thẻ đánh dấu cho hình ảnh, âm thanh, video và các thành phần multimedia khác...

A. NỘI DUNG

1.1. Chèn dữ liệu hình ảnh

1.1.1. Cú pháp và các thuộc tính

Cú pháp để chèn hình ảnh vào trang web:

```
<img src="đường dẫn ảnh"  border="viền" alt="Nội dung thay thế"/>
```

Các thuộc tính của thẻ img:

STT	Thuộc tính	Giá trị	Mô tả
1	src	Đường dẫn ảnh	Địa chỉ của tập tin hình ảnh, thường sử dụng địa chỉ tương đối, không phụ thuộc vị trí của tập tin ảnh trên đĩa. Đây là thuộc tính bắt buộc phải có.
2	alt	Text	Trình duyệt sẽ hiện thị đoạn text thay thế nếu ảnh không tồn tại. Nếu theo chuẩn thì đây là một thuộc tính bắt buộc. Nhưng nếu không có trình duyệt vẫn hiển thị ảnh bình thường.
3	width, height	pixel	Đây là độ rộng và cao mà bạn muốn hiển thị ảnh. Không phải là kích thước thực tế của ảnh. Nếu không có ảnh sẽ được hiện thị 100%. Nếu chỉ có 1 trong 2 thì chiều còn lại sẽ hiển thị theo tỉ lệ thuận. Ví dụ một bức ảnh có kích thước là 200px * 200px bạn có thể hiện thị trên webiste là width = 20px height = 30px hoặc chỉ một trong 2 thuộc tính này thuộc tính còn lại tự động tính cho bạn.
4	align	vị trí	Ví trị căn của ảnh so với văn bản
5	vspace	pixel	Độ rộng khoảng trắng 2 bên trái và phải của ảnh.
6	hspace	pixel	Độ rộng khoảng trắng 2 bên trên và dưới của ảnh.
7	border	pixel	Độ dày của đường viền

Bảng 2.1. Các thuộc tính của thẻ img

1.1.2. Đường dẫn của ảnh

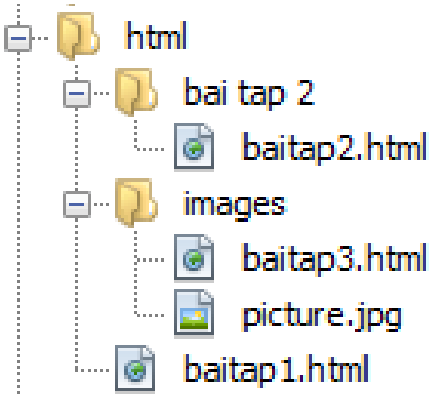
Với giá trị đường dẫn của ảnh bạn có thể để bất cứ đường dẫn nào kể cả các đường dẫn trên website như http://itplus.edu.vn/images.jpg hay đường dẫn local (trên máy tính). Đối với đường dẫn local thì các bạn phải hiểu được thế nào là đường dẫn tương tối và đường dẫn tuyệt đối.

Đường dẫn tuyệt đối được bắt đầu từ tên ổ cứng cho đến các thư mục con và đến tên file. Ví dụ `C:\html\picture.jpg`. Như vậy chỉ cần file ảnh `picture.jpg` ở cố định đúng tại thư mục html thuộc ổ C bạn có thể trỏ tới nó dù file HTML của bạn ở bất kỳ đâu.

Nhưng trong trường hợp bạn để file HTML và ảnh cùng trong thư mục html và thường là khi làm việc chúng ta luôn gom các file lại. Và bạn sử dụng đường dẫn tuyệt đối, hãy thử tưởng tượng bạn copy thư mục đó đi cho khác hay chia sẻ cho bạn bè và họ lưu thư mục đó ở ổ D. Chắc chắn bức ảnh sẽ không được hiển thị.

Vì vậy người ta luôn dùng đường dẫn tương đối cho trường hợp này và áp dụng cho nhiều thứ khác không chỉ riêng HTML. Đường dẫn tương đối là tùy thuộc vào ví trí của file trỏ tới một file khác.

Ví dụ ta có một cây thư mục như sau:



Hình 2.1. Ví dụ về cấu trúc cây thư mục

Nguyên tắc của đường dẫn tương đối là tính từ vị trí của file mà trỏ tới file khác với:

- `../tên thư mục/file`: cứ mỗi `../` bạn ra đi ra ngoài một cấp thư mục sau đó trỏ tới file thuộc cấu trúc thư mục được đặ trong đường dẫn.

- `tên thư mục/file`: tương ứng với bạn bạn trỏ tới file thuộc thư mục nằm cùng cấp với file của bạn.

Ví dụ:

- Để hiển thị ảnh `picture.jpg` trên file `baitap1.html` bạn có thể dùng đường dẫn sau: `"images/pictuer.jpg"` vì thư mục `images` nằm cùng cấp với file `baitap1.html`

- Để hiển thị ảnh `picture.jpg` trên file `baitap2.html` bạn có thể dùng đường dẫn sau: `"../images/pictuer.jpg"`. Dấu `../` ứng với bạn đã ra khỏi thư mục chứ file

baitap2.html 1 cấp nghĩ là thư mục html khi đó bạn tiếp tục đi từ thư mục html vào thư mục images và trỏ tới picture.jpg.

- Để hiển thị ảnh picture.jpg trên file baitap3.html bạn chỉ cần dùng đường dẫn sau: "pictuer.jpg" vì file picturer.jpg nằm cùng cấp với file baitap3.html

Ngoài ra phương pháp sử dụng đường dẫn tương đối cũng được sử dụng với môi trường internet với URL cũng như đường dẫn của file.

Tuy nhiên điều đặc biệt khi sử dụng đường dẫn ảnh với url có thể sử dụng dạng đường dẫn tương đối sau: "/path/picture" thì đường dẫn đó sẽ được hiểu là "domain/part/picture".

1.1.3. Căn chỉnh vị trí của ảnh

Khi một hình ảnh đi kèm với một đoạn văn hay thành phần nào đó bạn có thể có căn chỉnh vị trí của ảnh so với các thành phần đó với thuộc tính align. Ở đây thuộc tính align của ảnh khác với của thẻ p hay các thẻ khác. Nó có các giá trị khác và được hiểu là cách các thành phần khác hiển thị thế nào với bức ảnh đó.

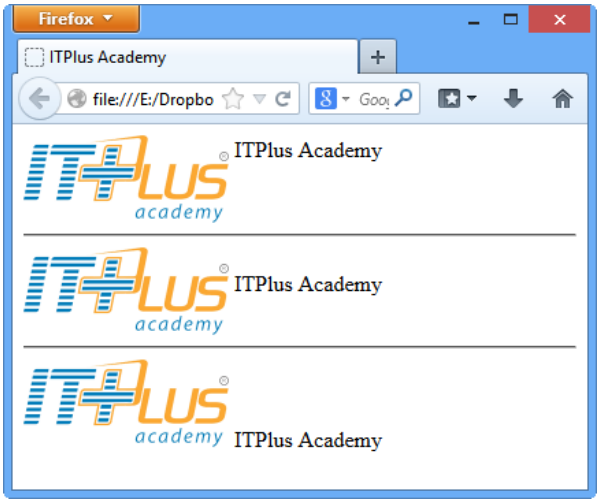
Thuộc tính align text có các giá trị:

- top: Văn bản sẽ hiển thị ngang với phần trên của bức ảnh.
- middle: Văn bản sẽ hiển thị ngang với phần trên của bức ảnh.
- bottom: Văn bản sẽ hiển thị ngang với phần dưới của bức ảnh.

Ví dụ: Sử dụng thuộc tính align của thẻ img:

```
<img src="images/itplus.png" align="top" width="150"/>
ITPlus Academy
<hr/>
<img src="images/itplus.png" align="middle" width="150"/>
ITPlus Academy
<hr/>
<img src="images/itplus.png" align="bottom" width="150"/>
ITPlus Academy
```

Kết quả:



Hình 2.2. Căn chỉnh vị trí của ảnh với thuộc tính align

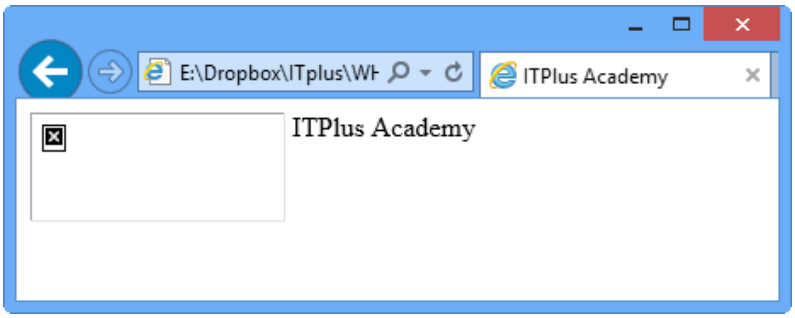
1.1.4. Thuộc tính Alt

Khi trang Web được xem bằng trình duyệt mà người sử dụng tắt việc trình bày hình ảnh để tiết kiệm thời gian download, hoặc đường dẫn ảnh không chính xác, nó sẽ không được hiển thị. Thuộc tính `alt = ""` cho phép thay thế vào vị trí hình ảnh một chuỗi văn bản mô tả nó.

Ví dụ**:**

```
<img src="images/itplus.png" align="top" width="150"/>
ITPlus Academy
```

Kết quả:



Hình 2.3. Thuộc tính alt của thẻ <img> trên trình duyệt Internet Explorer

Ta thấy với trình duyệt Internet Explorer, một bức ảnh không tồn tại sẽ được hiển thị như trên và được thay thế bởi từ "Ví dụ". Còn với các trình duyệt khác đơn thuần chỉ là thay thế bằng giá trị của thuộc tính alt.

1.1.5. Độ rộng và chiều cao của ảnh

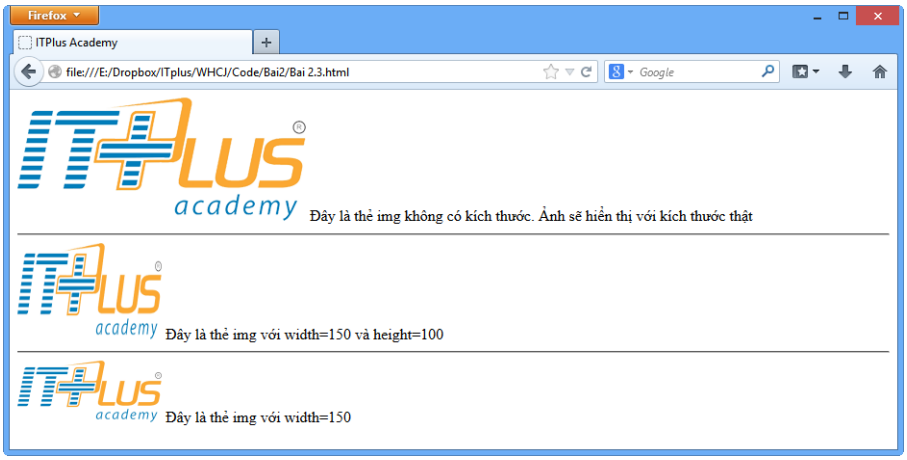
Trong quá trình biên tập website có thể bạn không chuẩn bị được những bức ảnh có kích thước chuẩn như bạn muốn thể hiện trên webiste vì vậy bạn cần sử dụng thuộc tính

width và height để chỉnh lại kích thước của ảnh dù to hay nhỏ về theo đúng ý mình để có được bố cục website đẹp mắt.

Ví dụ:

```
<html>
    <head>
        <meta charset="utf-8">
        <title>ITPlus Academy</title>
    </head>
    <body>
                <img src="images/itplus.png"/>
                Đây là thẻ img không có kích thước. Ảnh sẽ hiển thị với kích
thước thật
                <hr/>
                <img src="images/itplus.png" width="150" height="100"/>
                Đây là thẻ img với width=150 và height=100
                <hr/>
                <img src="images/itplus.png" width="150"/>
                Đây là thẻ img với width=150
    </body>
</html>
```

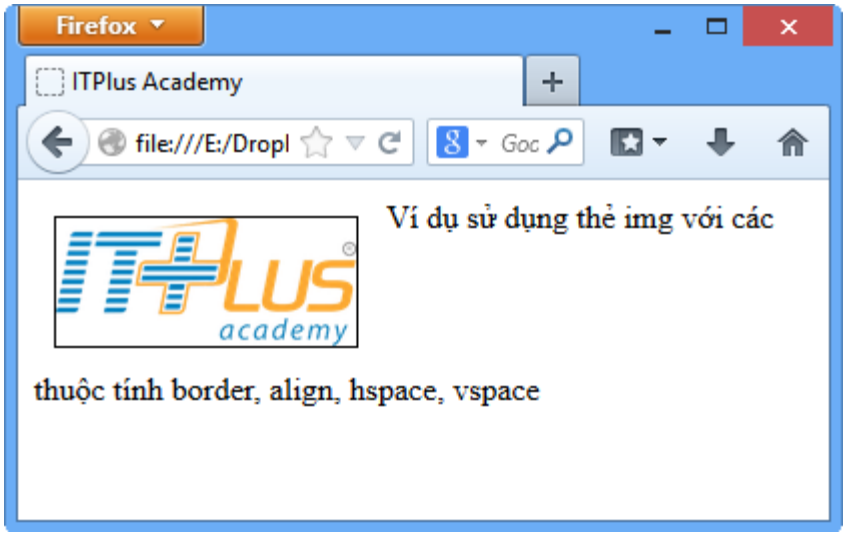
Kết quả:



Hình 2.4. Quy định độ rộng và chiều cao của ảnh với width và height

1.1.6. Khoảng cách xung quanh ảnh

Để giãn cách ảnh so với văn bản xung quanh nó tạo độ thông thoáng và dễ nhìn, chúng ta có thể sử dụng 2 thuộc tính vspace cho chiều dọc và hspace cho chiều ngang.



Hình 2.5. Giãn cách ảnh với thuộc tính vspace và hspace

1.2. Chèn dữ liệu âm thanh

Để chạy được dữ liệu âm thanh *(nhạc nền)* trên web không phải là vấn đề đơn giản. Cần phải biết các thủ thuật để đảm bảo rằng các tập tin âm thanh của bạn sẽ chạy tốt trên tất cả các trình duyệt *(Internet Explorer, Chrome, Firefox…)*. Trình duyệt khách nhau sẽ hỗ trợ các định dạng âm thanh khác nhau và một số trình duyệt cần phải thêm ứng dụng hỗ trợ. Ta dùng thẻ `<bgsound>` (đối với IE và Mosaic) hoặc `<embed>` (đối với IE và Netscape). Hai thẻ này không cần có thẻ đóng.

```
<bgsound src="filename" loop="infinite">
```

Trong đó:

- `src`: là đường dẫn tới tên tập tin âm thanh bạn muốn chèn vào (có đuôi .mid hoặc .wav).

- Thuộc tính `loop`: chỉ ra số lần phát lại bản nhạc này (1,2,3...) hoặc lặp lại vô tận infinite hay -1.

Thẻ `<bgsound>` sẽ bắt trang Web tự động nạp file nhạc được liên kết khi ta mở trang và tự động chơi bản nhạc đó, bạn không thể điều khiển dừng lại hay phát lại theo ý thích, bởi nó không hiện ra thanh điều khiển. Nhưng đối với thẻ `<embded>` thì khác, nó cho phép bạn chọn chế độ hiện thanh điều khiển để người truy cập có thể tùy ý chơi nhạc.

```
<embed src="filename" width="n" height="n" autostart="false" loop="false">
```

Trong đó:

Thuộc tính	Mô tả
`SRC="URL"`	Đường dẫn đến tệp âm thanh
`HEIGHT=n`	Chiều cao của khung nhìn cho giao diện điều khiển của

	phần mềm phát âm thanh
WIDTH=n	Chiều rộng của khung nhìn cho giao diện điều khiển của phần mềm phát âm thanh
AUTOSTART=true|false	Âm thanh tự động được kích hoạt khi bắt đầu nạp tài liệu hiện thị (nên đặt là true)
LOOP=true|false	Thiết lập việc lặp lại nhiều lần hay chỉ một lần
hidden	Dùng để làm ẩn thanh điều khiển này (true). Điều này sẽ tương tự như dùng thẻ <bgsound>

Đặt thẻ <embed> ở đâu thì khung nhìn của giao diện điều khiển việc phát tệp âm thanh xuất hiện ở đó.

Một điểm khác biệt nữa giữa hai thẻ này là: Thẻ <bgsound> sẽ tạm ngừng chơi nhạc nếu như cửa sổ trình duyệt bị thu nhỏ *(minimize)* và sẽ chơi tiếp tục khi cửa sổ được mở trở lại *(restore hay maximize)*. Nếu có nhiều trang web sử dụng thẻ <bgsound> thì chỉ có cửa sổ active sẽ chơi nhạc, các cửa sổ khác sẽ tạm ngưng phát nhạc. Còn thẻ <embed> thì vẫn chơi nhạc bất kể tình trạng của cửa sổ như thế nào.

Ví dụ: Nếu bạn hiện đang mở loa máy tính bạn sẽ được nghe bản nhạc (file audio.mid). Bản nhạc này được liên kết trong trang Web này bởi đoạn HTML sau:

```
<bgsound src="audio.mid">
```

Thế nhưng, nếu bạn thay vào đó bằng thẻ <embed>, bạn sẽ thấy như kết quả ở hình dưới. Và đoạn mã HTML này liên kết đến file nhạc *audio2.imd*, tuy nhiên bạn phải bấm nút Play mới có thể nghe được bởi thuộc tính autostart được gán giá trị bằng false.

```
<embed src="audio2.mid" autostart="false" loop="false" hidden="false">
```

1.3. Chèn dữ liệu video

Việc chèn tệp video vào tài liệu HTML cũng giống như chèn tệp âm thanh. Tuy nhiên, cần lưu ý các tệp video đều có kích thước khá lớn nên hãy cân nhắc về tốc độ đường truyền để có thể tải nhanh được video.

Để chèn video vào trang web ta sử dụng cú pháp sau:

```
<embed src=”film/film.mov” height=”44” width=”60”>
```

Trong đó, các thuộc tính cũng tương tự như chèn âm thanh:

Thuộc tính	Mô tả
SRC=”URL”	Đường dẫn đến video
HEIGHT=n	Chiều cao của khung nhìn cho giao diện điều khiển của phần mềm phát video

WIDTH=n	Chiều rộng của khung nhìn cho giao diện điều khiển của phần mềm phát video
AUTOPLAY=true|false	Đặt cho video tự chạy khi bắt đầu nạp tài liệu (không bắt buộc)
CONTROLLER=true|false	Cho hiện giao diện điều khiển của phần mềm video ở liền ngay phía dưới khung hình. Thuộc tính này không bắt buộc. Mặc định là true.
LOOP=true|false	Thiết lập việc lặp lại nhiều lần hay chỉ một lần

1.4. Chèn Flash và các thành phần Multimedia khác

Ngoài các đối tượng như hình ảnh, âm thanh, video ta còn có thể chèn các đối tượng Multimedia khác như các tập tin *Real Media (*.ra), QuickTime (*.mov), WindowMedia (*.asf)*... nhưng máy tính bạn phải cài đặt những chương trình hỗ trợ này.

Để chèn các thành phần này bạn dùng thẻ <object> (với IE) hoặc thẻ <embed> (với Netscape).

Cú pháp để chèn các đối tượng này vào website như sau:

```
<OBJECT CLASSID=classID CODEBASE=đường_dẫn ARCHIVE=tên_file WIDTH=chiều_rộng
HEIGHT=chiều_cao STANDBY="Waiting...">
  <PARAM NAME=tên VALUE=giá_trị>
  <PARAM NAME=tên VALUE=giá_trị>
  <EMBED SRC=tên_file WIDTH=chiều_rộng HEIGHT=chiều_cao></EMBED>
</OBJECT>
```

Trong đó, các thuộc tính của thẻ <OBJECT>:

- CLASSID: Tên địa chỉ của Object trong registry của máy tính

- CODEBASE: Đường dẫn mặc định của các thuộc tính khác

- ARCHIVE: Nếu có dử dụng file kèm theo, thì đây là đường dẫn chứa các file đó được nén lại thành một file bởi Winzip.

- WIDTH, HEIGHT: chiều rộng và cao của Object.

- STANDBY: Dòng text hiển thị trong quá trình chờ download file.

Các thuộc tính của thẻ <PARAM>:

- NAME: Tên của biến

- VALUE: Giá trị của biến

Thẻ <EMBED> được lồng vào trong thẻ <OBJECT> nhằm để trình duyệt nào không hiểu thẻ OBJECT (Netscape) sẽ tìm kiếm bên trong nó sẽ được thẻ EMBED và thi hành nó. Ngược lại trình duyệt nào hiểu thẻ OBJECT (IE) sẽ bỏ qua thẻ EMBED nằm trong nó. Đây chỉ là cách chèn cơ bản, các dòng thẻ <PARAM...> cụ thể do những chương trình chèn vào quyết định.

B. BÀI TẬP

2.1. Câu hỏi ôn tập

Câu 1. Để chèn hình ảnh vào website sử dụng thẻ nào dưới đây?

D. img
E. alt
F. src
G. embed

Câu 2. Sử dụng thẻ nào để chèn âm thanh vào website?

E. bgsound
F. src
G. object
H. loop

Câu 3. Sử dụng thẻ nào sau đây để chèn video vào website?

A. embed
E. insert
F. src
G. object

2.2. Bài tập

Bài 1. Sử dụng các kiến thức đã được học để thực hiện một số yêu cầu sau:

1. Chèn hình ảnh vào website
2. Chèn nhạc nền cho website
3. Chèn video vào website

Bài 2. Tạo giao diện như sau, trình bày văn bản và ảnh đúng vị trí.(lưu ý: Dùng ảnh tuỳ ý để thay thế)

BÀI 3. LÀM VIỆC VỚI TABLE VÀ LIST

Việc sử dụng bảng *(table)* trong HTML là một vấn đề khá cần thiết. Nó không chỉ được dùng để hiển thị một định dạng bảng thông thường trên bài viết của bạn như những bảng thống kê mà trước khi CSS ra đời, dùng để tạo bộ cục cho trang web như chia các cột các ô chủ đạo của website.

Bên cạnh đó, danh sách trong HTML không còn dừng lại ở dạng danh sách có thứ tự hoặc không có thứ tự mà còn kết hợp với CSS để tạo ra nhiều dạng menu đa cấp đẹp mắt.

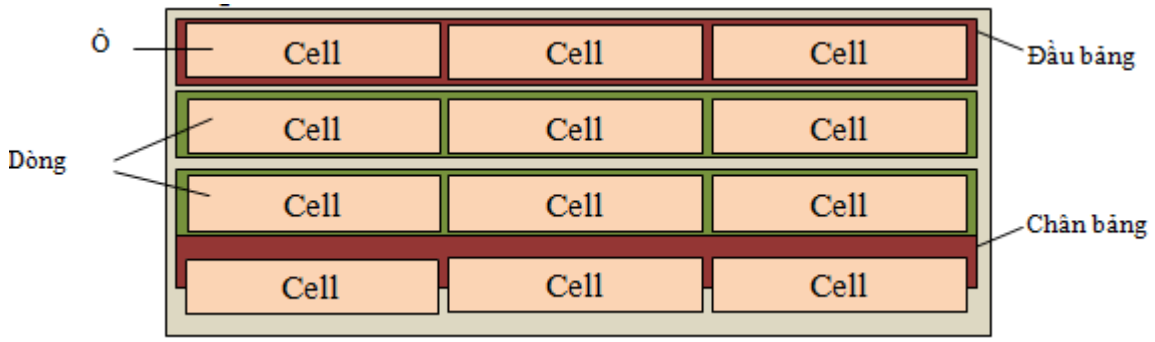
Bài này của giáo trình sẽ hướng dẫn các bạn sử dụng bảng với mục đích là tạo ra một thành phần dạng bảng để liệt kê một danh sách nào đó và sử dụng dụng danh sách để tạo ra một ứng dụng menu một cấp hoặc đa cấp.

A. NỘI DUNG

1.1. Làm việc với table

1.1.1. Những thẻ cơ bản của table

Một bảng trong HTML được chia theo bố cục: đầu bảng, các dòng, và chân bảng, mỗi thành phần đó lại chứa các ô, nhưng ô được đặt trên cùng sẽ có ảnh hưởng tới các các ô của các dòng dưới nó.



Hình 3.1. Cấu trúc bảng trong HTML

1.1.2. Thẻ được sử dụng để đánh dấu bảng

Bảng trong HTML được đánh dấu mở đầu bằng thẻ `<table>` và kết thúc bằng `</table>`

- Cặp thẻ `<thead></thead>` để đánh dấu đầu bảng.
- Cặp thẻ `<tbody></tbody>` để đánh dấu thân bảng.
- Cặp thẻ `<tfoot></tfoot>` để đánh dấu chân bảng.

Trong mỗi cặp thẻ này lại chia thành các dòng với các cặp thẻ `<tr></tr>`. Mỗi cặp thẻ `<tr></tr>` lại chứa các cặp thẻ `<td></td>` hoặc `<th></th>` đánh dấu các ô. Trong đó `<th>` sẽ mặc định văn bản chứa trong nó được in đậm và căn giữa.

Ba cặp thẻ trên có hoặc không khi có một trong 2 cặp thẻ `<thead></thead>` hoặc `<tfoot></tfoot>` thì nên có thêm cặp thẻ `<tbody></tbody>` để phân biệt rõ các dòng. Nếu không các dòng trong bảng vẫn được hiển thị bình thường.

Các thuộc tính của thẻ `<table>`:

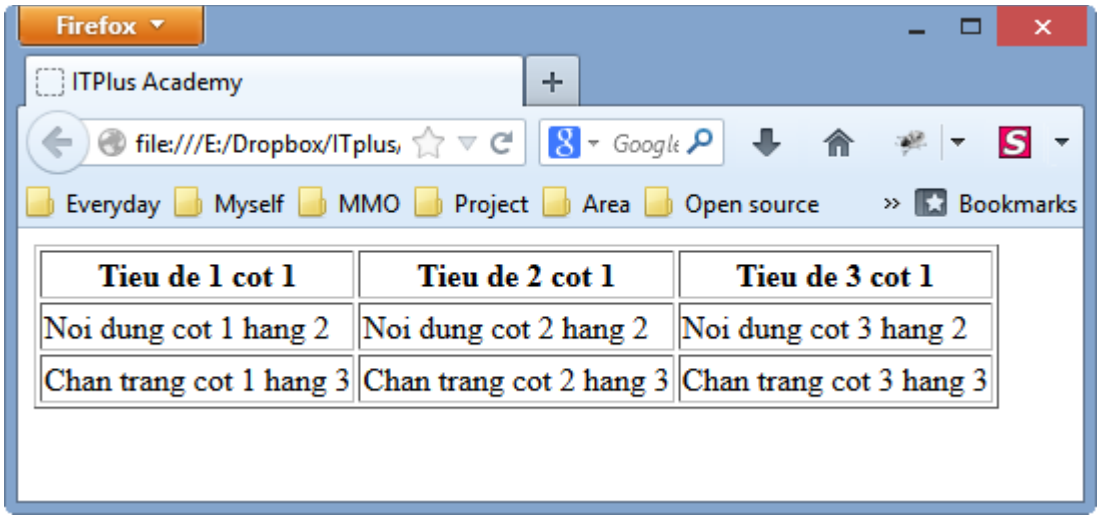
STT	Tên thuộc tính	Giá trị	Mô tả
1	`bgcolor`	- rgb(x,y,z) - #xxxxxx - tên mầu	Màu nền của bảng
2	`border`	Số pixel	Độ dày của đường viền
3	`cellspacing`	Số pixel	Khoảng cách giữa các ô
4	`cellpadding`	Số pixel	Khoảng cách giữa đường viền của ô tới nội dung của nó
5	`Width`	Số pixel, số phần trăm	Độ rộng của bảng. Có thể áp dụng cho thẻ `<th>` và `<td>`
6	`height`	Số pixel, số phần trăm	Chiều cao của bảng. Có thể áp dụng cho thẻ `<th>` và `<td>`

Bảng 3.1. Các thuộc tính định dạng bảng

Ví dụ với đoạn mã sau:

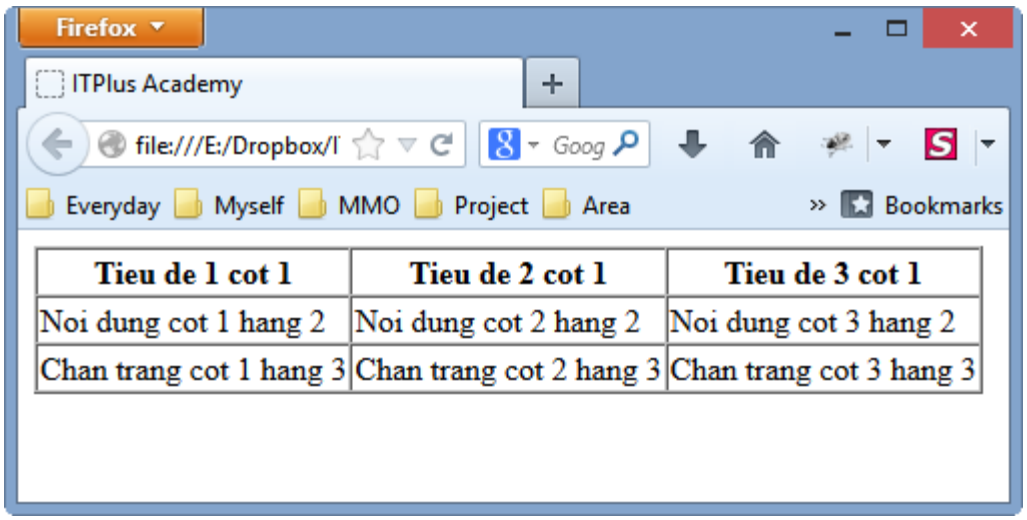
```
<table border = "1px" width="500px">
        <thead>
                <tr>
                        <th>Tieu de 1 cot 1</th>
                        <th>Tieu de 2 cot 1</th>
                        <th>Tieu de 3 cot 1</th>
                </tr>
        </thead>
        <tbody>
                <tr>
                        <td>Noi dung cot 1 hang 2</td>
                        <td>Noi dung cot 2 hang 2</td>
                        <td>Noi dung cot 3 hang 2</td>
                </tr>
        </tbody>
        <tfoot>
                <tr>
                        <td>Chan trang cot 1 hang 3</td>
                        <td>Chan trang cot 2 hang 3</td>
                        <td>Chan trang cot 3 hang 3</td>
                </tr>
        </tfoot>
</table>
```

Ta sẽ được một bảng như sau:



Hình 3. 2. Cấu trúc của bảng trong HTML

Chú ý: thuộc tính cellspacing mặc định là 2px nên ta sẽ thấy khoảng cách giữa các ô như trên nếu chúng ta gán giá trị cụ thể cho nó ví dụ là 0px ta sẽ được:



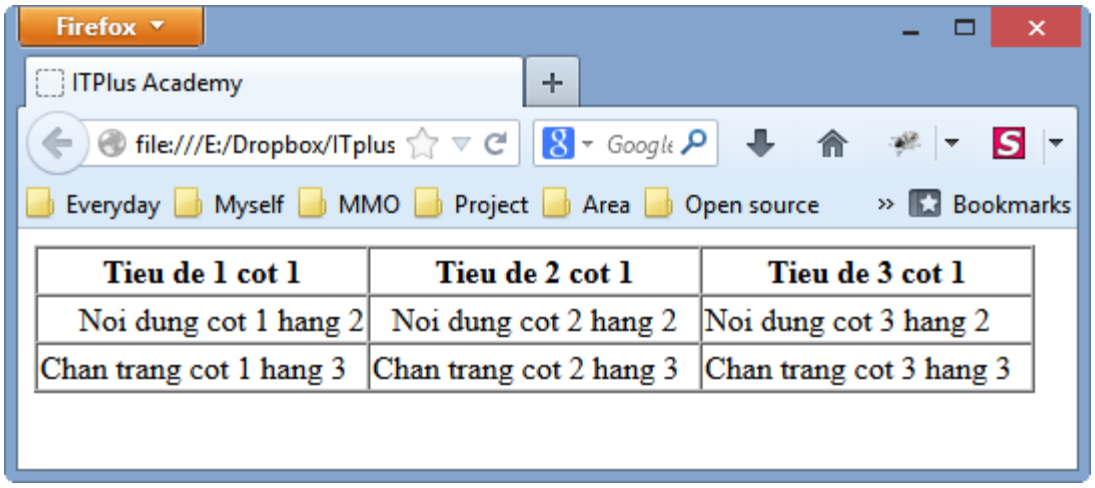
Hình 3. 3. Thẻ `<table>` với thuộc tính `cellspacing` bằng 0

1.1.3. Các thuộc tính của thẻ tr và td

a. Căn lề trong một ô

Đối với từng ô trong một bảng ta có thể căn lề cho chúng bằng thuộc tính `align` của thẻ `<td>`. Các giá trị của thuộc tính align cũng bao gồm: `left`, `right`, `center` và `justify`.

Ví dụ như bảng sau đây đã được căn lề cho một số ô bất kỳ:



Hình 3.4. Các thẻ **`<td>`** với thuộc tính **`align`**

b. Gom nhóm các ô theo cột

Như đã thấy các bảng mà chúng ta đã tạo trong phần nay luôn có độ rộng của các ô hay các cột là như nhau. Vậy có các nào để chúng ta tạo một bảng có định dạng như sau:

Tieu de 1 cot 1	Tieu de 2 cot 2	Tieu de 3 cot 3
Noi dung cot 1 gop cot 2 hang 2		Noi dung cot 3 hang 2
Chan trang cot 1 hang 3	Chan trang cot 2 hang 3	Chan trang cot 3 hang 3

Muốn như vậy chúng ta sẽ phải sử dụng đến thuộc tính `colspan` của thẻ `<td>` hoặc `<th>`

Thuộc tính `colspan=x` có tác dụng mở rộng các ô của bảng. Nó sẽ ghép `x` ô theo chiều từ từ trái sang phải để gom thành 1 ô. Và một điều các bạn nên để ý rằng nếu chung ta gom bao nhiêu ô lại với nhau thì trong mã HTML chúng ta phải xóa đi số các cột bị gom lại để tránh trường hợp bị phá cấu trúc của bảng. Trường hợp này hay sảy ra đối với những người mới bắt đầu học HTML.

Ví dụ với đoạn mã trên ta thêm thuộc tính `colspan = "2"` và thẻ `<td>` đầu tiên ta sẽ được:

Hình 3.5. Bảng với thẻ <td> đầu tiên có thuộc tính colspan bằng 2

Các bạn đã thấy được dòng đầu tiên đã bị thưa ra một cột hay nói chính xác hơn do ô đầu tiên có độ lớn bằng 2 ô cộng lại. Như vậy để giữ cấu trúc bảng ta phải xóa đi 1 cặp thẻ `<td></td>` cùng hàng với nó.

Hình 3.6. Bảng với thẻ `<td>` có thuộc tính `colspan` hàng 2 và bỏ đi một ô cùng hàng

Mã HTML để có được giao diện bảng như trong hình 3.6:

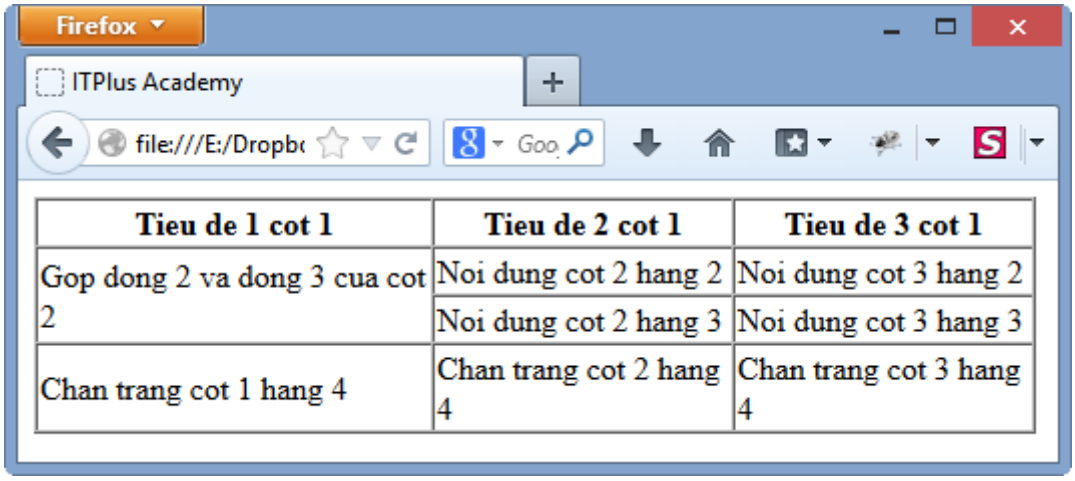
```
<table border = "1px" width="500" cellspacing="0">
        <thead>
                <tr>
                    <th>Tieu de 1 cot 1</th>
                    <th>Tieu de 2 cot 1</th>
                    <th>Tieu de 3 cot 1</th>
                </tr>
        </thead>
        <tbody>
                <tr>
                    <td colspan = "2">Noi dung cot 1 gop cot 2 hang 2</td>
                    <td>Noi dung cot 3 hang 2</td>
                </tr>
        </tbody>
        <tfoot>
                <tr>
                    <td>Chan trang cot 1 hang 3</td>
                    <td>Chan trang cot 2 hang 3</td>
                    <td>Chan trang cot 3 hang 3</td>
                </tr>
        </tfoot>
</table>
```

c. Gom nhóm các ô theo dòng

Tương tự như gom nhóm các ô theo cột để gom nhóm các ô theo dòng như định dạng sau:

Tieu de 1 cot 1	Tieu de 2 cot 2	Tieu de 3 cot 3
Gop dong 2 va dong 3 cua cot 2	Noi dung cot 2 hang 2	Noi dung cot 3 hang 2
	Noi dung cot 2 hang 3	Noi dung cot 3 hang 3
Chan trang cot 1 hang 4	Chan trang cot 2 hang 4	Chan trang cot 3 hang 4

Ở đây ta cần sử dụng thuộc tính `rowspan = x` của thẻ `<td>` hoặc `<th>`. Cũng như ở trên ta cần lưu ý về cấu trúc của bảng khi gom nhóm các ô. Như định dạng ở trên ta chỉ sử dụng 1 cặp thẻ `<td></td>` cho hàng thứ 2 và 2 cho hàng thứ 3.



Hình 3.7. Gom cột theo dòng với thuộc tính rowspan

Chú ý**:** Thuộc tính rowspan chỉ có tác dùng với các dòng trong cùng phần của bảng như: `<thead>` hoặc `<tbody>` hoặc `<tfoot>`

1.2. Làm việc với list

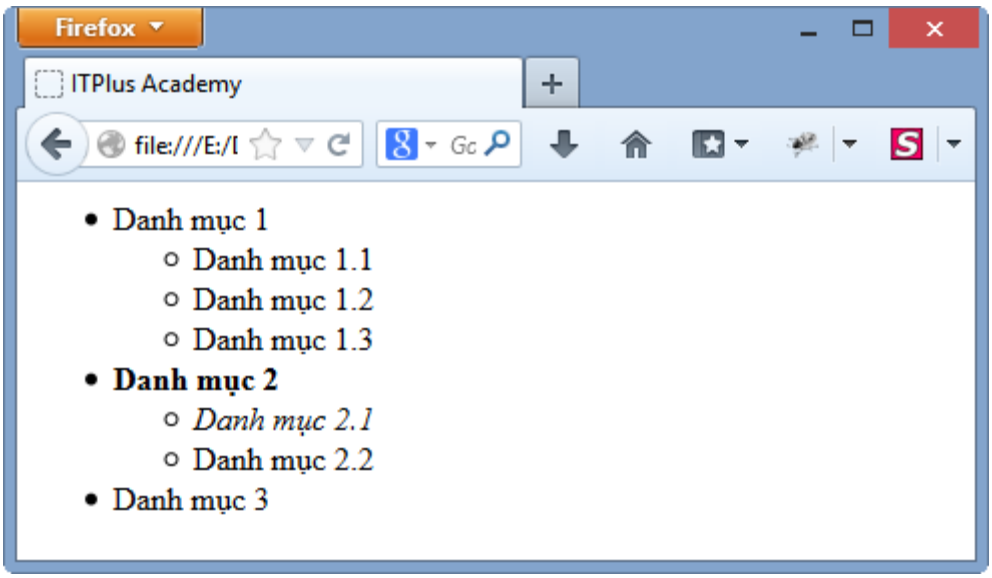
Để tạo ra một danh sách liệt kê kiểu bullet hoặc đánh số trong HTML chúng ta có thể sử dụng thẻ `<ul>` cho danh sách kiểu bullet *(Unordered Lists)* và `<ol>` cho danh sách kiểu đánh số *(Ordered Lists)*. Kèm theo đó là một dạng danh sách đặc biệt nữa là bao gồm các mục mỗi mục sẽ có phần mô tả đi kèm đó là *Definition Lists* được sử dụng cặp thẻ `<dl></dl>`

1.2.1. Thẻ UL

Danh sách *Unordered List* được bắt đầu với thẻ `<ul>` và kết thúc bằng thẻ `</ul>` và mỗi mục của danh sách được đặt trong cặp thẻ `<li></li>`. Đối với các danh sách đa cấp chúng ta có thể lồng một danh sách khác trong cặp thẻ `<li></li>` nghĩa là đặt cặp thể `<ul></ul>` vào trong cặp thẻ `<li></li>`.Ngoài ra trong cặp thẻ `<li></li>` chúng ta vẫn có thể kết hợp với các thẻ khác. Ví dụ với đoạn mã sau:

```
<ul>
    <li> Danh mục 1
        <ul>
              <li>Danh mục 1.1</li>
              <li>Danh mục 1.2</li>
              <li>Danh mục 1.3</li>
        </ul>
    </li>
    <li><b>Danh mục 2</b>
        <ul>
              <li><i>Danh mục 2.1</i></li>
              <li>Danh mục 2.2</li>
        </ul>
    </li>
    <li>Danh mục 3</li>
</ul>
```

Ta sẽ được kết quả như sau:



Hình 3.8. Kết hợp thẻ ul, li với các thẻ khác

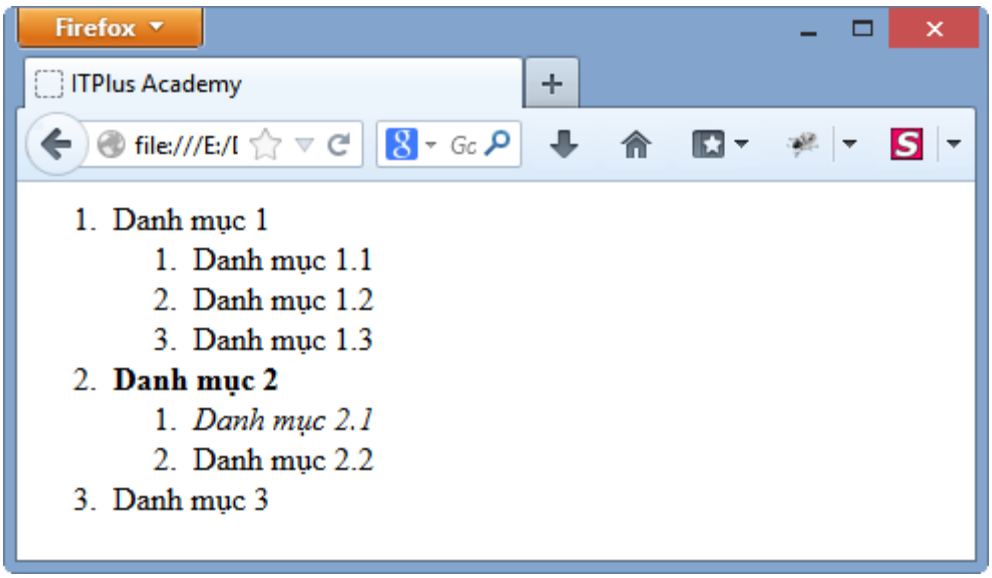
1.2.2. Thẻ OL

Tương tự như *Unorder List* về cấu trúc chỉ khác là *Order List* bắt đầu với thẻ `<ol>` và kết thúc bằng thẻ `</ol>`. Ngoài ra order list không tự động xếp cấp độ như *unorder list (thể hiện ở dấu bullet)* mà các mục trong một danh sách luôn được đánh số từ 1 hoặc có thể hiểu *order list* chỉ có tác dụng với một cấp.

Ví dụ với đoạn mã sau:

```
<ol>
    <li> Danh mục 1
        <ol>
              <li>Danh mục 1.1</li>
              <li>Danh mục 1.2</li>
              <li>Danh mục 1.3</li>
        </ol>
    </li>
    <li><b>Danh mục 2</b>
        <ol>
              <li><i>Danh mục 2.1</i></li>
              <li>Danh mục 2.2</li>
        </ol>
    </li>
    <li>Danh mục 3</li>
</ol>
```

Ta sẽ được kết quả như sau:



Hình 3.9. Sử dụng thẻ ol

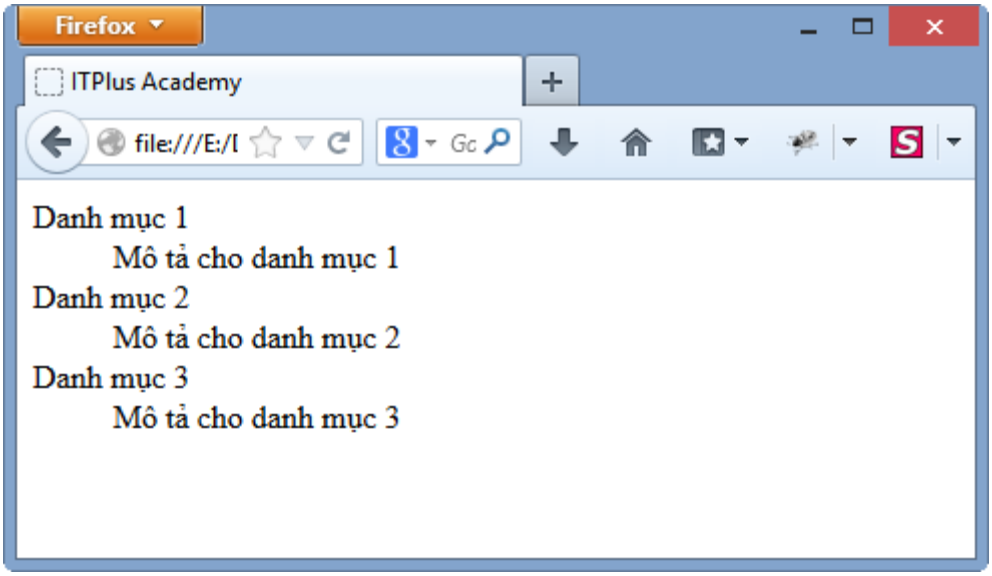
1.2.3. Thẻ DL

Definition Lists có cấu trúc phức tạp hơn một chút so với *Unordered Lists* và *Ordered Lists* và cũng ít được sử dụng hơn. Cấu trúc của nó có dạng như sau:

Trong đó cặp thẻ `<dt></dt>` là để dánh dấu tiêu đề cho một mục và `<dd></dd>` luôn phải đi kèm ngay sau đó nếu muốn có phần mô tả.

```
<dl>
        <dt>Danh mục 1</dt>
        <dd>Mô tả cho danh mục 1</dd>
        <dt>Danh mục 2</dt>
        <dd>Mô tả cho danh mục 2</dd>
        <dt>Danh mục 3</dt>
        <dd>Mô tả cho danh mục 3</dd>
</dl>
```

Ta sẽ được danh sách có dạng như sau:



Hình 3.10. Sử dụng thẻ DL

B. BÀI TẬP

2.1. Câu hỏi ôn tập

Câu 1: Nhóm thẻ nào trong các thẻ sau đều được sử dụng để tạo ra bảng (table)?

E. <table><head><tfoot>
F. <table><tr><tt>
G. <thead><body><tr>
H. <table><tr><td>

Câu 2. Sử dụng thẻ nào dưới đây để tạo ra danh sách dạng số thứ tự ?

A. <ul>
B. <list>
C. <ol>
D. <dl>

Câu 3. Sử dụng thẻ nào dưới đây để tạo ra danh sách không có số thứ tự?

A. <dl>
<list>
B. <ol>
<ul>

2.2. Bài tập

Bài 1. Tạo các bảng sau đây bằng mã HTML sử dụng `table`:

STT	Mã nhân viên	Họ và tên	Thông tin	
			Nơi sinh	Ngày sinh
1	A001	Nguyễn Văn A	Phú Thọ	10/11/1987
2	B002	Nguyễn Văn B	Quảng Ngãi	15/8/1989
3	C003	Nguyễn Văn C	Cà Mau	2/1/1990

Bảng 1

Tên miền	Phí cài đặt	Chi phí
.com	miễn phí	200.000đ/năm
.net	miễn phí	200.000đ/năm
.org	miễn phí	200.000đ/năm
.info	miễn phí	200.000đ/năm
.co.uk	miễn phí	miễn phí

Bảng 2

Header (#9C9)

Left (#FF0)	**Right** (#99C)
Footer (#999)	

Bảng 3

Bài 2. Viết đoạn mã HTML cho phép trang web hiển thị danh sách như sau:

1. Monday
   i. Introduction to HTML
   ii. Creating Lists
2. Tusday
   A. Creating Table
   B. Inserting Images
3. Wednesday
   I. Creating Forms
   II. Working with Link
4. Thursday
5. Friday

BÀI 4. LÀM VIỆC VỚI FORM

Bài này chúng ta sẽ tiếp tục tìm hiểu thêm một yếu tố quan trọng nữa trong HMTL đó là form. Thể hiện quen thuộc nhất của form chính là lúc các bạn muốn đăng nhập vào một trang quen thuộc nào đó, bao giờ cũng là 2 hộp văn bản *(textbox)* một cho tài khoản đăng nhập, một cho mật khẩu của bạn, một nút *(button)* đăng nhập và có thể có thêm một hộp kiểm *(checkbox)* để bạn đánh dấu lưu trạng thái đăng nhập. Đó chính là một form.

Cách thức hoạt động của form là bạn điền dữ liệu vào các thành phần của form sau đó nhấn nút gửi hoặc có thể nhấn Enter để gửi *(submit dữ liệu lên server).* Dữ liệu sẽ được sử lý tại đó với một ngôn lập trình nào đó như PHP chẳng hạn. Như vậy ở đây, với HTML chúng ta sẽ tìm hiểu các đánh dấu một form như thế.

Một form bao gồm các yếu tố sau: cách thức gửi dữ liệu, đường dẫn sẽ nhận dữ liệu, các thành phần trong form mỗi thành phần bao gồm 3 thuộc tính quan trọng để máy chủ có thể phân biệt và xử lý dữ liệu, đó là: tên, loại và giá trị của nó. Chúng ta sẽ đi tìm hiểu chi tiết từng thành phần.

A. NỘI DUNG

1.1. Thẻ form

1.1.1. Cấu trúc Form

Một form trong HMTL được đánh dấu bởi cặp thẻ `<form></form>` và có cấu trúc như sau:

```
<form method="value" action="path" enctype= "value">
   <input type="…"/>
   …
</form>
```

Trong đó:

- *Method*: thuộc tính bắt buộc. Nó quy định các thức gửi dữ liệu lên server. Thuộc tính này có 2 giá trị get và post.

- *Action*: thuộc tính bắt buộc là đường dẫn mà dữ liệu sẽ được submit tới. Nếu để giá trị trống dữ liệu sẽ được mặc định gửi tới chính URL hiện tại.

- *Enctype*: cách thức mã hóa dữ liệu trước khi gửi lên server.. Thuộc tính này không bắt buộc. Nhưng bạn cần sử dụng nó nếu bạn muốn upload một file lên server. Các giá trị:

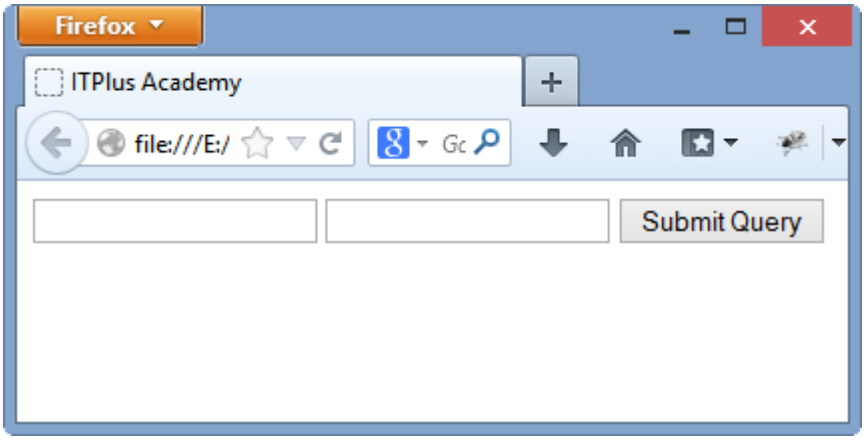
- *application/x-www-form-urlencoded*: Giá trị mặc định.
- *multipart/form-data*: Giá trị này được sử dụng khi bạn muốn upload file.
- *text/plain*: Các dấu khoảng trắng sẽ được chuyển thành dấu +.

1.1.2. Sự khác nhau giữa method post và get

Với post, dữ liệu sẽ được gửi ngầm định, với get dữ liệu sẽ được hiển thị tại url dưới dạng các tham số đường link. Ví dụ tôi tạm thời sử dụng một form đăng nhập đơn giản sau:

```
<form method="get" action="xuly.php">
        <input type="text" name="username" />
        <input type="password" name="password" />
        <input type="submit" name="btnSubmit"/>
</form>
```

form này sẽ được hiển theo hình sau:



Hình 4.1. Một form đơn giản trên trình duyệt

Với post sẽ không có gì nhận ra được ngoài việc sau khi bạn submit trình duyệt sẽ chuyển tới đường dẫn được khai báo trong với thuộc tính action. Còn với get bạn sẽ nhận ra được các giá trị của mình ngay trên đường dẫn đó. Ví dụ bạn nhập username là *"itplus"*, password là *"123456"* và nhất nút "Submit Query". Trình duyệt sẽ chuyển tới đường dẫn:

`xuly.php?username=itplus&password=12356&btnSubmit=Submit+Query`

Như vậy mọi giá trị của các thành phần trong form đều được hiển thị trên URL và chúng ta nhận ra rằng các tham số đường link nối với phần chính của URL bằng dấu "?" phân cách với nhau bằng dấu "&" và có dạng "tên = giá trị".

1.2. Các thẻ input

Thẻ input dùng để tạo ra cách thành phần trên form bao gồm: text, password, checkbox, radio, button, submit, reset. Loại thẻ input được quy định với thuộc tính type với các giá trị tương ứng. Các thẻ input tuy có các tác dụng khác nhau nhưng đều có chung một cấu trúc:

```
<input type= "kiểu" name= "tên" value= "giá trị"/>
```

Trong đó thuộc tính name là bắt buộc, cần phải đặt tên khác nhau để server xử lý chính xác. Giá trị của thuộc tính type là text. Thuộc tính value mặc định là rỗng hoặc *"Submit Query"* đối với thẻ input submit.

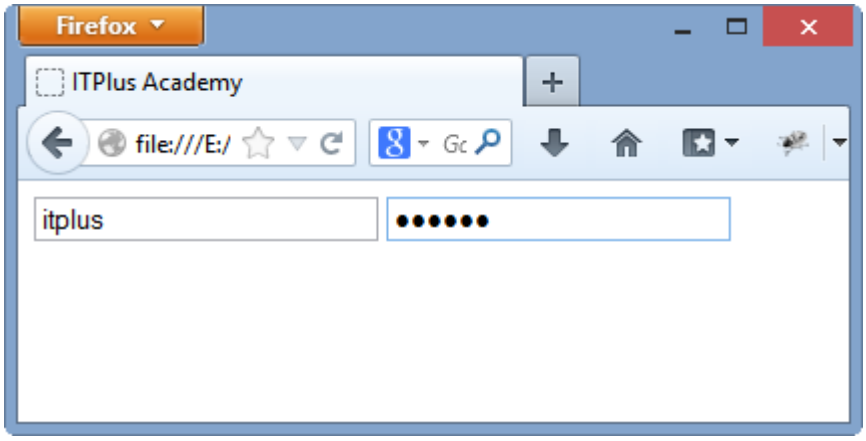
1.2.1. Thẻ input text và password

Về cơ bản thẻ input text và password có định dạng giống nhau nhưng thẻ input password sẽ thay thế các ký tự nhập vào bằng các dấu * hoặc ●. Ngoài thuộc tính name và value, 2 thẻ này còn có thêm thuộc tính `size= "giá trị"` để đánh dấu độ rộng của text box.

Ví dụ:

```
<input type= "text" name= "username" value= "itplus" size="25"/>
<input type= "pass" name= "password" value= "123456" size="25"/>
```

Ta sẽ có được form như sau:



Hình 4.2. Thẻ input text và password

1.2.2. Thẻ input checkbox

Thẻ input checkbox dùng để tạo ra các hộp kiểm cho bạn đánh dấu một hoặc nhiều lựa chọn không bắt buộc nào đó. Ví dụ như bạn có muốn lưu trạng thái đăng nhập không.

Các bạn thường thấy 1 checkbox luôn đi kèm text mô tả cho nó nhưng trên thực tế bạn chỉ nhận được duy nhất một checkbox có dạng ☐ mà thôi. Và bạn có thể sử dụng bất kỳ thẻ định dạng văn bản nào để tạo phần mô tả cho checkbox đó.

Tôi xin được giới thiệu thêm 1 thẻ nữa thường được dùng trong form cho mục đích này. Nó không chỉ áp dụng cho thẻ check box mà có thể áp dụng các thẻ input khác. Đó là thẻ `<label>`:

```
<label for= "tên của thẻ input"> Mô tả</label>
```

Tuy thẻ label sinh ra là để mô tả cho thẻ input nhưng thực tế nó chẳng có tác dụng gì ngoài hiển thị một đoạn văn bản. Tùy vào cách bạn sắp xếp và định dạng mà tạo ra được một *form* cân đối.

Trong một số trường hợp bạn cần dùng nhiều checkbox cho cùng một mục đích ví dụ bạn muốn xóa ngẫu nhiên một số bài viết trong danh mục quản lý tin tức. Bạn sẽ đánh dấu các *checkbox* và *submit*. Khi ấy bạn không thể đặt cho mỗi *checkbox* một tên được vì các *checkbox* được chọn ngẫu nhiên mà tên thì không linh động. Và hướng giải quyết HTML cung cấp đó là tạo một nhóm các *checkbox* bằng cách đặt tên chúng như nhau. Điều mà thẻ *input text* và *password* không nên áp dụng. Với cách này dữ liệu sẽ được gộp lại dưới một cái tên và tùy vào ngôn ngữ lập trình mà cách sử lý sẽ khác nhau. Như với bài toán xóa bài viết như trên server sẽ lấy ra các mã bài viết cần xóa và thực hiện thao tác.

Ví dụ như ta có các checkbox như sau:

```
<input type= “checkbox” name= “article_id” value= “1”/>
<input type= “checkbox” name= “article_id” value= “2”/>
<input type= “checkbox” name= “article_id” value= “3”/>
```

Hoặc

```
<input type= “checkbox” name= “article_id[]” value= “1”/>
<input type= “checkbox” name= “article_id[]” value= “2”/>
<input type= “checkbox” name= “article_id[]” value= “3”/>
```

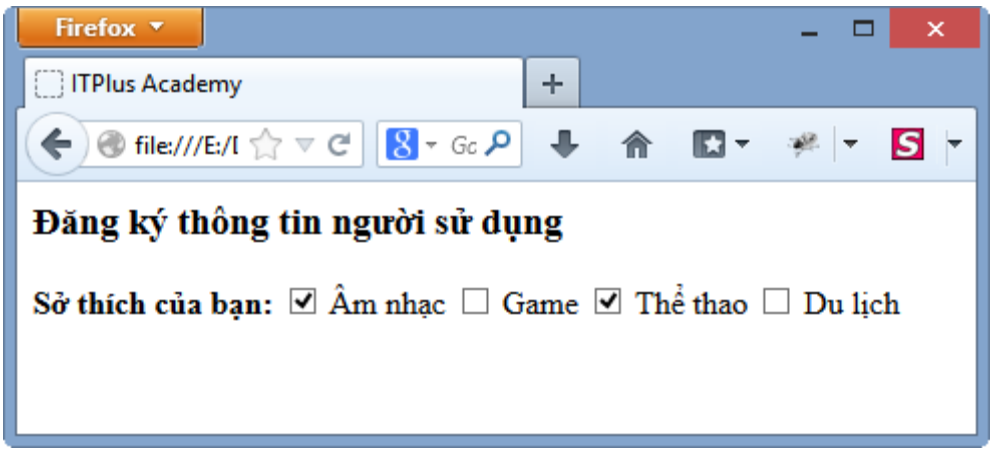
Về sự khác nhau của 2 cách đặt tên này sẽ được nói trong phần lập trình PHP của giáo trình.

Ngoài ra một thuộc tính quan trọng nữa của thẻ input checkbox là *checked* và giá trị duy nhất của nó là *checked*. Nếu ta đặt thêm *checked= “checked”* cho thẻ input checkbox thì checkbox đó sẽ được đánh dấu.

Ví dụ:

```
<h3>Đăng ký thông tin người sử dụng</h3>
<form method="get" action="">
        <b>Sở thích của bạn:</b>
        <input type="checkbox" name="hobbies" value="0" checked="checked"/>
        <label for="gender">Âm nhạc</label>
        <input type="checkbox" name="hobbies" value="1"/>
        <label for="gender">Game</label>
        <input type="checkbox" name="hobbies" value="2" checked="checked"/>
        <label for="gender">Thể thao</label>
        <input type="checkbox" name="hobbies" value="3"/>
        <label for="gender">Du lịch</label>
        <br/>
</form>
```

Form chúng ta nhận được trên trình duyệt là:



Hình 4.3. Thẻ input checkbox

1.2.3. Thẻ input radio

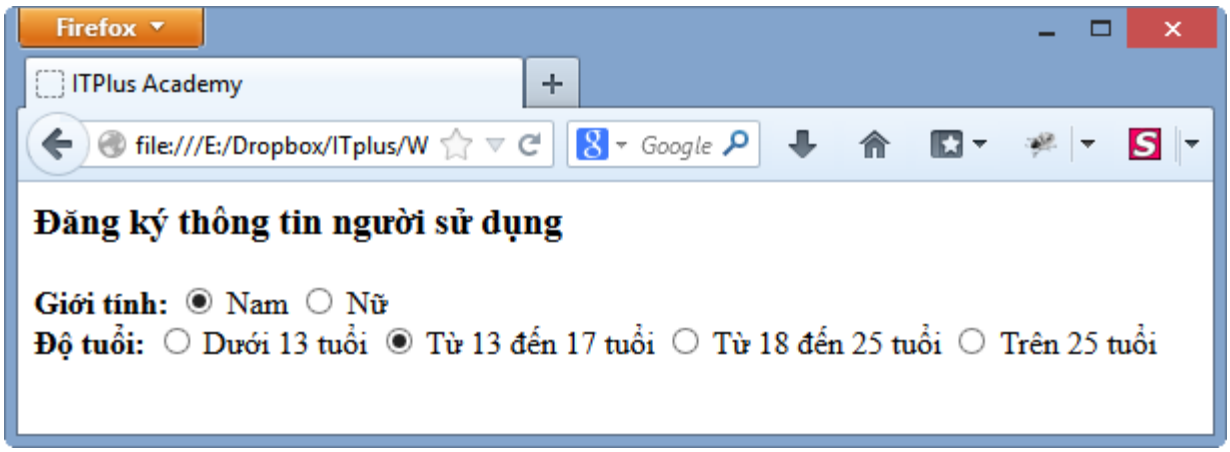
Giống như thẻ *input checkbox*, thẻ *input radio* cũng dùng để tạo ra các lựa chọn nhưng khác ở chỗ trong các lựa chọn đó bạn bắt buộc phải chọn 1 và input radio luôn có từ 2 cái trở lên. Các *radio* trong cùng một nhóm cũng phải được đặt tên giống nhau mới có tác dụng.

Thẻ *input radio* cũng có thuộc tính *checked* giống *input checkbox* nhưng nó chỉ đánh dấu radio cuối cùng có thuộc tính này của nhóm radio.

Ví dụ ta có đoạn mã sau:

```
<h3>Đăng ký thông tin người sử dụng</h3>
<form method="get" action="">
        <b>Giới tính:</b>
        <input type="radio" name="gender" value="0" checked= “checked”/>
        <label for="gender">Nam</input>
        <input type="radio" name="gender" value="1"/>
        <label for="gender">Nữ</input><br/>
        <b>Độ tuổi:</b>
        <input type="radio" name="age" value="0" checked= “checked”/>
        <label for="gender">Dưới 13 tuổi</input>
        <input type="radio" name="age" value="1" checked= “checked”/>
        <label for="gender">Từ 13 đến 17 tuổi</input>
        <input type="radio" name="age" value="2"/>
        <label for="gender">Từ 18 đến 25 tuổi</input>
        <input type="radio" name="age" value="3"/>
        <label for="gender">Trên 25 tuổi</input>
</form>
```

Form hiển thị trên trình duyệt là:



Hình 4.4. Thẻ input radio

1.2.4. Thẻ input file

Như đã nhắc tới ở bên trên với thuộc tính *enctype= "multipart-formdata"* của thẻ *<form>* chúng ta có thể upload file lên server. Để có thể chọn được một file trên máy tính chúng ta cần dùng đến thẻ input file

Thẻ input file có riêng cho mình một thuộc tính đó là *accept = "kiểu file"*. Thuộc tính này dùng để quy định loại file mà bạn được upload. Có 3 kiểu file cơ bản là:

- *audio/** : Các file âm thanh.

- *video/** : Các file video

- *image/**: Các file hình ảnh

<u>Lưu ý:</u>

- Dấu "*" có nghĩ là chấp nhất tất các đuôi mở rộng, bạn có thể thay thế nó bằng đuôi mở rộng nhất định như *audio/mp3, video/avi, image/ipg.*

- Thuộc tính này không áp dụng cho trình duyệt Internet Explorer và Safari

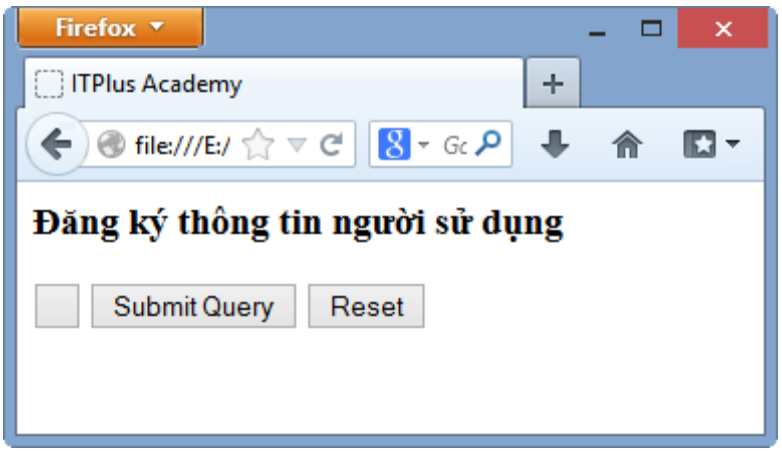
- Bạn có thể không sử dụng thuộc tính này để upload các kiểu file khác như rar, zip.

1.2.5. Input button, submit và reset

Đây là các thẻ tạo ra các nút nhấn trên *form*. Cấu trúc của chúng cũng như các thẻ *input* khác. Tuy nhiên chúng ta cần lưu ý một chút về thuộc tính value của chúng. Giá trị của thuộc tính value chính là text hiện thị trên nút. Nếu bạn để không sử dụng thuộc tính value thì mặc định đối với thẻ input submit là *"Submit Query"* của thẻ *input reset* là *"Reset"* còn với input button thì không có gì.

```
<h3>Đăng ký thông tin người sử dụng</h3>
<form method="get" action="">
        <input type="button" name="btnButton"/>
        <input type="submit" name="btnSubmit"/>
        <input type="reset" name="btnReset"/>
</form>
```

Form được hiển thị là:



Hình 4.5. Thẻ input button, submit và reset

Tác dụng của 3 thẻ input này là:

- *Button*: chỉ tác dụng khi đi kèm với các thuộc tính sự kiện và *Javascript*. Như khi được click thì xử lý một chức năng nào đó.

- *Reset*: một dạng *button* đặc biệt, khi sử dụng sẽ khởi tạo lại toàn bộ dữ liệu ban đầu của các thành phần trong bảng

- *Submit*: một dạng button đặc biệt, gửi dữ liệu lên server và điều đặc biệt là nếu trong form có 1 *input submit* thì chúng ta có thể ấn enter tại các *input textbox* hay *password* để gửi dữ liệu đi.

1.3. Danh sách với thẻ select

Một thành phần khác của form ngoài các thẻ input là thẻ select dùng để tạo ra một hộp chọn các giá trị theo danh sách được thả xuống hay người ta hay gọi là *Drop Down List*. Cũng có thể nói có tác dụng như là một nhóm các thẻ *input radio* vậy.

Cấu trúc:

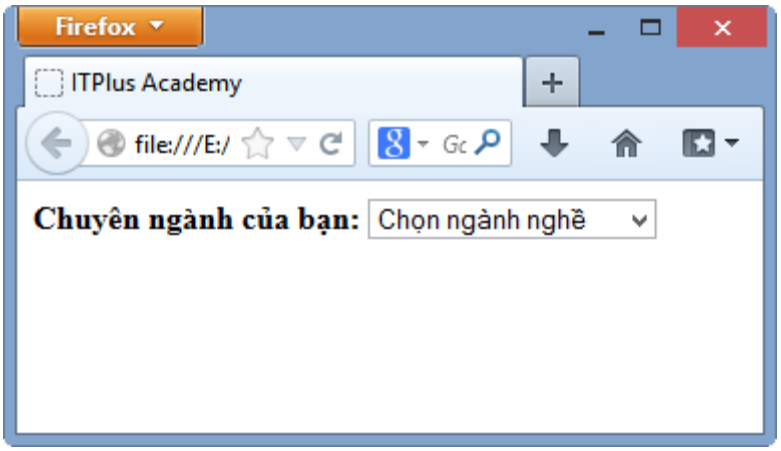
```
<select name="select1">
        <option value="1">Lựa chọn 1</option>
        <option value ="2" selected="selected">Lựa chọn 2</option>
...
</select>
```

Trong cặp thẻ `<select></select>` ta có thể có nhiều cặp thẻ option tương ứng với 1 mục trong danh sách lựa chọn. Với input checkbox thì chúng ta có thuộc tính `checked = "checked"` thì với option của select chúng ta có `selected = "selected"`. Và cũng chỉ có option có thuộc tính đó cuối cùng mới được chọn hiển thị trên hộp chọn, đó cũng là giá trị sẽ được submit.

Ví dụ:

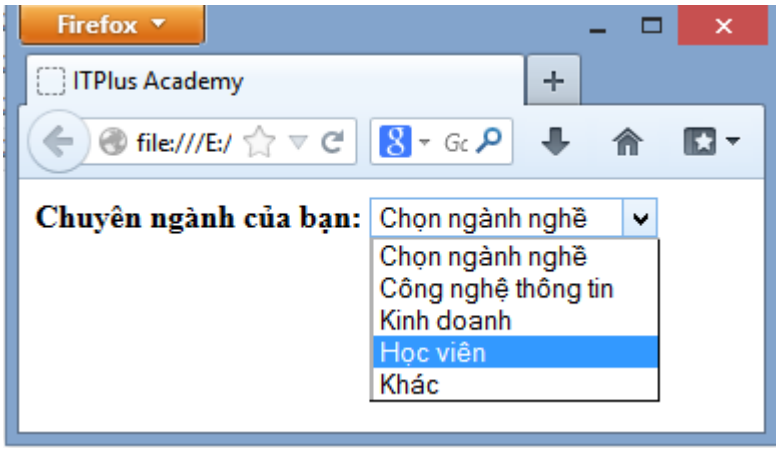
```
<form method="get" action="">
        <b>Chuyên ngành của bạn:</b>
        <select name="job">
                <option value="0">Chọn ngành nghề </option>
                <option value="1">Công nghệ thông tin </option>
                <option value="2">Kinh doanh</option>
                <option value="3">Học viên</option>
                <option value="4">Khác</option>
        </select>
</form>
```

Ta sẽ có form như sau:



Hình 4.6. Danh sách thả xuống với thẻ select

Khi click vào hộp chọn một danh sách sẽ được thả xuống:



Hình 4.7. Danh sách hiện ra khi click vào select box

1.4. Vùng nhập văn bản với textarea

Để nhập được những thông tin văn bản lớn và cần trực quan hơn một input text là không đủ, dù ta có thể có cách tăng chiều cao của nó lên nhưng vẫn chỉ có 1 dòng văn bản mà thôi. Do đó HTML có thêm một loại thẻ nữa đó là textarea. Nó tạo và một vùng nhập văn bản hay một textbox có nhiều dòng. Nó hơi khác với các thành phần form khác là nội dung đặt trong cặp thẻ <textarea></textarea> chính là giá trị được submit. Nó cũng

có thuộc tính name và 2 thuộc tính không bắt buộc khác là cols quy định số cột hay số chữ cái trên 1 dòng và rows quy định số dòng.

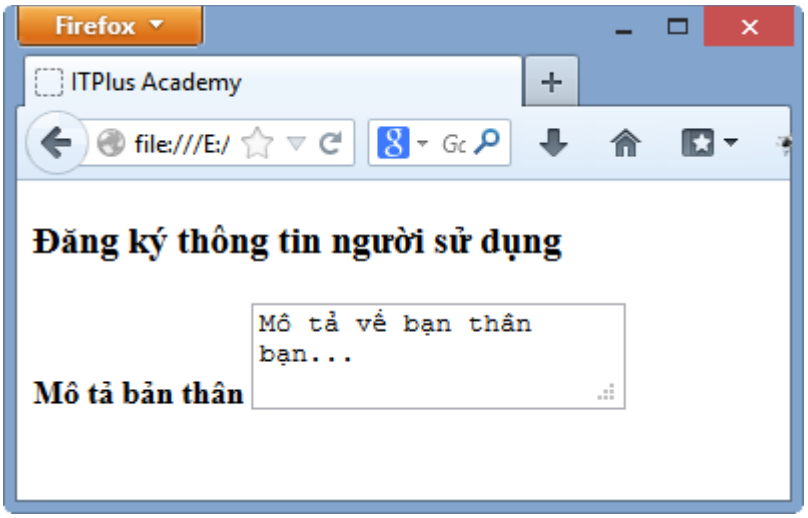
Cấu trúc của nó là:

```
<textarea name="name">Text<textarea>
```

Ví dụ:

```
<form method="get" action="">
        <b>Mô tả bản thân</b>
        <textarea name="description">Mô tả về bạn thân bạn...</textarea>
</form>
```

Hiển thị trên trình duyệt là:



Hình 4.8. Sử dụng Textarea

1.5. Các thuộc tính đặc biệt trong form

Trong một số trường hợp nào đó bạn chỉ muốn hiển thị một textbox có nội dung trong đó nhưng không cho phép người dùng chỉnh sửa thông tin trong đó hay một nhóm các checkbox không muốn submit dữ liệu lên server. Chúng ta sẽ sử dụng tới các thuộc tính đặc biệt của các thẻ thành phần trong form là: readonly, disabled.

STT	Tên thuộc tính	Giá trị	Mô tả	Áp dụng cho
1	readonly	readonly	Không cho phép chỉnh sửa thông tin	text, password, textarea
2	disabled	disabled	Vô hiệu hóa thuộc tính không cho phép chỉnh sửa cũng như submit lên server	Tất cả các thành phần của form

Bảng 4. 1. Các thuộc tính đặc biệt trong form

B. BÀI TẬP

2.1. Câu hỏi ôn tập

Câu 1. Đoạn mã nào dưới đây hiển thị một hộp textbox để nhập liệu?

H. <input textarea="text" type="username">

I. <input name="username" value="text">

J. <input="text" user="username">

K. <input type="text" name="username">

Câu 2. Thẻ nào cho phép tạo ra một nhóm các tùy chọn trong 1 danh sách?

H. <group>

I. <optgroup>

J. <option>

K. <select>

Câu 3. Đoạn mã nào dưới đây sẽ tạo ra một nút submit?

D. <input name="button" type="form">

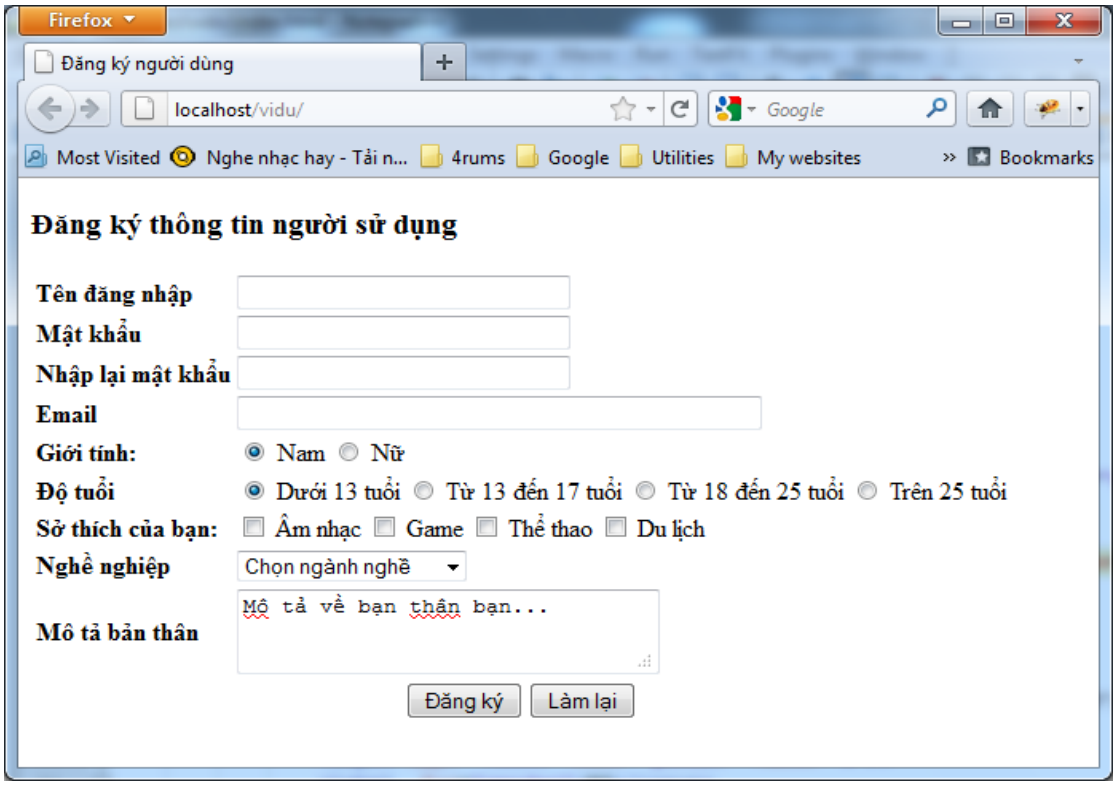
E. <input type="submit" value="Submit">

F. <input type="text" name="Submit">

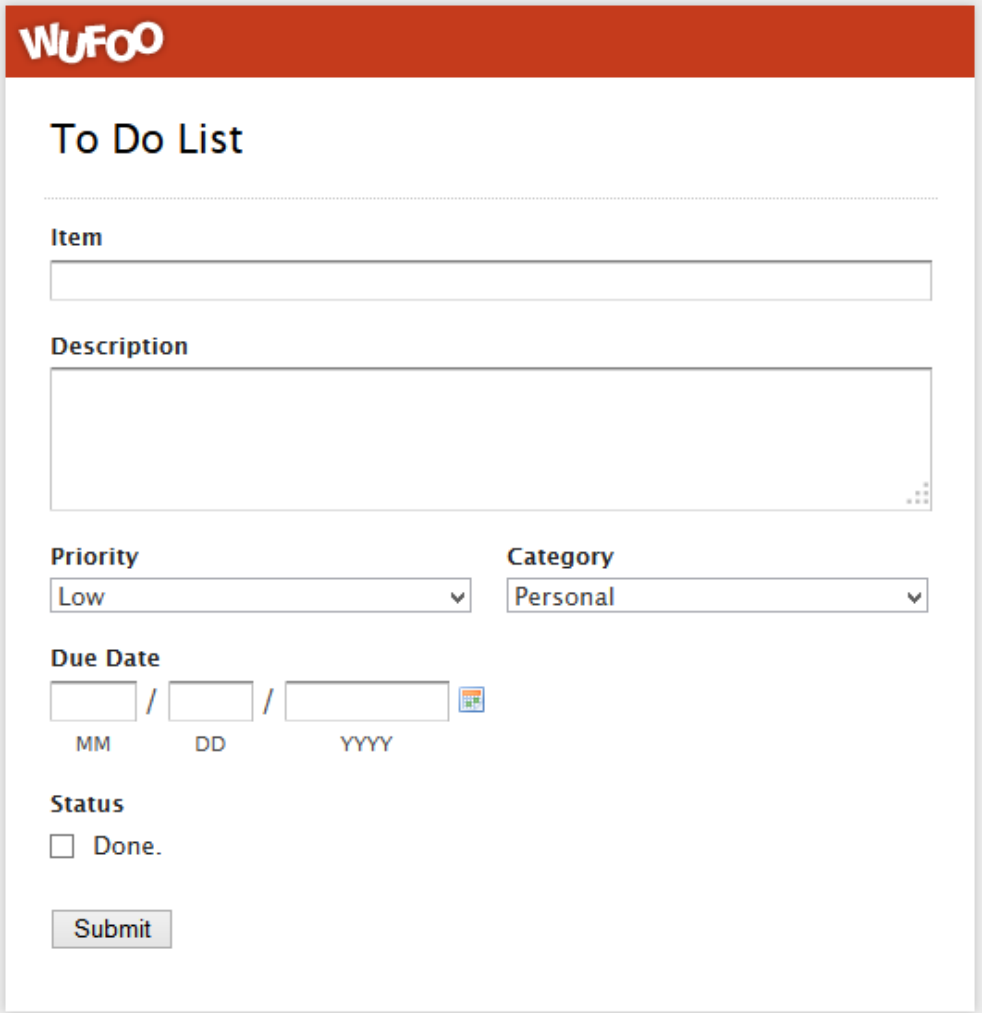
G. <submit type="special">

2.2. Bài tập

Bài 1. Sử dụng các kiến thức đã được học về HTML Form, xây dựng chương trình sau:



Bài 2. Sử dụng các kiến thức đã được học về HTML Form, xây dựng chương trình sau:



BÀI 5. HTML5

Có lẽ chúng ta đã nghe khá nhiều về thuật nghữ HTML5 mà chưa biết đó là gì và có ích lợi như thế nào cho công nghệ web. Trong phạm vi bài học này, giáo trình sẽ giới thiệu về lịch sử hình thành HTML5, nêu lên những ưu điểm của HTML5 và những ứng dụng cơ bản của HTML5.

A. NỘI DUNG

1.1. Tổng quan về HTML5

1.1.1. Lịch sử phát triển

HTML5 là một chuẩn mới của HTML. Phiên bản trước của HTML là HTML 4.01, được đưa ra từ năm 1999. Từ đó tới giờ, chuẩn web không có gì thay đổi. HTML5 vẫn còn đang trong quá trình xây dựng. Tuy vậy, các trình duyệt lớn hiện nay đã hỗ trợ rất nhiều thẻ HTML5 và các APIs *(Application Programming Interface - giao diện lập trình ứng dụng)*.

HTML5 ra đời bởi sự hợp tác giữa World Wide Web Consortium *(W3C)* và Web Hypertext Application Technology Working Group *(WHATWG)*.

WHATWG nghiên cứu trên nền web và ứng dụng, còn W3C nghiên cứu XHTML 2.0. Vào năm 2006, họ quyết định hợp tác và cho ra đời phiên bản mới của HTML.

1.1.2. Ưu điểm của HTML5

- Những tính năng mới, dựa trên nền tảng HTML, CSS, DOM, and JavaScript
- Giảm thiểu việc xử dụng plugin từ bên ngoài (ví dụ Flash, Slivelight…)
- Quản lý lỗi tốt hơn
- Thêm nhiều thẻ, giảm bớt script
- HTML5 độc lập thiết bị

1.1.3. HTML5 DOCTYPE

Khác với dòng `doctype` dài lê thê, với `schema xml` như trước đây, giờ `doctype` html5 vô cùng đơn giản: `<!DOCTYPE html>`

Một HTML5 document đơn giản, bắt buộc có những thẻ như dưới đây:

```
<!DOCTYPE html>
<html>
<head>
 <title>[I]Title of the document[/I]</title>
</head>
<body>
 [I]The content of the document......[/I]
</body>
</html>
```

1.1.4. Những tính năng mới của HTML5

- Dưới đây là một vài tính năng mới thú vị của HTML5:

- Thẻ `<canvas>` cho đồ họa 2D

- Thẻ `<video>` và `<audio>` dành cho media

- Hỗ trợ lưu trữ (dữ liệu, file, sql...) local

- Nhiều thẻ mới gần gũi nội dung hơn: `<article>`, `<footer>`, `<header>`, `<nav>`, `<section>`

- Nhiều điều khiển form mới, như calendar, date, time, email, url, search

- Kết hợp hoàn hảo với CSS3, tạo các hiệu ứng bo góc, 2D, 3D, đổ bóng, xoay hình, các animation và hỗ trợ fonts...

1.1.5. Các trình duyệt hỗ trợ HTML5

Hình 5.1. HTML5 và các trình duyệt hỗ trợ

HTML5 vẫn chưa phải chuẩn chính thức, nên chưa có trình duyệt nào hỗ trợ HTML5 đầy đủ. Nhưng các trình duyệt lớn hiện nay (Safari, Chrome, Firefox, Opera, Internet Explorer) vẫn đang tiếp tục bổ sung những tính năng mới của HTML5 vào các phiên bản mới của họ.

1.2. Các thẻ mới trong HTML5

Bản thân internet, và mục đích sử dụng internet, đã thay đổi rất nhiều kể từ khi HTML 4.01 trở thành chuẩn năm 1999.

Ngày nay, một vài thẻ trong HTML 4.01 đã lỗi thời, không còn được dùng nữa, hoặc không dùng đúng với mục đích khi nó được tạo ra. Những thẻ này sẽ được gỡ bỏ và viết mới lại trong HTML5.

Để thích ứng với việc sử dụng internet hiện nay, HTML5 cũng thêm vào nhiều thẻ mới có cấu trúc tốt hơn, quản lý tốt hơn, hỗ trợ đồ họa và media.

1.2.1. Những cấu trúc thẻ mới

STT	Thẻ	Mô tả
1	<article>	Bài viết
2	<aside>	Mô tả đoạn nội dung liên quan tới bài viết, nhưng có tính độc lập so với bài viết.
3	<bdi>	Tách một phần văn bản được định dạng khác hướng so với văn bản bên ngoài nó (tác dụng đối với các văn bản ngôn ngữ viết ngược)
4	<command>	Nút lệnh
5	<details>	Chi tiết thêm, người dùng có thể xem hoặc ẩn đi
6	<figcaption>	Chú thích cho thẻ <figure>
7	<figure>	Chỉ rõ nội dung bên trong (ví dụ: ảnh, code, biểu đồ...)
8	<footer>	Chân của một section (phần trên trang)
9	<header>	Phần đầu của một document hoặc section
10	<hgroup>	Nhóm các thẻ tiêu đề <h1> tới <h6> khi tiêu đề có nhiều cấp
11	<mark>	Đánh dấu chữ trọng tâm (highlight)
12	<meter>	Đơn vị đo lường
13	<nav>	Link điều hướng
14	<progress>	Quá trình thực thi tác vụ
15	<rp>	Hiển thị thay thế lên trình duyệt nếu không hỗ trợ ruby annotations
16	<rt>	Giải thích/cách phát âm ký tự (for East Asian typography)
17	<ruby>	Ruby annotation (for East Asian typography)
18	<section>	Một đoạn/phần của văn bản html
19	<summary>	Tạo tiêu đề cho thẻ details
20	<time>	Ngày/giờ
21	<wbr>	Chỗ có thể ngắt trong từ (nếu từ quá dài, dựa vào thẻ này có thể ngắt nếu hiển thị không hết từ)

1.2.2. Các thẻ media mới

STT	Thẻ	Mô tả
1	<audio>	Nội dung âm thanh
2	<embed>	Cho plugin bên ngoài
3	<source>	Nguồn của <video> và <audio>

4	<track>	Thông tin dạng text cho <video> và <audio> *(có thể là phụ đề)*
5	<video>	Nội dung video hoặc phim

1.2.3. Thẻ <canvas>

STT	Thẻ	Mô tả
1	<canvas>	Dùng để vẽ đồ họa, sử dụng javascript

1.2.4. Các phần tử form mới

STT	Thẻ	Mô tả
1	<datalist>	Danh sách dữ liệu gợi ý cho thẻ input
2	<keygen>	Tạo một cặp khóa cho form
3	<output>	Kết quả của một phép tính

1.2.5. Các thẻ được gỡ bỏ

Trong HTML5, một số thẻ cũ không còn được hỗ trợ như: <acronym>, <applet>, <basefont>, <big>, , <dir>, <font>, <frame>, <frameset>, <noframes>, <strike>, <tt>.

1.3. Thẻ Canvas và hiển thị các đối tượng đồ họa

1.3.1. Khái niệm

Thẻ <canvas> trong HTML5 dùng để vẽ đồ họa, thường sử dụng javascript. Thẻ <canvas> chỉ là đối tượng chứa của đồ họa. Bạn phải sử dụng javascript (hoặc ngôn ngữ script khác) để vẽ đối tượng đồ họa thật sự.

Canvas hỗ trợ vài phương thức để vẽ paths (đường), boxes (hộp), circles (hình tròn), characters (ký tự), và chèn ảnh.

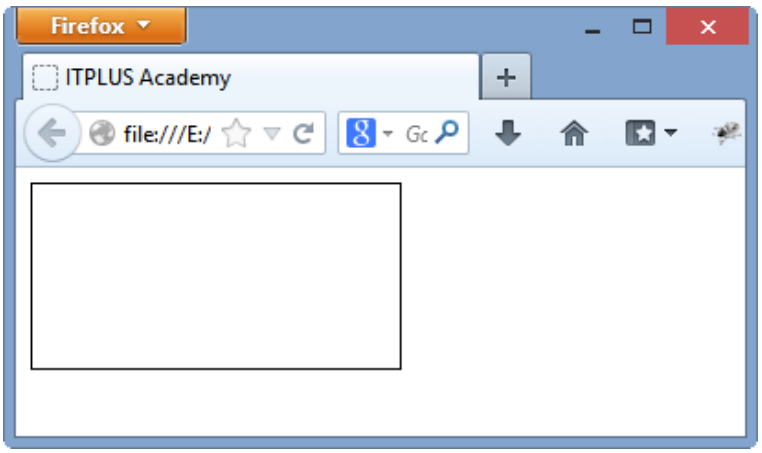
Trình duyệt hỗ trợ: Internet Explorer 9, Firefox, Opera, Chrome, và Safari. Internet Explorer 8 và các trình duyệt phiên bản trước đó không hỗ trợ <canvas>.

1.3.2. Cách tạo canvas

Canvas là một vùng hình chữ nhật trên trang HTML, được định nghĩa bởi thẻ <canvas>. Thẻ <canvas> không có viền *(border)* hoặc nội dung bên trong. Cần luôn đặt thuộc tính ID cho thẻ <canvas> để script có thể tham chiếu đối tượng và thuộc tính cao, rộng để xác định kích thước cho <canvas>. Có thể có nhiều thẻ <canvas> trong một trang HTML. Muốn canvas có đường viền, sử dụng style như sau:

```
<canvas id="myCanvas" width="200" height="100" style="border:1px solid
#000000;"></canvas>
```

Kết quả có được như hình vẽ:



Hình 5.2. Sử dụng canvas với HTML5 và CSS

1.3.3. Sử dụng javascript vẽ nội dung canvas

Khác với nội dung ví dụ nêu trên, trong ví dụ dưới đây, ta sẽ dùng script để vẽ nội dung của canvas.

```
<canvas id="myCanvas"></canvas>
<script>
        var c=document.getElementById("myCanvas");
        var ctx=c.getContext("2d");
        ctx.fillStyle="#FF0000";
        ctx.fillRect(0,0,150,75);
</script>
```

Đầu tiên, script tìm đến thẻ `<canvas>` có id là `myCanvas`:

`var c=document.getElementById("myCanvas");`

Sau đó, gọi phương thức `getContext()` của đối tượng canvas đó (truyền tham số "`2d`" là bắt buộc): `var ctx=c.getContext("2d");`

`getContext("2d")` là một đối tượng HTML5 được xây dựng sẵn, với rất nhiều thuộc tính và phương thức để vẽ paths, boxes, circles, text, images, và hơn thế nữa.

Hai dòng tiếp theo vẽ ra một hình chữ nhật màu đỏ:

`ctx.fillStyle="#FF0000";`

`ctx.fillRect(0,0,150,75);`

Thuộc tính `fillStyle` có thể là mã màu CSS, gradient (đổi màu), hoặc pattern (hoa văn). Mặc định `fillStyle` là mã màu #000000 (đen).

Phương thức `fillRect(x,y,width,height)` vẽ một hình chữ nhật, sử dụng style khai báo ở dòng trên.

Tọa độ của Canvas: Canvas là một grid (lưới) hai chiều. Góc trên bên trái của canvas có tọa độ là `(0,0)`. Hàm `fillRect()` ở trên có tham số là `(0,0,150,75)` có nghĩa là bắt đầu vẽ canvas từ tọa độ `(0,0)` và vẽ hình chữ nhật rộng và cao là `150x75 pixel`.

1.3.4. Canvas paths

Để vẽ đường thẳng trong canvas, ta sử dụng hai hàm sau:

- `moveTo(x,y)` định nghĩa điểm khởi đầu của đường thẳng

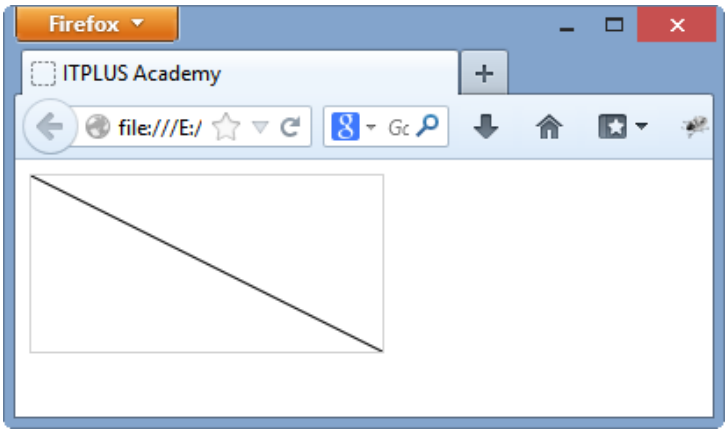
- `lineTo(x,y)` định nghĩa điểm kết thúc của đường thẳng

- Để vẽ được đường thẳng thật sự, ta phải sử dụng một trong các hàm `"ink"`, ví dụ `stroke()`.

Ví dụ để định nghĩa điểm đầu có tọa độ `(0,0)` và điểm cuối có tọa độ `(200,100)`. Sau đó sử dụng hàm `stroke()` để vẽ một đường thẳng ta làm như sau:

```
<canvas id="myCanvas" width="200" height="100" style="border:1px solid
#d3d3d3;"></canvas>
<script>
        var c=document.getElementById("myCanvas");
        var ctx=c.getContext("2d");
        ctx.moveTo(0,0);
        ctx.lineTo(200,100);
        ctx.stroke();
</script>
```

Kết quả ta được hình như sau:

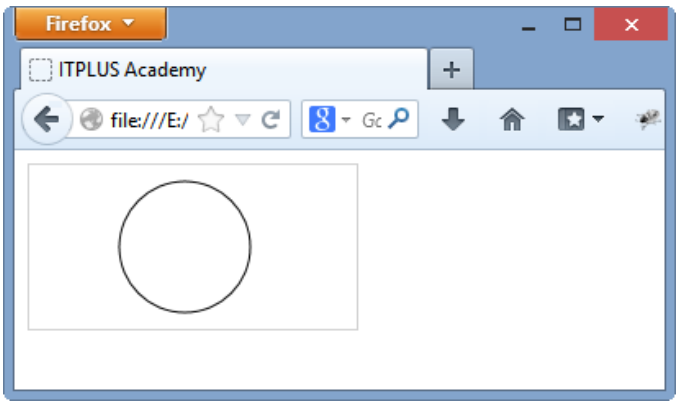


Hình 5. 3. Vẽ đường thẳng với hàm stroke()

Tương tự như ví dụ trên, nếu ta muốn vẽ đường tròn ta sử dụng phương thức: `arc(x,y,r,start,stop)`. Kết hợp với phương thức `"ink"`, chẳng hạn như `stroke()` hoặc `fill()`. Xem ví dụ dưới:

```
<canvas id="myCanvas"></canvas>
<script>
        var c=document.getElementById("myCanvas");
        var ctx=c.getContext("2d");
        ctx.beginPath();
        ctx.arc(95,50,40,0,2*Math.PI);
        ctx.stroke();
</script>
```

Kết quả thu được như sau:



Hình 5. 4. Vẽ đường tròn sử dụng hàm arc()

1.3.5. Canvas Text

Để vẽ text trong canvas, những thuộc tính và phương thức quan trọng nhất là:

- `Font`: chỉ định font chữ

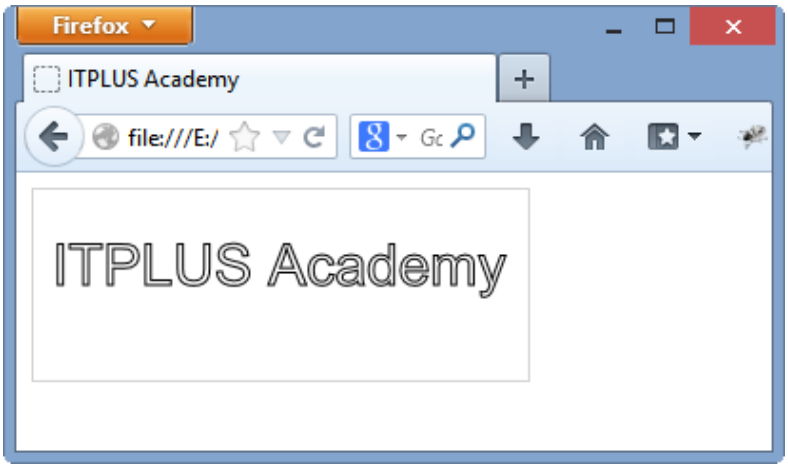
- `fillText(text,x,y)`: Vẽ text đổ màu

- `strokeText(text,x,y:` Vẽ text không đổ màu

Hai ví dụ dưới dây sử dụng `fillText()` và `strokeText()` để vẽ các đối tượng Canvas Text:

```
<canvas id="myCanvas" width="260" height="100" style="border:1px solid
#d3d3d3;"></canvas>
<script>
        var c=document.getElementById("myCanvas");
        var ctx=c.getContext("2d");
        ctx.font="30px Arial";
        ctx.strokeText("ITPLUS Academy",10,50);
</script>
```

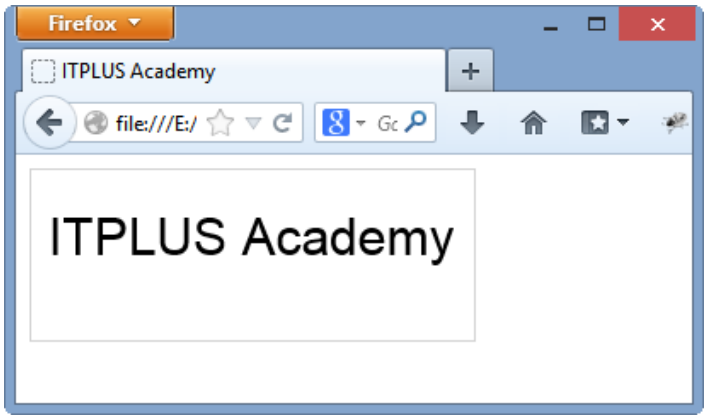
Kết quả:



Hình 5. 5. Vẽ đối tượng canvas text sử dụng strokeText()

```
<canvas id="myCanvas" width="260" height="100" style="border:1px solid
#d3d3d3;"></canvas>
<script>
        var c=document.getElementById("myCanvas");
        var ctx=c.getContext("2d");
        ctx.font="30px Arial";
        ctx.fillText("ITPLUS Academy",10,50);
</script>
```

Kết quả:



Hình 5. 6. Vẽ đối tượng canvas text sử dụng fillText()

1.3.6. Canvas Gradient

Gradients có thể sử dụng để đổ màu cho rectangles, circles, lines, text, ... Hình khối trong canvas không giới hạn màu. Có hai loại gradients:

- `createLinearGradient(x,y,x1,y1)`: Tạo hiệu ứng linear gradient

- `createRadialGradient(x,y,r,x1,y1,r1)`: Tạo hiệu ứng radial/circular gradient

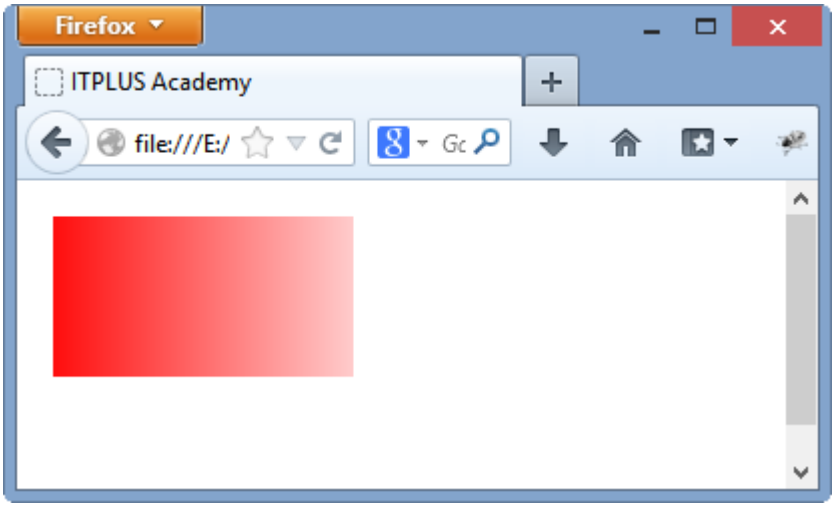
Sử dụng gradient, ta phải thêm ít nhất hai điểm dừng màu. Phương thức `addColorStop()` dùng để thêm điểm dừng màu, và vị trí của nó trong gradient. Vị trí có thể là bất kỳ giá trị nào trong khoảng từ 0 tới 1.

Để sử dụng gradient, ta gán thuộc tính `fillStyle` hoặc `strokeStyle` thành gradient, sau đó vẽ hình, ví dụ hình chữ nhật, text, hay một đường thẳng.

Dùng `createLinearGradient()`:

```
<canvas id="myCanvas"></canvas>
<script>
        var c=document.getElementById("myCanvas");
        var ctx=c.getContext("2d");
        // Tạo gradient
        var grd=ctx.createLinearGradient(0,0,200,0);
        grd.addColorStop(0,"red");
        grd.addColorStop(1,"white");
        // Đổ màu gradient
        ctx.fillStyle=grd;
        ctx.fillRect(10,10,150,80);
</script>
```

Kết quả:

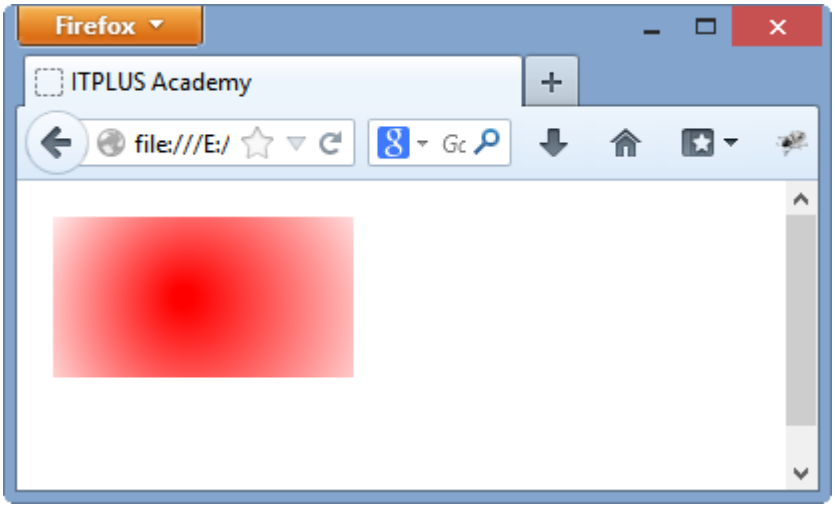


Hình 5. 7. Sử dụng `createLinearGradient()`

Dùng `createRadialGradient()`:

```
<canvas id="myCanvas"></canvas>
<script>
        var c=document.getElementById("myCanvas");
        var ctx=c.getContext("2d");
        // Tạo gradient
        var grd=ctx.createRadialGradient(75,50,5,90,60,100);
        grd.addColorStop(0,"red");
        grd.addColorStop(1,"white");
        // Đổ màu gradient
        ctx.fillStyle=grd;
        ctx.fillRect(10,10,150,80);
</script>
```

Kết quả:



Hình 5. 8. Sử dụng `createRadialGradient()`

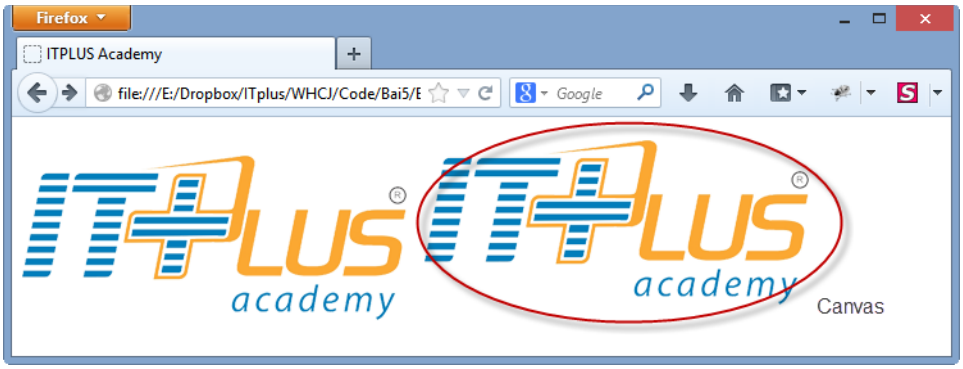
1.3.7. Canvas Images

Để chèn ảnh trong canvas, sử dụng phương thức: `drawImage(image,x,y)`

Ví dụ sử dụng hình ảnh tạo canvas:

```
<img id="itplus" src="itplus.png" alt="ITPlus Academy">
<canvas id="myCanvas" width="400" height="150"></canvas>
<script>
        var c=document.getElementById("myCanvas");
        var ctx=c.getContext("2d");
        var img=document.getElementById("itplus");
        ctx.drawImage(img,10,10);
</script>
```

Kết quả:



Hình 5. 9. Sử dụng canvas để vẽ image

1.4. Form và các thành phần của form trong HTML5

Form 2.0 - một tên gọi khác của form trong HTML5, là một trong những thành phần hấp dẫn nhất. Form 2.0 có nhiều bổ sung đáng giá cho các phần tử `<input>`, khiến chúng trở nên mạnh mẽ và đáp ứng được các yêu cầu thiết kế phức tạp mà không cần đến một giải pháp công nghệ bên ngoài nào như *Flash* và *Silverlight*.

```
<form>
        <label>Text Box:</label><br/>
        <input name="TextBox" type="text" maxlength="25" required
placeholder="Enter your text here" autofocus><br/><br/>
        <label>Numeric:</label><br/>
        <input name="Numeric" type="number" min="18" max="100"><br/><br/>
        <label>Slider:</label><br/>
        <input type="range" name="points" min="1" max="10" />
        <label>Date:</label><br/>
        <input name="Date" type="date"><br/><br/>
        <button type=submit>Submit</button>
</form>
```

Có lẽ bạn sẽ dễ dàng nhận ra sự xuất hiện nhiều thuộc tính mới trong thẻ `<input>`: thuộc tính required đánh dấu một field là bắt buộc, `placeholder` sẽ hiện một dòng text mờ trên textbox (dòng text này sẽ biến mất khi click vào *textbox*) và *autofocus* xác định phần tử nhận *focus* đầu tiên trên *form*. Với HTML4, việc hiện thực các tính năng này đều yêu cầu kĩ năng lập trình JavaScript, nhưng nay thì chúng đã được xây dựng trực tiếp trong HTML5.

Bên cạnh đó, thuộc tính type của `<input>` cung cấp nhiều kiểu *control* mới cho form. Chẳng hạn như ta có thể dễ dàng tạo một date control để lựa chọn ngày tháng bằng

cách set thuộc tính `type="date"`. HTML5 thậm chí còn có sẵn các field được thiết kế riêng cho địa chỉ email hoặc điện thoại. Cái hay ở chỗ HTML5 cho khả năng tương thích ngược khá tốt. Khi bạn thử đoạn code trên trên một trình duyệt không hỗ trợ HTML5 thì các field sẽ được hiển thị như các textbox thông thường mà không gây ra lỗi khó chịu nào. Tất cả các hỗ trợ này khiến cho việc xây dựng form trở nên dễ dàng hơn bao giờ hết.

1.4.1. Các thẻ mới của form

HTML5 có những thẻ mới sau: `<datalist>`, `<keygen>`, `<output>`. Không phải hầu hết các trình truyệt đều hỗ trợ những thẻ mới này của form. Tuy nhiên, bạn có bắt đầu sử dụng chúng. Nếu chúng không được hỗ trợ thì nó sẽ hoạt động dưới dạng những textbox bình thường.

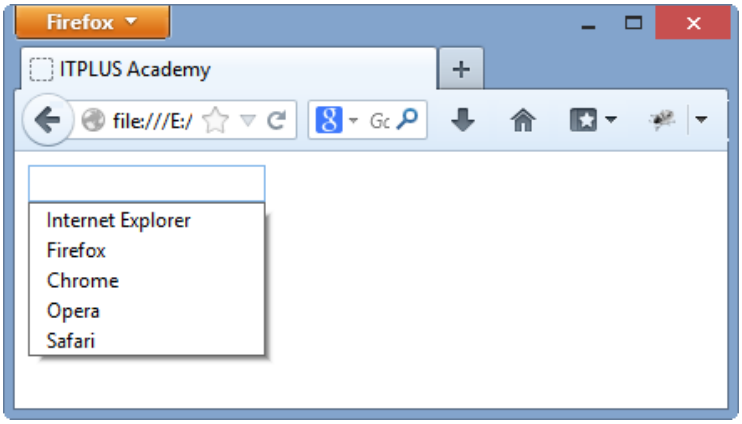
a. Thẻ <datalist>

Thẻ `<datalist>` chỉ ra một danh sách tùy chọn được định nghĩa sẵn cho thẻ `<input>`. Thẻ `<datalist>` được sử dụng để cung cấp tính năng *"auto complete"* cho thẻ `<input>`. Người sử dụng sẽ nhìn thấy một danh sách sổ xuống của những tùy chọn được định sẵn đó.

Ví dụ sau đây giới thiệu về thẻ `<datalist>`:

```
<form>
   <input list="browsers">
      <datalist id="browsers">
         <option value="Internet Explorer">
         <option value="Firefox">
         <option value="Chrome">
         <option value="Opera">
         <option value="Safari">
 </datalist>
</form>
```

Kết quả như sau:



Hình 5.10. Datalist bổ sung thuộc tính cho thẻ input

b. Thẻ <keygen>

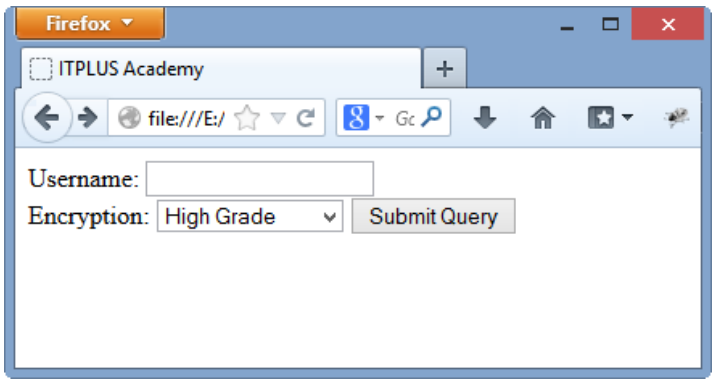
Mục đích của thẻ `<keygen>` là để cung cấp một cách an toàn để xác thực người dùng. Thẻ `<keygen>` quy định cụ thể một lĩnh vực mà từ khóa sẽ hiển thị. Khi form được được submit, hai phím được tạo ra, một public và một private.

Khóa private được lưu trữ tại máy cục bộ và các khóa public được gửi đến máy chủ. Khóa public có thể được sử dụng để tạo ra một chứng chỉ để xác thực người dùng.

Ví dụ minh họa cho thẻ `keygen`:

```
<form action="demo_keygen.php" method="get">
        Username: <input type="text" name="usr_name"><br/>
        Encryption: <keygen name="security">
        <input type="submit">
</form>
```

Kết quả như sau:



Hình 5. 11. Sử dụng thẻ <keygen>

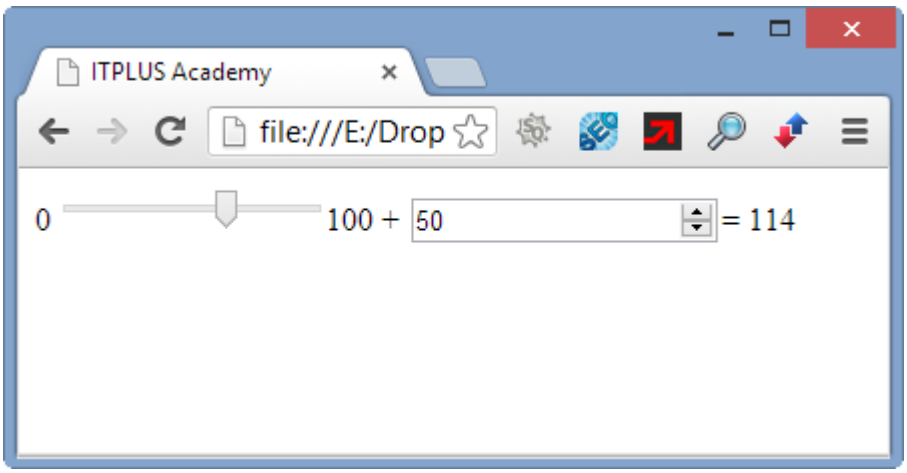
c. Thẻ output

Thẻ `<output>` trả về kết quả đầu ra của một phép tính *(giống như thực hiện đối với script)*:

Ví dụ sử dụng thẻ `<output>`:

```
<form oninput="x.value=parseInt(a.value)+parseInt(b.value)">0
        <input type="range" id="a" value="50">100 +
        <input type="number" id="b" value="50">=
        <output name="x" for="a b"></output>
</form>
```

Kết quả như sau:



Hình 5. 12. Sử dụng thẻ output

1.4.2. Các thuộc tính mới của form

a. Thuộc tính autocomplete

Thuộc tính *autocomplete* là thuộc tính của form, khi bạn bật *autocomplete* thì trình duyệt tự động điền lại giá trị trước đó bạn đã nhập vào form của bạn.

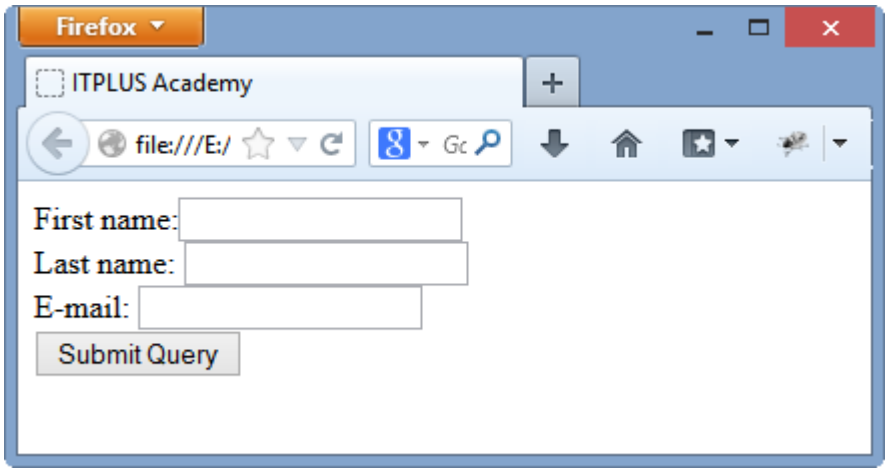
Ta có thể gán giá trị cho thuộc tính autocomplete=”on” cho form và tắt automcomplete = "off" cho thẻ input bên trong, hoặc ta có thể làm ngược lại.

Thuộc tính autocomplete dùng được cho `<form>` và các loại thẻ `<input>` sau: text, search, url, tel, email, password, datepickers, range, and color.

Ví dụ một form HTML dùng autocomplete (tắt autocomplete cho 1 thẻ trong đó):

```
<form action="demo_form.php" autocomplete="on">
        First name:<input type="text" name="fname"><br>
        Last name: <input type="text" name="lname"><br>
        E-mail: <input type="email" name="email" autocomplete="off"><br>
        <input type="submit">
</form>
```

Kết quả:



Hình 5. 13. Ví dụ về thuộc tính autocomplete của form

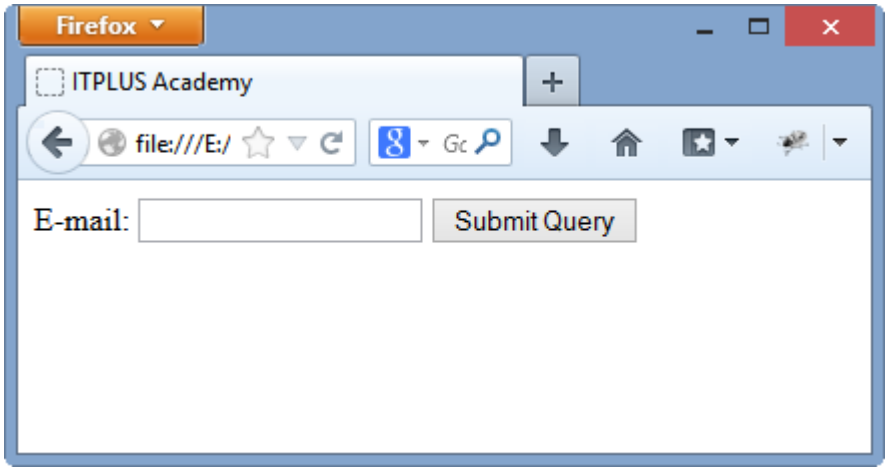
b. Thuộc tính novalidate

Thuộc tính novalidate là thuộc tính boolean, nghĩa là có hoặc không. Thuộc tính này khiến cho form không bắt lỗi nhập liệu khi submit.

Ví dụ form không bắt lỗi khi nhập sai email:

```
<form action="demo_form.php" novalidate="novalidate">
        E-mail: <input type="email" name="user_email">
        <input type="submit">
</form>
```

Kết quả:



Hình 5. 14. Ví dụ về thuộc tính novalidate

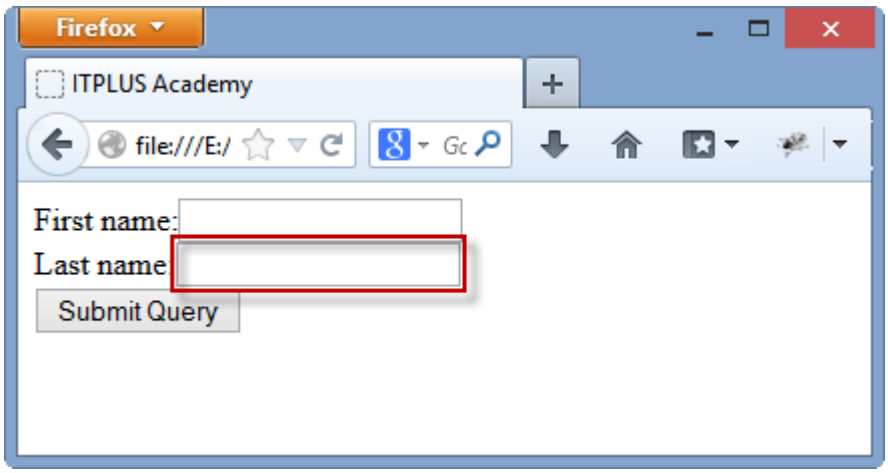
c. Thuộc tính autofocus

Thuộc tính autofocus là thuộc tính boolean, nghĩa là có hoặc không. Khi load một trang web, con trỏ tự động nằm tại thẻ `<input>` nào có thuộc tính autofocus.

Ví dụ tự động focus trường textbox Lastname khi load trang:

```
<form action="demo_form.php">
        First name:<input type="text" name="fname">
 Last name:<input type="text" name="lname" autofocus>
        <input type="submit">
</form>
```

Kết quả:



Hình 5. 15. Ví dụ về thuộc tính autofocus

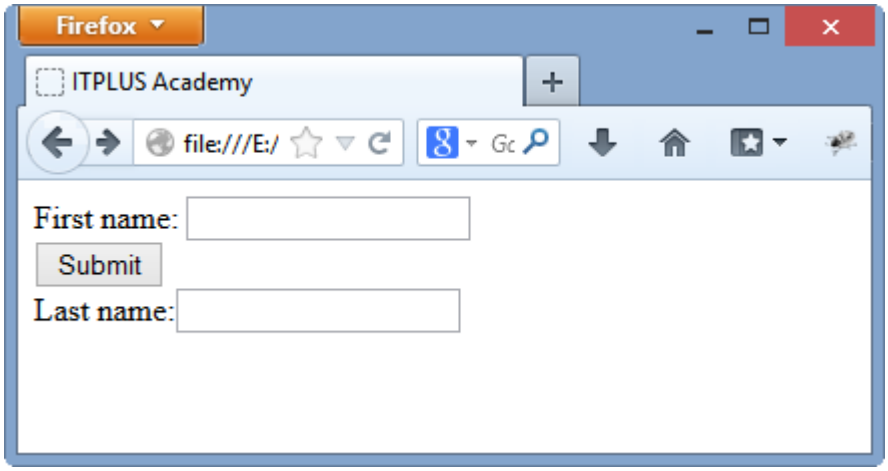
d. Thuộc tính form

Trước đây `<input>` thuộc form nào thì nằm trong form đó, giờ đây với thuộc tính form ta có nhiều cách hơn để quy định `<input>` là của form nào.

Ví dụ về việc một input nằm ngoài form nhưng vẫn là một phần của form:

```
<form action="demo_form.php" id="form1">
        First name: <input type="text" name="fname"><br>
        <input type="submit" value="Submit">
</form>
        Last name:<input type="text" name="lname" form="form1">
```

Kết quả:



Hình 5. 16. Ví dụ về thuộc tính form

e. Thuộc tính formaction

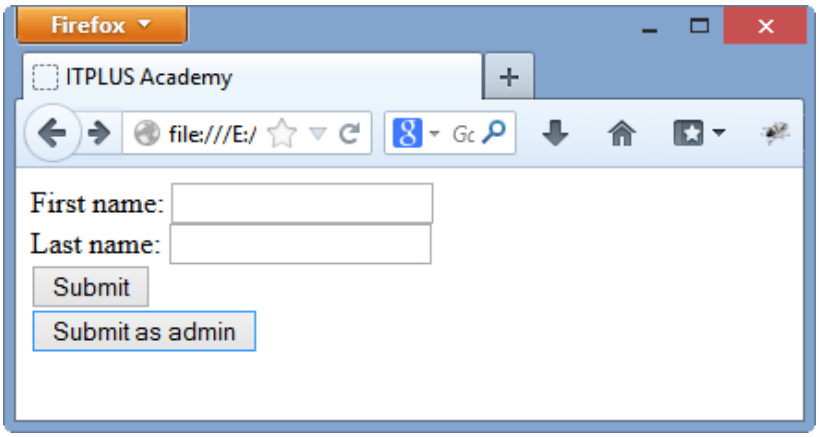
Thuộc tính formaction giúp ta có thể tùy chọn file nào sẽ xử lý dữ liệu nhập vào. Thuộc tính formaction sẽ ghi đè lên thuộc tính action của thẻ form.

Thuộc tính formaction được dùng cho 2 kiểu `<input>` là kiểu `type="submit"` và kiểu `type="image"`.

Ví dụ một HTML form có 2 nút submit để gửi đi xử lý ở 2 nơi khác nhau:

```
<form action="demo_form.php">
   First name: <input type="text" name="fname"><br>
   Last name: <input type="text" name="lname"><br>
   <input type="submit" value="Submit"><br>
   <input type="submit" formaction="demo_admin.php" value="Submit as admin">
</form>
```

Kết quả:



Hình 5. 17. Ví dụ về thuộc tính formaction

f. Thuộc tính formenctype

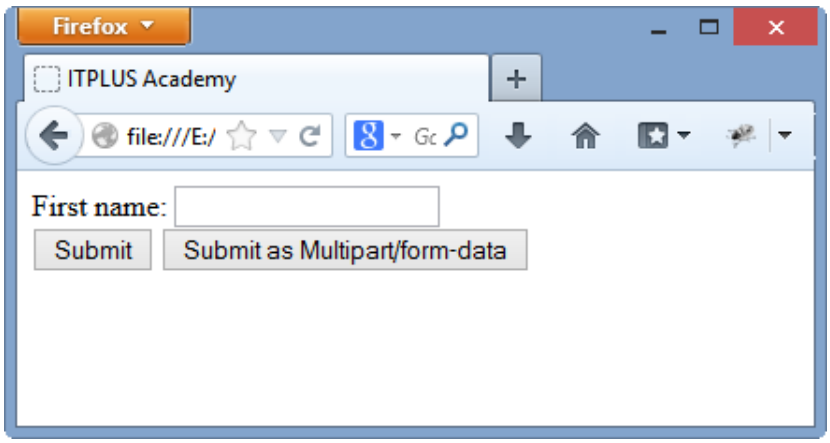
Thuộc tính formenctype giúp ta mã hóa dữ liệu `"multipart/form-data"` khi gửi đến server với yêu cầu `method="post"`. Thuộc tính formenctype sẽ ghi đè lên thuộc tính enctype của phần tử `<form>`.

Thuộc tính formenctype được dùng cho 2 kiểu `<input>` là `type="submit"` and `type="image"`.

Ví dụ gửi dữ liệu với kiểu mã hóa mặc định và kiểu mã hóa `"multipart/form-data”`:

```
<form action="demo_post_enctype.php" method="post">
   First name: <input type="text" name="fname"><br>
   <input type="submit" value="Submit">
   <input type="submit" formenctype="multipart/form-data" value="Submit as
Multipart/form-data">
</form>
```

Kết quả:



Hình 5. 18. Ví dụ về thuộc tính formenctype

g. Thuộc tính formmethod

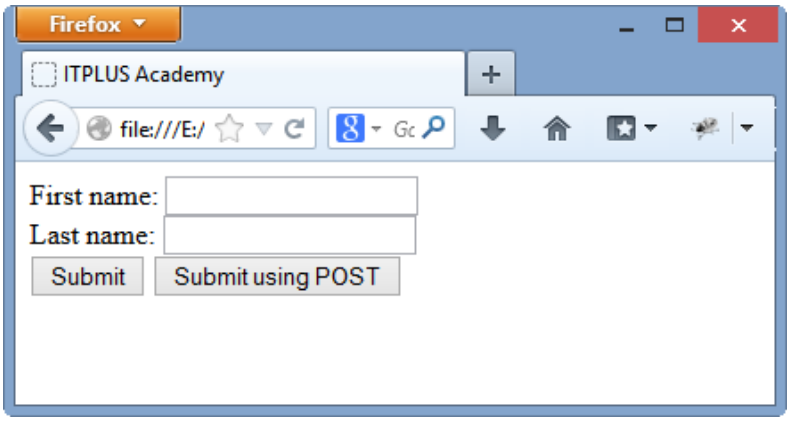
Thuộc tính formmethod quy định phương thức gửi dữ liệu từ form đến phần xử lý trên sever. Thuộc tính formmethod ghi đè lên thuộc tính method của phần tử form.

Thuộc tính formmethod được dùng cho 2 kiểu `<input>` là `type="submit"` and `type="image"`.

Ví dụ nút submit thay đổi lại phương thức get thành post:

```
<form action="demo_form.php" method="get">
        First name: <input type="text" name="fname"><br>
        Last name: <input type="text" name="lname"><br>
        <input type="submit" value="Submit">
        <input type="submit" formmethod="post" formaction="demo_post.php"
value="Submit using POST">
</form>
```

Kết quả:



Hình 5. 19. Ví dụ về thuộc tính formmethod

h. Thuộc tính formnovalidate

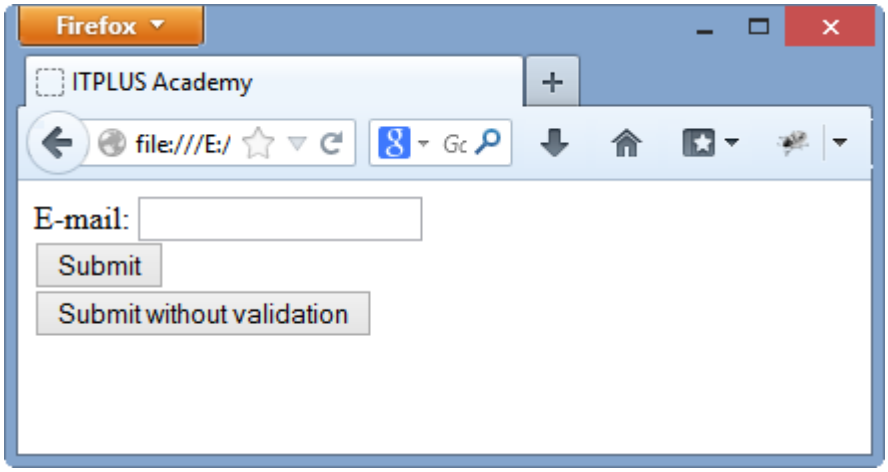
Nếu như thuộc tính novalidate có 2 giá trị là có hoặc không, nó làm cho form không bắt lỗi dữ liệu người dùng nhập vào thì huộc tính formnovalidate sẽ ghi đè lên thuộc tính novalidate của `<form>`.

Thuộc tính formnovalidate dùng cho kiểu `<input> type="submit"`.

Ví dụ một form với 2 nút submit (bắt và không bắt lỗi):

```
<form action="demo_form.php">
        E-mail: <input type="email" name="userid"><br>
        <input type="submit" value="Submit"><br>
        <input type="submit" formnovalidate="formnovalidate" value="Submit
without validation">
</form>
```

Kết quả:



Hình 5. 20. Form với 2 nút submit, một kiểm tra dữ liệu, một không kiểm tra dữ liệu nhập vào

i. Thuộc tính formtarget

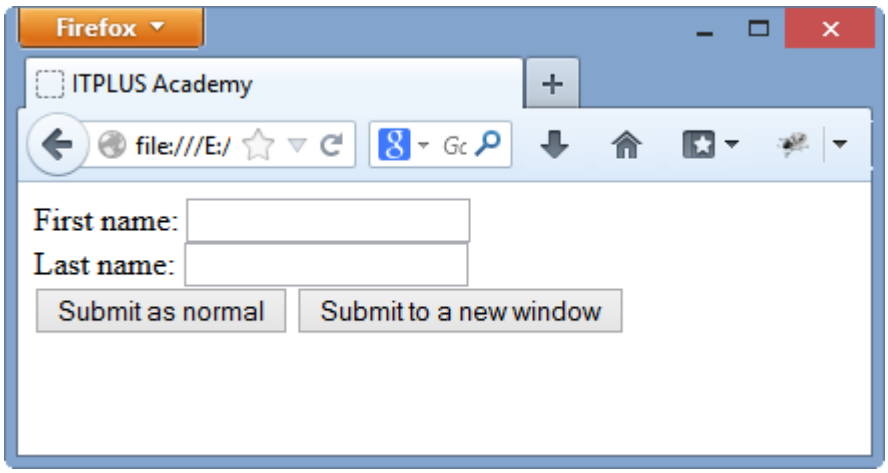
Giống thuộc tính target của form, tùy theo giá trị của target sau khi submit dữ liệu thì sẽ mở trang mới hoặc cửa sổ mới,.. Thuộc tính formtarget ghi đè thuộc tính target của `<form>`.

Thuộc tính formtarget được dùng cho 2 kiểu `<input>` là `type="submit"` và `type="image"`.

Ví dụ một form với 2 nút submit 2 kiểu target:

```
<form action="demo_form.php">
   First name: <input type="text" name="fname"><br>
   Last name: <input type="text" name="lname"><br>
   <input type="submit" value="Submit as normal">
   <input type="submit" formtarget="_blank" value="Submit to a new window">
</form>
```

Kết quả:



Hình 5. 21. Ví dụ về thuộc tính formtarget

j. Thuộc tính height và width

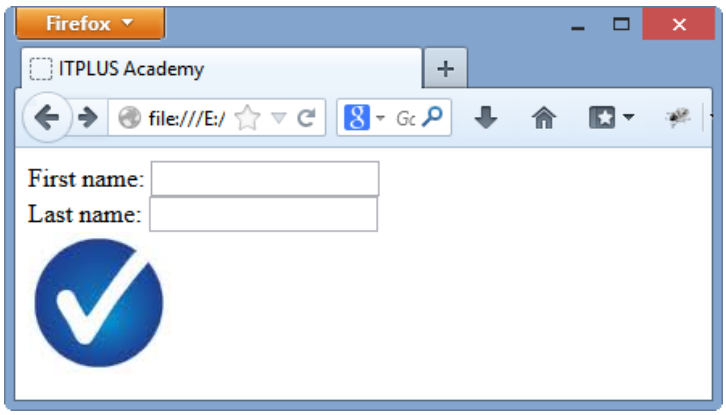
Thuộc tính height và width quy định chiều cao và chiều rộng của thẻ `<input>`. Thuộc tính height and width chỉ dùng cho thẻ `<input type="image">`.

Đối với một hình ảnh, ta nên ghi rõ chiều cao và chiều rộng vì khi tải trang sẽ dành riêng không gian cho nó và chất lượng hình ảnh không bị ảnh hưởng.

Ví dụ quy định chiều cao và chiều rộng cho ảnh:

```
<form action="demo_form.php">
        First name: <input type="text" name="fname"><br>
        Last name: <input type="text" name="lname"><br>
        <input type="image" src="img_submit.jpg" alt="Submit" width="88"
height="88">
</form>
```

Kết quả:



Hình 5. 22. Ví dụ về thuộc tính width và height với thẻ input type="image"

k. Thuộc tính list

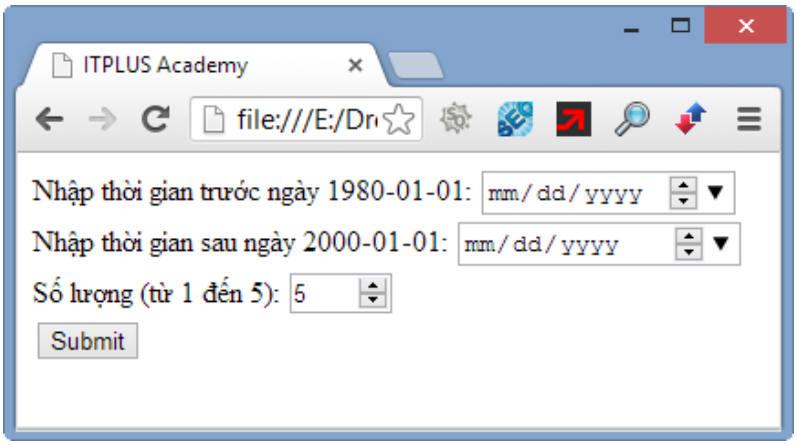
Thuộc tính list hiển thị ra một `<datalist>` chứa các dữ liệu mà ta định sẵn để thuận tiện cho việc nhập liệu *(đã được trình bày trong phần các thẻ mới của form).*

l. Thuộc tính min và max

Thuộc tính min và max xác định giá trị nhỏ nhất và lớn nhất của thẻ `<input>`. Thuộc tính min và max dùng cho các loại <input> sau: `number`, `range`, `date`, `datetime`, `datetime-local`, `month`, `time` và `week`. Ví dụ phần tử `<input>` với giá trị min và max:

```
<form action="demo_form.php">
        Nhập thời gian trước ngày 1980-01-01:
        <input type="date" name="bday" max="1979-12-31"> <br/>
        Nhập thời gian sau ngày 2000-01-01:
        <input type="date" name="bday" min="2000-01-02"> <br/>
        Số lượng (từ 1 đến 5):
        <input type="number" name="quantity" min="1" max="5">  <br/>
        <input type="submit">
</form>
```

Kết quả:



Hình 5. 23. Ví dụ về thuộc tính min và max

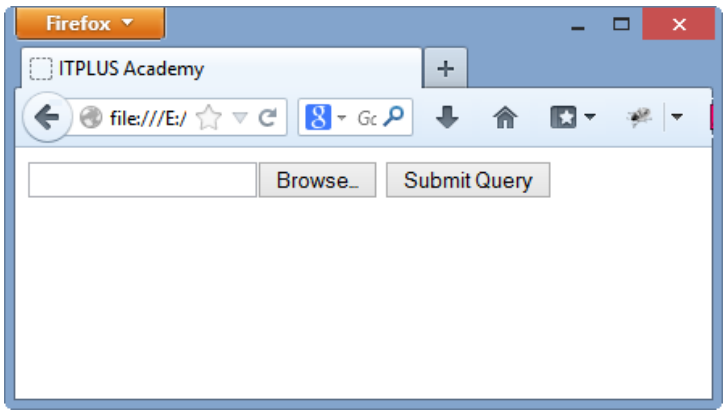
m. Thuộc tính multiple

Thuộc tính multiple có 2 giá trị là có hoặc không. Cho phép ta nhập nhiều hơn một giá trị vào thẻ `<input>`. Thuộc tính multiple được dùng cho 2 kiểu `<input>` là: `email` và `file`.

Ví dụ cho phép chọn nhiều file ảnh:

```
<form action="demo_form.php">
        <input type="file" name="img" multiple="multiple">
        <input type="submit">
</form>
```

Kết quả:



Hình 5. 24. Ví dụ về thuộc tính multiple

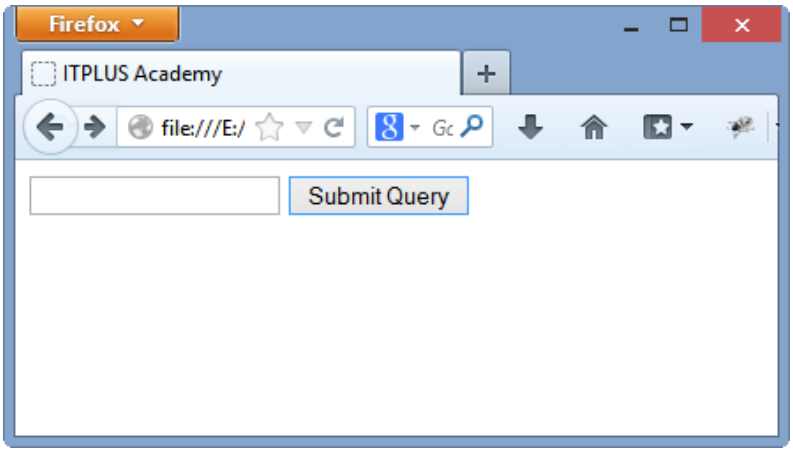
n. Thuộc tính pattern

Tạo ra sự ràng buộc đối với dữ liệu nhập vào. Thuộc tính pattern được dùng cho các kiểu: `text, search, url, tel, email, và password.`

Ví dụ trường nhập liệu chỉ tối đa 3 từ và không cho phép nhập các ký tự đặc biệt:

```
<form action="demo_form.php">
        <input type="text" name="country_code" pattern="[A-Za-z]{3}"
title="3 ký tự chữ nhé">
        <input type="submit">
</form>
```

Kết quả:



Hình 5. 25. Ví dụ về thuộc tính pattern

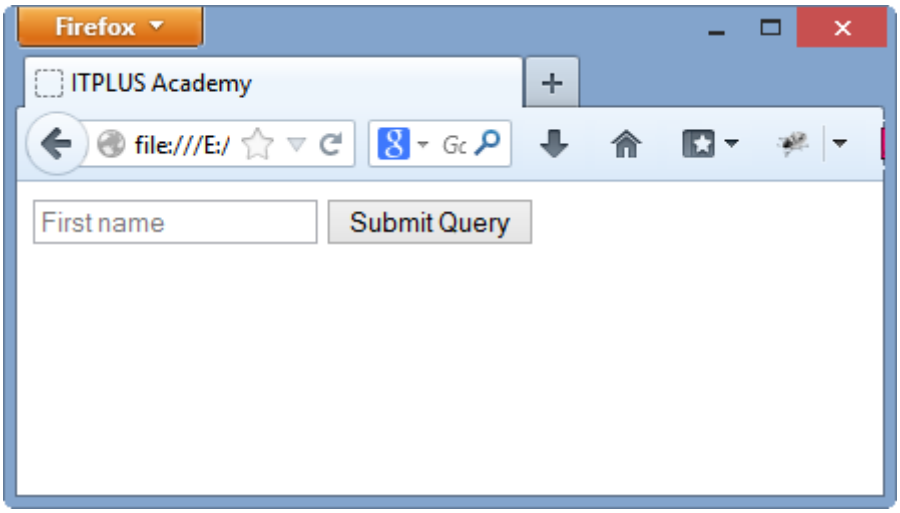
o. Thuộc tính placeholder

Thuộc tính placeholder tạo ra giá trị mẫu trên thẻ `<input>`, giá trị mẫu sẽ bị xóa đi nếu ta chọn đến trường của nó. Thuộc tính placeholder dùng cho các kiểu `<input>` sau: `text, search, url, tel, email, và password.`

Ví dụ trường nhập liệu có chú thích giá trị mẫu:

```
<form action="demo_form.php">
        <input type="text" name="fname" placeholder="First name">
        <input type="submit">
</form>
```

Kết quả:



Hình 5. 26. Ví dụ về thuộc tính placeholder

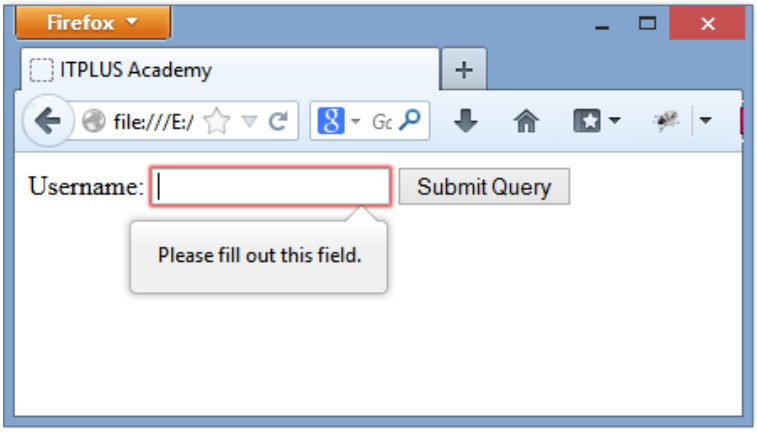
p. *Thuộc tính required*

Thuộc tính required có 2 giá trị có hoặc không. Thuộc tính này quy định một trường nhập liệu bắt buộc không được rỗng. Thuộc tính required được dùng cho các kiểu nhập liệu: `text`, `search`, `url`, `tel`, `email`, `password`, `date pickers`, `number`, `checkbox`, `radio`, và `file`.

Ví dụ về việc bắt buộc nhập liệu:

```
<form action="demo_form.php">
        Username: <input type="text" name="username" required="required">
        <input type="submit">
</form>
```

Kết quả:



Hình 5. 27. Ví dụ về thuộc tính required

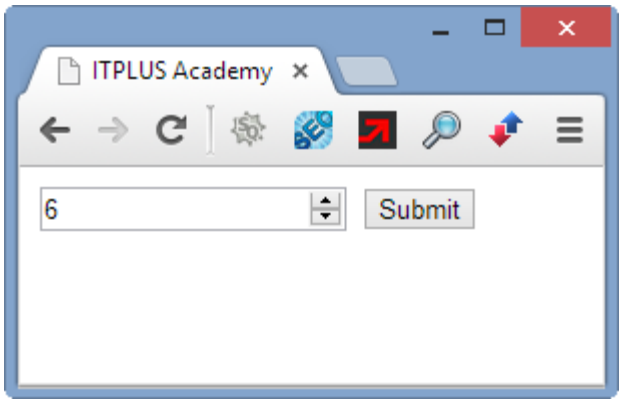
q. Thuộc tính step

Thuộc tính step quy định bước số khi nhập vào `<input>`. Thuộc tính step có thể sử dụng kết hợp với max và min để tạo ra một khoảng giá trị các số mà các số kề bên hơn kém nhau "`step`" đơn vị. Thuộc tính step được dùng cho các kiểu dữ liệu: `number`, `range`, `date`, `datetime`, `datetime-local`, `month`, `time` và `week`.

Ví dụ tạo một trường cho nhập dữ liệu với bước số là 3:

```
<form action="demo_form.php">
        <input type="number" name="points" step="3">
        <input type="submit">
</form>
```

Kết quả:



Hình 5. 28. Ví dụ về thuộc tính step

1.4.3. HTML5 input type

HTML5 cung cấp thêm một số loại cho thẻ `<input>`. Đây là những loại input cho phép điều khiển và kiểm soát dữ liệu tốt hơn.

Các loại đó là: `color`, `date`, `datetime`, `datetime-local`, `email`, `month`, `number`, `range`, `search`, `tel`, `time`, `url`, `week`

STT	HTML5 input type	Công dụng
1	`color`	Cho phép lựa chọn màu sắc
2	`date`	Cho phép lựa chọn ngày
3	`datetime`	Cho phép lựa chọn ngày giờ với time zone
4	`datetime-local`	Cho phép lựa chọn ngày, tháng không time zone
5	`email`	Cho phép nhập liệu 1 email (bắt buộc)
6	`month`	Cho phép lựa chọn tháng
7	`number`	Cho phép nhập liệu kiểu số
8	`range`	Cho phép lựa chọn khoảng
9	`search`	Định nghĩa một trường tìm kiếm

10	`tel`	Nhập liệu kiểu số điện thoại
11	`time`	Cho phép lựa chọn thời gian
12	`url`	Cho phép nhập liệu một đường dẫn URL (bắt buộc)
13	`week`	Cho phép lựa chọn tuần nào đó trong năm

1.5. Trình diễn đa phương tiện với HTML5

1.5.1. HTML5 Video

Tới tận bây giờ, vẫn chưa có chuẩn hiển thị video/phim trên web. Hầu hết video được hiển thị thông qua một *plug-in* (ví dụ như flash). Những trình duyệt khác nhau có thể có những plug-in khác nhau.

HTML5 định nghĩa một phần tử mới để nhúng *video/movie* vào trang web: đó là thẻ `<video>`.

Trình duyệt hỗ trợ: Internet Explorer 9, Firefox, Opera, Chrome, và Safari. Internet Explorer 8 và các version trước đó không hỗ trợ.

a. HTML5 Video hoạt động như nào?

```
<video width="320" height="240" controls>
        <source src="movie.mp4" type="video/mp4">
        <source src="movie.ogg" type="video/ogg">
</video>
```

- Thuộc tính `control` để thêm các điều khiển cho video, ví dụ như `play`, `pause` và `volume` (âm lượng).

- Thêm thuộc tính `width` và `height` (rộng, cao) cũng là một ý tưởng hay. Nếu thuộc tính `width` và `height` được gán trước, trang web sẽ đặt trước không gian cho video. Nếu không có, trình duyệt sẽ không biết kích thước của video, và sẽ không đặt trước khoảng trống cho nó. Layout của trang sẽ thay đổi trong suốt quá trình tải video.

- Ta cũng có thể chèn text vào giữa cặp thẻ `<video>` và `</video>` để hiển thị nếu trình duyệt không hỗ trợ thẻ `<video>`.

- Thẻ `<video>` cho phép nhúng nhiều `<source>`. Thẻ `<source>` có thể link tới nhiều video khác nhau. Trình duyệt sẽ sử dụng video đầu tiên có định dạng hợp lệ.

b. Các định dạng video và trình duyệt hỗ trợ

Có 3 định dạng video được hỗ trợ tính tới thời điểm này:

STT	Trình duyệt	MP4	WebM	Ogg
1	Chrome 6+	Yes	Yes	Yes
2	Firefox 3.6+	No	Yes	Yes
3	Internet Explorer 9+	Yes	No	No

4	Opera 10.6+	No	Yes	Yes
5	Safari 5+	Yes	No	No

- MP4 = MPEG 4 files with H264 video codec and AAC audio codec
- WebM = WebM files with VP8 video codec and Vorbis audio codec
- Ogg = Ogg files with Theora video codec and Vorbis audio codec

c. MIME Types for Video Formats

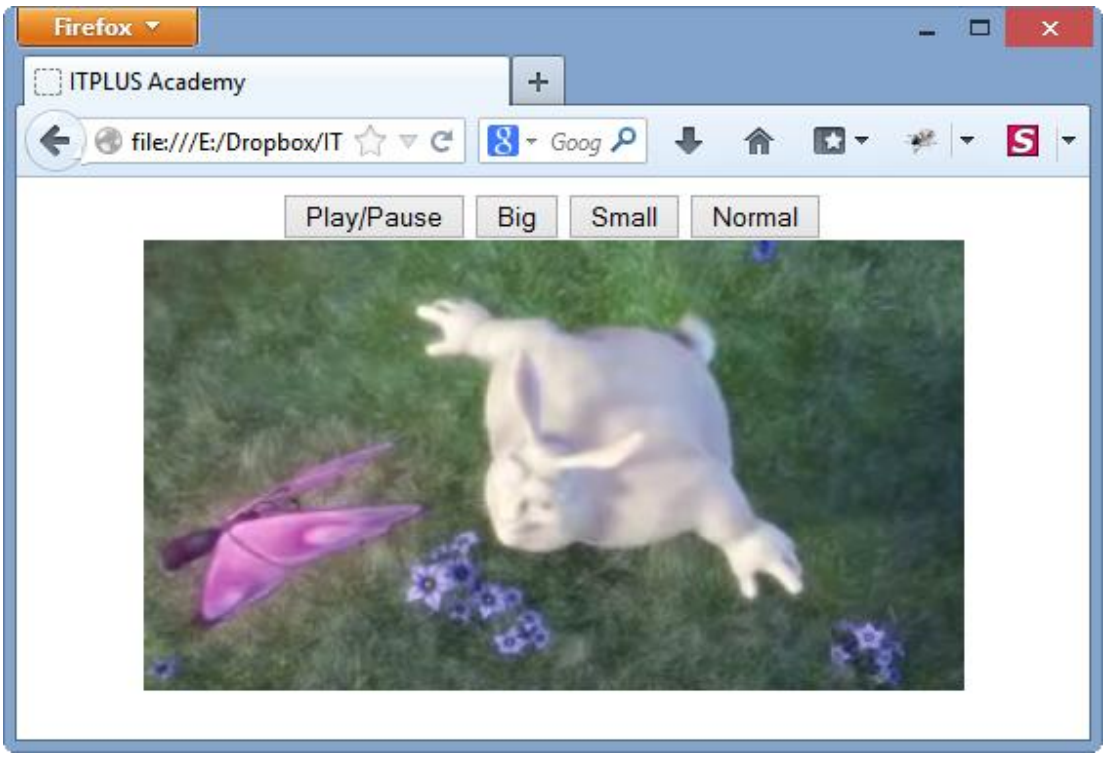
STT	Format	MIME-Type
1	MP4	video/mp4
2	Ogg	video/ogg
3	WebM	video/webm

HTML5 có các phương thức, thuộc tính, và sự kiện DOM cho các phần tử `<video>` và `<audio>`. Những phương thức, thuộc tính, sự kiện này cho phép ta điều khiển/thao tác với `<video>` và `<audio>` thông qua javascript.

Ví dụ dưới đây sẽ mình họa điều này:

```
<div style="text-align:center">
        <button onclick="playPause()">Play/Pause</button>
        <button onclick="makeBig()">Big</button>
        <button onclick="makeSmall()">Small</button>
        <button onclick="makeNormal()">Normal</button><br/>
        <video id="video1" width="420">
                <source src="mov_bbb.mp4" type="video/mp4">
                <source src="mov_bbb.ogg" type="video/ogg">
        </video>
</div>
<script>
        var myVideo=document.getElementById("video1");
        //Hàm chạy hoặc dừng video
        function playPause()
        {
                if (myVideo.paused) myVideo.play();
                else myVideo.pause();
        }
        //Hàm phóng to
        function makeBig()
        {
                myVideo.width=560;
        }
        //Ham thu nhỏ
        function makeSmall()
        {
                myVideo.width=320;
        }
         //Hàm đưa về trạng thái bình thường
        function makeNormal()
        {
                myVideo.width=420;
        }
</script>
```

Kết quả như sau:



Hình 5. 29. Video trong HTML5

1.5.2. HTML5 Audio

Tương tự video, tới bây giờ vẫn chưa có chuẩn nào để phát nhạc/âm thanh trên website. Audio thường được phát thông qua plug-in như flash, và chúng khác nhau trên các trình duyệt. HTML5 cung cấp một phần tử mới để nhúng nhạc vào website, đó là `<audio>`.

Trình duyệt hỗ trợ: Internet Explorer 9, Firefox, Opera, Chrome, và Safari. Internet Explorer 8 và các version trước đó không hỗ trợ.

a. Cách thức <audio> hoạt động

```
<audio controls>
        <source src="audio.ogg" type="audio/ogg">
        <source src="audio.mp3" type="audio/mpeg">
</audio>
```

Thuộc tính control sử dụng để thêm điều khiển cho phần tử, như play, pause, and volume. Có thể thêm text giữa cặp thẻ `<audio>` và `</audio>` để trình duyệt hiển thị khi không hỗ trợ `<audio>`.

Cho phép nhúng nhiều phần tử `<source>` như video. Thẻ `<source>` có thể link tới nhiều audio khác nhau. Trình duyệt sẽ sử dụng audio đầu tiên có định dạng hợp lệ.

b. Các định dạng audio và trình duyệt hỗ trợ

Có 3 định dạng audio được hỗ trợ tính tới thời điểm này:

STT	Trình duyệt	MP3	Wav	Ogg

1	Chrome 6+	Yes	Yes	Yes
2	Firefox 3.6+	No	Yes	Yes
3	Internet Explorer 9+	Yes	No	No
4	Opera 10+	No	Yes	Yes
5	Safari 5+	Yes	Yes	No

c. MIME Types for Audio Formats

STT	Format	MIME-Type
1	MP3	audio/mpeg
2	Ogg	audio/ogg
3	Wav	audio/wav

1.6. Lưu trữ dữ liệu tại trình duyệt

HTML5 cung cấp một tính năng lưu trữ dữ liệu tại client với dung lượng giới hạn lớn hơn nhiều so với cookie. Tính năng này được gọi là *Web Storage* và được chia thành hai đối tượng là *localStorage* và *sessionStorage*. Bài viết này sẽ giúp bạn nắm được các kiến thức đầy đủ về sử dụng hai đối tượng này trong việc lập trình web.

Người ta dựng sẵn hai kiểu đối tượng web storage trong HTML5:

- *Đối tượng sessionStorage* lưu trữ dữ liệu cho một phiên làm việc duy nhất. Nếu người dùng đóng trang hoặc trình duyệt, dữ liệu bị xóa hết.

- *Đối tượng localStorage* lưu trữ dữ liệu không thời hạn. Dữ liệu vẫn còn được lưu trữ khi đóng trang web hoặc trình duyệt, tùy thuộc vào giá trị thiết lập dung lượng lưu trữ cho trình duyệt của người dùng.

Cả hai đối tượng lưu trữ đều có cùng các phương thức và các thuộc tính. Để thống nhất, bài này sử dụng đối tượng localStorage trong suốt các ví dụ.

1.6.1. Các trình duyệt hỗ trợ

Tất cả các phiên bản mới của các trình duyệt mới nhất, bao gồm Firefox, Google Chrome, Safari, Opera và Microsoft® Windows® Internet Explorer® 8+. Thật không may, phiên bản Internet Explorer 7 và trước đó không hỗ trợ tính năng web storage. Hình 5.30 cho thấy các phiên bản của từng trình duyệt máy tính có hỗ trợ tính năng web storage của HTML5.

Chrome	Firefox	Safari	Opera	Internet Explorer
4+	4+	4+	11+	8+

Hình 5. 30. Các trình duyệt hỗ trợ web storage

Các trình duyệt di động, ngoài Opera Mini, cũng hỗ trợ tính năng web storage của HTML5. Hình 5.31 cho thấy các phiên bản của từng trình duyệt di động có hỗ trợ tính năng web storage của HTML5.

iOS	Android	Opera Mini	Opera Mobile
5+	3+	NA	11+

Hình 5. 31. Trình duyệt di động hỗ trợ tính năng web storage

Việc các trình duyệt hỗ trợ web storage của HTML5 là khá ấn tượng. Tuy nhiên, các trình duyệt cũ hơn cần có một kiểu kiểm tra trình duyệt nào đó về hỗ trợ tính năng web storage trước khi bạn định sử dụng nó. Việc kiểm tra một trình duyệt có hỗ trợ tính năng web storage hay không rất dễ dàng. Bạn có thể sử dụng một câu lệnh có điều kiện đơn giản để xem liệu đối tượng lưu trữ dữ liệu của HTML5 đã được định nghĩa chưa. Nếu nó đã được định nghĩa thì việc tiếp tục viết kịch bản lệnh với web storage là an toàn. Nếu nó vẫn chưa được định nghĩa, bạn phải sử dụng một phương pháp thay thế khác, chẳng hạn như các cookie của JavaScript, nếu việc lưu trữ dữ liệu là cần thiết.

```
if(typeof(Storage)!== "undefined") {
  // Web storage is supported
}
else {
  // Web storage is NOT supported
}
```

1.6.2. Các phương thức web storage của HTML5

Có một vài phương thức dễ sử dụng cung cấp các chức năng web storage của HTML5. Những phương thức này hỗ trợ thiết lập một cặp khóa/giá trị, hai tùy chọn để lấy ra một giá trị dựa trên một khóa, xóa tất cả các cặp khóa/giá trị cùng một lúc và loại bỏ một cặp khóa/giá trị cụ thể. Bảng 3 cho thấy các phương thức web storage của HTML5 có sẵn.

STT	Phương thức	Mô tả
1	`setItem(key,`	Thêm một cặp khóa/giá trị vào đối tượng web storage để

	`value)`	sử dụng sau này. Giá trị này có thể là bất kỳ kiểu dữ liệu nào: một chuỗi ký tự, số, mảng và v.v
2	`getItem(key)`	Lấy ra một giá trị dựa trên khóa đã được sử dụng để lưu trữ nó ở vị trí đầu tiên.
3	`clear()`	Xóa tất cả các cặp khóa/giá trị khỏi đối tượng web storage.
4	`removeItem(key)`	Loại bỏ một cặp khóa/giá trị cụ thể khỏi đối tượng web storage dựa trên một khóa.
5	`key(n)`	Lấy ra giá trị cho key[n].

B. BÀI TẬP

2.1. Câu hỏi ôn tập

Câu 1: Phiên bản HTML trước phiên bản HTML5 là phiên bản nào?

I. HTML 4.1
J. HTML 4.9
K. HTML 4
L. HTML 4.01

Câu 2: Thành phần HTML5 nào cho phép chạy video trên web?

A. <movie>
B. <media>
C. <video>

Câu 3: Thành phần HTML5 nào là đúng để cho phép chạy âm thanh trên web?

A. <audio>
B. <sound>
C. <mp3>

Câu 4: Thẻ <canvas> trong HTML5 được sử dụng để làm gì?

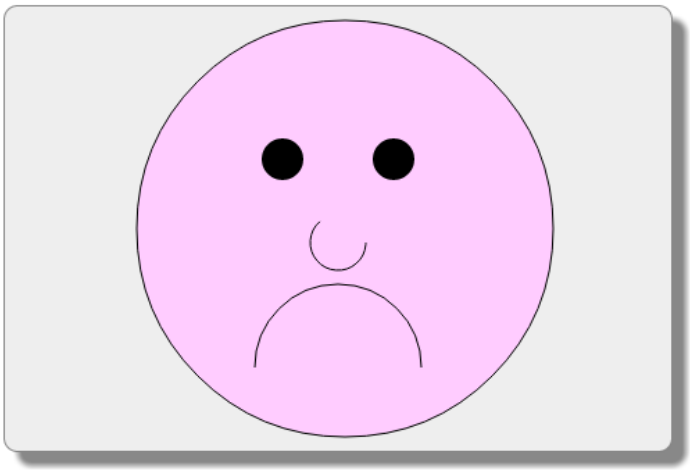
A. Tạo ra các yếu tố kéo thả
B. Hiển thị dữ liệu trong database
C. Thao tác dữ liệu trong MySQL
D. Vẽ các đối tượng đồ họa

Câu 5: Thành phần nào trong HTML5 được sử dụng để vẽ trong canvas?

A. getContent
B. getContext
C. getGraphics
D. getCanvas

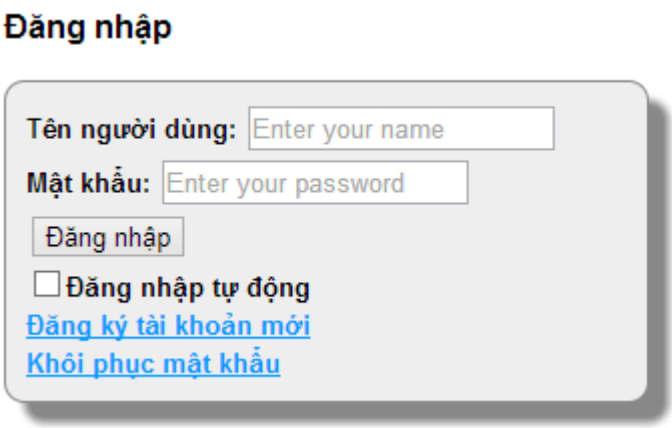
2.2. Bài tập

Bài 1. Sử dụng các kiến thức đã được học về HTML5 canvas và các đối tượng đồ họa, thực hiện vẽ hình sau:



Bài 2: Sử dụng các kiến thức đã được học về HTML5 thực hiện xây dựng form theo các yêu cầu sau:

- Hiển thị form đăng nhập như hình vẽ ở bên.
- Tự động autocomplete đối với trường nhập liệu "Tên người dùng"
- Kiểm tra dữ liệu nhập vào có hợp lệ hay không?



BÀI 6. NGÔN NGỮ CSS

Trong công việc thiết kế trang web thì một trong những vấn đề quan trọng đó là làm sao đảm bảo trang web có thể hiển thị tốt trên hầu hết các trình duyệt. Hiểu rõ vấn đề này W3C *(World Wide web Consortium)* - một tổ chức có nhiệm vụ quản lý và đặt ra các tiêu chuẩn web cùng với các đối tác khác như Microsoft, Mozilla Foundation,… đã đặt ra các tiêu chuẩn về mã cho web. Nó cho phép nhà phát triển web có thể tự tin hơn khi thực hiện dự án cũng như đảm bảo cho các trang web thỏa mãn tiêu chuẩn có thể hiển thị tốt trên nhiều trình duyệt khác nhau. Mục đích của việc cho ra đời CSS cũng là để đáp ứng nhu cầu trên.

Nội dung bài học cung cấp những kiến thức tổng quan về CSS, sự tương tác giữa CSS và HTML và vận dụng các thuộc tính của CSS để trình bày giao diện trang web.

A. NỘI DUNG

1.1. Tổng quan về CSS

1.1.1. CSS là gì?

CSS là từ viết tắt của cụm từ tiếng Anh Cascading Style Sheets, các tập tin định kiểu theo tầng, được dùng để miêu tả cách trình bày các tài liệu HTML, XHTML, XML v.v...hay nói đơn giản hơn, CSS được dùng để trình bày giao diện trang web.

Tính hữu dụng của CSS:

- Hạn chế tối thiểu việc làm rối mã HTML của trang web bằng các thẻ quy định kiểu dáng (chữ đậm, chữ in nghiêng, chữ có gạch chân, chữ màu...), khiến mã nguồn của trang web được gọn gàng hơn, tách nội dung của trang web và định dạng hiển thị, dễ dàng cho việc chỉnh sửa, cập nhật nội dung.

- Tạo ra các kiểu dáng có thể áp dụng cho nhiều trang web, giúp tránh phải lặp lại việc định dạng cho các trang web giống nhau.

- Thay vì đặt các thẻ qui định kiểu dáng cho văn bản HTML (hoặc XHTML) ngay trong nội dung của nó, bạn nên sử dụng CSS trong việc thiết kế và xây dựng trang web.

1.1.2. Cú pháp và quy ước viết CSS

Cú pháp của CSS bao gồm 3 thành phần chính là đối tượng *(selector)*, thuộc tính *(attribute)* và giá trị tương ứng của thuộc tính *(value)*.

Cú pháp của CSS được thể hiện như sau:

```
Selector{
     attribute 1: Value1;
     ...
     attribute n: Value n;
}
```

Trong đó:

Selector là bộ chọn, dùng để xác định đoạn mã CSS tương ứng được bao trong phần từ { đến } sẽ được áp dụng cho những thành phần nào đó trong trang web. Selector là các đối tượng như các thẻ *HTML (tag), Class* hay *Id* mà chúng ta sẽ áp dụng các thuộc tính trình bày. Property là thuộc tính trình bày trong đó mỗi một thuộc tính cần phải có một giá trị. Một thuộc tính và giá trị của nó được phân cách bởi dấu hai chấm. Hai cặp thuộc tính - giá trị được phân cách nhau bởi dấu chấm phẩy.

Ví dụ: `#top {padding-top:10px;}`

Trong ví dụ trên ta có thể thấy đối tượng được đặt tên là top và được ép cho cách mép trên 10px. Dấu "#" đứng trước tiên, đó là quy định của CSS để hiểu đối tượng được đánh dấu trên tập tin (X)HTML là một Id, có nghĩa là trên toàn bộ trang web thì đối tượng này là duy nhất.

Các cách viết tên Selector:

- Viết `Selector` theo tên `Tag, Class, Id`
- Viết tên `selector` theo phân cấp đối tượng.

Một lối viết tên `selector` nữa đó là dựa trên tên các thuộc tính có trong HTML. Ví dụ trong HTML ta có đoạn mã như như sau: `<input name="Search" type="Text" value="Key word">`. Để áp dụng thuộc tính CSS cho riêng ô tìm kiếm này chúng ta sẽ dùng selector `input[name="Search"]`.

Ngoài việc viết tên selector cụ thể, chúng ta cũng có thể dùng một `selector` đại diện như `*{ color:red }` sẽ tác động đến tất cả các thành phần có trên trang web làm cho chúng có text màu đỏ.

1.1.3. Cách sử dụng CSS

Đến đây, bạn đã tìm hiểu về cú pháp viết CSS, nhưng câu hỏi đặt ra là *CSS được đặt ở đâu trong tập tin (X)HTML?*

Chúng ta có ba cách khác nhau để nhúng CSS vào trong một tài liệu (X)HTML:

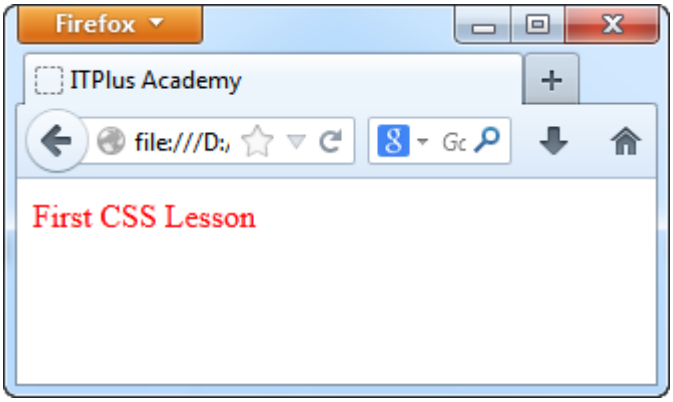
a. Nội tuyến (kiểu thuộc tính)

Đây là một phương pháp truyền thống để nhúng CSS vào một tài liệu HTML bằng cách nhúng vào từng thẻ HTML muốn áp dụng. Trường hợp này chúng ta sẽ không cần Selector trong cú pháp.

Ví dụ:

```
<body style="background-color:white;">
    <p style="color:red">First CSS Lesson</p>
</body>
```

Chạy đoạn mã HTML trên trình duyệt ta được kết quả là trang web có màu nền xanh lá cây và câu *"First CSS Lesson"* có màu đỏ.



Hình 6.1. Kết quả áp dụng CSS nội tuyến

- Nhược điểm: Không thể áp dụng cho nhiều thẻ HTML khác nhau.

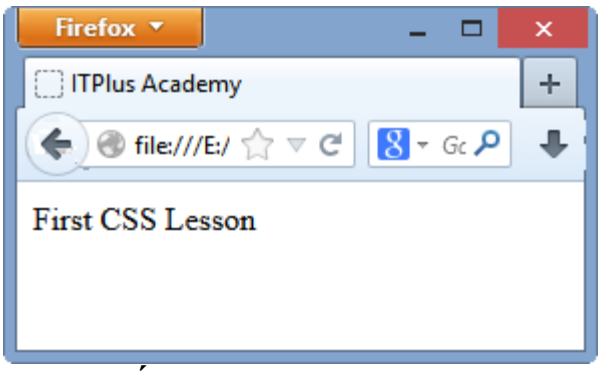
b. Bên trong thẻ style

Với cách này, thay vì viết thuộc tính CSS cho mỗi thẻ HTML thì ta gom tất cả các thuộc tính CSS vào trong thẻ style để giúp cho việc thay đổi mã CSS được dễ dàng hơn.

Cũng ví dụ làm trang web có màu nền trắng, dòng chữ màu đen, chúng ta có đoạn mã như sau:

```
<html>
    <head>
        <meta charset="utf-8">
        <title>ITPlus Academy</title>
                <style type="text/css">
                        body { background-color:white; }
                        p { color:black; }
                </style>
    </head>
    <body>
        <p> First CSS Lesson <p>
    </body>
</html>
```

Kết quả như sau:



Hình 6.2. Kết quả áp dụng CSS bên trong thẻ Style

Lưu ý: Thẻ style nên đặt trong thẻ `head`.

Đối với những trình duyệt cũ, không thể nhận ra thẻ `<style>`. Theo mặc định, thì khi một trình duyệt không nhận ra một thẻ thì nó sẽ hiện ra phần nội dung chứa trong thẻ. Như ở ví dụ trên, nếu trình duyệt không hỗ trợ thẻ style thì 2 dòng CSS:

`body {background-color:black } p { color:red }` sẽ hiện ra trên trình duyệt. Để tránh tình trạng này, bạn nên đưa vào thêm dấu `<!--` ở trước và `-->` ở sau khối code CSS. Như ví dụ trên sẽ viết lại là:

```
<style type="text/css">
  <!-- body { background-color:black }
  p { color:red } -->
</style>
```

c. Ngoại tuyến (Tập tin CSS sẽ được đặt bên ngoài tập tin HTML)

Tương tự như cách 2 nhưng thay vì đặt tất cả các mã CSS trong thẻ style chúng ta sẽ đưa chúng vào trong một tập tin CSS (có phần mở rộng .css) bên ngoài và liên kết nó vào trang web bằng thuộc tính `href` trong thẻ `link`. Đây là cách đặt CSS thường được dùng nhất và tốt nhất vì nó đặc biệt hữu ích cho việc đồng bộ hay bảo trì một website lớn sử dụng cùng một kiểu mẫu.

Ví dụ:

Đầu tiên, tạo ra một tập tin vidu2.html có nội dung như sau:

```
<html>
    <head>
        <meta charset="utf-8">
        <title>ITPlus Academy</title>
        <link rel="stylesheet" type="text/css" href="style.css" />
        <!--dòng trên để liên kết tới tập tin CSS-->
    </head>
    <body>
        <p> First CSS Lesson <p>
    </body>
</html>
```

Sau đó hãy tạo một tập tin style.css với nội dung:

```
body {
   background-color:black;
}
p{
   color:red;
}
```

Lưu ý**:** Trong CSS chúng ta còn có thể sử dụng thuộc tính `@import` để nhập một tập tin CSS vào tập tin CSS hiện hành với cú pháp: `@import url(link)` và trong tập tin HTML ta không cần gọi tập tin CSS đã được gọi trong theo `@import url(link)` nữa.

Ví dụ ta có 2 tập tin CSS và một tập tin HTML như sau:

Tập tin CSS thứ nhất style1.css:

```
#main
{
        width:300px;
        height:100px;
        margin:0px;
        padding:0px;
        float:left;
        background-color:#CCFFFF;
}
#main-left
{
        color:#F00;
        text-align:center;
}
```

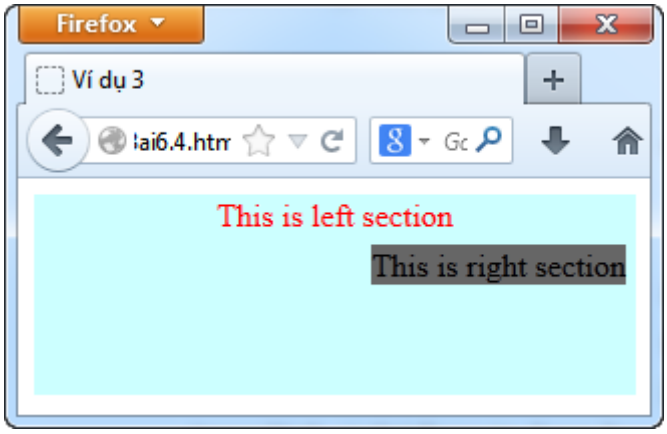
Tập tin CSS thứ 2 style2.css:

```
@import "style.css";

#main-right
{
   margin:5px;
   background-color:#666;
   float:right;
}
```

Tập tin HTML vidu3.html:

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Ví dụ 3</title>
<link href="style2.css" rel="stylesheet" type="text/css" />
</head>
<body>
<div id="main">
        <div id="main-left">This is left section</div>
        <div id="main-right">This is right section</div>
</div>
</body>
</html>
```

Kết quả khi chạy đoạn mã HTML trên lên trình duyệt thì ta sẽ nhận được kết quả như sau:



Hình 6. 3. Sử dụng @import trong CSS

d. Độ ưu tiên áp dụng mã CSS của trình duyệt

Trước khi thực thi CSS cho một trang web, trình duyệt sẽ đọc toàn bộ CSS mà trang web có thể được áp dụng, bao gồm: CSS mặc định của trình duyệt, tập tin CSS bên ngoài liên kết vào trang web, CSS nhúng trong thẻ `<style>` và các CSS nội tuyến. Sau đó, trình duyệt sẽ tổng hợp toàn bộ CSS này vào một CSS ảo, và nếu có các thuộc tính CSS giống nhau thì thuộc tính CSS nào nằm sau sẽ được ưu tiên sử dụng. Theo nguyên tắc đó thì độ ưu tiên của trình duyệt đối với mã CSS như sau: **CSS nội tuyến > CSS bên trong > CSS bên ngoài > CSS mặc định của trình duyệt**.

Ví dụ trong một trang web có liên kết tới tập tin style.css có nội dung như sau:

```
p {
    color: #333;
    text-align: left;
    width: 500px;
}
```

Trong thẻ `<style>` giữa thẻ `<head>` cũng có một đoạn CSS liên quan:

```
p {
    background-color: #ccc;
    text-align: right;
    width: 350px;
    height: 500px;
}
```

Trong phần nội dung trang web đó cũng có sử dụng CSS nội tuyến:

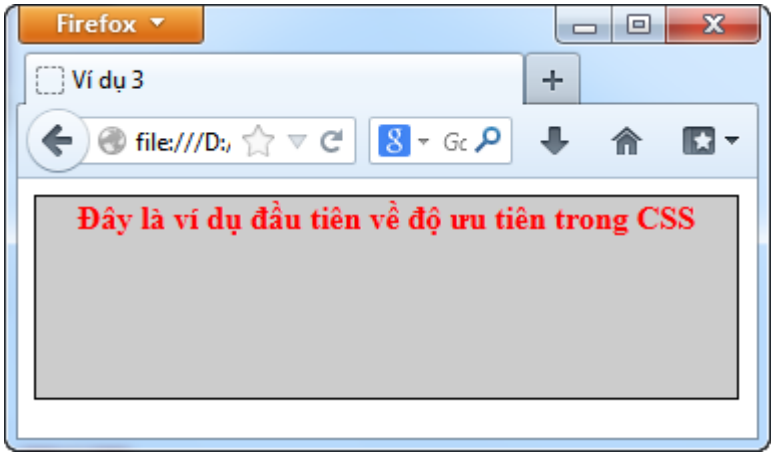
```
<p style="height:100px; text-align:center; border:1px solid #000;
color:#F00;font-weight:bold">
Đây là ví dụ đầu tiên về độ ưu tiên trong CSS
</p>
```

Khi thực thi CSS trình duyệt sẽ đọc hết tất cả các nguồn chứa style rồi sẽ tổng hợp lại vào một CSS ảo và nếu có sự trùng lắp các thuộc tính CSS thì nó sẽ lấy thuộc tính

CSS có mức ưu tiên cao hơn. Như ví dụ trên chúng ta sẽ thấy mã CSS cuối cùng mà phần tử p nhận được là:

```
p {
    background-color: #ccc;
    width: 100%;
    height: 100px;
    text-align: center;
    border: 1px solid #000;
    color: #F00;
    font-weight:bold
}
```

Kết quả hiển thị trên trình duyệt:



Hình 6. 4. Ví dụ về độ ưu tiên trong CSS

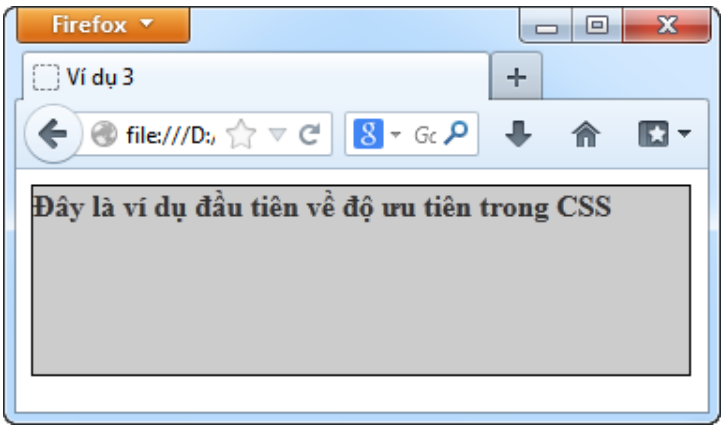
Thứ tự về độ ưu tiên trên là mặc định của trình duyệt khi đọc mã CSS. Tuy nhiên nếu muốn thay đổi độ ưu tiên cho một thuộc tính nào đó thì ta sử dụng thuộc tính `!important`. Chỉ cần đặt thuộc tính này sau một thuộc tính nào đó theo cú pháp `selector { property:value !important }` thì trình duyệt sẽ hiểu đây là một thuộc tính được ưu tiên. Bây giờ, chúng ta cùng xét lại ví dụ trên nhưng có đặt thêm thuộc tính `!important` vào sau các thuộc tính để xem kết quả.

```
p {
    color: #333 !important;
    text-align: left !important;
    width: 500px;
}
```

Phần CSS sẽ tác động lên thuộc tính p là:

```
p {
    background-color: #ccc;
    width: 100%;
    height: 100px;
    text-align: left !important;
    border: 1px solid #000;
    color: #333 !important;
    font-weight:bold
}
```

Kết quả khi thêm thuộc tính !important**:**



Hình 6. 5. Kết quả khi thêm thuộc tính !important

Lưu ý: Cùng một thuộc tính cho một `selector` thì nếu cả hai thuộc tính đều đặt `!important` thì giá trị thứ hai sẽ được áp dụng.

e. Chú thích trong CSS

Trong CSS ta sử dụng `/* */` (tương tự như trong C, C++) để ghi chú hay bỏ qua một lệnh nào đó mà ta không muốn thực hiện và đoạn chú thích sẽ được trình duyệt bỏ qua.

Ví dụ:

```
body {
    background: #ffffff;
    /* trang web sẽ có nền màu trắng */
    font-family: Verdana;
    /* font chữ mặc định là Verdana */
    color: #ff0000;
    /* màu chữ mặc định
    là màu đỏ */
}
```

1.1.4. Đơn vị tính

CSS hỗ trợ các đơn vị tính gồm đơn vị tính chiều dài, đơn vị đo góc, thời gian, cường độ âm thanh và màu sắc. Tuy nhiên, đơn vị tính chiều dài và màu sắc là 2 loại thường dùng nhất. Dưới đây là bảng liệt kê các đơn vị tính theo chiều dài và màu sắc.

Bảng liệt kê các đơn vị chiều dài:

STT	Đơn vị	Mô tả
1	%	Phần trăm
2	In	Inch (1 inch = 2.54 cm)
3	Cm	Centimet
4	Mm	Milimet

STT	Đơn vị	Mô tả
5	`Em`	1 em tương đương kích thước font hiện hành, nếu font hiện hành có kích cỡ 14px thì 1 em = 14 px. Đây là một đơn vị rất hữu ích trong việc hiển thị trang web.
6	`Pt`	Point (1 pt = 1/72 inch)
7	`Ex`	1 ex bằng chiều cao của chữ x in thường của font hiện hành. Do đó, đơn vị này không những phụ thuộc trên kích cỡ font chữ mà còn phụ thuộc loại font chữ vì cùng 1 cỡ 14px nhưng chiều cao chữ x của font Times New Roman và font Tahoma là khác nhau.
8	`Pc`	Pica (1 pc = 12 pt)
9	`Px`	Pixels (điểm ảnh trên màn hình máy tính)

Bảng 6.1. Bảng liệt kê các đơn vị chiều dài

Bảng liệt kê các đơn vị màu sắc:

STT	Đơn vị	Màu sắc
1	`Color-name`	Tên màu theo Tiếng Anh, Ví dụ black, white, green, red, blue…
2	`RGB (r, g, b)`	Màu RGB với 3 giá trị R, G, B có giá trị từ 0 đến 255 kết hợp với nhau tạo ra vô số màu.
3	`RGB (%r, %g, %b)`	Màu RGB với 3 giá trị R, G, B có giá trị từ 0% đến 100% kết hợp với nhau.
4	`Hexadecimal RGB`	Mã màu RGB dạng hệ thập lục, Ví dụ #000000 - đen, #FFFFFF - trắng…

Bảng 6.2. Bảng liệt kê các đơn vị màu sắc

1.2. Selector trong CSS

1.2.1. Bộ chọn toàn cục (Universal Selector)

Bộ chọn toàn cục tên tiếng Anh là *Universal Selector*, ký hiệu là `*`. Đây là `selector` cơ bản nhất, nó có tác dụng chọn tất cả những phần tử trên trang, chúng ta thường sử dụng bộ chọn này để canh lề cho trang (*margin và padding*), tuy nhiên, kĩ thuật này lại làm cho trình duyệt trở nên chậm chạp hơn, vì vậy nên tránh lạm dụng kỹ thuật này.

Ví dụ:

```
* {
margin: 0;
padding: 0;
}
```

Dấu * cũng có thể áp dụng với những phần tử con của một phần tử, như ví dụ dưới đây, bộ chọn sẽ chọn hết tất cả các phần tử con của div container:

```
#container * {
    border: 1px solid black;
}
```

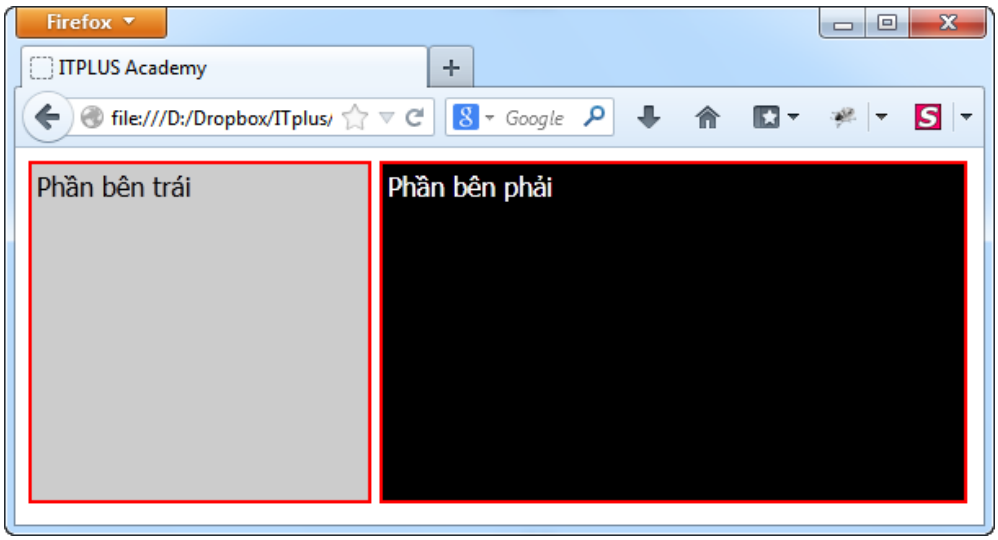
Ví dụ ta có tập tin CSS có tên là style.css chứa đoạn mã trên như sau:

```
#container * {
    border: 2px solid red;
    font-family: tahoma;
    padding: 3px;
}

#left {
    width: 200px;
    height: 200px;
    float: left;
    background-color: #CCCCCC;
}

#mid {
    width: 350px;
    height: 200px;
    float: left;
    background-color: #000000;
    margin-left: 5px;
    color: #FFFFFF;
}
```

Cùng với đoạn mã trong phần body của tập tin HTML như sau:

```
<div id="container">
        <div id="left">Phần bên trái</div>
        <div id="mid">Phần bên phải</div>
</div>
```

Khi chạy tập tin HTML trên trình duyệt thì ta sẽ nhận được kết quả như sau:



Hình 6. 6. Minh họa cách dùng Universal Selector

1.2.2. Bộ chọn phân cấp (Descendant Selector)

Bộ chọn descendant là bộ chọn có cú pháp X Y (dấu cách ở giữa X và Y), nó cho phép bạn có thể áp dụng cho 1 nhóm các phần tử cùng loại trong một phần tử cha, như ví dụ trên, chúng ta áp dụng cho tất cả các thẻ `<a>` trong một unordered list, và các thẻ ngoài unordered list này, sẽ không có hiệu lực.

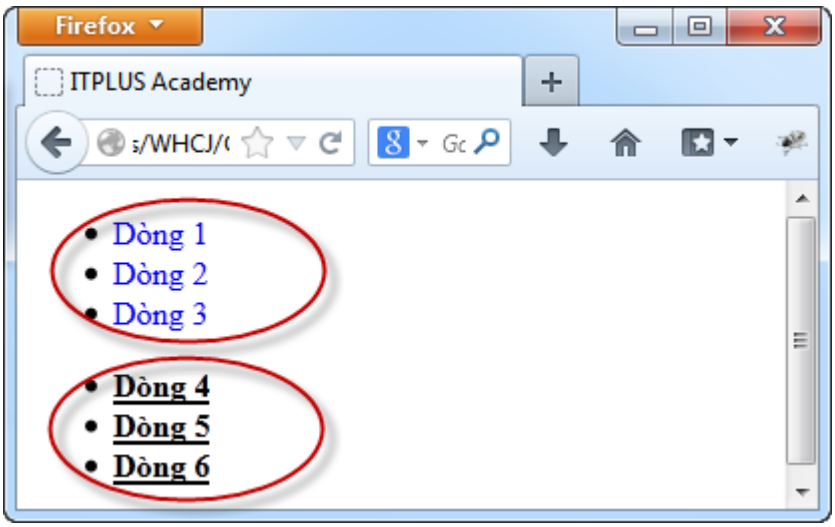
Ví dụ giả sử ta có tập tin HTML là *index.html* với khối mã trong thẻ body như sau:

```
<div id="top">
      <ul>
          <li><a href="#" id="1">Dòng 1</a></li>
          <li><a href="#" id="2">Dòng 2</a></li>
          <li><a href="#" id="3">Dòng 3</a></li>
      </ul>
</div>
<div id="center">
      <ul>
          <li><a href="#" id="4">Dòng 4</a></li>
          <li><a href="#" id="5">Dòng 5</a></li>
          <li><a href="#" id="6">Dòng 6</a></li>
      </ul>
</div>
```

Và một tập tin CSS có tên style.css như sau:

```
#top li a {
    text-decoration: none;
}

#center li a {
    text-decoration: underline;
    color: #000000;
    font-weight: bold;
}
```

Khi chạy trên trình duyệt tập tin index.html ta nhận được kết quả như sau:



Hình 6. 7. Minh họa cách dùng Descendant Selector

1.2.3. Bộ chọn ID

Sử dụng cách này để chọn một Id, và bạn nên nhớ rằng, Id là duy nhất, và bạn không thể sử dụng lại khai báo này nữa.

Ví dụ:

```
#container {
    width: 960px;
    margin: auto;
}
```

1.2.4. Bộ chọn Class

Ở phần này, chúng ta sẽ được tìm hiểu cụ thể hơn về tác dụng của Class và ID trong CSS. Trước tiên ta hãy xem qua cách sử dụng hai thẻ Id và Class trong một đoạn HTML.

```
<div id="banner">
    <!-- thẻ div chứa banner của trang web -->
</div>
<div id=”container”>
    <!-- thẻ div chứa các bài viết của trang web -->
    <div class="article" id=”featured”>
        <!-- thẻ div chứa bài viết nổi bật-->
    </div>
    <div class="article">
        <!-- thẻ div chứa bài viết thường -->
    </div>
    <div class="article">
        <!-- thẻ div chứa bài viết thường -->
    </div>
</div>
```

Qua hai đoạn HTML chúng ta có thể dễ dàng nhận thấy Id được sử dụng cho các thành phần mang tính tuyệt đối và duy nhất trong toàn bộ trang web. Nó thường được gán cho *heading, container, leftcol, rightcol, content-warpper, footer* ...

Còn với Class sẽ được gán cho các thành phần sẽ lặp lại nhiều lần như *bài viết, module, content title, readon, ul li* ... bởi khi gán Class cho nhiều thành phần của trang

web thì bạn chỉ việc khai báo style trên tập tin CSS cho các thành phần này một lần duy nhất.

Ví dụ sau thể hiện cách khai báo thuộc tính cho Id và Class trong tập tin CSS:

```
#container {
    width: 80%;
    margin: auto;
    padding: 20px;
    border: 1px solid #666;
    background: #ffffff;
}

.article {
    font-size: 13pt;
    color: #008080;
    font-weight: bold;
}

#featured {
    font-size: 15pt !important;
}
```

1.2.5. Bộ chọn Type Selector

Type selector là các thẻ X(HTML) thông thường như các thẻ h, ul, p, em, li, a…với cú pháp bộ chọn: X (,Y,Z,…) {attribute: value;…}.

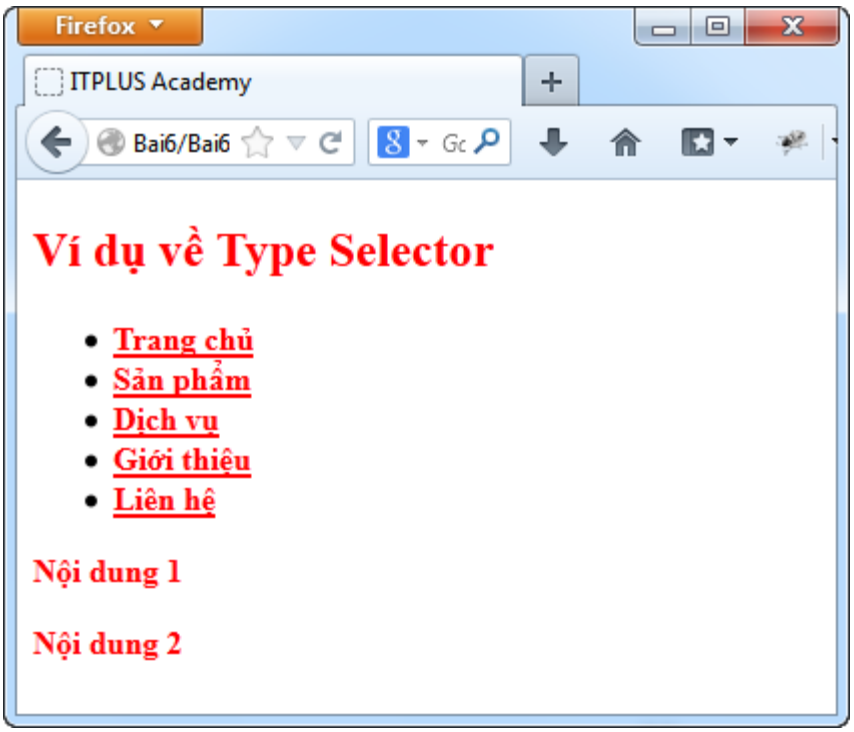
Trong việc viết mã HTML, không phải lúc nào ta cũng dùng Id hay Class cho tất cả các thành phần trên trang web, nếu muốn áp dụng cho một tập các thẻ nào đó thì cách tốt nhất là dùng type selector.

Ví dụ, ta có đoạn mã HTML trong thẻ body như sau:

```
<div id="container">
    <h2>Ví dụ về Type Selector</h2>
    <ul>
        <li><a href="#" id="1">Trang chủ</a></li>
        <li><a href="#" id="2">Sản phẩm</a></li>
        <li><a href="#" id="3">Dịch vụ</a></li>
        <li><a href="#" id="4">Giới thiệu</a></li>
        <li><a href="#" id="5">Liên hệ</a></li>
    </ul>
    <p>Nội dung 1</p>
    <p>Nội dung 2</p>
</div>
```

Bây giờ ta muốn thiết lập màu chữ của các thẻ `h2, a, p` cùng màu đỏ, kiểu chữ đậm, ta thêm dòng mã CSS: `h2, a, p { color:red; font-weight:bold; }`

Kết quả hiển thị trên trình duyệt:



Hình 6.8. Minh họa cách dùng Type Selector

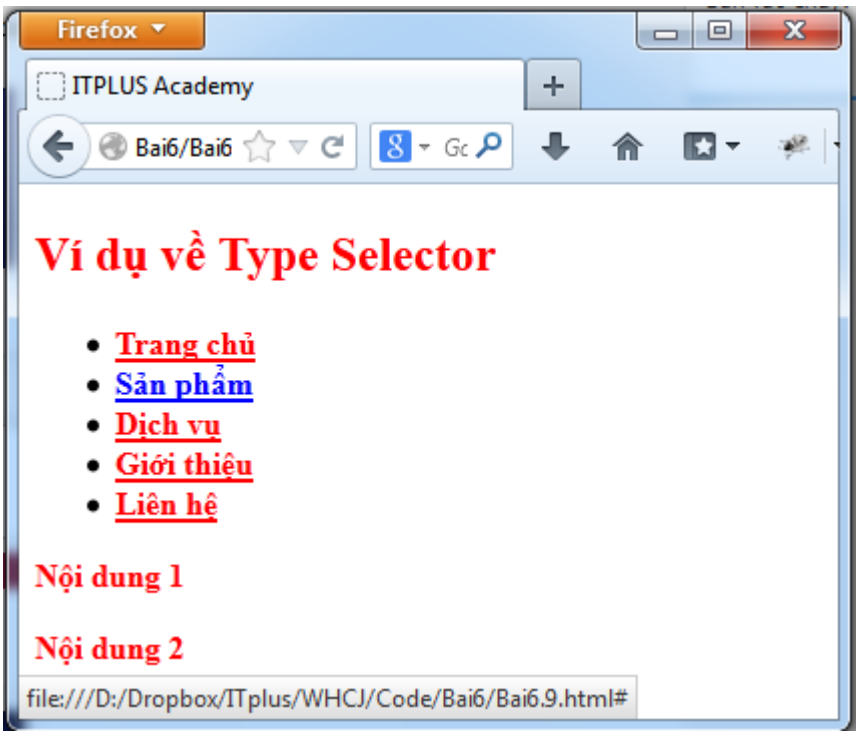
1.2.6. Bộ chọn Pseudo (Classes Selector)

Là bộ chọn có cú pháp: X:Y, trong đó: X là các thẻ (tag) như thẻ a, thẻ p…và Y là các thuộc tính như `link, visited, hover,…`

Ví dụ:

```
a:link {
    color: #F00;
}

a:visited {
    color: #9F0;
}

a:hover {
    color: #00F;
}
```

Khi ta rê chuột qua chữ *Sản phẩm* kết quả của đoạn mã trên khi chạy tập tin HTML trên trình duyệt như sau:



Hình 6.9. Minh họa cách dùng Pseudo-classes Selector

1.2.7. Bộ chọn liền kề (Adjacent Sibling Selector)

Adjacent sibling selectors có dạng X + Y hoặc X ~ Y, trong đó X và Y là 2 thẻ HTML và X, Y phải là 2 thẻ cùng cấp quan hệ.

Trường hợp X + Y thì nó sẽ chỉ chọn phần tử nào đứng ngay sau thẻ X.

Ví dụ, ta có đoạn mã phần body trong tập tin HTML như sau:

```
<div id="container">
    <ul>
        <li><a href="#" id="1">Trang chủ</a></li>
        <li><a href="#" id="2">Sản phẩm</a></li>
        <li><a href="#" id="3">Dịch vụ</a></li>
        <li><a href="#" id="4">Giới thiệu</a></li>
        <li><a href="#" id="5">Liên hệ</a></li>
    </ul>
    <p>Nội dung 1</p>
    <p>Nội dung 2</p>
</div>
```

Bây giờ ta muốn chỉ thẻ `<p>` nằm ngay sau thẻ `<ul>` mới có màu xám, ta viết mã CSS như sau:

```
ul + p {
    color: #999999;
    font-weight: bold;
    font-style: italic
}
```

Dưới đây là hình ảnh minh họa kết quả của việc áp dụng Selector này:

Hình 6.10. Minh họa cách dùng Adjacent sibling Selector

Trường hợp X ~ Y thì nó sẽ chọn tất nhửng những thẻ Y đứng sau thẻ X

Ví dụ

`ul ~ p {color: blue;}` nghĩa là bộ chọn sẽ áp dụng màu xanh da trời cho tất cả các thẻ p đứng sau thẻ ul

1.2.8. Bộ chọn Child Selector

Là bộ chọn có cú pháp là X > Y. Sự khác biệt giữa X Y và X > Y là X > Y sẽ chỉ chọn những phần tử là con trực tiếp của nó, còn X Y là chọn tất cả các phần tử Y là con cháu của X.

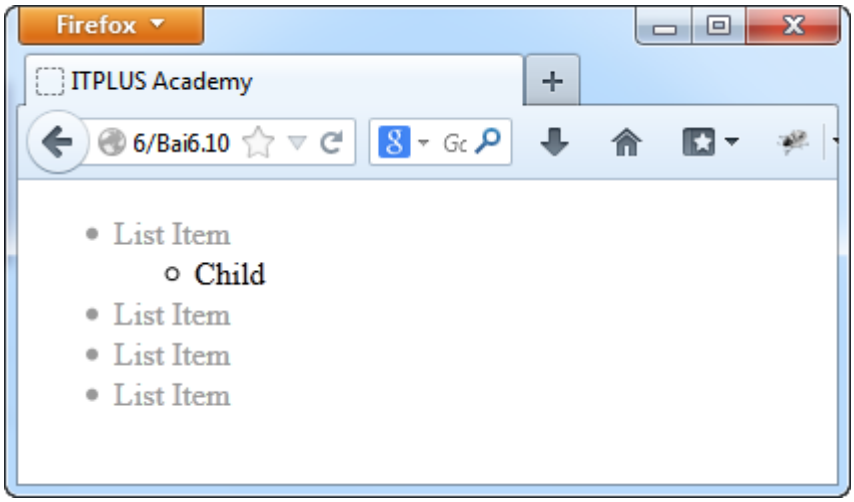
Ví dụ với tập tin HTML có đoạn mã sau:

```
<div id="container">
    <ul>
        <li>List Item
            <ul>
                <li> Child</li>
            </ul>
        <li>List Item</li>
        <li>List Item</li>
        <li>List Item</li>
    </ul>
</div>
```

Bộ chọn `#container > ul` sẽ chỉ chọn thẻ `<ul>` là con của nó, còn thẻ `<ul>` cháu của nó sẽ không được chọn. Mã CSS:

```
ul { color: #000000;}
#container > ul { color: #999999;}
```

Và đây là kết quả hiển thị trên trình duyệt khi chạy tập tin HTML

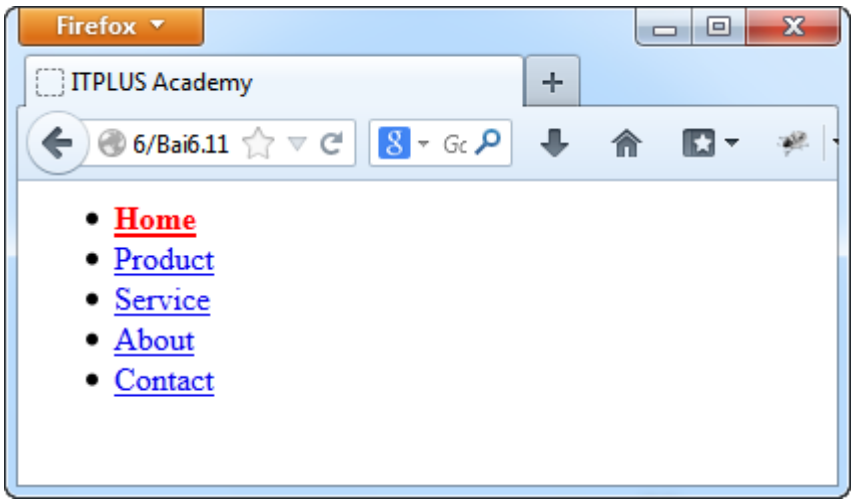


Hình 6.11. Minh họa cách dùng Child Selector

1.2.9. Bộ chọn theo thuộc tính (Attribute Selector)

Là bộ chọn có cú pháp X[Y] trong đó X là các thẻ, Y là thuộc tính của thẻ. Với ví dụ dưới đây, bộ chọn dạng `a[title] {color: red; font-weight:bold}` chỉ chọn những thẻ `<a>` có thuộc tính title.

```
<ul>
    <li><a href="#" title="Go to Homepage">Home</a></li>
    <li><a href="#">Product</a></li>
    <li><a href="#">Service</a></li>
    <li><a href="#">About</a></li>
    <li><a href="#">Contact</a></li>
</ul>
```



Hình 6.12. Minh họa cách dùng Attribute Selector

Chú ý: Do CSS phân biệt hoa thường trong việc viết tên *Selector* nên *Selector* đặc biệt là Id và Class nên được viết chữ in thường nhằm thống nhất, tránh nhầm lẫn.

1.3. Các thuộc tính cơ bản của CSS

1.3.1. Nhóm các thuộc tính background

Nhóm thuộc tính Background cho phép xác định màu nền hoặc ảnh nền cho trang web với 5 thuộc tính: background-color, background-image, background-repeat, background-attachment, background-position. Sau đây, chúng ta sẽ đi tìm hiểu từng thuộc tính trong nhóm 5 thuộc tính này.

a. Background-color

Thuộc tính background-color giúp xác định màu nền cho một thành phần trên trang web. Các giá trị mã màu của background-color cũng giống như color nhưng có thêm giá trị transparent để tạo nền trong suốt.

Ví dụ :

```
body {
    background-color: cyan;
}
h1 {
    background-color: red;
}
h2 {
    background-color: orange;
}
```

b. Background-image

Việc sử dụng ảnh nền với thuộc tính Background-image giúp trang web trông sinh động và bắt mắt hơn. Để chèn ảnh nền vào một thành phần trên trang web chúng ta sử dụng thuộc tính background-image.

Ví dụ:

```
body {
    background-image: url(logo.png);
}
```

Ta sẽ phải chỉ định đường dẫn của ảnh trong cặp ngoặc đơn sau url. Do ảnh đặt trong cùng thư mục với tập tin .css nên chúng ta chỉ cần ghi `logo.png`. Nhưng nếu chúng ta tạo thêm một thư mục img trong thư mục thì chúng ta sẽ phải ghi là `background-image:url(img/logo.png)`. Nếu không chắc lắm bạn có thể dùng đường dẫn tương đối cho ảnh.

c. Background-repeat

Thuộc tính `background-repeat` cung cấp cho chúng ta các điều khiển giúp kiểm soát trình trạng lặp lại của ảnh nền. Thuộc tính này có 4 giá trị:

- `repeat-x`: Chỉ lặp lại ảnh theo phương ngang.

- `repeat-y`: Chỉ lặp lại ảnh theo phương dọc.

- `repeat`: Lặp lại ảnh theo cả 2 phương, đây là giá trị mặc định.

- `no-repeat`: Không lặp lại ảnh.

Lưu ý**:** Thuộc tính này rất hay dùng để thiết kế banner hoặc thiết kế hình khối menu cho website

d. Background-attachment (Khóa ảnh nền)

`Background-attachment` là một thuộc tính cho phép bạn xác định tính cố định của ảnh nền so với với nội dung trang web. Thuộc tính này có 2 giá trị:

- `scroll`: Ảnh nền sẽ cuộn cùng nội dung trang web, đây là giá trị mặc định.

- `fixed`: Cố định ảnh nền so với nội dung trang web. Khi áp dụng giá trị này, ảnh nền sẽ đứng yên khi bạn đang cuộn trang web.

e. Background-position (định vị ảnh nền)

Theo mặc định, ảnh nền khi được chèn sẽ nằm ở góc trên, bên trái màn hình. Tuy nhiên với thuộc tính `background-position` bạn sẽ có thể đặt ảnh nền ở bất cứ vị trí nào (trong không gian của thành phần mà nó làm nền).

`Background-position` sẽ dùng một cặp 2 giá trị để biểu diễn tọa độ đặt ảnh nền. Có khá nhiều kiểu giá trị cho thuộc tính `position`. Như đơn vị chính xác như `centimeters, pixels, inches,`… hay các đơn vị qui đổi như %, hoặc các vị trí đặt biệt như `top, bottom, left, right.`

STT	Giá trị	Ý nghĩa
1	Background-position:5cm 2cm	Ảnh được định vị 5cm từ trái qua và 2cm từ trên xuống
2	Background-position:20% 30%	Ảnh được định vị 20% từ trái qua và 30% từ trên xuống
3	Background-position:bottom left	Ảnh được định vị ở góc trái phía dưới

Bảng 6. 3. Bảng giá trị background-position

Thuộc tính background rút gọn: Khi sử dụng quá nhiều thuộc tính CSS sẽ gây khó khăn cho người đọc, việc chỉnh sửa cũng tốn nhiều dung lượng ổ cứng cho nên CSS đưa ra một cấu trúc rút gọn cho các thuộc tính cùng nhóm.

Chúng ta có thể nhóm lại đoạn CSS sau:

```
background-color:transparent;
background-image: url(logo.png);
background-repeat: no-repeat;
background-attachment: fixed;
background-position: right bottom;
```

Thành một dòng ngắn gọn:

```
background:transparent url(logo.png) no-repeat fixed right bottom;
```

1.3.2. Nhóm các thuộc tính border

Border là một thành phần quan trọng trong một trang web. Nó thường được dùng trong trang trí, đóng khung cho một đối tượng cần nhấn mạnh, phân cách các đối tượng giúp trang web trông dễ nhìn hơn,… Chúng ta sẽ tìm hiểu về cách dùng CSS để định `border` cho các đối tượng web với 3 thuộc tính `border` là `border-width, border-color, border style.`

a. Border-width

Border-width là một thuộc tính CSS quy định độ rộng cho viền của một đối tượng web. Thuộc tính này có các giá trị: `thin` (mảnh), `medium` (vừa), `thick` (dày), hay là một giá trị đo cụ thể như `pixels`.

b. Border-color

`Border-color` là thuộc tính CSS quy định màu viền cho một đối tượng web. thuộc tính này nhận tất cả đơn vị màu CSS hỗ trợ.

c. Border-style

Border-style là thuộc tính CSS quy định kiểu viền thể hiện của một đối tượng web. CSS cung cấp tất cả 8 kiểu viền tương ứng với 8 giá trị: `dotted, dashed, solid, double, groove, ridge, inset và outset`. Ngoài ra, hai giá trị `none` hay `hidden` dùng để ẩn đường viền.

Ví dụ:

```
h1 {
    border-width: thin;
    border-color: #FF0000;
    border-style: solid
}
```

Thuộc tính border rút gọn:

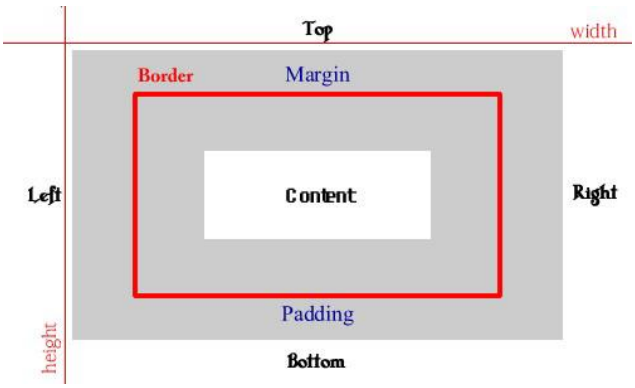
Để dễ nhớ chúng ta có thể sử dụng cấu trúc rút gọn của CSS như sau:

`Border:<border-width> |<border-color> |<border-style>`

1.3.3. Box Model, Margin và Padding

a. Box Model

Trong CSS, box model (mô hình hộp) mô tả cách mà CSS định dạng khối không gian bao quanh một thành phần. Nó bao gồm padding (vùng đệm), border (viền) và margin (canh lề) và các tùy chọn. Hình bên dưới mô tả cấu trúc minh họa mô hình hộp cho một thành phần web.



Hình 6. 13. Hình minh họa Box Model trong CSS

Chúng ta sẽ đi xét mô hình hộp của một đối tượng web cụ thể:

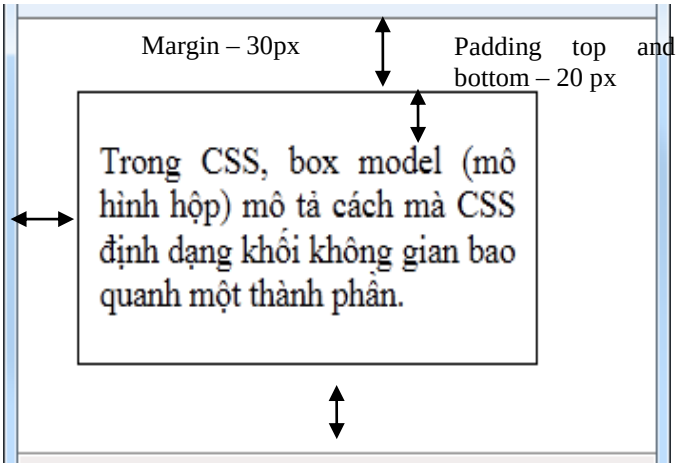
Ví dụ, chúng ta có một đoạn HTML sau:

```
<p>
Trong CSS, box model (mô hình hộp) mô tả cách mà CSS định dạng khối không
gian bao quanh một thành phần.
</p>
```

Phần CSS cho đoạn HTML trên:

```
p {
    width: 200px;
    margin: 30px 20px;
    padding: 20px 10px;
    border: 1px solid #000;
    text-align: justify
}
```

Với ví dụ trên chúng ta sẽ khái quát được mô hình hộp như sau:



Hình 6. 14. Khái quát Box Model

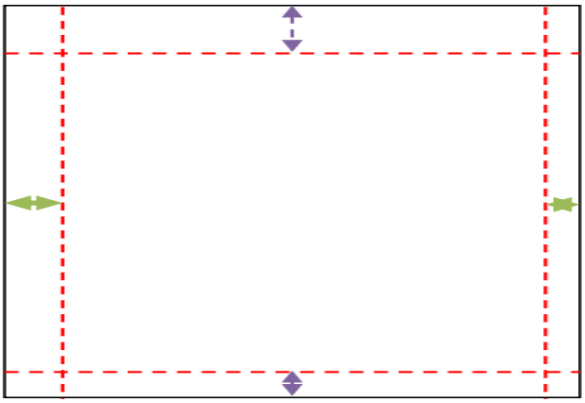
b. Margin

Margin là thuộc tính canh lề có vai trò tương tự như margin trong Microsoft Word, dùng để canh chỉnh lề cho cả trang web, cho một thành phần nào đó đối với các thành phần khác hoặc canh lề đối với viền trang.

Ví dụ ta có đoạn css sau:

```
div {
    margin-top: 80px;
    margin-bottom: 40px;
    margin-left: 50px;
    margin-right: 30px;
    border: 1px dotted #FF0000;
}
```

Đoạn mã trên được mô hình hóa như sau:



Hình 6. 15. Mô hình hóa margin

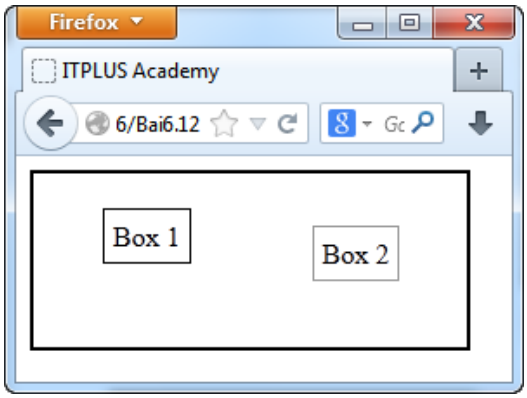
Lưu ý: Ta có thể viết gọn lại theo nguyên tắc **Top-Right-Bottom-Left**, tức là: **margin:<margin-top>|<margin-right>|<margin-bottom>|<margin-left>**

Ví dụ:

```
body {
    margin: 80px 30px 40px 50px;
    border: 1px dotted #FF0000;
}
```

Ví dụ dùng thuộc tính margin để canh lề cho các đối tượng trong trang web:

```
#container {
    width: 250px;
    height: 100px;
    float: left;
    margin: 0px;
    padding: 0px;
    border: 2px solid #000000;
}
#box1 {
    margin: 20px 30px 20px 40px;
    border: 1px solid #000000;
    float: left;
    padding: 5px;
}
#box2 {
    margin: 30px 30px 20px 40px;
    border: 1px solid #999999;
    float: left;
    padding: 5px;
}
```



Hình 6. 16. Ví dụ minh họa sử dụng margin

c. Padding

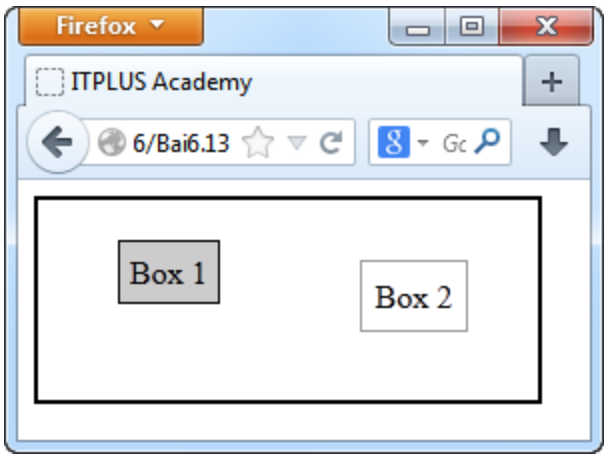
Padding có thể hiểu như là một thuộc tính đệm. Padding không ảnh hưởng tới khoảng các giữa các các đối tượng như margin mà nó chỉ quy định khoảng cách giữa phần nội dung và viền của một đối tượng (xem lại ảnh minh họa về boxmodel)

Về cú pháp, tương tự thuộc tính margin:

Padding:<padding-top> | <padding-right> | <padding-bottom> | <padding-left>

Ví dụ cũng là đoạn mã ở ví dụ trên, ta thêm `background-color` cho `box 1` và thêm thuộc tính `padding` vào `box 2`, ta thấy phần text trong `box 2` bị đẩy xuống một khoảng theo padding:

```
#box1 {
    margin: 20px 30px 20px 40px;
    border: 1px solid #000000;
    float: left;
    background: #CCCCCC;
    padding: 5px;
}

#box2 {
    margin: 30px 30px 20px 40px;
    border: 1px solid #999999;
    float: left;
    padding: 10px;
    background: #CCCCCC;
}
```



Hình 6. 17. Ví dụ minh họa sử dụng padding

1.3.4. Nhóm các thuộc tình font

Nhóm thuộc tính font thuộc nhóm thuộc tính định dạng văn bản bao gồm 5 thuộc tính: font-family, font-style, font-variant, font-weight, font-size cho phép xác định kiểu chữ, độ cao chữ và kích thước chữ.

a. Font-family

Thuộc tính font-family định nghĩa một danh sách ưu tiên các font sẽ được dùng để hiển thị một thành phần trang web. Theo đó, thì font đầu tiên được liệt kê trong danh sách sẽ được dùng để hiển thị trang web. Nếu như trên máy tính truy cập chưa cài đặt font này thì font thứ hai trong danh sách sẽ được ưu tiên…cho đến khi có một font phù hợp.

Có hai loại tên font được dùng để chỉ định trong `font-family`: `family-names` và `generic families`.

- `Family-names`: Tên cụ thể của một font, Ví dụ Arial, Verdana, Tohama,…
- `Generic families`: Tên của một họ gồm nhiều font, Ví dụ sans-serif, serif,…

Khi lên danh sách font dùng để hiển thị một trang web bạn sẽ chọn những font mong muốn trang web sẽ được hiển thị để đặt ở các vị trí ưu tiên. Tuy nhiên, có thể những font này sẽ không thông dụng lắm nên bạn cũng cần chỉ định thêm một số font thông dụng dự phần như *Arial, Tohama hay Times New Roman* và bạn cũng được đề nghị đặt vào danh sách font của mình một `generic families` (thường thì nó sẽ có độ ưu tiên thấp nhất). Thực hiện theo cách này thì sẽ đảm bảo trang web của bạn có thể hiển thị tốt trên bất kỳ hệ thống nào.

Ví dụ ta muốn quy định font chữ dùng cho cả trang web là *Times New Roman, Tohama, sans-serif*, và font chữ dùng để hiển thị các tiêu đề h1, h2, h3 sẽ là Arial, Verdana và các font họ serif thì ta viết mã CSS như sau:

```
body {
    font-family: "Times New Roman", Tohama, sans-serif;
}
```

Chú ý**:** Đối với các font có khoảng trắng trong tên như Times New Roman cần được đặt trong dấu ngoặc kép.

b. Font-style

Thuộc tính font-style định nghĩa việc áp dụng các kiểu in thường (normal), in nghiêng (italic) hay xiên (oblique) lên các thành phần trang web. Trong ví dụ bên dưới chúng ta sẽ thử thực hiện áp dụng kiểu in nghiêng cho thành phần h1 và kiểu xiên cho h2.

```
h1 {
    font-style: italic;
}

h2 {
    font-style: oblique;
}
```

c. Font-variant

Thuộc tính font-variant được dùng để chọn giữa chế độ bình thường và small-caps của một font chữ.

Một font small-caps là một font sử dụng chữ in hoa có kích cỡ nhỏ hơn in hoa chuẩn để thay thế những chữ in thường. Nếu như font chữ dùng để hiển thị không có sẵn font small-caps thì trình duyệt sẽ hiện chữ in hoa để thay thế.

Trong ví dụ sau chúng ta sẽ sử dụng kiểu small-caps cho phần h1:

```
h1 {
     font-variant:small-caps ;
   }
```

d. Thuộc tính font-weight

Thuộc tính font-weight mô tả cách thức thể hiện của font chữ là ở dạng bình thường (normal) hay in đậm (bold). Ngoài ra, một số trình duyệt cũng hỗ trợ mô tả độ in đậm bằng các con số từ 100 – 900.

Thử in đậm phần p:

```
p {
    font-weight:bold;
  }
```

e. Font-size

Kích thước của một font được định bởi thuộc tính font-size. Thuộc tính này nhận các giá trị đơn vị đo hỗ trợ bởi CSS bên cạnh các giá trị xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger. Tùy theo mục đích sử dụng của website bạn có thể lựa chon những đơn vị phù hợp. Ví dụ trang web của bạn phục vụ chủ yếu là những người già, thị lực kém hay những người dùng sử dụng các màn hình máy tính kém

chất lượng thì bạn có thể cân nhắc sử dụng các đơn vị qui đổi như em hay %. Như vậy sẽ đảm bảo font chữ trên trang web của bạn luôn ở kích thước phù hợp.

Ở ví dụ sau trang web sẽ có kích cỡ font là 20px, h1 là 3em = 3 x 20 = 60px, h2 là 2em = 40px.

```
body {
    font-size: 20px;
}

h1 {
    font-size: 3em;
}

h2 {
    font-size: 2em;
}
```

Thuộc tính font rút gọn:

Tương tự như các thuộc tính background, chúng ta cũng có thể rút gọn các thuộc tính font lại thành một thuộc tính đơn như ví dụ sau:

```
h1 {
    font-style: italic;
    font-variant: small-caps;
    font-weight: bold;
    font-size: 35px;
    font-family: arial, verdana, sans-serif;
}
```

Thành:

```
h1 {
    font: italic bold 35px arial, verdana, sans-serif;
}
```

Cấu trúc rút gọn cho các thuộc tính nhóm font:

font :*<font-style>* | *< font-variant>* | *<font-weight>* | *<font-size>* |*< font-family>*

1.3.5. Width và Heigh

a. Width

Width là thuộc tính CSS dùng để quy định chiều rộng cho một thành phần web. Ví dụ sau chúng ta sẽ định chiều rộng cho thành phần p của một trang web.

```
p {
    width: 700px;
}
```

- Thuộc tính `max-width`: `Max-width` là thuộc tính CSS dùng để quy định chiều rộng tối đa cho một thành phần web.

- Thuộc tính `min-width`: `Min-width` là thuộc tính CSS dùng để quy định chiều rộng tối thiểu cho một thành phần web.

b. Height

Thuộc tính height là thuộc tính CSS dùng để quy định chiều cao cho một thành phần web. Ví dụ sau chúng ta sẽ định chiều cao cho thành phần p của một trang web.

```
p {
    height: 300px
}
```

- Thuộc tính `max-height`: `Max-height` là thuộc tính CSS dùng để quy định chiều cao tối đa cho một thành phần web.

- Thuộc tính `min-height`: `Min-height` là thuộc tính CSS dùng để quy định chiều cao tối thiểu cho một thành phần web.

<u>Lưu ý khi sử dụng Height</u>**:** Thông thường chiều cao của thành phần trang web/trang web do văn bản trong trang web đó quy định, giá trị của nó có thể là một con số cụ thể có thể là giá trị auto.

1.3.6. Nhóm các thuộc tính text

Nhóm thuộc tính Text cũng thuộc nhóm thuộc tính định dạng văn bản tương tự như Font bao gồm các thuộc tính color, text-align, text-indent, text-decoration, letter-spacing, text-transform.

a. Color

Để định màu chữ cho một thành phần nào đó trên trang web chúng ta sử dụng thuộc tính color. Giá trị của thuộc tính này là các giá trị màu CSS hỗ trợ.

Ví dụ sau chúng ta sẽ viết CSS để định màu chữ chung cho một trang web là đen, cho tiêu đề h1 màu xanh da trời, cho tiêu đề h2 màu xanh lá chúng ta sẽ làm như sau:

```
body {
    color: #000;
}

h1 {
    color: #0000FF;
}

h2 {
    color: #00FF00;
}
```

b. Text-align

Thuộc tính text-align giúp bạn thêm các canh chỉnh văn bản cho các thành phần trong trang web. Cũng tương tự như các lựa chọn canh chỉnh văn bản trong các trình soạn thảo văn bản thông dụng như MS Word, thuộc tính này có tất cả 4 giá trị : `left` (canh trái – mặc định), `right` (canh phải), `center` (canh giữa) và `justify` (canh đều).

Trong ví dụ sau chúng ta sẽ thực hiện canh phải các thành phần h1, h2 và canh đều đối với thành phần <p>:

```
h1, h2 {
    text-align: right
}

p {
    text-align: justify
}
```

c. Text-indent

Thuộc tính text-indent cung cấp khả năng tạo ra khoảng thụt đầu dòng cho dòng đầu tiên trong đoạn văn bản. Giá trị thuộc tính này là các đơn vị đo cơ bản dùng trong CSS.

Trong ví dụ sau chúng ta sẽ định dạng thụt đầu dòng một khoảng 30px cho dòng văn bản đầu tiên trong mỗi đoạn văn bản đối với các thành phần `<p>`:

```
p {
    text-indent:30px
  }
```

d. Text-decoration

Thuộc tính text-decoration giúp bạn thêm các hiệu ứng gạch chân `(underline)`, gạch xiên `(line-through)`, gạch đầu `(overline)`, và một hiệu ứng đặc biệt là văn bản nhấp nháy `(blink)`.

Ví dụ sau chúng ta sẽ định dạng gạch chân cho thành phần h1, gạch giữa cho thành phần h2 giống như: text và ~~text~~

```
h1 { text-decoration: underline;}
h2 { text-decoration: line-through;}
```

e. Letter-spacing

Thuộc tính letter-spacing được dùng để định khoảng cách giữa các ký tự trong một đoạn văn bản.

Muốn định khoảng cách giữa các ký tự trong thành phần h1, h2 là 7px và thành phần <p> là 5px chúng ta sẽ viết CSS sau:

```
h1, h2 {
    letter-spacing: 7px
}

p {
    letter-spacing: 5px
}
```

f. Text-transform

Text-transform là thuộc tính qui định chế độ in hoa hay in thường của văn bản mà không phụ thuộc vào văn bản gốc trên HTML. Thuộc tính này có tất cả 4 giá trị: `uppercase` (in hoa), `lowercase` (in thường), `capitalize` (in hoa ở ký tự đầu tiên trong mỗi từ) và `none` (không áp dụng hiệu ứng – mặc định).

Trong ví dụ dưới đây chúng ta sẽ định dạng cho thành phần h1 là in hoa, h2 là in hoa đầu mỗi ký tự.

```
h1 {
    text-transform: uppercase
}

h2 {
    text-transform: capitalize
}
```

1.3.7. Float và Clear

a. Float

Float (theo như nghĩa tiếng Việt là thả trôi, trôi nổi) là một thuộc tính CSS dùng để cố định một thành phần web về bên trái hay bên phải không gian bao quanh nó. Đây là một thuộc tính rất cần thiết khi dàn trang (như tạo các trang web 2 column layout hay 3 column layout), hiển thị văn bản thành cột (giống như kiểu format > Columns trong MS Word vậy), hay thực hiện việc định vị trí ảnh và text (như một số kiểu text wrapping của MS Word). Để rõ hơn chúng ta hãy xem hình minh họa sau về cơ chế hoạt động của float:

Thuộc tính float có 3 giá trị:

- `Left`: Cố định phần tử về bên trái.
- `Right`: Cố định phần tử về bên phải.
- `None`: Bình thường.

Nhìn vào hình minh họa trên chúng ta thấy là ban đầu trong box lớn có hai thành phần là Box B và phần Content. Lúc đầu Box B nằm bên trên và Content nằm bên dưới, nhưng khi chúng ta đặt thuộc tính float cho Box B thì Box B bị cố định về bên trái và chừa lại khoảng trống bên trái nó. Còn phần Content thì vốn nằm bên dưới sẽ tự động tràn lên để lấp đầy khoảng trống do Box B tạo ra.

Ví dụ:

```
<div id= "logo"></div>
<div>ITPLUS Academy</div>
```

Với đoạn mã HTML trên, chúng ta sẽ thực hiện float ảnh logo sang trái để phần nội dung bên dưới tràn lên nằm cạnh logo.

```
#logo {
    background: url(Untitled.png);
    width: 300px;
    height: 128px;
    margin: 0px;
    padding: 0px;
    float: left;
}
```

So sánh kết quả hiển thị khi có và không có float trong mã CSS khi chạy tập tin HTML:

Trường hợp không có thuộc tính float:

Hình 6. 18. Thẻ div với id logo không có thuộc tính float

Trường hợp có thuộc tính float trong #logo:

Hình 12.7: Thẻ div với id logo có thuộc tính float left

b. Clear

Đi cùng với thuộc tính float, trong CSS còn có một thuộc tính là clear. Thuộc tính clear là một thuộc tính thường được gán vào các phần tử liên quan tới phần tử đã được float để quyết định hướng xử sự của phần tử này.

Ở ví dụ trên, khi chúng ta float tấm ảnh qua trái thì mặc nhiên văn bản sẽ được tràn lên để lấp vào chỗ trống. Nhưng khi chúng ta đặt vào văn bản thuộc tính clear thì chúng ta có quyền quyết định xem phần văn bản đó có được tràn lên hay không. Thuộc tính clear có tất cả 4 thuộc tính: left (tràn bên trái), right (tràn bên phải), both (không tràn) và none.

Thử dùng thuộc tính clear với các giá trị khác nhau đối với đoạn văn bản trong ví dụ float ảnh logo ở trên.

1.3.8. Position

Kết hợp với thuộc tính float đã học, thuộc tính position mang lại nhiều khả năng để tạo một cách trình bày tiên tiến và chính xác cho trang web với 2 giá trị là Absolute và Relative.

Nguyên lý hoạt động của position:

Hãy tưởng tượng cửa sổ trình duyệt của bạn giống như một hệ tọa độ và với position bạn có thể đặt một đối tượng web ở bất cứ vị trí nào trên hệ tọa độ này.

Giả sử chúng ta muốn định vị một ảnh ở vị trí 70px cách đỉnh và 90px từ bên trái tài liệu, chúng ta sẽ viết CSS như sau:

```
img { position:absolute; top:70px; left:90px }
```

a. Absolute Position

Định vị tuyệt đối là sự định vị mà trong đó các thành phần được định vị không để lại bất cứ một khoảng trống nào trong tài liệu. Một thành phần được định vị tuyệt đối sẽ nhận giá trị position là absolute. Các đối tượng đã định vị tuyệt đối sẽ dùng kết hợp với các thuộc tính top, left, right, bottom để xác định tọa độ.

Ví dụ sau sẽ chỉ cho chúng ta cách đặt bốn ảnh ở bốn góc tài liệu bằng định vị tuyệt đối.

```
#logo1 {
    position: absolute;
    top: 50px;
    left: 70px
}
#logo2 {
    position: absolute;
    top: 0;
    right: 0
}
#logo3 {
    position: absolute;
    bottom: 0;
    left: 0
}
#logo4 {
    position: absolute;
    bottom: 70px;
    right: 50px
}
```

b. Relative Position

Sự định vị tương đối cho một thành phần là sự định vị được tính từ vị trí gốc trong tài liệu. Các thành phần đã được định vị tương đối sẽ để lại khoảng không trong tài liệu. Các thành phần được định vị tương đối sẽ nhận giá trị position là relative. Chúng ta hãy làm lại ví dụ trên nhưng thay absolute thành relative. Các bạn ghi nhận lại vị trí 4 ảnh logo lúc áp dụng thuộc tính position là none, absolute và relative rồi rút ra nhận xét.

B. BÀI TẬP

2.1. Câu hỏi ôn tập

Câu 1: Thuộc tính nào được sử dụng để tạo ra khoảng cách giữa các thành phần HTML?

M. margin
N. spacing
O. padding
P. border

Câu 2: Khi người sử dụng phím tab để di chuyển giữa các đối tượng trên website, thuộc tính nào chỉ ra sự di chuyển từ thành phần này sang thành phần trong hành vi này?

D. a:link
E. a:focus
F. a:active
G. a:visited

Câu 3: Thuộc tính nào được sử dụng để tạo màu cho chữ?

D. text-color
E. color
F. text:color
G. font-color

Câu 4: CSS là viết tắt của cụm từ nào sau đây?

E. Cascading Style Sheets
F. Carrot Sytem Style
G. Correlated Styling System
H. Canvas Styling System

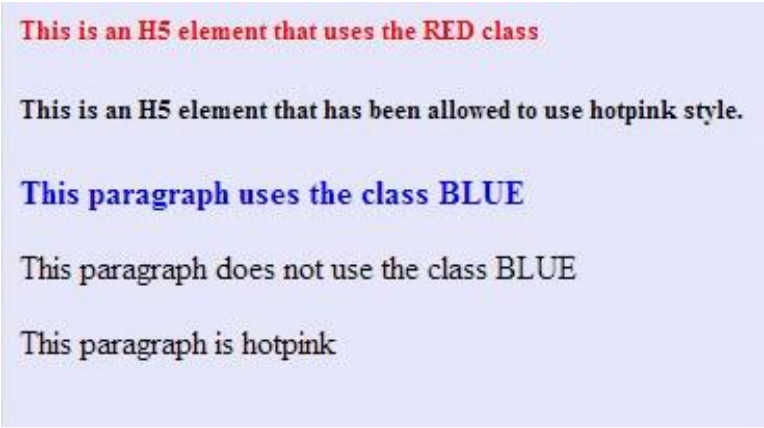
Câu 5: Thuộc tính và giá trị nào cho phép chuyển đổi danh sách dạng La Mã (roman) sang dạng danh sách bình thường?

E. list-style-type:upper-roman;
F. list-type:roman;
G. list-bullet-type:roman-numerals;
H. list-style:roman;

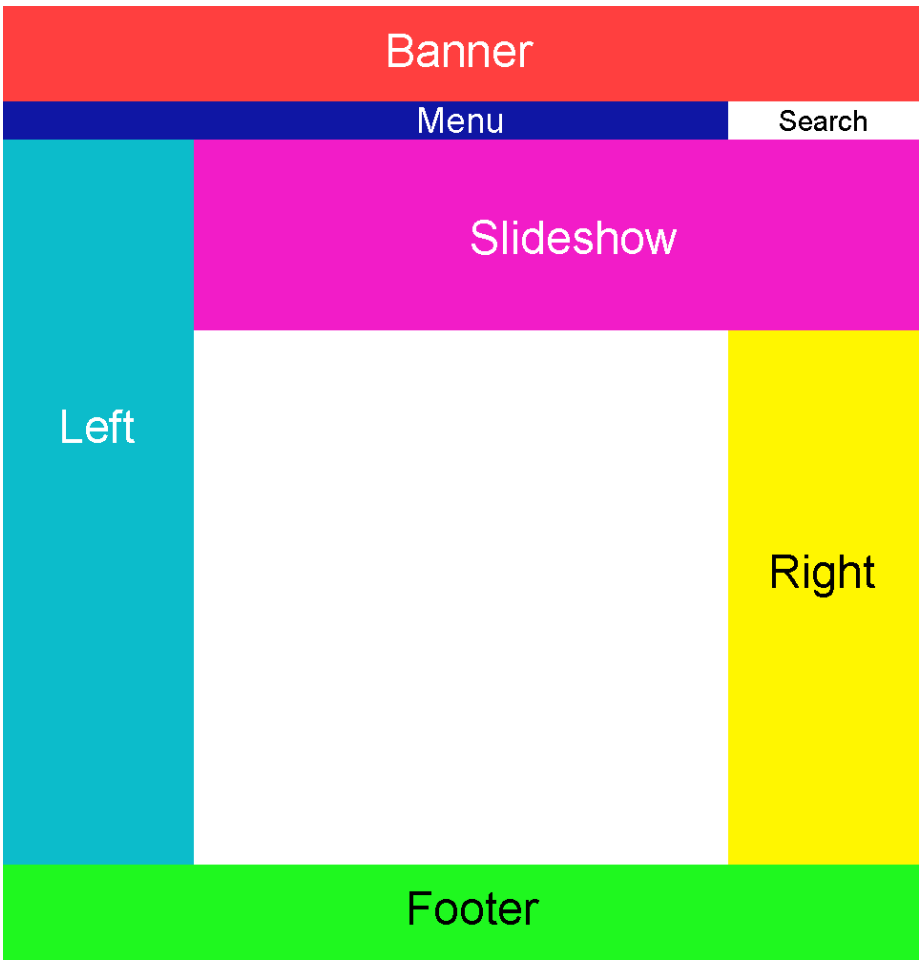
2.2. Bài tập

Bài 1. Sử dụng CSS và mã HTML tạo một trang với các yêu cầu như sau.

- Có giao diện như hình vẽ
- Áp dụng CSS cho các phần tử trong bài tập trên bằng ít nhất 3 cách viết.
- Tăng giảm kích cỡ của các phần tử tuỳ ý, đồng chứng minh thẻ h1 có thể có cỡ chữ nhỏ hơn thẻ h5?



Bài 2. Sử dụng CSS và HTMML tạo website có bố cục như sau:



- Đặt viền border (cả 4 phía) cho mỗi vùng màu là 10px, tìm cách sửa lại giao diện để tránh "vỡ". Mỗi vùng màu dùng một loại nét viền khác nhau.

Bài 3. Thêm một ảnh vào trang web với các yêu cầu cụ thể như sau:

- Đặt một ảnh vào vị trí vùng màu màu trắng.
- Điều khiển sao cho ảnh nằm chính giữa vùng màu trắng.
- Đặt lại vị trí ảnh vào vị trí cách lề trên vùng trắng 20pixel.
- Tiếp tục thay đổi CSS sao cho ảnh đó nằm đè lên vùng màu hồng.

BÀI 7. NGÔN NGỮ JAVASCRIPT

Với HTML và CSS chúng ta đã biết cách tạo ra giao diện các trang web đơn giản với hình ảnh và màu sắc phong phú và bắt mắt. Tuy nhiên trang web chỉ mới ở mức biểu diễn thông tin ở dạng tĩnh chứ chưa có khả năng đáp ứng các sự kiện từ phía người dùng, kiểm tra độ chính xác của dữ liệu nhập vào hoặc làm cho website trở nên sinh động hơn.

Bài này sẽ cùng tìm hiểu về ngôn ngữ, cách thức sử dụng, các đối tượng, khả năng kiểm tra, xử lý dữ liệu trên form và một số hiệu ứng trên trang web của javascript.

A. NỘI DUNG

1.1. Tổng quan về Javascript

1.1.1. Javascript là gì?

a. Khái niệm

Javascript là một ngôn ngữ lập trình trên môi trường internet hay còn gọi là client script phổ thông nhất hiện nay. Gọi nó là client script vì nó được thực hiện ngay máy tính của bạn mà không phải trên server.

b. Các tác dụng của javascript

- Xử lý các sự kiện khi người dùng tương tác với HTML qua các thành phần HTML
- Có thể hiệu chỉnh hoặc bổ sung thành phần của HTML như thay đổi độ rộng, chiều cao hoặc nội dụng một thẻ nào đó.
- Kiểm tra dữ liệu trên web đặc biệt là dữ liệu form.
- Có thể nhận biết được loại trình duyệt bạn dùng từ đó có những xử lý khác nhau.
- Có thể tạo được cookies *(sẽ được giới thiệu trong những phần sau)*.

c. Những hạn chế của javascript

- Chỉ được thực thi trên client side và nó không thể giao tiếp với server.
- Không thể tạo được các hình ảnh đồ họa.
- Có thể làm việc khác nhau trên trình duyệt khác nhau do cách thực thi javascript của mỗi trình duyệt là khác nhau.

1.1.2. Cách sử dụng javascript

Để sử dụng javascript cho trang HTML của bạn. Các bạn cần dùng cặp thẻ `<script></script>` của HTML:

```
<script type="text/javascript" src="path"></script>
```

Hoặc

```
<script type= "text/javascript">
   //mã script đặt ở đây
</script>
```

Cách thứ nhất dùng để trỏ tới một file javascript ngoài được lưu với đuôi mở rộng .js. Thuộc tính `type = "text/javascript"` là bắt buộc để khai báo mã lập trình bạn dùng là javascript còn giá trị của thuộc tính src chính là đường dẫn URL của file javascript. Cách này thường được đặt ngay trong cặp thẻ `<head></head>`.

Các thứ hai là bạn sẽ viết trực tiếp mã javascript trên file HTML các này thường được sử dụng trong cắp thẻ `<body></body>` và thường được viết ngay cạnh thành phần mà nó tác động.

1.1.3. Biến và cách khai báo javascript

Nếu theo cấu trúc dữ liệu thì biến là một vùng nhớ dùng để lưu trữ dữ liệu hoặc chúng ta có thể hiểu theo nghĩa đơn giản hơn biến dùng để lưu trữ dữ liệu theo môt kiểu nào đó mà chúng ta có thể thay đổi giá trị đó.

Biến cần phải khai báo trước khi gán giá trị và phải gán giá trị trước khi sử dụng nó. Với javascript một biến được khai báo với từ khóa `var`, khi khai báo bạn có thể gán ngay dữ liệu cho nó. Ví dụ ta khai báo biến i: `var i = 1;`

Tương đương với: `var i; i= 1;`

Đối với javascript khi khai báo biến không phân biệt kiểu dữ liệu tuy nhiên kiểu dữ liệu trong javascript không nằm ngoài các kiểu sau:

- Kiểu số có thể là kiểu nguyên `(int)`, kiểu thập phân `(float)`
- Kiểu chuỗi: `string`
- Kiểu logic: `boolean`
- Kiểu mảng: `array`
- Kiểu đối tượng: `object`

Ví dụ:

```
<script type="text/javascript">
   var i = 1.5;
   var str = "Xin chào";
   var arr = new array();
</script>
```

Quy tắc đặt tên cho biến

- Không được là từ khóa
- Ký tự đầu tiên phải là một chữ cái, các ký tự tiếp theo có thể là chữ số, gạch dưới, chữ cái.
- Không được có khoảng trắng.
- Phân biệt chữ hoa và chữ thường.

Đó là nguyên tắc chung để đặt tên biến cho hầu hết các ngôn ngữ lập trình. Còn người ta đưa ra một chuẩn cho các đặt tên biến cũng nhưng phương thức là **CamelCase**. Chuẩn này nghĩa là ký tự đầu sẽ viết thường và ký tự đầu của mỗi từ tiếp theo sẽ viết hoa.

Ví dụ: `numberOfStudent` thay vì `NumberOfStudent`, `numberofstudent`

1.1.4. Function và các khai báo

Một function dùng để xử lý một sự kiện hay một khối lượng câu lệnh nào đó trong javascript. Bạn có thể sử dụng function ở bất kỳ nơi nào trong cặp thẻ `<body></body>` của HTML miễn là sau khi bạn đã khai báo nó. Vì vậy nó thường được đặt trong cặp thẻ `<head></head>`.

Cách khai báo:

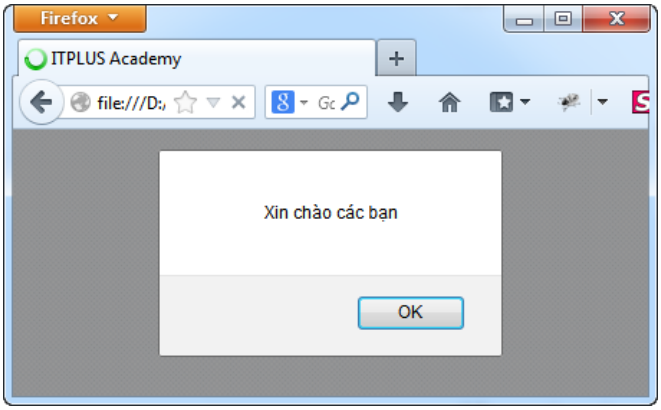
```
function functionName(var1,var2,...,varx)
{
   //some code
}
```

Trong đó `function` là từ khóa, `functionName` là tên của *function* theo **CamelCase**, và `var1, var2 ...` được gọi là các tham số dùng để truyền dữ liệu cho *function* đó.

Một ví dụ đơn giản với 1 file HTML sau:

```
<script type="text/javascript">
    function hello(str) {
        alert(str);
    }
    hello("Xin chào các bạn");
</script>
```

Lưu lại và mở file này trên trình duyệt bạn sẽ nhận được một hộp thông báo với nội dung *"Xin chào các bạn"* và nó lại được hiển thị mỗi lần bạn load lại trang này.



Hình 7.1. Ví dụ về sử dụng function trong javascript

1.1.5. Toán tử trong javascript

Toán tử trong ngôn ngữ lập trình là các phép tính như công trừ nhân chia, hay các phép so sánh lớn hơn nhỏ hơn hay bằng nhau. Toán tử phổ biến (mặc định x = 10, y = 3):

STT	Toán tử	Mô tả	Ví dụ	Kết quả
1	+	Cộng	var z=x+y;	z = 13
2	-	Trừ	var z=y-x;	z = -7
3	*	Nhân	var z=x*y;	z = 30
4	/	Chia *(nếu chia số nguyên cho số nguyên ta sẽ được một số nguyên)*	var z=x/y;	z = 3
5	%	Chia lấy dư	var z=x%y;	z = 1
6	++	Tăng thêm một đơn vị	x++;	x = 11
7	--	Giảm đi một đơn vị	x--;	x = 9
8	=	Gán giá trị	var y=x; var x=5;	Y= 10; x = 5
9	+=	Cộng sau đó gán lại giá trị của biến	x+=2;	x = 12
10	-=	Trừ sau đó gán lại giá trị của biến	x-=2;	x=8
11	*=	Nhân sau đó gán lại giá trị của biến	x*=2;	x=20
12	/=	Chia sau đó gán lại giá trị của biến	x/=2;	x=5
13	%=	Chia lấy dư sau đó gán lại giá trị của biến	x%=3	x=1
14	==	So sánh bằng	x==10 y==4	true false
15	>	So sánh lớn hơn	x>y	true
16	>=	So sánh lớn hơn hoặc bằng	x>=10	true
17	<	So sánh nhỏ hơn	x<y	false
18	>=	So sánh nhỏ hơn hoặc bằng	y<=4	true
19	!=	So sánh khác	x!=y	true
20	||	Phép hoặc. Nếu một trong các phép so sánh có 1 cái đúng thì sẽ trả về đúng còn tất cả đều sai sẽ trả về sai	x<y||y<4	true
21	&&	Phép và. Nếu một trong các phép so sánh có 1 cái sai thì sẽ trả về sai còn tất cả đều đúng sẽ trả về đúng	x==10 && y==4	false

Bảng 7.1. Toán tử trong javascript

Ngoài ra toán tử "+" còn dùng để nối các chuỗi với nhau.

Ví dụ:

```
<script type="text/javascript">
    var str = "Xin chào";
    str += " các"; str = str + " bạn!";
</script>
```

Kết quả `str = "Xin chào các bạn!";`

1.2. Phát biểu có điều kiện

Bây giờ chúng ta sẽ tìm hiểu tới xương sống của javascript nói riêng và các ngôn ngữ lập trình nói chung đó là các câu lệnh điều khiển điều kiện, vòng lặp chúng ta sẽ sử dụng rất nhiều trong lập trình.

Để bắt đầu phần này chúng ta nên biết trước một số các phương thức của javascript như `alert(str)` hiện hộp thoại thông báo có nội dung là giá trị của chuỗi truyền vào hoặc `document.write(str)` chèn chuỗi vào tài liệu HTML, có thể chính là các mã HTML.

1.2.1. Câu lệnh if - else

Câu lệnh này hầu như được sử dụng liên tục trong lập trình, nó được dùng khi bạn muốn thực hiện những công việc khác nhau trong các trường hợp khác nhau của một sự kiện.

Cú pháp:

```
if (biểu thức điều kiện) {
        // Lệnh thực hiện khi biểu thức điều kiện đúng
    }
    else {
        // Lệnh thực hiện khi biểu thức điều kiện sai
    }
```

Trong đó bạn có thể chỉ cần sử dụng vế thứ nhất là `if` hoặc cũng có thể sử dụng thêm một, hoặc nhiều câu lệnh `if - else` nữa ngay sau từ khóa `else` hoặc bên trong khối lệnh của `if` hoặc của `else`. Và nếu sau `if` hoặc `else` chỉ có một câu lệnh bạn có thể không cần sử dụng cặp dấu `{}`.

Ví dụ:

```
var i = 1;
if (i == 1)
    alert("Xin chào");
else
    if (i == 2)
       alert("Tạm biệt");
    else
       alert("");
```

Đoạn mã trên có nghĩa là nếu i có giá trị bằng 1 sẽ thông báo *"Xin chào"* nếu là 2 sẽ thông báo *"Tạm biệt"* còn các trường hợp khác sẽ thông báo một chuỗi rỗng.

1.2.2. Câu lệnh while

While là vòng lặp không xác định được dùng khi công việc của bạn cần thực hiện cho đến khi một điều kiện nào đó đúng mà không thể xác định rõ số lần lặp và có thể dẫn tới vòng lặp vô tận.

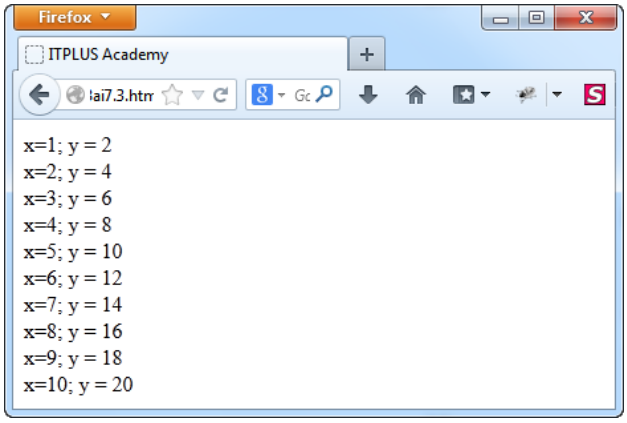
Cú pháp:

```
while (<điều kiện>)
{
    //Các câu lệnh thực hiện trong khi lặp
}
```

Ví dụ:

```
var x = 1;
var y;
while (x <= 10) {
  y = x * 2
  document.write("x=" + x + "; y = " + y + "</br>");
  x++;
}
```

Kết quả:



Hình 7.2. Ví dụ về câu lệnh while

1.2.3. Câu lệnh for

Chúng ta sử dụng vòng lặp for khi đã xác định được số vòng lặp.

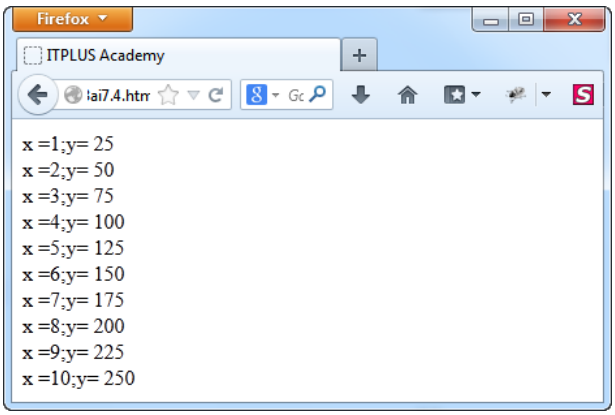
Cú pháp:

```
for (biến=giá trị khởi tạo; biến = giá trị kết thúc; biến + = giá trị tăng)
{
   //Các lệnh được thực hiện trong khi lặp
}
```

Ví dụ:

```
var x;
var y;
for (x = 1; x <= 10; x++) {
  y = x * 25;
  document.write("x =" + x + ";y= " + y + "<br/>");
}
```

Mở file trên trình duyệt ta sẽ có:



Hình 7.3. Ví dụ về vòng lặp for

Đặc biệt ta có thể khai báo biến ngay trong vòng for như sau:

```
for(var x = 1; x<=10; x++)
```

1.2.4. Câu lệnh switch – case

Câu lệnh này cũng có tác dụng phân chia trường hợp để sử lý giống như if else. Nhưng các thức làm việc có phần khác với if else.

Cú pháp:

```
switch (n) {
   case 1:
      //some code
      break;
   case 2:
      //somecode
      break;

...
   default:
     //somecode
}
```

Câu lệnh `switch case` có nhiệm vụ so sánh giá trị của n *(có thể là một biểu thức hoặc một biến)* với các giá trị trong các vế `case`. Nó sẽ duyệt lần lượt các trường hợp nếu nó đúng với trường hợp nào thì sẽ thực thi khối lệnh của trường hợp đó. Sau đó sẽ kết thúc nếu ta sử dụng `break` còn nếu không giá trị của biểu thức được tiếp tục so sánh tiếp và thưc hiện cả những mã lệnh thuộc trường hợp khác sau đó cho đến khi gặp một `break` khác.

Trường hợp `default` dùng để thực thi mã lệnh nếu tất cả các trường hợp trên đều không xảy ra.

Ví dụ:

```
var i = 3;
    switch (i) {
        case 1:
        case 2:
            alert("2");
        case 3:
            alert("3");
        case 10:
            alert("10");
        case 11:
            alert("11");
            break;
        default:
            alert("default");
            break;
    }
```

Thực thi đoạn mã này trên trình duyệt ta sẽ nhận được lần lượt các thông báo với nội dung “123”, “10”, “11”. Đoạn mã trên đã hoạt động như sau. Giá trị của biến i với 1, 2 và thực thi alert khi gặp trường hợp 3 vì i có giá trị bằng 3. Nhưng do ta không đặt break các trường hợp tiếp theo được so sánh tiếp tuy giá trị so sánh không đúng nhưng 2 lệnh `alert` tiếp theo vẫn được thực hiện cho đến khi gặp lệnh `break` do đó lệnh `alert` cuối cùng không dược thực thi nữa.

1.2.5. Phát biểu break và continue

a. Câu lệnh beak

Câu lệnh `break` dùng để kết thúc việc thực hiện của vòng lặp `for` hay `while` khi bạn gặp điều kiện cần mà vòng lặp vẫn chưa chạy hết có nghĩa là những đoạn mã của vòng lặp sau `break` không được thực thi.

Cú pháp của nó rất đơn giản chỉ với từ khóa `break`

Đoạn mã sau lặp cho đến khi x lớn hơn hoặc bằng 10. Tuy nhiên nếu giá trị x đã vào vòng lặp nhỏ hơn 5, vòng lặp sẽ kết thúc:

Ví dụ:

```
var x = 1;
    while (x < 10) {
        if (x < 5) break;
        x++;
    }
```

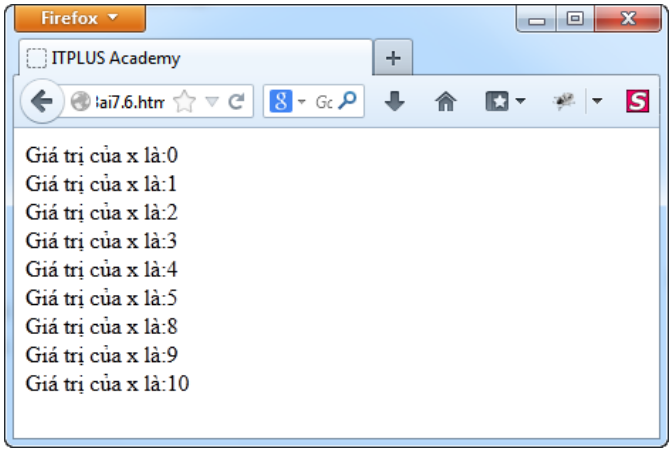
b. Câu lệnh continue

Lệnh `continue` giống lệnh `break` nhưng khác ở chỗ vòng lặp sẽ nhảy đến bước lặp tiếp theo ngay và đoạn mã sau `continue` sẽ không được thực thi nữa.

Ví dụ đoạn mã sau tăng x từ 0 lên 5, nhảy lên 8 và tiếp tục tăng lên 10:

```
var x = 0;
    while (x <= 10) {
        document.write("Giá trị của x là:" + x + "</br>");
        if (x == 5) {
            x = 8;
            continue;
        }
        x++;
    }
```

Sử dụng đoạn mã này trong file HTML ta sẽ được kết quả là:



Hình 7.4. Ví dụ về câu lệnh continue

1.3. Xử lý form với javascript

Khi bạn cho phép người dùng nhập liệu trên form nhằm submit dữ liệu lên server, bạn cần viết một hàm để kiểm tra dữ liệu có hợp lệ hay không trước khi cho phép dữ liệu trong form được submit.

1.3.1. Javascript và thẻ Text, Password

Trong hàm, bạn sử dụng phát biểu `return` để trả về giá trị true nếu dữ liệu nhập là hợp lệ, giá trị false dữ liệu nhập không hợp lệ. Nếu giá trị bằng `false` thao tác `submit` sẽ không thực hiện và yêu cầu người dùng sửa lại dữ liệu. Phương thức này gọi trong biến cố `onsubmit` của `form` như ví dụ sau

Ví dụ: Javascript kiểm tra dữ liệu nhập vào thẻ text, password

```
function checkInput() {
        //kiểm tra Username
        if (document.frmLogin.Username.value == "") {
            Alert("InvalidEmail, Please enter your own Email");
            Document.frmLogin.Username.focus();
            return false;
        }
        //kiểm tra Password
        if (document.frmLogin.Password.value == "") {
            alert("Please enter your own Password");
            Document.frmLogin.Password.forcus();
            return flase;
        }
        return true;
    }
```

Trong ví dụ trên, trước tiên bạn kiểm tra giá trị nhập vào trong thẻ text có tên là *Username*, nếu giá trị rỗng, một thông báo xuất hiện và chuyển con nháy vào thẻ này bằng phương thức `focus()`;

Kế tiếp, phương thức trả về giá trị false. Điều này có nghĩa là form sẽ không được `submit` cho đến khi phương thức trả về giá trị `true`.

Tương tự nhu vậy, bạn có thể kiểm tra dữ liệu trên thẻ `pasword` có tên là `Password`, nếu giá trị rỗng một thông báo hiện ra vả chuyển con nháy vào thẻ này bằng phương thức `focus()`.

Tiếp theo phương thức này sẽ trả về giá trị `false`. Điều này có nghĩa là ỏm sẽ không được `submit` cho đến khi phương thức trên trả về gái trị `true`.

Nếu cả hai trường hợp trên đều hợp lệ, có nghĩa là người dùng có nhập dữ liệ vào hai thẻ `Username` và `Passowrd`, phương thức trên trả về giá trị `true`, `form` sẽ được `submit` theo yêu cầu của người dùng.

1.3.2. Javascript và thẻ Radio, Checkbox

Bằng cách sử dụng phát biểu `return` như trong trường hợp `login` để trả giá trị `true` nếu dữ liệu chọn là hợp lệ giá trị `false` dữ liệu chọn không hợp lệ. Bạn có thể tham khảo ví dụ sau:

Ví dụ: javascript kiểm tra dữ liệu của thẻ radio, checkbox:

```
function checkInput() {
        var isOK = false;
        for (var i = 0; i < document.frmReg.rdGender.length; i++) {
            if (frmReg.rdGender[i].checked == true) {
                isOK = true;
                break;
            }
        }
        if (isOK == false) {
            alert("Please enter choose your gender");
            return false;
        }
        if (document.frmReg.chkBA.checked == false ||
document.frmReg.chkMBA.checked == false || document.frmReg.chkBS.checked ==
false || document.frmReg.chkMBS.checked == false) {
            alert("Please enter at least one certificate");
            return false;
        }
        return true;
    }
```

Để khai báo nội dung trang web, bạn có thể thực hiện quá trình khao báo có nội dung giống như ví dụ sau:

```
<table width="251">
    <form name="frmReg" method="post" action="jsdoreg.php?goto="
onSubmit="return checkInput();">
        <tr>
            <td>Gender</td>
        </tr>
        <tr>
            <td><input type="radio" name="rdGender" value="M">Male <input
type="radio" name="rdGender" value="F">Female
            </td>
        </tr>
        <tr>
            <td>Marital Status</td>
        </tr>
        <tr>
            <td>
                <input type="radio" name="rdMar" value="S" checked>Single
                <input type="radio" name="rdMar" value="M">Married
            </td>
        </tr>
        <tr>
            <td>Graduate</td>
        </tr>
        <tr>
            <td>
                <input type="checkbox" name="chkBA" value="BA">BA
                <input type="checkbox" name="chkBS" value="BS">BS
                <input type="checkbox" name="chkMBA" value="MBA">MBA
                <input type="checkbox" name="chkMBS" value="MBS">MBS
            </td>
        </tr>
        <tr>
            <td align="left" class="text_normal" valign="top"><br>
                <input type="submit" style="width=80px" name="Register"
value="Register" class="button">
                <input type="reset" name="Reset" value="Reset"
class="button">
            </td>
        </tr>
    </form>
</table>
```

- Đối với thẻ `input` dạng `radio` có tên `rdMar`, bạn không cần kiểm tra trạng thái của chúng, bởi vì trong khai báo HTML đã có giá trị mặc định, điều này có nghĩa là nếu người dùng không chọn, thì họ sẽ có tình trạng gia đình mặc định là `singe`.

1.3.3. Javascript với thẻ <textarea>

- Thẻ `<textarea>` không có thuộc tính value, tuy nhiên khi tham chiếu đến giá trị của nó, bạn lại sử dụng thuộc tính này như cách tham chiếu đến thẻ input dạng text.

- Thẻ `<textarea>` không giới hạn chiều dài chuỗi. Có nghĩa là bạn phải viết một đoạn javacript để khống chế chiều dài người dùng nhập cho

phù hợp với chiều dài của cột dữ liệu đã khai báo trong CSDL. Bằng cách thiết kế trang web có hai thẻ `<textarea>`, thẻ thứ nhất có tên là `txtdesc` không khai báo giá trị mặc định trong thẻ `<textarea>`, thẻ thứ hai có tên là `txtreq` với chuỗi mặc định trong thẻ.

- Thêm vào đó, bạn có thể thêm một thẻ input dang text để trình bày số kí tự người dùng đã nhập vào thẻ `<textarea>` có tên txtdesc trên biến cố `onkeypress`

- Để kiểm tra chiều dài của thẻ `txtdesc`, bạn có thể sử dụng thuộc tính `length` của `value`.

Ví dụ: Kiểm tra chiều dài của thẻ `<textarea>`:

```
function checkInput() {
        if (document.frmEdit.txtDesc.value == "") {
            alert("Please enter your Description");
            document.frmEdit.txtDesc.focus();
            return false;
        }
        if (document.frmEdit.txtDesc.value.length > 200) {
            alert("Please enter your Description < 200 characters");
            document.frmEdit.txtDesc.focus();
            return false;
        }
        return true;
    }
```

- Cũng tương tự như các trường hợp kiểm tra dữ liệu trên, trong trường hợp người dùng không nhập dữ liệu vào thẻ `txtdesc`, hay chuỗi nhập lớn hơn 200 kí tự, trình duyệt xuất hiện thông báo.

Tuy nhiên để kiểm tra chiều dài của thẻ `textarea`, bạn sử dụng tham chiếu `txtdesc.value.length`, sau đó so sánh chiều dài này với chiều dài được giới hạn, với mục đích ghi chú cho người dùng biết số kí tự họ có thể thêm vào thẻ `txtdesc`:

```
function count() {
        Document.frmEdit.txtcount.value = 200
        document.frmEdit.txtdesc.value.length;
    }
```

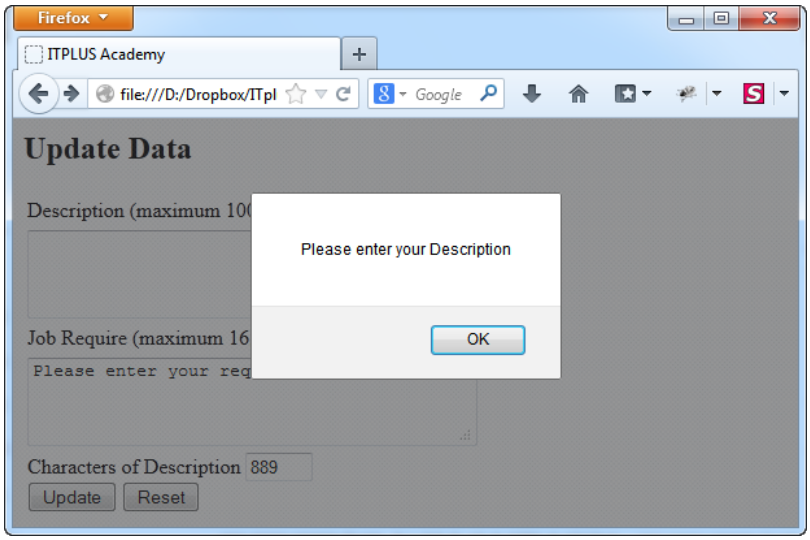
Bước kế tiếp bạn sẽ khai báo phương thức `count()` trên biến cố `onkeypress` của thẻ `txtdesc`, trở lại trình duyệt bạn nhập một chuỗi, trong thẻ text có tên `txtcount` sẽ xuất hiện số kí tự còn lại :

Để hoàn thiện đoạn javascript kiểm tra thẻ textarea mời bạn xem ví dụ dưới đây:

```
<html>
<head>
    <title>ITPLUS Academy</title>
    <script type="text/javascript">
        function checkInput() {
            if (document.frmEdit.txtDesc.value == "") {
                alert("Please enter your Description");
                document.frmEdit.txtDesc.focus();
                return false;
            }
            if (document.frmEdit.txtDesc.value.length > 200) {
                alert("Please enter your Description < 200 characters");
                document.frmEdit.txtDesc.focus();
                return false;
            }
            return true;
        }
        //De thuc hien tinh so ky tu con lai, ban nhap doan phuong thuc co
ten Count()
        function count() {
            Document.frmEdit.txtcount.value = 200
            document.frmEdit.txtdesc.value.length;
        }

    </script>
</head>
<body>
<h2>Update Data</h2>
<table width="288">
    <form name="frmEdit" method="post" action="jsdoreg.php?goto="
onSubmit="return checkInput();">
        <tr>
            <td>Description (maximum 100 characters)</td>
        </tr>
        <tr>
            <td valign="top"><textarea name="txtDesc" rows="3" cols="40"
class="textbox” onKeyPress="
                                        count();"> </textarea></td>
        </tr>
        <tr>
            <td valign="top">Job Require (maximum 16,000 characters)</td>
        </tr>
        <tr>
            <td valign="top"><textarea name="txtReq" rows="3" cols="40"
class="textbox">Please enter your
                requirement</textarea></td>
        </tr>
        <tr>
            <td align="left" class="text_normal" valign="top"> Characters of
Description
                <input type="text" name="txtCount" size="5" maxlength="5"
class="textbox"><br>
                <input type="submit" style="width=80px" name="Update"
value="Update" class="button">
                <input type="reset" name="Reset" value="Reset"
class="button">
            </td>
        </tr>
    </form>
</table>
</body>
</html>
```

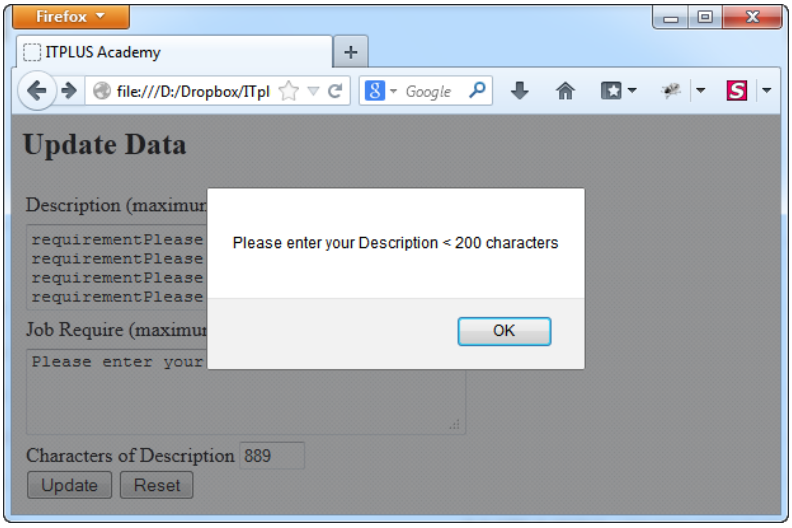
Kết quả:

- Trường hợp không nhập liệu:



Hình 7.5. Trường hợp không nhập dữ liệu

- Trường hợp nhập dữ liệu vượt quá 200 ký tự:



Hình 7.6. Dữ liệu vượt quá 200 ký tự

1.3.4. Javascript với thẻ Checkbox cùng tên

Như bạn đã biết, mỗi khi trang web của bạn có một số lượng mẫu tin từ kết quả tìm kiếm hay liệt kê, muốn cho người dùng xóa hay cập nhật trên nhiều mẫu tin, bạn sẽ trình bày dữ liệu như cách trình bày dữ liệu liệt kê mail trong hộp mail điện tử yahoo hay hotmail.

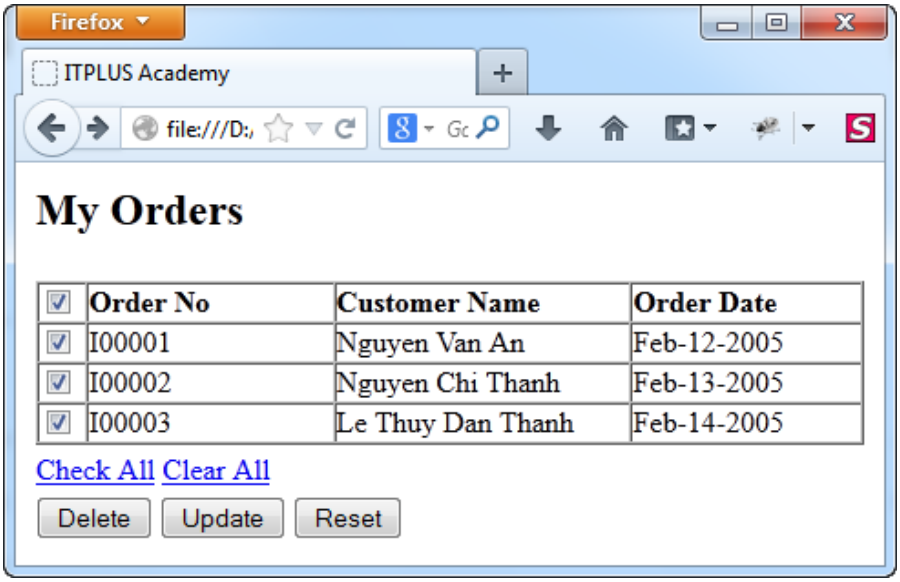
Trong phần này chúng ta giả định rằng một mẫu tin cố định, bằng cách tạo ra thẻ checkbox, cho phép người dùng chọn những mẫu tin họ cần cập nhật hay xóa chúng khỏi bảng dữ liệu.

```
<html>
<head>
    <title>ITPLUS Academy</title>
    <script type="text/javascript">
        function docheck(status, from_) {
            var alen = document.frmList.elements.length;
            //tính số phần tử có trên form
            alen = (alen > 5) ? document.frmList.chkid.length : 0;
            if (alen > 0) {
                //duyệt các phần tử nếu số mẫu tin nhiều hơn một
                for (var i = 0; i < alen; i++)
                    document.frmList.chkid[i].checked = status;
            } else {
                document.frmList.chkid.checked = status;
            }
            //gán trạng thái của thẻ chkall
            if (from_ > 0) {
                document.frmList.chkall.checked = status;
            }
        }

        function docheckone() {
            var alen = document.frmList.elements.length;
            //bạn cho rằng các thẻ chkid đều checked
            var isChecked = true;
            alen = (alen > 5) ? document.frmList.chkid.length : 0;
            if (alen > 0) {
                // kiểm tra nếu có ít nhất một thẻ chkid không checked, biến
ischecked có giá trị false
                for (var i = 0; i < alen; i++)
                    if (document.frmList.chkid[i].checked == false)
                        isChecked = false;
            } else {
                //trong trường hợp chỉ có một mẫu tin
                if (document.frmList.chkid.checked == false)
                    isChecked = false;
            }
            //gán trạng thái của thẻ chkall bằng với giá trị của biến
ischecked
            document.frmList.chkall.checked = isChecked;
        }

        function checkInput() {
            var alen = document.frmList.elements.length;
            var isChecked = false;
            alen = (alen > 5) ? document.frmList.chkid.length : 0;
            if (alen > 0) {
                for (var i = 0; i < alen; i++)
                    if (document.frmList.chkid[i].checked == true)
                        isChecked = true;
            } else {
                if (document.frmList.chkid.checked == true)
                    isChecked = true;
            }
            //nếu không check chkid, thông báo cho người dùng biết
            if (!isChecked)
                alert("Please select at least one of them");

            return isChecked;
        }
    </script>
</head>
<body>
```

```
<table width="100%">
    <form name="frmList" method="post" action="jsdo.php?goto="
onSubmit="return checkInput();">
        <tr>
            <td><h2>My Orders</h2></td>
        </tr>
        <tr>
            <td valign="top" height="79">
                <table width="100%" border="1" cellspacing="0"
cellpadding="0">
                    <tr>
                        <td width="6%"><input type="checkbox" name="chkall"
value="0"

onClick="docheck(document.frmList.chkall.checked,0);">
                        </td>
                        <td width="30%"><b>Order No</b></td>
                        <td width="36%"><b>Customer Name</b></td>
                        <td width="28%"><b>Order Date</b></td>
                    </tr>
                    <tr>
                        <td width="6%">
                            <input type="checkbox" name="chkid"
value="I00001" onClick="docheckone();">
                        </td>
                        <td width="30%">I00001</td>
                        <td width="36%">Nguyen Van An</td>
                        <td width="28%">Feb-12-2005</td>
                    </tr>
                    <tr>
                        <td width="6%"><input type="checkbox" name="chkid"
value="I00002" onclick="docheckone();"></td>
                        <td width="30%">I00002</td>
                        <td width="36%">Nguyen Chi Thanh</td>
                        <td width="28%">Feb-13-2005</td>
                    </tr>
                    <tr>
                        <td width="6%">
                            <input type="checkbox" name="chkid"
value="I00003" onClick="docheckone();">
                        </td>
                        <td width="30%">I00003</td>
                        <td width="36%">Le Thuy Dan Thanh</td>
                        <td width="28%">Feb-14-2005</td>
                    </tr>
                </table>
            </td>
        </tr>
        <tr>
            <td align="left" class="text_normal" valign="top"><a
href="javascript:docheck(true,1);">Check All</a>
                <a href="javascript:docheck(false,2);">Clear All</a></td>
        </tr>
        <tr>
            <td align="left" class="text_normal" valign="top">
                <input type="submit" style="width=80px" name="submit"
value="Delete" class="button">
                <input type="submit" style="width=80px" name="submit"
value="Update" class="button">
                <input type="reset" name="Reset" value="Reset"
class="button">
            </td>
        </tr>
```

```
    </form>
</table>
</body>
</html>
```

Bạn có thể chọn Check All để chọn cả danh sách hoặc Clear All để không chọn checkbox nào.



Hình 7.7. Trường hợp chọn Check All

1.4. Các đối tượng trong javascript

1.4.1. Giới thiệu về lập trình hướng đối tượng

Lập trình hướng đối tượng *(gọi tắt là OOP - Object Oriented Programming)*, lưu trữ các tập hợp dữ liệu dưới dạng các đối tượng với các thuộc tính hay phương thức của chính đối tượng đó. Các đối tượng được hiểu như là các thực thể ngoài đời thực như con người có thuộc tính chiều cao cân năng, phương thức là các hành động nói năng, đi đứng.

Có 2 khái niệm quan trọng mà bạn cần phải biết đối với lập trình hướng đối tượng là:

- `Class` *(lớp):* có thể hiểu là một kiểu đối tượng giống như kiểu của biến. Class quy định các thuộc tính và phương thức.
- `Object` *(đối tượng)*: là một thực thể thuộc một lớp nào đó có các thuộc tính có thể khác nhau con phương thức là giống nhau.

Ví dụ con người là một Class còn Anh Nguyễn Văn Nam và Chị Trần Thị Lan là 2 đối tượng với 2 thuộc tính Tên và giới tính là khác nhau nhưng đều có thể nói hay đi lại.

Và muốn làm việc với các đối tượng ta cũng phải khai báo nó thuộc lớp nào. Sau đó chúng ta có thể lấy hoặc gán giá trị cho các thuộc tính của nó. Như vậy đa số chúng ta sẽ làm việc với các đối tượng. Tuy nhiên đôi khi chúng ta cũng có thể sử dụng những thuộc tính chung hoặc phương thức của chính class đó. Đó là các thuộc tính và phương thức static. Chúng ta có thể gọi trực tiếp bằng class mà không cần phải gọi qua đối tượng.

Ngoài ra lập trình hướng đối tượng rất đa dạng, đối tượng này có thể là thuộc tính của đối tượng kia, các lớp lại có thể kế thừa nhau... Chúng ta sẽ cùng tìm hiểu một số đối tượng chính trong javascript.

1.4.2. Đối tượng String

Như tôi đã nói ở trên string là một kiểu dữ liệu, nhưng nó cũng chính là một đối tượng. Đối tượng String được xây dựng nội tại trong javascript cung cấp nhiều phương thức thao tác trên chuỗi. Đối tượng này có thuộc tính duy nhất là độ dài (length) và các phương thức xử lý chuỗi.

Khai báo một chuỗi. Giống như cách khai báo các đối tượng các chúng ta sử dụng từ khóa `new` để khai báo một đối tượng.

```
var str = new String("Xin chào các bạn");
```

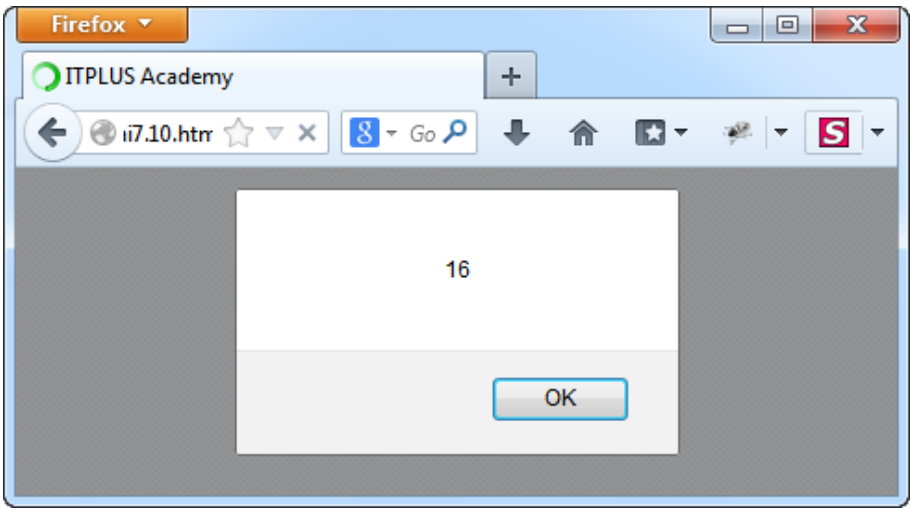
Hoặc đơn giản chỉ cần `var str = "Xin chào các bạn";` đối với đối tượng String.

Thuộc tính length là số các ký tự của chuỗi kể cả dấu cách. Nhưng vậy chuỗi `"Xin chào các bạn"` có 16 ký tự.

Ví dụ:

```
var str = "Xin chào các bạn";
alert(str.length);
```

Ta sẽ nhân được thông báo có nội dung là "16":



Hình 7.8. Sử dụng đối tượng string

Các phương thức của đối tượng String (chúng ta sẽ dùng chuỗi `"Xin chào các bạn"` để mô tả cho phần này):

STT	Tên phương thức	Mô tả
1	charAt(index)	index là chỉ số thứ tự của chữ cái muốn tìm có giá trị từ 0 đến str.length-1. Ví dụ: `str.charAt(0)` là 'X'. `str.charAt(str.length-1)` hay `str.charAt(15)` là 'n'.
2	charCodeAt(index)	Tương tự như charAt nhưng giá trị trả về là mã ASCII của ký tự với kết quả là số nguyên. Ví dụ: `str.chartCodeAt(0)` là 88 (Mã ASCII là 01011000)
3	concat(string)	Nối 2 chuỗi lại với nhau. Ví dụ: `str.concat("đã đến với học viện ITPLUS!");` tương đương với `str+= "đã đến với học viện ITPLUS!"` Kết quả biến str có giá trị là: `"Xin chào các bạn đã đến với học viện ITPLUS!".`
4	fromCharCode(x,x2, ...)	Tham số là các số nguyên. Phương thức này trả về một chuỗi có các ký tự có mã ASCII lần lượt ứng với các tham số chuyền vào. Phương thức này là phương thức static được gọi trực tiếp từ lớp String.

STT	Tên phương thức	Mô tả
		Ví dụ: `String.fromCharCode(72,69,76,76,79)` ta được chuỗi "HELLO"
5	indexOf(string)	Lấy chỉ số đầu của của chuỗi hoặc truyền vào trong chuỗi được tìm kiếm. Nếu chuỗi đó không thuộc chuỗi được tìm kiếm thì sẽ trả về -1. Phương thức này được áp dụng cho việc tìm kiếm ký tự trong chuỗi. Ví dụ: `str.indexOf("in")` là 0, `str.indexOf("x")` là -1.
6	lastIndexOf(string)	Tương tự như `indexOf` nhưng giá trị trả về là chỉ số cuối cùng của chuỗi truyền vào hoặc là -1 nếu không tìm thấy. Ví dụ: `str.lastIndexOf("n")` là 15.
7	match(regexp/string)	Dùng để so sánh một chuỗi với biểu thức quy tắc. Tôi sẽ nói rõ phương thức này tại phần giới thiệu về biểu thức quy tắc trong javascript
8	replace(regexp/str, subject)	Tìm chuỗi con ứng với biểu thức quy tắc hoặc chuỗi truyền vào nếu có sẽ thay toàn bộ chuỗi con đó bằng chuỗi subject. Giá trị trả về là chuỗi đã được thay thế. Ví dụ: `str = str.replace(" ", "").` Giá trị của str sẽ là "Xinchàocácbạn".
9	search (regexp/str)	Phương thức này cũng giống với `indexOf` tuy nhiên nó có thể sử dụng biểu thức quy tắc để tìm kiếm
10	slice(begin, end)	Tách chuỗi con có chỉ số bắt đầu từ begin kết thúc tại end sau đó trả về chuỗi con đó. Begin là bắt buộc còn end là tùy chọn. Nếu không có end thì chuỗi trả về bắt đầu từ begin cho tới cuối cùng. Nếu begin có giá trị âm chuỗi trả về được tính ngược lại theo chiều từ phải qua trái. Ví dụ `str.slice(4)` là "chào các bạn", `str.slice(4,7)` là "chào", và `str.slice(-3)` là "bạn"

STT	Tên phương thức	Mô tả

11	split(separator, litmit)	Tách chuỗi ra thành các chuỗi con được lưu trữ trong mảng theo ký tự được truyền vào và có thể có giới hạn nếu ta truyền vào số nguyên limit.
		Ví dụ: `var arr = str.split();`
		Ta được mảng có 1 phần tử là "Xin chào các bạn".
		`var arr = str.split("");`
		Ta được một mảng có 15 phần tử là là tất cả các ký tự của chuỗi.
		`var arr= str.split(" ")`
		Ta được mảng có 4 phần tử "Xin", "chào", "các", "bạn".
		`var arr = str.split(" ",2);`
		Ta đươc mảng 2 phần tử: "Xin", "chào"

12	substr(begin, length)	Trả về chuỗi con bắt đầu từ chỉ số `begin` và có độ dài bằng `length`. `Begin` là bắt buộc , `length` là tùy chọn nếu không truyền vào sẽ mặc định là từ begin cho tới hết chuỗi. Ví dụ `str = str.substr(4, 4).` Ta có giá trị của str là "chào"
13	substring(begin, end)	Tác dụng giống substr tuy nhiên chuỗi con kết thúc bằng chỉ số bằng end -1. Ví dụ `str = str.substring(4,8).` Ta được giá trị của str là "chào"
14	toLowerCase()	Trả về giá trị là chuỗi có các ký tự viết thường. Ví dụ `str = str.toLowerCase().` Giá trị của str sẽ là "xin chào các bạn"
15	toUpperCase()	Trả về giá trị là chuỗi có các ký tự viết hoa. Ví dụ `str = str.toLowerCase().` Giá trị của str sẽ là "XIN CHÀO CÁC BẠN"
16	valueOf()	Trả về giá trị của đối tượng chuỗi. Ví dụ `var a = new String("Xin chào các bạn");` `a.valueOf ()` sẽ trả về "Xin chào các bạn");

Bảng 7.2. Các phương thức trong đối tượng String

1.4.3. Đối tượng Array

Array *(mảng)* là một đối tượng đặc biệt nó có thể chứa nhiều giá trị cùng một lúc. Và mỗi giá trị sẽ được đánh chỉ số riêng. Array được áp dụng khi bạn muốn lưu trữ một nhóm các giá trị như danh sách tên các học viên, giả chỉ có 10 học viên bạn có thể khai báo 10 biến cũng không sao nhưng nếu có 100 hoặc thậm chí không xác định số học viên bạn cần phải dùng tới mảng.

Kiểu dữ liệu của mỗi phần tử trong mảng là bất kỳ và phải giống nhau.

a. Cách khai báo mảng

Cách 1:

```
var arr = new Array();
arr[0] = 1; arr[1] = 2; ...
```

Cách 2:

```
var arr = new Array("a", "b", "c");
```

Cách 3:

```
var arr = [1,2,3];
```

Chúng ta có thể truy cập các phần tử thông qua chỉ số của chúng. Chỉ số được đánh dấu bắt đầu từ 0.

Như cách khai báo thứ nhất ta truy cập vào phần tử thứ nhất và gán nó bằng 1 với câu lệnh `arr[0] =1`. Ta cũng có thể sử dụng các phần tử của mảng trong mọi câu lệnh như một biến: `arr[2] = arr[0] + arr[1];` Ta có giá trị của `arr[2]` là 3.

Array cũng có một thuộc tính là length trả về số phần tử của array.

b. Các phương thức của mảng:

STT	Phương thức	Mô tả
1	concat(array1, array 2, ...)	Nối các mảng lại với nhau. Phương thức này trả về mảng kết quả của phép nối. Ví dụ: `var arr1 = [1,2,3];` `var arr2 = [3,4,5]; arr3 = arr1.concat(arr2).` Ta được mảng `arr3= [1,2,3,3,4,5]`
2	join(separator)	Nối tất các cả phần tử trong mảng thành 1 chuỗi và các thành phần được cách nhau bởi ký tự truyền vào. Nếu không có sẽ mặc định là dấu "," Ví dụ: `var arr=["Xin","chào","các","bạn"]';`

STT	Phương thức	Mô tả
		`var str = arr.join(" ");` Ta được giá trị của `str` là `"Xin chào các bạn"`
3	pop()	Lấy ra phần tử cuối cùng của mảng và trả về phần tử đó. Ví dụ: `var arr=["Xin","chào","các","bạn"];` `var str = arr.pop().` Ta được str có giá trị là `"bạn", arr` còn 3 phần tử `["Xin", "chào", "các"]`
4	push(var1, var2, ...)	Đặt các phần tử vào cuối mảng trả về độ dài mới của mảng. Ví dụ: `var arr = ["Xin", "chào"];` `var i = arr.push("các", "bạn");` Ta được `i = 4, arr=["Xin","chào","các","bạn"]`
5	reverse	Đảo ngược thứ tự các phần tử trong mảng. Trả về mảng đã được đảo ngược
6	shift	Có tác dụng ngược với pop. Lấy ra phần tử đầu tiên của mảng và trả về phần tử đó.
7	splice(index, howmany, element1, element2, ...)	Chèn hoặc xóa các phần tử trong mảng. Với tham số bắt buộc index là chỉ số bắt đầu xóa hoặc chèn, tham số bắt buộc howmany là số các phần tử bị xóa, các tham số element là các phần tử được chèn vào. Trả về mảng các phần tử bị xóa.
8	sort()	Sắp xếp các phần từ trong mảng theo chiều tăng dần đối với số và alphabe đối với chuỗi. Trả về mảng đã được sắp xếp
9	slice(begin, end)	Trả về một phần của mảng bắt đầu từ chỉ số begin và kết thúc tại chỉ số end. Nếu chỉ số end không được truyền vào sẽ mặc địch mảng con tính từ chỉ số begin cho tới hết mảng. Ví dụ: `var arr=["Xin","chào","các","bạn"].` `arr = arr.slice(1,2);` Lúc này `arr= ["chào","các"]`
10	toString()	Gần như phương thức join. Nhưng toString chỉ có một các biến mảng thành 1 chuỗi. Đó là các từ cách nhau bằng 1 dấu ,
11	unshift	Có tác dụng giống với push nhưng theo chiều ngược lại.
12	valueOf()	Trả về giá trị nguyên thủy của mảng. Phương thức này được ngầm định.

STT	Phương thức	Mô tả

		Ví dụ: `var arr = ["Xin","chào","các","bạn"]`
		`document.write(arr)`
		tương đương với
		`document.write(arr.valueOf()).`
		Kết quả là `Xin,chào,các,bạn.`

Bảng 7.3. Các phương thức của đối tượng mảng

1.4.4. Đối tượng Date

Trong javascript chúng ta có thể làm việc với ngày tháng và thời gian thông qua đối tượng Date(). Có 4 cách khai báo đối tượng date:

- `var d = new Date()`: lấy thời gian hiện tại.

- `var d = new Date(miliseconds)`: Lấy thời gian kể từ ngày 1/1/1970 cộng thêm số mili giây được truyền vào. Ví dụ `var d = new Date(86400000)` ta sẽ được ngày 2/1/1970

- `var d = new Date(dateString)`: Lấy thời gian với chuỗi ngày tháng được truyền vào. Ví dụ d = new Date("12/1/2011"). Chú ý định dạng ngày tháng trong javascript là tháng/ngày/năm

- `var d = new Date(year, month, day, hours, minutes, seconds, milisecond)`: Lấy thời gian với đầy đủ tham số truyền vào.

Một số phương thức của đối tượng Date (ta sẽ lấy ngày 1/12/2011 để ví dụ cho phần này. Khai báo: `var d = new Date("12/1/2011");`)

STT	Tên phương thức	Mô tả
1	getDate()	Trả về giá trị ngày. Có giá trị từ 1-31. Ví dụ d.getDate() trả về 1.
2	getDay()	Trả về chỉ số của ngày ứng với một mảng các ngày trong tuần. Giá trị từ 0-6. Ví dụ d.getDay trả về 3. Vì ngày 1/12/2011 là thứ 5. Mảng ngày tháng bắt đầu từ thứ 2.
3	getFullYear()	Trả về giá trị năm với 4 chữ số.
4	getMonth()	Trả về chỉ số của tháng ứng với mảng các tháng trong năm. Giá trị từ 0-11. Ví dụ d.getMonth() trả về 11.
5	getHours()	Trả về giá trị giờ từ 0 – 23
6	getMinutes()	Trả về giá trị phút từ 0 – 59

STT	Tên phương thức	Mô tả
7	getSeconds()	Trả về giá trị giây từ 0 – 59
8	getTime()	Trả về số miliseconds kể từ 0 giờ ngày 1/1/1970 cho tới thời điểm hiện tại
9	setDate(date)	Thiết lập giá trị ngày cho đối tượng Date
10	setFullYear(year)	Thiết lập giá trị năm cho đối tượng Date
11	setMonth(month)	Thiết lập giá trị tháng cho đối tượng Date
12	setHours(hours)	Thiết lập giá trị giờ cho đối tượng Date
13	getMinutes(minutes)	Thiết lập giá trị phút cho đối tượng Date
14	getSeconds(seconds)	Thiết lập giá trị giây cho đối tượng Date
15	setTime(miliseconds)	Thiết lập thời gian bằng số mili giây tính từ 0 h ngày 1/1/1970

16	parse(dateString)	Chuyển chuỗi ngày tháng được truyền vào ra số mili giấy tính từ 0 h ngày 1/1/1970. Phương thức static ta có thể gọi trực tiếp như sau:
		`var d1 = Date.parse("December 1, 2011");`
		`document.write(d1);`
		Kết quả 1322672400000

17	toDateString()	Trả về một chuỗi ngày tháng của đối tượng. Ví dụ: `document.write(d.toDateString);` Kết quả Thu Dec 1 2011
18	valueOf()	Trả về giá trị nguyên thủy của đối tượng giống `getTime()`. Ví dụ: `document.write(d)` là *Thu Dec 01 2011 00:00:00 GMT+0700 (SE Asia Standard Time)* và `document.write(d.valueOf())` là *1322672400000*
19	toLocaleString()	Trả về chuỗi ngày tháng theo định dạng vùng miền. Ví dụ `d.toLocalString()` trả về *Thursday, December 01, 2011 12:00:00 AM*
20	toString()	Trả chuỗi ngày tháng giống khi ta sử dụng trực tiếp đối tượng Date như document.write(d).
21	toTime()	Giống toString() nhưng chỉ trả về phần thời gian (giờ,

STT	Tên phương thức	Mô tả
		phút, giây)

Bảng 7.4. Các phương thức của đối tượng Date

1.4.5. Đối tượng Math

Để thực hiện các phép toán phức tạp như căn, log hay sin trong javascript. Chúng ta có thể sử dụng đối tượng Math. Đối tượng Math không cần khởi tạo chúng ta có thể sử dụng các thuộc tính và phương thức của nó trực tiếp.

Ví dụ:

```
var pi = Math.PI; // số PI
var x = Math.sqrt(16) //căn của 16
```

Bảng các thuộc tính:

STT	Tên thuộc tính	Mô tả
1	E	Trả về giá trị của số Euler. (2.718)
2	LN2	Trả về giá trị của ln2 (0.693)
3	LN10	Trả về giá trị của ln10 (2.302)
4	LOG2E	Trả về giá trị của $\log_2 e$ (1.442)
5	LOG10E	Trả về giá trị của lne (0.434)
6	PI	Trả về giá trị của số PI (3.14)
7	SQRT1_2	Trả về căn của ½ (0.707)
8	SQRT2	Trả về căn của 2 (1.414)

Bảng 7.5. Các thuộc tính của đối tượng Math

1.4.6. Đối tượng Window

Đối tượng Window đại diện của của một cửa sổ của trình duyệt. Đối tượng window có rất nhiều thuộc tính và phương thức giúp chúng ta có thể thao tác với trang web cũng như tác động tới trình duyệt.

Đa số các phương thức chúng ta sử dụng đề là phương thức của đối tượng window, điều đặc biệt đối tượng window được ngầm định. Phương thức `alert()` là một Ví dụ ta có viết viết `alert("a")` hay cũng có thể viết là `window.alert("a")`. Tuy nhiên các viết đơn giản bao giờ cũng là cách được chọn lựa.

Chú ý: để sự dụng thuộc tính và phương thức của đối tượng `window` ta viết là "`window`" chứ không phải là "`Window`" như `window.alert()`; giống như String hay Date là `Date.parse(string)`.

Các thuộc tính của đối tượng window:

STT	Tên thuộc tính	Mô tả
1	defaultStatus	Thiết lập hoặc lấy về text mặc định trên thanh trạng thái của cửa sổ
2	document	Trả về đối tượng Document.
3	frames	Trả về mảng các Frames trong tài liệu HTML.
4	history	Trả về đối tượng History
5	innerHeight	Trả về độ cao của khung hiển thị trang web của trình duyệt
6	innerWidth	Trả về độ rộng của khung hiển thị trang web của trình duyệt
7	length	Trả về số các Frames trong tài liệu HTML.
8	location	Trả về đối tượng Location.
9	name	Trả về tên của cửa sổ
10	navigator	Trả về đối tượng Navigator
11	outerHeight	Trả về độ cao của cả cửa sổ trình duyệt
12	outerWidth	Trả về độ rộng của cả cửa sổ trình duyệt
13	pageXOffset	Trả về số pixel mà nội dụng của trang web được cuộn đi theo chiều ngang so với góc trái trên của của sổ.
14	pageYOffset	Trả về số pixel mà nội dụng của trang web được đi cuộn theo chiều dọc so với góc trái trên của của sổ.
15	parent	Trả về đối tượng window là cha của đối tượng window hiện hành
16	screenTop, screenY	Trả về tọa độ y của góc trái trên của sổ so với màn hình
17	screenLeft, screenX	Trả về tạo độ x của góc trái trên của sổ so với màn hình
18	status	Gán text cho thanh trạng thái của cửa sổ

Bảng 7.6. Các thuộc tính của đối tượng window

1.4.7. Đối tượng Navigator

Đối tượng navigator dùng để lấy thông tin về trình duyệt.

Các thuộc tính của đối tượng Navigator:

STT	Tên thuộc tính	Mô tả

1	appCodeName	Trả về code name của trình duyệt
2	appName	Trả về tên của trình duyệt
3	appVersion	Trả về vesion của trình duyệt
4	cookieEnabled	Trả về giá trị boolean xem trình duyệt có hỗ trợ cookie hay không
5	platform	Trả về platform (hay hệ điều hành) trình duyệt hỗ trợ
6	userAgent	Trả về chuỗi thông tin mà trình duyệt gửi về cho server

Bảng 7.7. Các thuộc tính của đối tượng Navigator

Ví dụ tôi đang sử dụng trình duyệt Firefox 7.0.1 trên hệ điều hành Window 7 32 bit thì giá trị của các thuộc tính là:

STT	Tên thuộc tính	Mô tả
1	navigator.appCodeName	Mozilla
2	navigator.appName	Netscape
3	navigator.appVersion	5.0 (Windows)
4	navigator.cookieEnabled	true
5	navigator.platform	Win32
6	navigator.userAgent	Mozilla/5.0

Bảng 7. 8. Các thuộc tính và giá trị của đối tượng Navigator

Từ kết quả này chúng ta thấy được rằng thông tin appVersion không hẳn đúng với version thực tế của trình duyệt.

1.4.8. Đối tượng Location

Đối tượng Location lưu trữ thông tin về URL, chuyển tiếp trang hay reload lại trang web. Chúng ta có thể gọi

Các thuộc tính của Location (chúng ta sẽ sử dụng URL: http://itplus.edu.vn cho phần này)

STT	Tên thuộc tính	Mô tả
1	hash	Trả về phần id mà sau khi nhấn các siêu liên kết nội bộ tính cả dấu #
2	host	Trả về chuỗi bao gồm tên host và cổng của server
3	hostname	Trả về chuỗi tên host

STT	Tên thuộc tính	Mô tả
4	href	Trả về toàn bộ URL hiện tại
5	pathname	Trả về phần URL sau tên miền
6	port	Trả về cổng của server
7	protocol	Trả về protocol của url
8	search	Lấy về chuỗi tham số đường link bắt đầu từ dấu ?

Bảng 7.9. Các thuộc tính của đối tượng Location

Ví dụ ta đang ở trang : http://domain.com:8080/index.html?u=abc&p=123#top

Thì các giá trị lần lượt của các thuộc tính là:

STT	Tên thuộc tính	Mô tả
1	location.hash	#top
2	location.host	domain.com:8080
3	location.hostname	domain.com
4	location.port	8080
5	location.protocol	http:
6	location.pathname	index.html
7	location.search	?u=abc&p=123
8	location.href	http://domain.com:8080/index.html?u=abc&p=123#top

Bảng 7.10. Các thuộc tính và giá trị của đối tượng Location

Ngoài tác dụng lấy ra các thông tin về URL chúng ta có thể chuyển từ URL này sang URL khác bằng javascript thông qua thuộc tính href của đối tượng location.

Ví dụ ta gán `location.href = "domain.com"` Lập tức trình duyệt sẽ đưa bạn tới trang `domain.com`.

Các phương thức của đối tượng location

TT	Tên thuộc tính	Mô tả
1	assign(string)	Nếu string là tên đầy đủ của tên miền bao gồm cả protocol http:// thì trình duyệt sẽ chuyển tới URL đó nếu không thì Thay thế phần sau của pathname của URL bằng chuỗi string và chuyển tới URL đó.
2	replace(string)	Giống assign

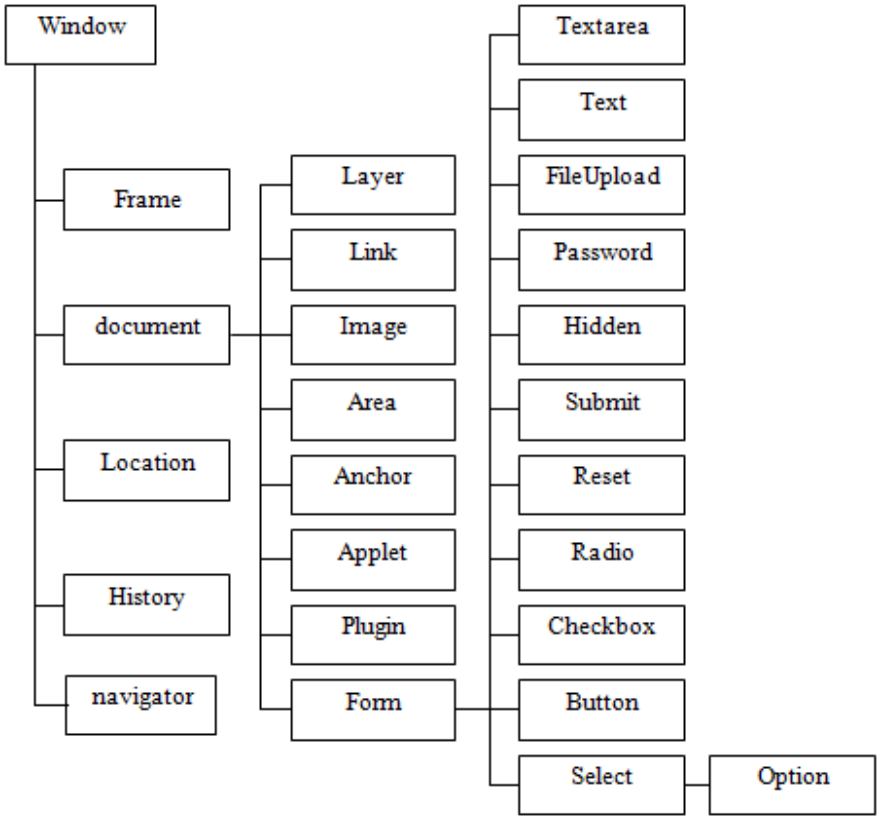
3	reload()	Load lại trang

Bảng 7.11. Các phương thức của đối tượng Location

1.4.9. Đối tượng Document

Trước khi đi vào giới thiệu đối tượng document. Tôi xin giới thiệu với các bạn mô hình DOM *(Document Object Model)* trong HTML. DOM là mô hình ánh xạ toàn bộ các thành phần của HTML thành các đối tượng. Như ta đã biết các đối tượng như location, navigator ... cùng là thuộc tính của đối tượng window.

Mô hình DOM *(Document Object Model)*:



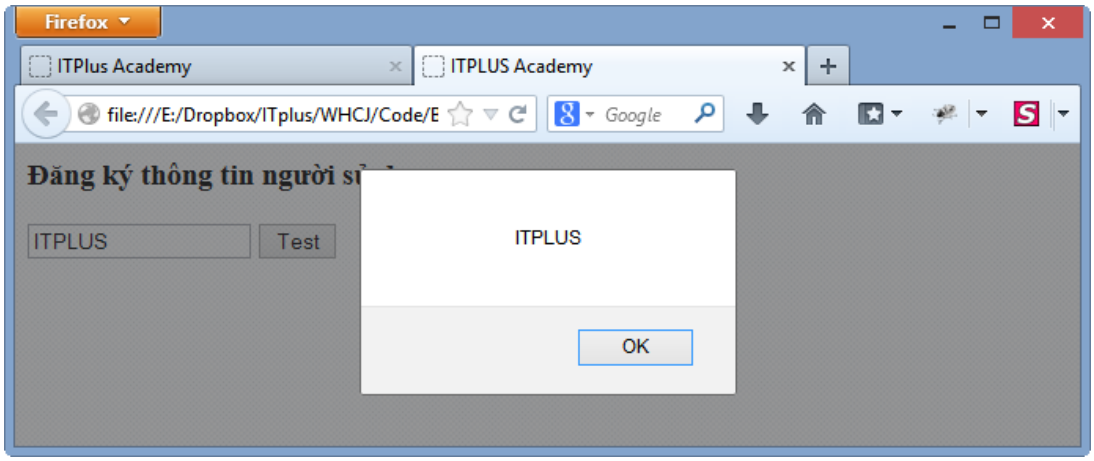
Hình 7.9. Cấu trúc DOM trong javascript

Với sơ đồ trên đây. Từ mỗi đối tượng cha ta có thể truy cập vào đối tượng con của nó thông qua thuộc tính name của thẻ con như là một thuộc tính của đối tượng cha. Và mỗi thuộc tính của thẻ cũng được coi là thuộc tính của đối tượng ánh xạ của thẻ đó.

Ví dụ ta có một file HTML như sau:

```
<html>
<head>
    <title>ITPLUS Academy</title>
</head>
<body>
<h3>Đăng ký thông tin người sử dụng</h3>

<form name="frmRegister" id="frmRegister" method="get" action="">
    <input type="text" name="txtTest"/>

    <input type="button" value="Test"
           onclick="alert(window.document.frmRegister.txtTest.value);"/>
</form>
</body>
</html>
```

Sau khi bạn điền dữ liệu vào `textbox` sẽ và nhấn nút bạn sẽ nhận được thông báo có nội dung là chuỗi bạn vừa tạo.



Hình 7.10. Lấy dữ liệu thông qua đối tượng DOM

Ở đây ta đã đi theo đúng mô hình DOM từ đối tượng gốc window cho tới thuộc value của đối tượng input có `name = "txtTest"` thuộc form có tên `"frmRegister"`. Tuy nhiên thuộc tính name thường chỉ dùng cho form. Vì vậy chúng ta cần phải dùng một số phương thức khác của đối tượng document để lấy ra thành phần mong muốn.

Mỗi tài liệu HTML đều có được ánh xạ bằng đối tượng `document`. Với đối tượng `document` bạn có thể truy cập toàn bộ các thành phần trên trang HTML

Các thuộc tính của đối tượng document:

STT	Tên thuộc tính	Mô tả
1	anchors	Trả về mảng các siêu liên kết trên trang HTML
2	forms	Trả về mảng các form
3	images	Trả về mảng các ảnh -thẻ <img> trên trang HTML
4	links	Trả về mảng các thẻ <link>
5	cookie	Trả về chuỗi cookie của trình duyệt

6	domain	Trả về tên miền
7	title	Thiết lập hoặc lấy về chuỗi tiêu đề của trang web
8	URL	Trả về chuỗi URL của trang web

Bảng 7.12. Các thuộc tính của đối tượng document

Các phương thức của document:

STT	Tên phương thức	Mô tả
1	getElementById(id)	Trả về một đối tượng DOM có thuộc tính ID bằng chuỗi truyền vào.
2	getElementsByName(name)	Trả về tập đối tượng DOM có thuộc tính name là chuỗi truyền vào.
3	getElementsByTagName(tagName)	Trả về tập đối tượng DOM có tên thẻ là chuỗi truyền vào. Ví dụ: `document.getElementsByTagName("a")` trả tập các thẻ <a>
4	write(string)	Chèn chuỗi tham số vào nội dung của thẻ `<body></body>`. Có thể là mã HTML hoặc mã javascirpt
5	writeln(string)	Giống `writeln` nhưng kèm theo ký tự xuống dòng.

Bảng 7.13. Các phương thức của đối tượng document

Với ví dụ lấy giá trị textbox ở trên ta có thể sửa thành:

```
<html>
<head>
    <title>ITPLUS Academy</title>
</head>
<body>
<h3>Đăng ký thông tin người sử dụng</h3>
<form name="frmRegister" id="frmRegister" method="get" action="">
    <input type="text" name="txtTest"/>
    <input type="button" value="Test" onclick= ”getValue()”/>
</form>
<script type= “text/javascript”>
    function getValue(){
    var frm = document.getElementById(“frmRegister”);
    alert(frm.txtTest.value);
    }
</script>
</body>
</html>
```

1.4.10. Đối tượng Cookie

Cookie là dữ liệu được lưu trữ trên máy trạm – client. Nó được đồng gửi đi khi chúng ta request tới server. Trường hợp dễ thấy là trình duyệt yêu cầu bạn có muốn lưu lại thông tin đăng nhập mỗi khi bạn đăng nhập và trang web hoặc bạn đánh dấu vào checkbox ghi nhớ trạng thái đăng nhập. Đó chính là việc lưu trữ thông tin đăng nhập của bạn bằng cookie và mỗi lần bạn và lại trang web đó server sẽ kiểm tra xem nếu có cookie rồi thì sẽ tự động đăng nhập luôn cho bạn.

Để tạo cookie ta sử dụng thuộc tính cookie của đối tượng document như sau:

```
document.cookie = “cookiename=namestring;expires=time”
```

Trong đó:

- `cookiname` là tên bạn muốn đặt cho cookie

- `namestring` là giá trị của cookie

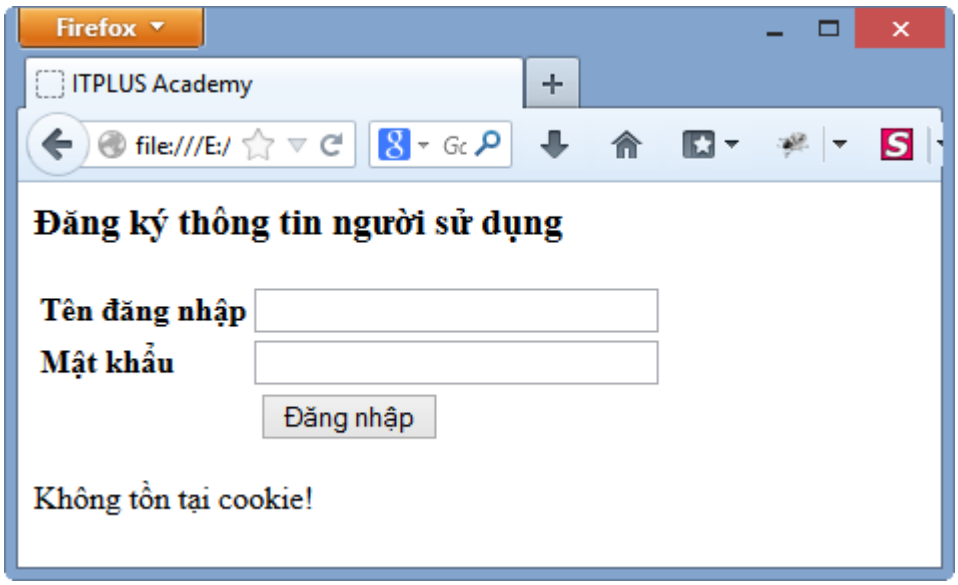
- `expries` là thời gian cho phép cookie tồn tại tính bằng đơn vị ngày. Đây là thuộc tính không bắt buộc.

Để có thể lấy được giá trị của cookie ta phải tự xây dựng phương thức. Ví dụ đăng nhập sau mô tả việc tạo cookie và lấy cookie bằng javascript:

```
<html>
<head>
    <title>ITPLUS Academy</title>
</head>
<body>
<h3>Đăng ký thông tin người sử dụng</h3>

<form id="frmRegister" method="get" action="">
    <table>
        <tr>
            <td><b>Tên đăng nhập</b></td>
            <td><input type="text" name="username" size="30"/></td>
```

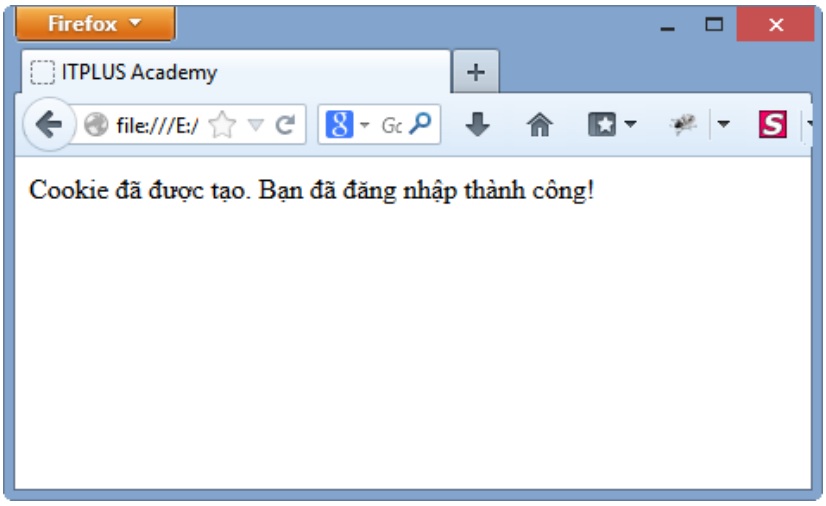
```
        </tr>
        <tr>
            <td><b>Mật khẩu</b></td>
            <td><input type="password" name="password" size="30"/></td>
        </tr>
        <tr>
            <td colspan="2" align="center">
                <input type="button" value="Đăng nhập"
onclick="createCookie();"/>
            </td>
        </tr>
    </table>
</form>
<script type="text/javascript">
    var username = getCookie("username1");
    var password = getCookie("password");
    if (username != "" && password != "") {
        document.body.innerHTML = "Cookie tồn tại. Chào mừng " + username +
" quay trở lại.";
    } else {
        document.write("Không tồn tại cookie!");
    }
    function getCookie(c_name) {
        var i, x, y, arrCookies = document.cookie.split(";");
        for (i = 0; i < arrCookies.length; i++) {
            x = arrCookies[i].substr(0, arrCookies[i].indexOf("="));
            y = arrCookies[i].substr(arrCookies[i].indexOf("=") + 1);
            if (x == c_name) return y;
            else return "";
        }
    }
    function createCookie() {
        var frm = document.getElementById("frmRegister");
        var strUsername = frm.username.value;
        var strPassword = frm.password.value;
        document.cookie = "username=" + strUsername + ";expires=30";
        document.cookie = "password=" + strPassword + ";expires=30";
        document.body.innerHTML = "Cookie đã được tạo. Bạn đã đăng nhập
thành công!";
    }
</script>
</body>
     </html>
```

Kết quả hiển thị trên trình duyệt như sau:



Hình 7.11. Ví dụ về cookie (cookie chưa tồn tại)

Sau khi bạn nhập tên đăng nhập, password và nhấn nút đăng nhập ta sẽ được:



Hình 7.12. Sau khi cookie đã được tạo

B. BÀI TẬP

2.1. Câu hỏi ôn tập

Câu 1. JavaScript là ngôn ngữ xử lý ở?

L. Client

M. Server

N. Server/client

O. Không ở dạng nào

Câu 2. Javascript là ngôn ngữ kịch bản có dấu được mã nguồn không?

I. Không dấu được vì các kịch bản chạy ở client

J. Dấu được vì chương trình hoạt động độc lập với trình duyệt

Câu 3. Trong Javascript sự kiện Onload thực hiện khi?

L. Khi bắt đầu chương trình chạy

M. Khi click chuột

N. Khi kết thúc một chương trình

O. Khi di chuyển chuột qua.

Câu 4. Lệnh lặp for có dạng như thế nào?

H. for (biến = Giá trị đầu; Điều kiện; Giá trị tăng)

I. for (biến = Giá trị đầu; Giá trị tăng; điều kiện)

J. for (biến = Điều kiện; Giá trị tăng; Giá trị cuối)

Câu 5: Hàm alert() dùng để làm gì?

Q. Dùng để hiện một thông báo

R. Dùng để hiện một thông báo nhập

S. Dùng để chuyển đổi số sang chữ

2.2. Bài tập

Bài 1. Viết chương trình nhập vào 2 số m(dòng), n(cột). Khi click chọn nút hiển thị thì xuất ra một bảng bao gồm m dòng và n cột.

Bài 2. Viết chương trình tính lương của nhân viên dựa theo thâm niên công tác (TNCT) tuỳ ý như sau; `Lương=hệ số * lương căn bản`, trong đó lương căn bản là 650.000 đồng.

- Nếu TNCT <12 tháng: hệ số =1.92
- Nếu TNCT 12 <= TNCT < 36 tháng : hệ số = 2.34
- Nếu 36<= TNCT <60 tháng: hệ số = 3
- Nếu TNCT >= 60 tháng: hệ số = 4.5

Bài 3: Viết chương trình kiểm tra dữ liệu của form nhập liệu như sau:

- Bắt buộc phải nhập username, password, email.
- Username và Password không được rỗng và chỉ bao gồm các ký tự a-z, 0-9, gồm 8 đến 15 ký tự và phải bắt đầu bằng chữ cái.
- Email phải đúng định dạng: tên mail bao gồm các ký tự từ a-z 0-9 dấu "_", "." Bắt đầu bằng ký tự chữ phân cách tên và tên miền bằng dấu "@". Tên miền email bao gồm các ký tự a-z, 0-9 dấu "-" và kết thúc bằng đuôi gồm dấu "." và các ký tự từ a-z ít nhất có 2 ký tự chữ
- Bắt buộc phải chọn một ngành nghề

BÀI 8. JQUERY VÀ CÁC HIỆU ỨNG ĐỘNG

Với sự phát triển nhanh chóng của Internet, người dùng ngày càng quan tâm hơn đến hình thức thể hiện của một trang web. Nếu như trước đây trang web chỉ cần có banner, nội dung và ít footer hời hợt là đã được cho là một trang web hoàn chỉnh. Nhưng bây giờ trang web đó phải có banner bắt mắt, nội dung hay và còn nhiều hiệu ứng lạ mắt khác nữa thì mới có thể thu hút được người đọc.

Chính vì thế những người lập trình web bắt đầu chú ý đến các thư viện Javascript mở như jQuery để tạo ra các hiệu ứng đẹp mắt có thể tương tác trực tiếp với người đọc một cách nhanh chóng và dễ dàng hơn rất nhiều là sử dụng thuần Javascript.

Bài này sẽ giới thiệu đến các bạn những nội dung cơ bản và cần thiết nhất đối với người lập trình web về jQuery.

A. NỘI DUNG

1.1. Tổng quan về jQuery

1.1.1. jQuery là gì?

a. Khái niệm

jQuery là một Javascript Framework, hỗ trợ các nhà lập trình web tạo ra các tương tác trên website một cách nhanh nhất. jQuery được khởi xướng bởi John Resig *(hiện là trưởng dự án của Mozzila)* vào năm 2006. jQuery là mã nguồn mở và hoàn toàn miễn phí và đặc biệt jQuery có một cộng đồng sử dụng đông đảo và được rất nhiều lập trình tham gia hoàn thiện, phát triển và viết Plugin.

Hình 8.1. jQuery và slogan "write less, do more"

b. Tại sao nên sử dụng jQuery

- jQuery đơn giản hóa cách viết Javascript và tăng tốc độ xử lý các sự kiện trên trang web, giúp tiết kiệm thời gian và công sức so với cách viết Javascript thông thường.

- Bên cạnh đó, việc sử dụng jQuery giúp giải quyết tốt vấn đề xung đột giữa các trình duyệt web.

- Học jQuery rất đơn giản, nếu như bạn nắm vững HTML & CSS, bạn có thể tiếp cận và sử dụng jQuery nhanh chóng.

c. Tác dụng của jQuery

- *Hướng tới các thành phần trong HTML*: nếu không sử dụng thư viện Javascript này, bạn phải viết rất nhiều dòng code mới có thể đạt được mục tiêu là di chuyển trong cấu trúc cây *(DOM - Document Object Model)* của một tài liệu HTML và chọn ra các thành phần liên quan. jQuery cho phép bạn chọn bất cứ thành phần nào của tài liệu một cách dễ dàng dựa vào jQuery selector.

- *Thay đổi giao diện của một trang web (thay đổi HTML và CSS)*: CSS là công cụ rất mạnh để định dạng một trang web nhưng nó có một nhược điểm là không phải tất cả các trình duyệt đều hiển thị giống nhau. Cho nên jQuery ra đời để lấp chỗ trống này, vì vậy bạn có thể sử dụng nó để giúp trang web có thể hiển thị tốt trên hầu hết các trình duyệt. Hơn nữa jQuery cũng có thể thay đổi class hoặc những định dạng CSS đã được áp dụng lên bất cứ thành phần nào của HTML ngay cả khi trang web đó đã được trình duyệt load thành công.

- *Thay đổi nội dung của tài liệu*: jQuery không chỉ thay đổi bề ngoài của trang web, mà còn có thể thay đổi nội dung của chính tài liệu đó. Nó có thể thêm hoặc bớt nội dung trên trang, thậm chí cả cấu trúc HTML của một trang web cũng có thể được viết lại và mở rộng.

- *Tương tác với người dùng*: jQuery cho bạn nhiều phương thức để tương tác với người dùng và tối giản các mã Event trong code HTML.

- *Tạo hiệu ứng động*: jQuery cho phép bạn sử dụng rất nhiều hiệu ứng động như mờ dần, slideUp, slideDown()…

- *Hỗ trợ Ajax*: đây chính là công nghệ ngày càng trở nên phổ biến *Asynchronous JavaScript And XML* (AJAX), nó giúp người thiết kế web tạo ra những trang web tương tác cực tốt và nhiều tính năng. Thư viện jQuery loại bỏ sự phức tạp của trình duyệt trong quá trình này và cho phép người phát triển web có thể tập trung vào các tính năng đầu cuối, đơn giản hoá các tác vụ Javascript.

d. Ưu điểm của jQuery

Ngoài jQuery còn có các thư viện javascript nổi tiếng khác như mooTools, Prototype, Dojo... thế nhưng jQuery có vẻ thu hút các lập trình web hơn là vì nó có các ưu điểm:

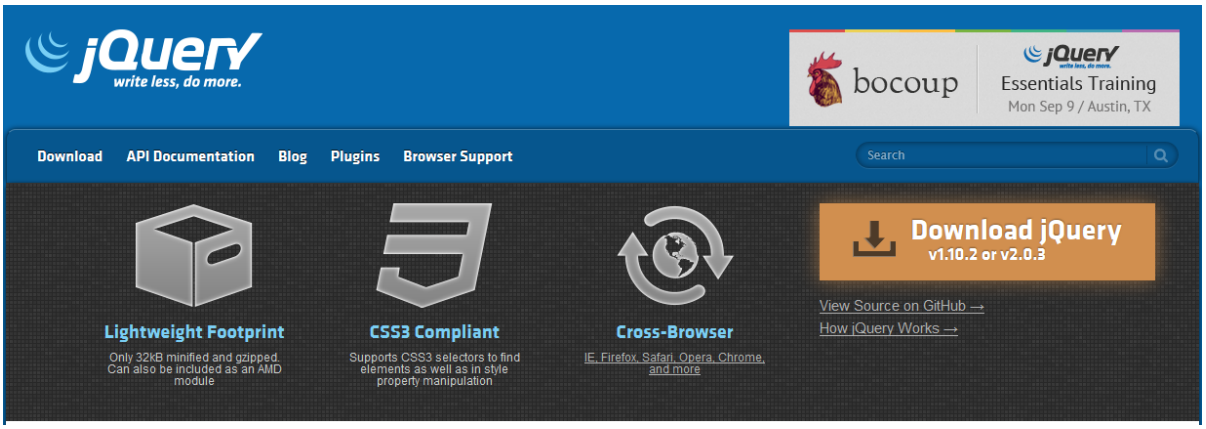
- Hỗ trợ tốt việc xử lí các vấn đề thường gặp (DOM, AJAX…)
- Tương thích nhiều trình duyệt web phổ biến.
- Nhỏ gọn, dễ dùng, có nhiều tài liệu hướng dẫn sử dụng chi tiết.
- Ít xung khắc với các thư viện Javascript khác.

- Plugin *(thư viện ứng dụng)* phong phú.

1.1.2. Cách sử dụng jQuery

Bởi vì jQuery là một thư viện JavaScript *(Javascript Framework)* do vậy để sử dụng nó bạn phải chèn nó vào trang web giống như làm với một file javascript bình thường. Có hai cách để chèn jQuery vào một trang web.

a. Tự host jQuery



Hình 8. 2. Giao diện website của jQuery

Vào trang chủ của jQuery *(http://jquery.com)* và download phiên bản mới nhất. Thường thì có 2 phiên bản của jQuery cho bạn download. Phiên bản chưa nén dành cho những người phát triển và đang học như bạn. Còn phiên bản nén kia dành cho phần sử dụng trực tiếp trên trang vì nó có dung lượng nhỏ hơn rất nhiều so với phiên bản chưa nén. Bạn không cần phải cài đặt jQuery, bạn chỉ cần đặt đường link tới thư viện đó là được. Bất cứ khi nào bạn cần sử dụng jQuery, bạn chỉ cần gọi nó trong tài liệu HTML đến nơi lưu trữ nó trên host của bạn.

Cú pháp như sau:

```
<script src="javascript/jquery.min.js" type="text/javascript"></script>
```

Cụ thể như trong đoạn mã trên, file jquery được tải về có tên là `jquery.min.js` được đặt trong thư mục `javascript`.

b. Dùng phiên bản có sẵn trên server của Google

Ngoài cách trên ra bạn cũng có thể sử dụng phiên bản nén của jQuery có sẵn trên server của Google. Sử dụng cách này vừa có thể tiết kiệm băng thông cho trang web của bạn và vừa được load nhanh hơn nếu máy của người dùng đã cache jQuery.

Cú pháp để chèn jQuery sử dụng file có sẵn trên server của Google như sau:

```
<script src=”
http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>
```

Lưu ý: Thư viện jQuery được đặt ở dưới Stylesheet. Đây là một điểm rất quan trọng bạn cần lưu ý là thứ tự của các file khi gọi. Ban đầu phải là CSS load trước, khi trang web đã load xong phần CSS thì chúng ta mới thêm vào phần thư viện jQuery cuối cùng mới là code jQuery chúng ta tự viết ra. Nếu không khi code jQuery của bạn sẽ không làm việc đúng như mong đợi nếu thư viện jQuery chưa được load.

1.2. Cú pháp cơ bản của jQuery

Cú pháp jQuery được dùng để chọn các thẻ HTML và tương tác các hành vi với các thẻ đó. Cú pháp cơ bản rất đơn giản như sau:

```
$(document).ready(function() {
 $(selector).action();
});
```

- `$(document).ready(function()`: Hàm sẵn sàng thực thi trong 1 tài liệu (The Document Ready Function). Hàm này dùng để ngăn chặn bất cứ code jQuery nào chạy trước khi trang HTML tải về hoàn thành. Như vậy hàm này đảm bảo dữ liệu chạy đúng kết quả.

- Dấu `$` là để định nghĩa jQuery.

- Phần `(selector)` là phần tìm các thẻ HTML thông qua tên thẻ, thuộc tính, lớp.

- Phần `action()` là hành vi cần thực hiện cho các thẻ HTML sau khi tìm được.

Ví dụ:

- `$(this).hide()`: Ẩn thẻ HTML hiện tại

- `$("p").hide()`: Ẩn tất cả các thẻ <p> (thẻ xuống dòng)

- `$("p.test").hide()`: Ẩn tất cả các thẻ `<p>` có `class = "test"`. Chẳng hạn: `<p class="test">thẻ này ẩn</p>` và `<p> thẻ này không ẩn vì không có class = "test"</p>`

- `$("#test").hide()`: Ẩn các thẻ có `id = "test"`

jQuery là sự kết hợp tuyệt vời giữa cú pháp chọn lựa của XPath và CSS vì vậy nếu bạn nắm vững 2 phần đó bạn sẽ học jQuery nhanh hơn rất nhiều.

1.3. Truy xuất các phần tử và CSS qua jQuery Selector

jQuery selector tận dụng thế mạnh của CSS selector nhằm giúp người mới tìm hiểu về jQuery có thể nắm bắt nhanh chóng khi đã làm việc và có kiến thức về CSS. Do đó jQuery selector tương tự như CSS selector.

jQuery selector giúp chúng ta dễ dàng truy vấn đến các phần tử DOM *(Document Object Model - Mô hình đối tượng tài liệu)* một cách nhanh nhất, code đơn giản và ngắn ngọn nhất.

Ví dụ: Để truy vấn đến phần tử có id là header, với cách viết javascript thông thường bạn phải viết như sau:

```
document.getElementById("header");
```

Thì với jQuery, công việc chỉ còn đơn giản như sau:

```
$("#header");
```

1.3.1. Cú pháp jQuery selector

jQuery selector có cấu trúc như sau:

```
$("CSS selector")
```

jQuery selector được bắt đầu bằng `$` kết hợp với các `CSS selector` trong cặp dấu `(' và ')` hoặc `(" và ")`. Bạn cũng có thể sử dụng `jQuery` để thay cho dấu `$`:

```
jQuery('CSS selector')
```

1.3.2. jQuery CSS selector

Cũng giống như CSS selector, 3 thành phần được sử dụng nhiều nhất trong jQuery selector là tên thẻ `(type selector)`, `ID` và `Class`: Bảng sau sẽ giới thiệu các thành phần jQuery Selector:

Thành phần Selector	CSS selector	jQuery selector	Mô tả
Tên thẻ HTML (type selector)	div	$('div')	Chọn tất cả các thẻ div trong tài liệu HTML
ID	#header	$('#header')	Chỉ chọn phần tử có id là header
Class	.content	$('.content')	Chọn tất cả các phần tử có class là content
Universal selector	*	$("*")	Chọn tất cả các phần tử trong tài liệu HTML
Descendant selector	form input	$('form input')	Chọn tất cả các thành phần input của form
Child selector	ul > li	$('ul > li')	Chọn tất cả các thành phần li (cấp 1) trong danh sách *(HTML List)*
Adjacent selector	div + em	$('div + em')	Chọn thành phần em liền kề phía sau của div trong tài liệu HTML

Sibling selector	div ~ em	$('div ~ em')	Chọn thành phần em liền kề phía trước của div trong tài liệu HTML

Bảng 8.1. Bảng jQuery Selector

1.3.3. jQuery Attribute selector

Attribute selectors là cách chọn các thành phần trong tài liệu HTML dựa vào thuộc tính của một hay nhiều thẻ HTML nào đó. Với Attribute selector chúng ta có thể chọn được các đối tượng mà không cần phải khai báo thêm các `Class` hoặc `Id` vào trong thẻ HTML mà vẫn có thể hướng được đến các thành phần đó. Tương tự bạn cũng có thể sử dụng các *CSS Attribute selectors* trong *jQuery selector*.

Ví dụ:

```
$('input[type="text"]') //Chọn tất cả các thành phần input có type là text
$('img[title]') //Chọn tất cả các hình ảnh được gán thuộc tính title
```

1.3.4. Các selector riêng của jQuery

Các selector riêng của jQuery khá giống với Pseudo selector (Ảo) của CSS, các selector riêng của jQuery là những selector được thêm vào sau dấu `(:)`:

```
$(':input') //Chọn tất cả các phần tử input
```

Hoặc chúng ta cũng có thể kết hợp để chọn các phần tử rõ ràng hơn như sau:

```
$('input:text') //Chọn tất cả các thành phần input có type="text"
```

- *Form selector*: jQuery cung cấp những selector để quá trình làm việc với Form trở nên đơn giản hơn. Bảng bên dưới mô tả các jQuery selector khi làm việc với Form:

Selector	jQuery selector	Mô tả

:text, :checkbox, :radio, :image, :file, :hidden, :password, :submit, :reset	`$(':selector')`
	Ví dụ: `$(':text')`
	//Chọn tất cả các thành phần
	`<input type="text" />`
		Chọn tất cả các thành phần input có thuộc tính type có giá trị tương ứng với selector

:input	`$(':input')`	Chọn tất cả các thành phần input của form
:button	`$(':button')`	Chọn tất cả các thành phần button hoặc các thành phần `<input type="button" />`
:enable	`$(':enable')`	Chọn tất cả các phần tử form được bật

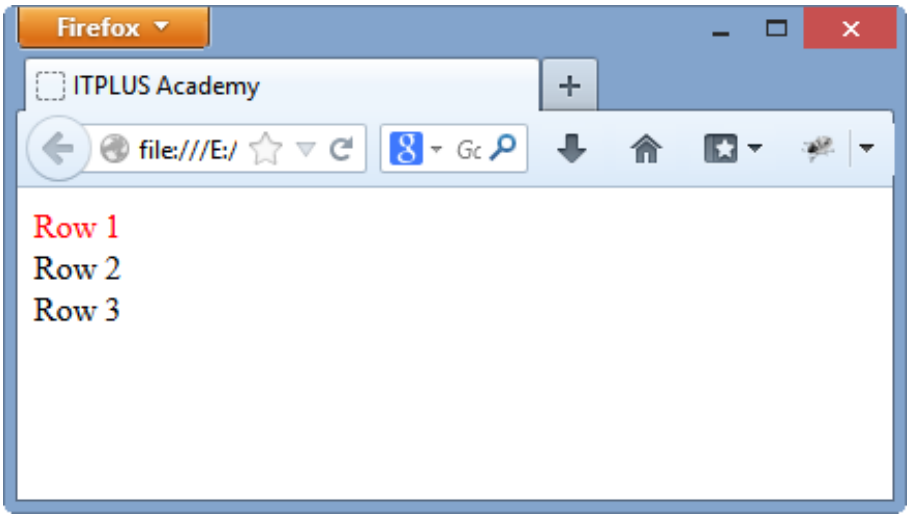
:disable	`$('':disable')`	Chọn tất cả thành phần Form được tắt
:checked	`$(:checked')`	Các nút radio hoặc các thành phần checkbox đang được chọn
:selected	`$('.selected')`	Các phần tử option của `<select>` đang được chọn

- *:first selector*: Cho phép chọn phần tử đầu tiên theo thứ tự xuất hiện của nó trong tài liệu HTML.

Ví dụ: Chúng ta sẽ sử dụng `:first selector` để chọn thành phần `<div>` đầu tiên ở ví dụ trên như sau:

```
<html>
<head>
  <title>ITPLUS Academy</title>
      <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>

  <script type="text/javascript">
      $(document).ready(function() {
            $("div:first").css("color","red");
      });
</script>
</head>
<body>
      <div>Row 1</div>
      <div>Row 2</div>
      <div>Row 3</div>
</body>
</html>
```

Kết quả là thành phần `<div>Row 1</div>` sẽ được chọn và hiện thị với màu đỏ:



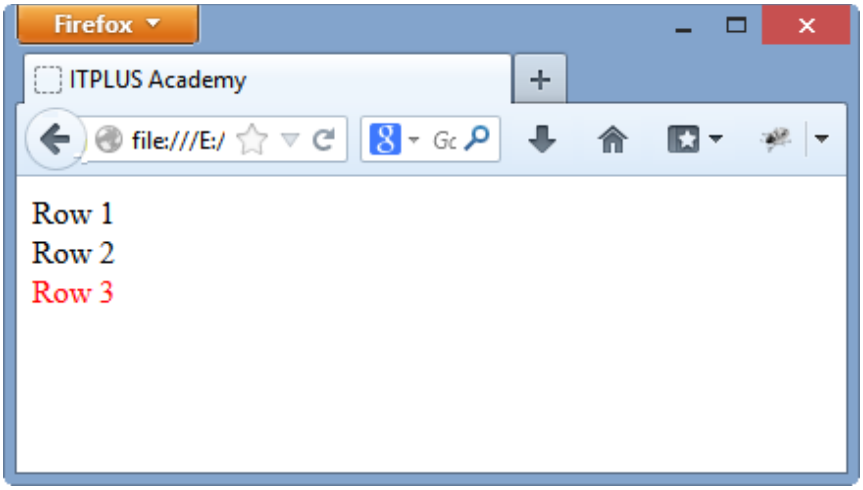
Hình 8.3. Sử dụng :first selector

- *:last selector*: Ngược lại với `:first selector`, `:last selecotor` sẽ chọn phần tử cuối cùng theo thứ tự xuất hiện của nó trong tài liệu HTML.

Ví dụ: Chúng ta sẽ chọn thành phần `<div>Row 3</div>` từ ví dụ trên như sau:

```
<script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>

  <script type="text/javascript">
      $(document).ready(function() {
            $("div:last").css("color","red");
      });
</script>
```

Kết quả thu được như sau:



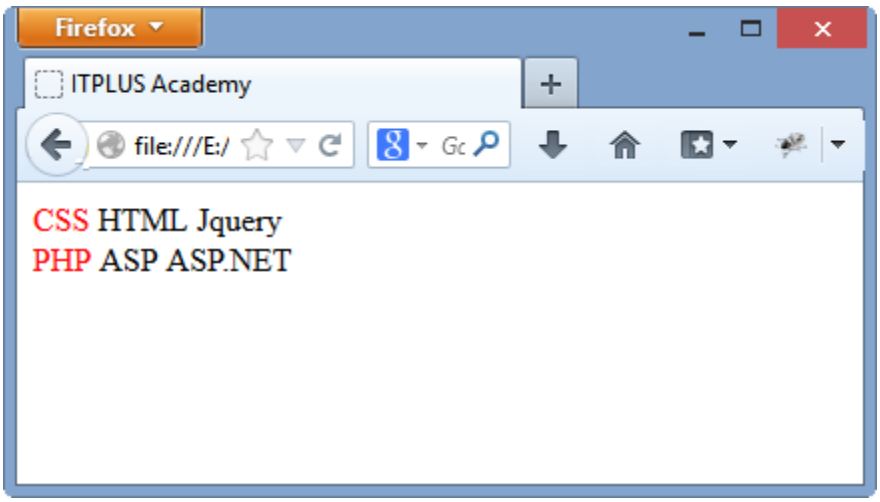
Hình 8.4. Sử dụng :last selector

- *:first-child selector*: Cho phép chọn thành phần con đầu tiên của một phần tử nào đó trong cấu trúc DOM của tài liệu HTML.

Ví dụ: Chúng ta sử dụng dụng `:first-child` để chọn thành phần `<span>CSS</span>` và `<span>PHP</span>` là con đầu tiên của `div` trong code HTML và cho hiện thị với màu đỏ như sau:

```
<script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>

  <script type="text/javascript">
      $(document).ready(function() {
            $("div span:first-child").css("color", "red");
      });
</script>
```

Kết quả:



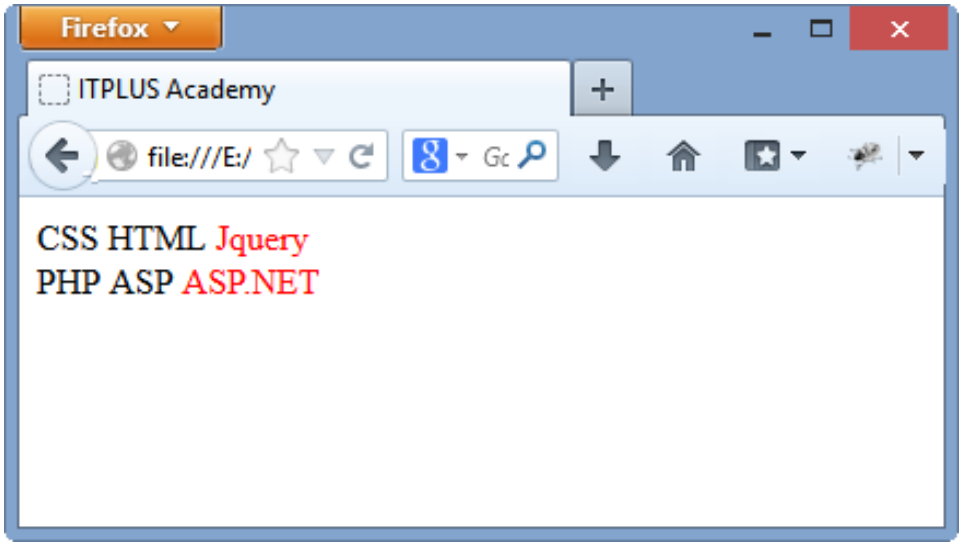
Hình 8.5. Sử dụng :first-child

- *:last-child selector*: Ngược lại với :first-child selector, :last-child chọn thành phần là con cuối cùng của một phần tử nào đó trong cấu trúc DOM

Ví dụ: Chúng ta sẽ sử dụng `:last-child` selector để chọn thành phần `<span>Jquery</span>` và `<span>ASP.NET</span>` ở ví dụ trên như sau:

```
<script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>

  <script type="text/javascript">
      $(document).ready(function() {
            $("div span:last-child").css("color", "red");
      });
</script>
```

Kết quả thu được như sau:



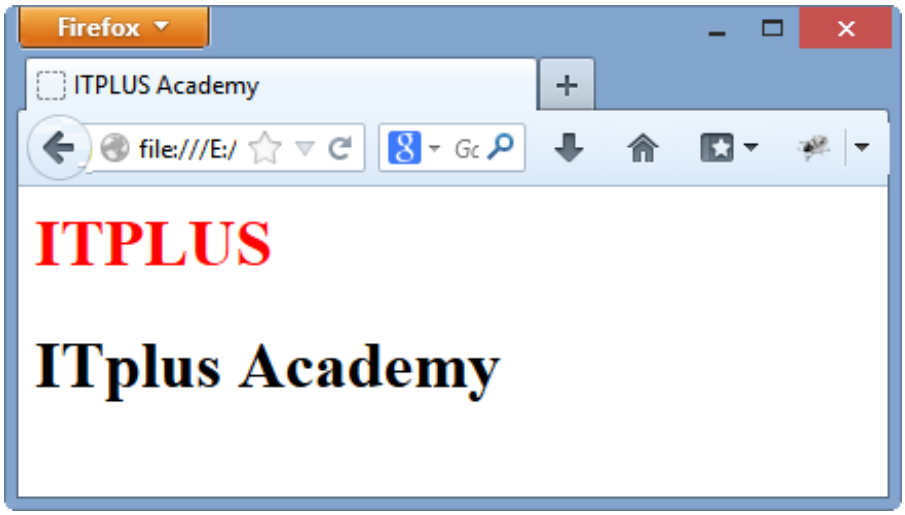
Hình 8.6. Sử dụng :last-child

- *:contains selector*: Cho phép chọn tất cả các phần tử mà nội dung của nó có chứa nội dung do chúng ta chỉ định. Nội dung chỉ định phân biệt chữ hoa và chữ thường ví dụ *ITPLUS* khác *itplus.*

Ví dụ: Ở ví dụ sau, chúng ta sẽ chọn tất cả các thành phần `h1` mà nội dung của nó có chứa từ ITPLUS.

```
<html>
<head>
  <title>ITPLUS Academy</title>
  <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $('h1:contains("ITPLUS")').css("color", "red");
    });
  </script>
</head>
<body>
      <h1>ITPLUS</h1>
  <h1>ITplus Academy</h1>
</body>
</html>
```

Kết quả:

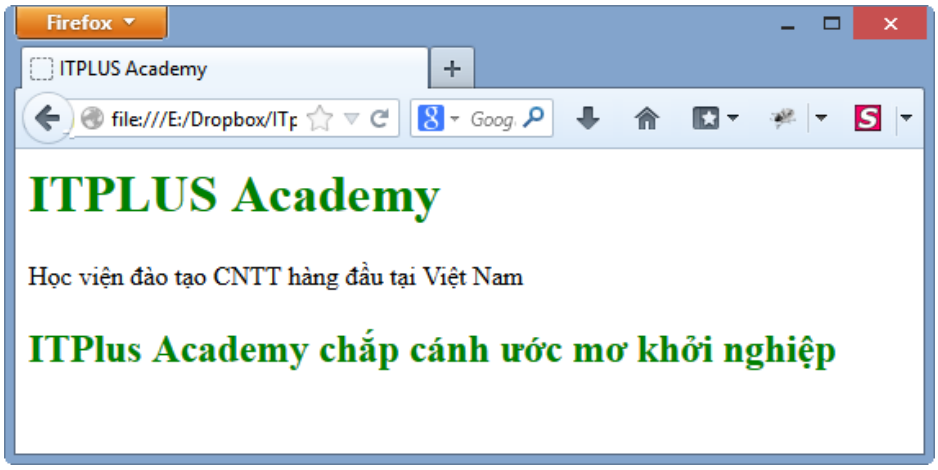


Hình 8.7. Sử dụng :contains selector

:header selector: sẽ chọn toàn bộ các thành phần Heading tags (Các thẻ từ h1 đến h6).

```
<html>
<head>
  <title>ITPLUS Academy</title>
  <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $(':header').css("color", "green");
    });
  </script>
</head>
<body>
      <h1>ITPLUS Academy</h1>
  <p>Học viện đào tạo CNTT hàng đầu tại Việt Nam</p>
  <h2>ITPlus Academy chắp cánh ước mơ khởi nghiệp</h2>
</body>
</html>
```

Ví dụ: Sử dụng `:header selector` để chọn thành phần h1 và h2 và cho hiện thị với màu xanh như sau:



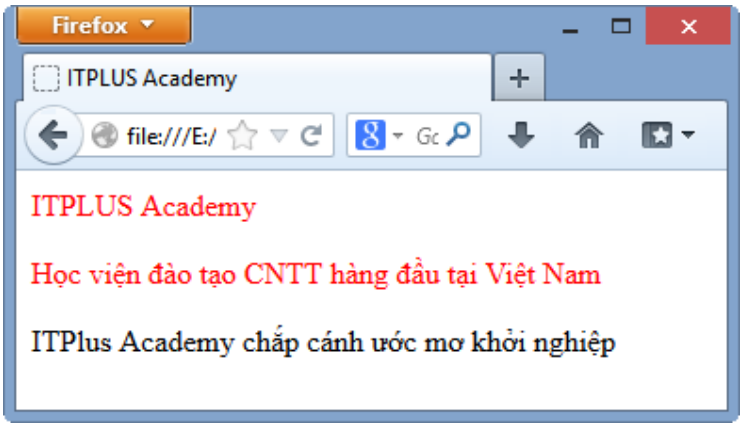
Hình 8.8. Sử dụng :header selector

- *:has() selector*: là cách chọn tất cả các phần từ mà nó có ít nhất một phần tử con được chỉ định.

Ví dụ: Chọn tất cả các thành phần div sao cho nó có chứa ít nhất một thành phần p:

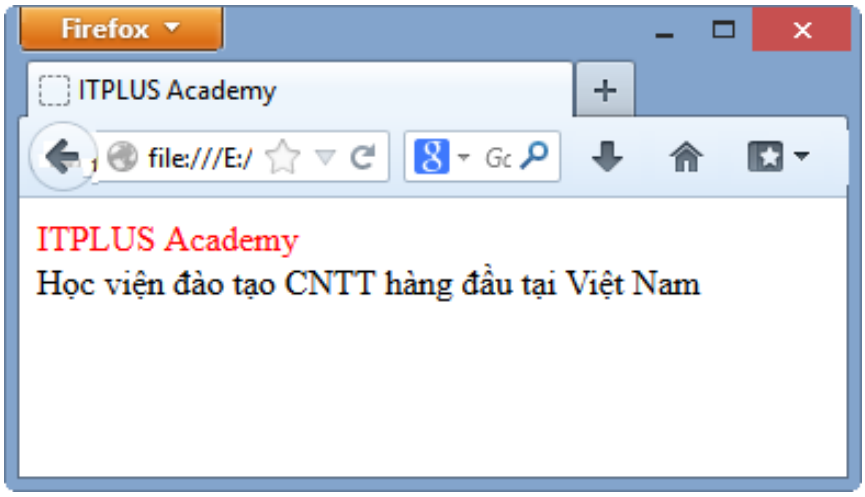
```
<html>
<head>
  <title>ITPLUS Academy</title>
  <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $('div:has(p)').css("color", "red");
    });
  </script>
</head>
<body>
      <div>ITPLUS Academy<p>Học viện đào tạo CNTT hàng đầu tại Việt
Nam</p></div>
  <div>ITPlus Academy chắp cánh ước mơ khởi nghiệp</div>
</body>
</html>
```

Kết quả thu được như sau:



Hình 8.9. Sử dụng :has() selector

- *:empty selector()*: sẽ chọn tất cả các thành phần mà nó không chứa các thành phần con, hoặc không có nội dung (empty)



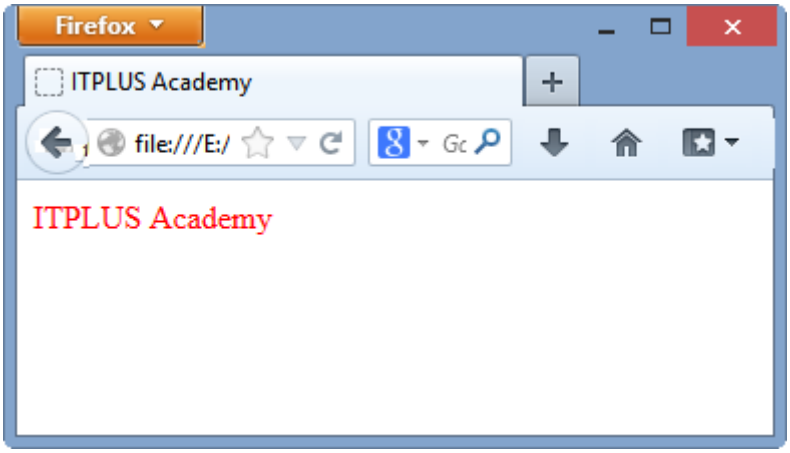
Hình 8.10. Sử dụng :empty selector

Ví dụ: Với thành phần `div` đầu tiên không có dữ liệu, ta sẽ sử dụng jQuery để thêm vào thành phần này một câu thông báo như sau:

```
<html>
<head>
  <title>ITPLUS Academy</title>
  <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>
  <script type="text/javascript">
    $(document).ready(function () {
      $('div:empty').text('ITPLUS Academy').css("color", "red");
    });
  </script>
</head>
<body>
      <div></div>
      <div>Học viện đào tạo CNTT hàng đầu tại Việt Nam</div>
</body>
</html>
```

- *:parent selector*: ngược lại với :empty, nó sẽ chọn tất cả các thành phần tồn tại nội dung bên trong nó hoặc chứa các thành phần con.

Cũng với ví dụ ở trên, kết quả nhận được khi thay `:empty` selector thành `:parent` selector là:



Hình 8.11. Sử dụng :parent selector

- *:eq() selector*: cho phép chúng ta chọn phần tự theo chỉ số (index) của nó trong cấu trúc HTML, tương tự như mảng trong javascript, các chỉ số được bắt đầu bằng vị trí 0

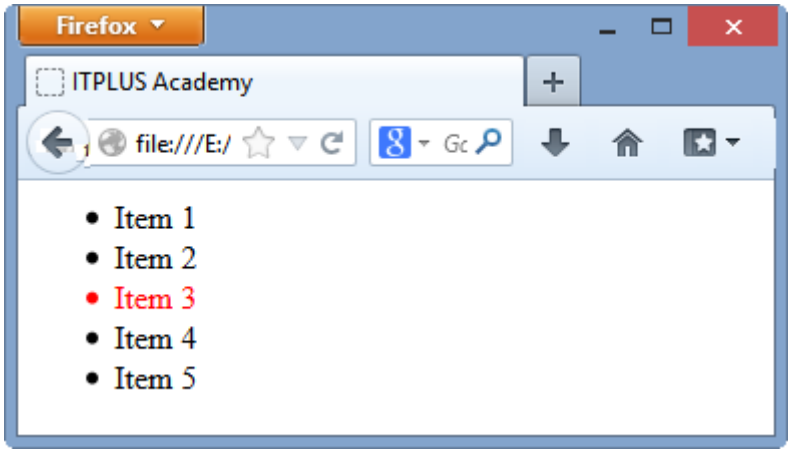
Ví dụ: Giả sử ta có đoạn mã HTML như sau:

```
<ul>
      <li>Item 1</li>
   <li>Item 2</li>
   <li>Item 3</li>
   <li>Item 4</li>
   <li>Item 5</li>
</ul>
```

Nếu chúng ta chọn thành phần `<li>Item 3</li>` và bôi màu đỏ cho nó thì cần thực hiện như sau:

```
$('li:eq(2)').css("color", "red");
```

Kết quả thu được như sau:



Hình 8.12. Ví dụ về :eq() selector

- *:lt() selector*: lt là viết tắt của less than *(nhỏ hơn, ít hơn)*. *lt() selector* là cách chọn các phần tử có vị trí nhỏ hơn giá trị index, index được đánh dấu từ vị trị 0

Tương tự như đối với *eq() selector*, nếu muốn chọn thành phần `<li>Item 1</li>` đến `<li>Item 4</li>` ta làm như sau:

```
$('li:lt(4)').css("color", "red");
```

- *:gt() selector*: gt là viết tắt của greater than (lớn hơn). gt() selector cho phép chọn các phần tử có vị trí lớn hơn giá trị index, index được đánh dấu từ vị trị 0.

Tương tự như đối với *eq() selector*, nếu muốn chọn thành phần `<li>Item 3</li>` đến `<li>Item 5</li>` ta làm như sau:

```
$('li:gt(1)').css("color", "red");
```

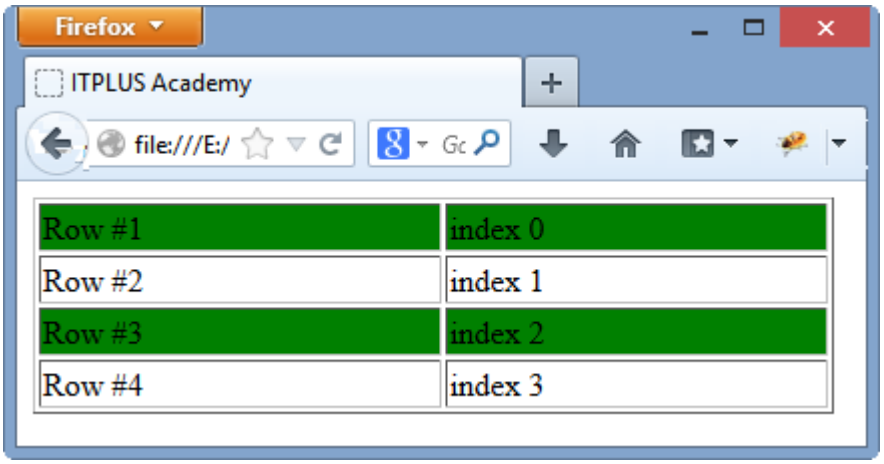
- *:odd selector và :even selector*: là cách chọn các đối tượng theo vị trí chẵn, lẽ. Thường được sử dụng để định dạng bảng theo kiểu kẻ sọc *(giữa các dòng chẵn lẻ)*.

Ví dụ: Ta có 1 bảng gồm có 4 dòng. Ta có thể tô màu xanh cho dòng 1 và dòng 3 như sau:

```
<html>
<head>
  <title>ITPLUS Academy</title>
  <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>
  <script type="text/javascript">
    $(document).ready(function () {
```

```
      $('tr:even').css("background", "green");
    });
  </script>
</head>
<body>
      <table width="400" border="1">
    <tr><td>Row #1</td><td>index 0</td></tr>
    <tr><td>Row #2</td><td>index 1</td></tr>
    <tr><td>Row #3</td><td>index 2</td></tr>
    <tr><td>Row #4</td><td>index 3</td></tr>
      </table>
</body>
</html>
```

Kết quả thu được:



Hình 8.13. Sử dụng :even selector

Chúng ta đang định dạng cho các ô lẻ, nhưng lại sử dụng even *(chẵn)*, Bởi vì chỉ số *(index)* của odd và even được đánh từ 0, do đó dòng Row #1 có chỉ số là 0 *(chẵn)*, Row 3 có chỉ số là 2 *(chẵn)* và chúng ta sử dụng even để chọn các dòng có chỉ số chẵn này, và trong thực tế đó là các dòng lẻ Row #1 và #3.

Tương tự, chúng ta có thể sử dụng `$("tr:odd")` để định dạng cho các dòng Row #2 và #4.

- *:nth-child selector*: cho phép chọn các phần tử theo vị trí của *index*, các giá trị của odd và even. Khác với eq, gt, lt, odd, even, chỉ số của :nth-child được bắt đầu từ số 1, và đây là selector duy nhất của jQuery có chỉ số được bắt đầu từ 1.

Cũng với ví dụ trên, ta có thể định dạng các dòng Row #1 và Row #3 *(các dòng lẻ)* bằng odd *(lẻ)* kết hợp với `:nth-child` như sau:

```
$('tr:nth-child').css("background", "green");
```

1.4. Các sự kiện với jQuery

jQuery cung cấp cho chúng ta khá nhiều các sự kiện đủ để thao tác với các thành phần trên website mà chúng ta mong muốn. Bên cạnh đó, jQuery còn loại bỏ hoàn toàn các event code ra khỏi mã HTML như cách viết javascript thông thường.

Trong javascript, để xuất hiện thông báo với người dùng khi click vào liên kết bạn phải viết như sau:

```
<a class="click" href="#" onclick="arlert('Welcome to ITPLUS')">Click me</a>
```

Bạn nhận thấy rằng chúng ta phải viết mã để gọi sự kiện onclick ngay trong liên kết `<a>`. Nhưng với jQuery thì chúng ta có thể sử dụng bộ selector để thao tác với các sự kiện như sau:

```
$("a.click").click(function() { alert("Welcome to ITPLUS"); });
```

1.4.1. Bộ quản lý sự kiện

Javascript cung cấp bộ quản lý sự kiện `window.onload` cho phép chúng ta thực thi một hàm nào đó. Trong khi đó jQuery cung cấp cho chúng ta bộ quả lý sự kiện `$(document).ready()` nhưng với những ưu điểm vượt trội hơn sự kiện `window.onload`.

`Window.onload` sẽ được thực thi sau khi trình duyệt tải xong toàn bộ tài liệu cần thiết bao gồm stylesheet, hình ảnh. Trong khi đó `$(document).ready()` sẽ được thực thi ngay khi các phần tử DOM được sẵn sàng mà không cần đợi các tài liệu như stylesheet hay hình ảnh tải xong. Do đó quá trình xử lý vào thao tác sẽ diễn ra nhanh hơn.

Cách đăng ký bộ quản lý sự kiện trong jQuery:

```
$(document).ready(function () {
     //Code jQuery o day
});
```

Hoặc

```
$().ready(function () {
     //Code jQuery o day
});
```

Hoặc

```
$(function(){
     //Code jQuery o day
})
```

<u>Lưu ý</u>: Để đảm bảo rằng trang web vẫn có định dạng trước khi code JavaScript được thực hiện, người ta thường đặt `<link rel="stylesheet">` đằng trước thẻ `<script>` trong phần `<head>` của tài liệu.

1.4.2. Một số sự kiện thường sử dụng trong jQuery

a. Sự kiện đối với trình duyệt

- *error()*: Xảy ra khi trình duyệt gặp lỗi trong quá trình tải tài liệu, ví dụ liên kết đến hình ảnh bị gãy, …

- *resize()*: khi người dùng thay đổi kích thước cửa sổ trình duyệt

- *scroll()*: Xử lý tác tác vụ khi người dùng cuộn trang

b. Sự kiện với Form

- *focus()*: khi con trỏ chuột đang tác động tại phần tử Form

- *blur()*: Thực hiện khi click chuột ra ngoài phần tử form

- *change()*: khi giá trị của phần tử form được thay đổi

- *select()*: khi phần tử form được chọn

- *submit()*: Khi form được submit

c. Sự kiện với bàn phím

- *focusin()* : tương tự focus()

- *focusout()*: tương tự blur()

- *keyup(), keydown(), keypress()*: Xảy ra khi đang nhập liệu vào Form

d. Sự kiện đối với chuột

- *click()*: Khi người dùng click chuột

- *hover()*: di chuyển chuột lên đối tượng

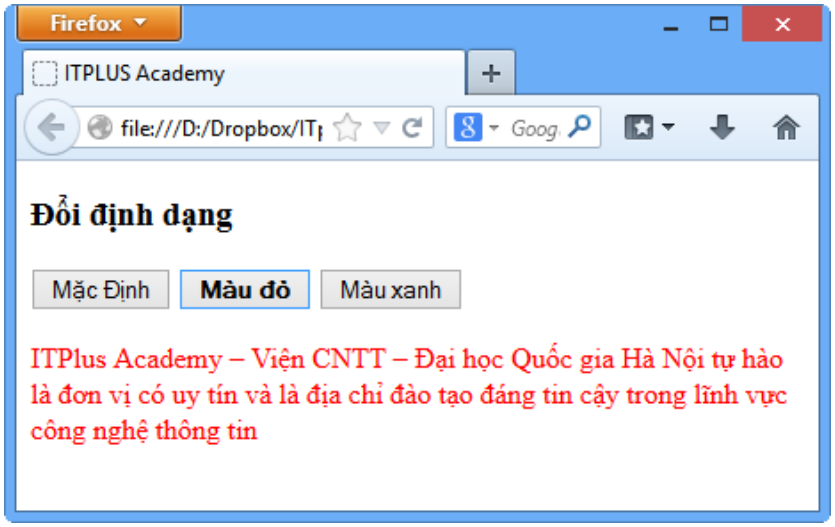
- *toggle()*: Thực hiện theo các lần click chuột của người dùng, ví dụ thay đổi giữa trạng thái ẩn hiện của phần tử

Ví dụ: Để minh hoạ cho các sự kiện với jQuery, giả sử ta muốn có một trang web có thể thay đổi màu sắc các đoạn văn tuỳ theo ý của người dùng. Chúng ta sẽ cho phép người dùng nhấp chuột vào 3 nút để thay đổi màu sắc theo kiểu. Mặc định, Màu đỏ và Màu Xanh. Khi nhấn vào nút Màu Đỏ, thì nội dung sẽ chuyển thành màu đỏ, khi nhấn vào nút Màu Xanh thì nội dung sẽ thành màu xanh và cuối cùng khi nhấn vào nút Mặc định thì nội dung quay về trạng thái ban đầu.

```
<html>
<head>
    <title>ITPLUS Academy</title>
    <style>
            .selected {font-weight: bold;}
            body.red .text {color: red;}
            body.green .text{color: green;}
    </style>
    <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>
    <script type="text/javascript">
        $(document).ready(function() {
            $('#switcher-default').bind('click', function() {
               $('body').removeClass('red').removeClass('green');
                $('#switcher .button').removeClass('selected');
                $(this).addClass('selected')
            });
            $('#switcher-red').bind('click', function(){
                $('body').addClass('red').removeClass('green');
                $('#switcher .button').removeClass('selected');
```

```
                $(this).addClass('selected');
            });
            $('#switcher-green').bind('click', function() {
                $('body').addClass('green').removeClass('red');
                $('#switcher .button').removeClass('selected');
                $(this).addClass('selected')
            });
        });
    </script>
</head>
<body>

<div id="switcher">
      <h3>Đổi định dạng</h3>
      <button id="switcher-default">Mặc Định</button>
    <button id="switcher-red">Màu đỏ</button>
    <button id="switcher-green">Màu xanh</button>
</div>
<div class="text">
    <p>
        ITPlus Academy - Viện CNTT - Đại học Quốc gia Hà Nội
        tự hào là đơn vị có uy tín và là địa chỉ đào tạo đáng tin cậy
        trong lĩnh vực công nghệ thông tin
    </p>
</div>
</body>
</html>
```

Kết quả:



Hình 8.14. Sự kiện trong jQuery

1.4.3. Gắn kết Event Handler sử dụng phương thức .bind()

Trong một trang web, trình duyệt sẽ quản lý cách thức hiển thị của trang tương ứng với mã HTML và CSS. Phần mã kịch bản chúng ta thêm vào trang sẽ định nghĩa các hành vi của giao diện đó. Các mã kịch bản này là dạng *event handler* hay còn được gọi là các *listener*. Chúng có nhiệm vụ lắng nghe và đáp ứng lại các sự kiện khác nhau trong suốt quá trình trang được hiển thị. Các sự kiện này có thể được sinh ra bởi hệ thống nhưng hầu hết chúng thường là kết quả của các thao tác từ người dùng *(như di chuyển hay nhấn chuột, gõ phím …).*

jQuery cung cấp cho ta nhiều phương thức để có thể gắn kết các event handler cho các thành phần trên trang, phổ biến nhất là sử dụng phương thức .bind().

a. Các dạng sử dụng phương thức .bind():

Các dạng sử dụng của phương thức .bind() gồm có:

```
.bind(eventType[,eventData],handler(eventObject))
```

hoặc

```
.bind(eventType[,eventData],preventBubble)
```

hoặc

```
.bind(events)
```

Tham số:

- *eventType*: là một chuỗi chứa một hoặc nhiều loại sự kiện DOM như *"click", "submit", ...*

- *eventData*: (Tùy chọn) Danh sách dữ liệu sẽ được truyền cho event handler

- *handler(eventObject):* Hàm được thực thi mỗi khi sự kiện xảy ra.

- *preventBubble*: mặc định = true. Nếu = false sẽ gắn kết một function ngăn chặn hành động mặc định khi xảy ra và ngăn chặn sự kiện bong bóng.

- *events*: là một sơ đồ của một hoặc nhiều kiểu sự kiện DOM và các hàm thực thi chúng.

Giá trị trả về: *.bind()* trả về một đối tượng jQuery, dùng cho mục đích sử dụng liên tục các phương thức theo kiểu mắt xích.

Phương thức *.bind()* là cách thức chủ yếu để gắn kết các hành vi vào tài liệu. Tất cả các sự kiện javascript được chấp nhận cho tham số *eventType* bao gồm*: blur, change, click, dbclick, error, focus, keydown, keypress, keyup, load, mousedown, mousemove, mouseout, mouseup resize, scroll, select, submit, unload…*

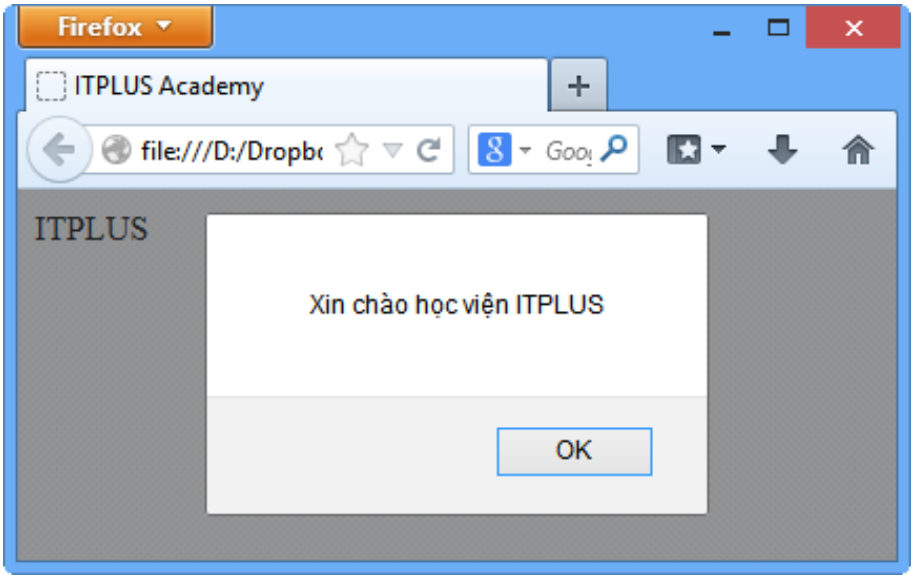
jQuery cung cấp các phương thức tắt để gắn kết mỗi loại sự kiện trên, nó hoạt động giống như khi sử dụng *.bind()* nhưng viết ngắn hơn được vài chữ. Ví dụ bạn có thể viết *.click()* thay cho *.bind('click')*

Khi sự kiện xảy ra trên một phần tử, tất cả handler đã gắn cho kiểu sự kiện của phần tử sẽ được kích hoạt. Nếu có nhiều handler được đăng ký, chúng sẽ được thực thi theo thứ tự khi chúng được gắn vào.

Ví dụ sau sẽ gắn kết sự kiện click vào cho phần tử có ID là `itplus`: khi người dùng click vào `#itplus`, thông báo *"Xin chào học viện ITPLUS"* sẽ hiện ra.

```
<html>
<head>
    <title>ITPLUS Academy</title>
    <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>
    <script type="text/javascript">
            $(document).ready(function () {
            $('#itplus').bind('click', function() {
                        alert('Xin chào học viện ITPLUS');
                  });
        });
    </script>
</head>
<body>
      <div id="itplus">ITPLUS</div>
</body>
</html>
```

Kết quả:



Hình 8.15. Sử dụng phương thức .bind()

b. Gắn kết nhiều sự kiện :

Nhiều loại sự kiện có thể được gắn kết đồng thời cho một phần tử bằng cách ngăn cách nhau bởi khoảng trắng. Ví dụ sau sẽ gắn kết sự kiện cho phần tử `#foo`:

- Thêm thuộc tính class = “entered” khi mouse enters

- Bỏ thuộc tính class khi mouse leaves.

```
$('#foo').bind('mouseenter mouseleave', function() {
    $(this).toggleClass('entered');
});
```

Đối với phiên bản jQuery 1.4, chúng ta có thể gắn kết nhiều event handler đồng thời bằng cách truyền vào một sơ đồ của một cặp event type/handlers:

```
$('#foo').bind({
    click: function() {
        // do something on click
    },
    mouseenter: function() {
        // do something on mouseenter
    }
});
```

c. Event Handlers

Tham số handler mang một hàm hồi quy. Bên trong handler, từ khóa this chỉ đến phần tử DOM mà hander đó tác động. Để sử dụng phần tử trong jQuery, nó có thể được truyền vào hàm $() như bình thường.

Ví dụ:

```
$('#foo').bind('click', function() {
    alert($(this).text());
});
```

Sau khi đoạn code trên được thực thi, khi người dùng click vào phần tử có ID là foo, nội dung text của nó sẽ được hiện lên trong thông báo.

d. Truyền dữ liệu cho sự kiện (Passing Event Data)

Tham số tùy chọn eventData thường không được sử dụng. Khi xuất hiện, nó cho phép chúng ta có thể truyền thêm thông tin cho handler. Một sự tiện lợi của việc sử dụng tham số này là nó có thể khắc phục vấn đề sự đóng kín handler.

Xét ví dụ sau:

```
<html>
<head>
    <title>ITPLUS Academy</title>
    <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>
    <script type="text/javascript">
                $(document).ready(function () {
                 var message = 'Hello foo!';
                 $('#foo').bind('click', function () {
                    alert(message);
                 });
                 message = 'Hello bar!';
                 $('#bar').bind('click', function () {
                     alert(message);
                 });
             });
    </script>
</head>
<body>
        <div id="foo">foo</div>
    <div id="bar">bar</div>
</body>
</html>
```

Khi bạn click vào foo hay bar thì đều hiện lên thông báo 'Hello bar'. Bởi vì các handler có tính đóng gói, giá trị của message đã thay đổi trước khi được gọi trong handler. Để giải quyết điều này, chúng ta có thể truyền message bằng cách sử dụng eventData như sau:

```
var message = 'Hello foo!';
    $('#foo').bind('click', { msg: message }, function (event) {
       alert(event.data.msg);
    });
    message = 'Hello bar!';
    $('#bar').bind('click', { msg: message }, function (event) {
       alert(event.data.msg);
    });
```

Trong trường hợp này, biến message không được chỉ đến trực tiếp bên trong các handler, thay vào đó, nó được truyền bởi giá trị (value) thông qua eventData, cái mà được ấn định giá trị tại thời điểm sự kiện xảy ra. Bây giờ,handler đầu tiên sẽ hiển thị *"Hello foo!"* , còn handler thứ hai sẽ hiển thị *"Hello bar!"*

Nếu eventData có mặt, nó sẽ là tham số thứ hai của phương thức .bind(); Nếu không thì hàm hồi quy được truyền vào như là tham số thứ hai và cuối cùng.

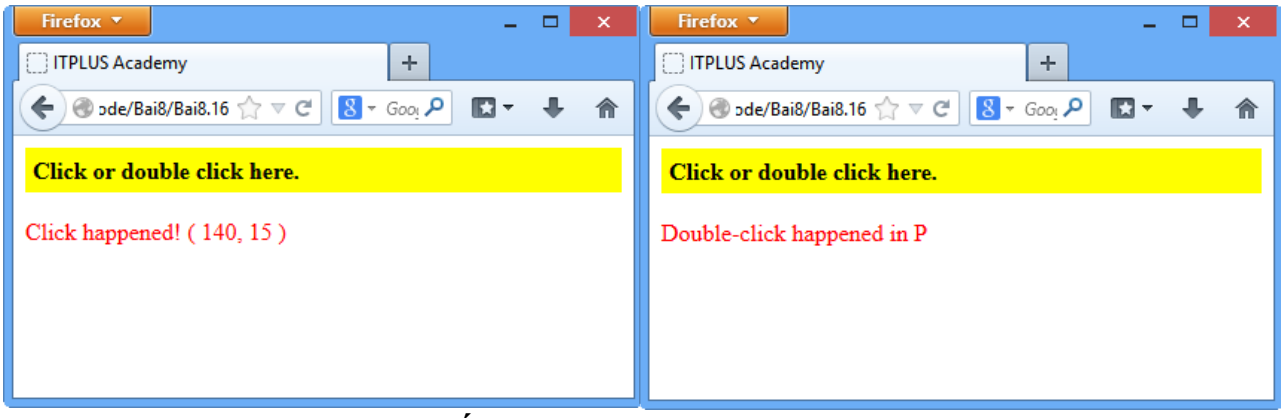
e. Một số ví dụ sử dụng phương thức .bind():

Ví dụ 1**:** Xử lý sự kiện click và double click vào một đoạn văn.

Thứ tự phản ứng sự kiện có thể khác nhau giữa các trình duyệt trong việc nhận 2 sự kiện click và 1 sự kiện dbclick. Độ nhạy của double click *(được tính bằng khoảng thời gian lớn nhất giữa các lần click chuột mà được phát hiện là một lần double click)* có thể khác nhau giữa các hệ thống hay trình duyệt và phụ thuộc vào cấu hình của người dùng.

```
<html>
<head>
    <title>ITPLUS Academy</title>
    <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>
</head>
<body>
    <p>Click or double click here.</p>
    <span></span>
    <script>
         $("p").bind("click", function(event){
             var str = "( " + event.pageX + ", " + event.pageY + " )";
             $("span").text("Click happened! " + str);
         });
         $("p").bind("dblclick", function(){
             $("span").text("Double-click happened in " + this.nodeName);
         });
         $("p").bind("mouseenter mouseleave", function(event){
             $(this).toggleClass("over");
         });
    </script>
</body>
</html>
```

Kết quả:

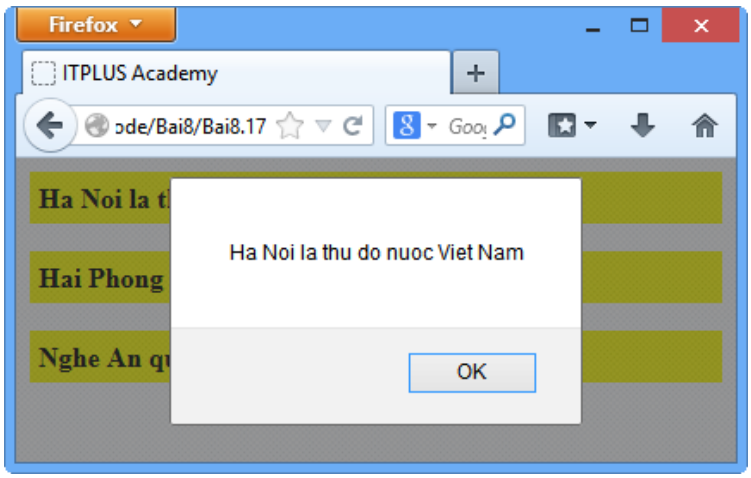


Hình 8.16. Kết quả khi thực hiện click và double click

Ví dụ 2**:** Hiển thị thông báo nội dung text của mỗi đoạn văn khi được click vào.

```
<html>
<head>
    <title>ITPLUS Academy</title>
            <style>
        p { background:yellow; font-weight:bold; cursor:pointer;
padding:5px; }
    </style>
        <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>
    <script type="text/javascript">
        $(document).ready(function () {
                function handler(event) {
                    alert(event.data.foo);
                }
                $("p").bind("click", {foo: "bar"}, handler)
        });
   </script>
</head>
<body>
    <p>Ha Noi la thu do nuoc Viet Nam</p>
    <p>Hai Phong thanh pho hoa phuong do</p>
    <p>Nghe An que huong Bac Ho</p>
</body>
</html>
```

Kết quả:



Hình 8.17. Lấy nội dung text của thẻ p

Ví dụ 3**:** Truyền thêm thông tin trước khi sự kiện xảy ra.

```
function handler(event) {
    alert(event.data.foo);
}
$("p").bind("click", {foo: "bar"}, handler)
```

Ví dụ 4: Thoát một hành động mặc định và ngăn chặn sự kiện bong bóng bằng cách return false:

```
$("form").bind("submit", function() { return false; })
```

Ví dụ 5**:** Chỉ thoát hành động mặc định bằng cách sử dụng phương thức *.preventDefault()*:

```
$("form").bind("submit", function(event) {
 event.preventDefault();
 });
```

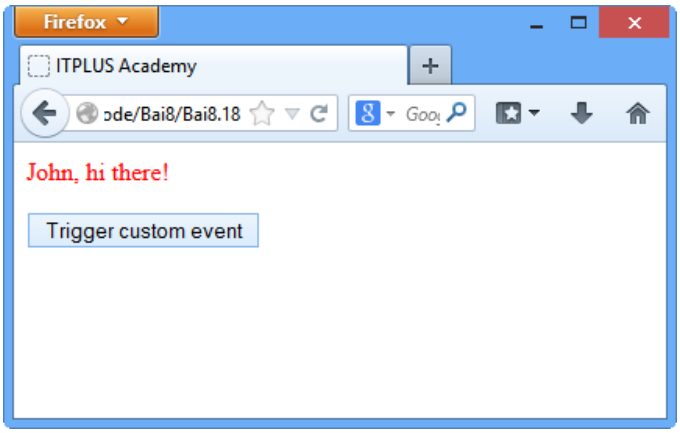
Ví dụ 6**:** Ngăn chặn sự kiện bong bóng mà không chặn hành động mặc định bằng cách sử dụng phương thức *.stopPropagation()*:

```
$("form").bind("submit", function(event) {
    event.stopPropagation();
});
```

Ví dụ 7**:** Gắn kết một sự kiện tùy chỉnh

```
<html>
<head>
    <title>ITPLUS Academy</title>
            <style>
        p { color:red; }
        span { color:blue; }
    </style>
        <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>
</head>
<body>
        <p>Has an attached custom event.</p>
    <button>Trigger custom event</button>
    <span style="display:none;"></span>
    <script>
        $("p").bind("myCustomEvent", function(e, myName, myValue){
            $(this).text(myName + ", hi there!");
            $("span").stop().css("opacity", 1)
            .text("myName = " + myName)
            .fadeIn(30).fadeOut(1000);
       });
       $("button").click(function () {
           $("p").trigger("myCustomEvent", [ "John" ]);
       });
    </script>
</body>
</html>
```

Kết quả:



Hình 8.18. Gắn kết một sự kiện tùy chỉnh

<u>Ví dụ 8</u>**:** Gắn kết nhiều sự kiện đồng thời:

```
$("div.test").bind({
    click: function(){
        $(this).addClass("active");
    },
    mouseenter: function(){
        $(this).addClass("inside");
    },
    mouseleave: function(){
        $(this).removeClass("inside");
    }
});
```

1.5. Các hiệu ứng với jQuery

Các hiệu ứng động của jQuery sẽ làm cho trang web của bạn thêm phần sinh động. jQuery cho phép bạn ẩn hiện, trượt lên trượt xuống các thành phần của trang web. Bạn cũng có thể cho nó xảy ra cùng một lúc hoặc theo thứ tự định trước. Trong phần này chúng ta sẽ tìm hiểu các hiệu ứng jQuery và kết hợp chúng để tạo ra những hiệu ứng đẹp.

1.5.1. Thay đổi Inline CSS

Trước khi chúng ta học những hiệu ứng jQuery, chúng ta cần xem lại một chút về CSS. Trong phần CSS, chúng ta thay đổi giao diện của các thành phần trên trang bằng cách khai báo thuộc tính của class trong một stylesheet riêng biệt. Sau đó chúng ta thêm hoặc loại bỏ những class đó bằng jQuery. Về cơ bản thì cách này nên được sử dụng để thêm CSS vào HTML bởi vì nó tôn trọng quy luật tách riêng phần trình bày và cấu trúc. Tuy nhiên, đôi khi chúng ta muốn áp dụng style cho những thành phần chưa được hoặc khó mà được định dạng bằng stylesheet. Nhưng rất may mắn là jQuery có phương thức .css() để sử dụng cho những trường hợp này.

Phương thức này hoạt động bằng cả hai cách lấy và đặt. Để lấy giá trị của một thuộc tính, chúng ta chỉ cần chuyển tên của thuộc tính đó thành một một chuỗi, dạng như `.css('backgroundColor')`. jQuery có thể hiểu được những thuộc tính kết hợp bởi nhiều từ và nối với nhau bằng dấu - như là trong CSS `('background-color')`, hoặc dạng viết hoa chữ cái đầu như là `('backgroundColor')`. Để định dạng thuộc tính style, phương thức .css() có hai cách sử dụng. Cách thứ nhất chỉ nhận một cặp thuộc tính - giá trị. Cách thứ hai là nhận một tập hợp các cặp thuộc tính - giá trị.

```
.css('property','value')
.css({'property1': 'value1', 'property-2': 'value2'})
```

<u>Chú ý</u>: Các giá trị số không dùng dấu ngoặc kép trong khi giá trị chuỗi phải có dấu ngoặc kép. Tuy nhiên, khi sử dụng bản đồ ký hiệu, dấu ngoặc kép không bắt buộc cho những tên thuộc tính được viết dưới dạng in hoa chữ cái đầu.

Chúng ta sử dụng phương thức *.css()* cũng giống như cách chúng ta đã sử dụng *.addClass()*. Bằng cách gán nó cho một Selector sau đó thì *Bind* nó vào một sự kiện. Để minh họa, chúng ta sử dụng ví dụ sau:

```
<html>
<head>
    <title>ITPLUS Academy</title>
    <style>
        p {color: red;}
        span {color: blue;}
    </style>
    <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>
    <script type="text/javascript">
        $(document).ready(function () {
```

```
            var $speech = $('div.speech');
            var defaultSize = $speech.css('fontSize');
            $('#switcher button').click(function () {
                var num = parseFloat($speech.css('fontSize'), 10);
                switch (this.id) {
                    case'switcher-large':
                        num *= 1.4;
                        break;
                    case'switcher-small':
                        num /= 1.4;
                        break;
                    default:
                        num = parseFloat(defaultSize, 10);
                }
                $speech.css('fontSize', num + 'px');
            });
        });
    </script>
</head>
<body>
<div id="switcher">
    <div class="label">Cỡ chữ</div>
    <button id="switcher-default">Mặc Định</button>
    <button id="switcher-large">Lớn</button>
    <button id="switcher-small">Nhỏ</button>
</div>
<div class="speech">
    <p>
        ITPlus Academy – Viện CNTT – Đại học Quốc gia Hà Nội
        tự hào là đơn vị có uy tín và là địa chỉ đào tạo đáng tin cậy
        trong lĩnh vực công nghệ thông tin
    </p>
</div>
</body>
</html>
```

1.5.2. Một số hiệu ứng sử dụng jQuery thường gặp

Thư viện jQuery cung cấp một số kỹ thuật để tạo nên các hiệu ứng chuyển động cho một trang web. Chúng bao gồm các chuyển động đơn giản, sử dụng thường xuyên, và cả các khả năng để xây dựng các hiệu ứng phức tạp. Trong phần này giáo trình sẽ giới thiệu một số hiệu ứng thường dùng để lập trình web với thư viện jquery.

Dưới đây là bảng ví dụ về các hàm hiệu ứng trong jQuery:

Tên hàm	Mô tả
`$(selector).hide()`	Ẩn các phần tử đã được chọn
`$(selector).show()`	Hiển thị các phần tử đã được chọn
`$(selector).toggle()`	Kết hợp ẩn và hiện của các phần tử đã được chọn
`$(selector).slideDown()`	Trượt xuống (sẽ hiển thị) các phần tử đã được chọn.
`$(selector).slideUp()`	Trượt lên (ẩn đi) các phần tử đã được chọn.
`$(selector).slideToggle()`	Kết hợp trượt lên và trượt xuống của các phần tử đã được chọn

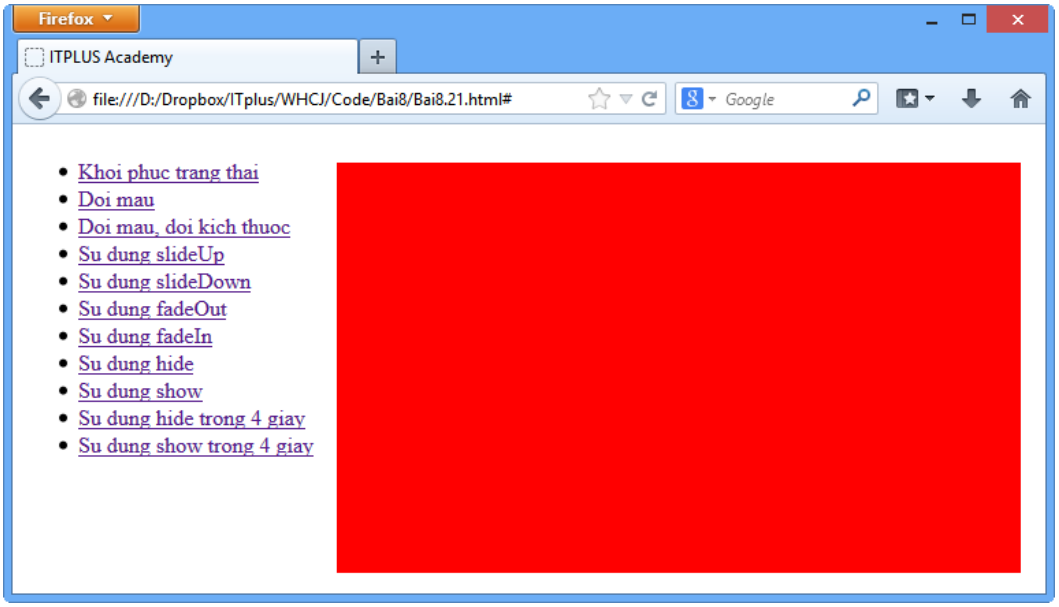
`$(selector).fadeIn()`	Làm rõ các phần tử đã được chọn
`$(selector).fadeOut()`	Làm mờ đi các phần tử đã được chọn
`$(selector).fadeTo()`	Làm mờ đi các phần tử đã được chọn theo một giá trị đã được thiết lập sãn.
`$(selector).animate()`	Thực thi một ảnh động tùy chỉnh trên các phần tử đã được chọn

Ví dụ về các hiệu ứng cơ bản trong jQuery:

```
<html>
<head>
    <title>ITPLUS Academy</title>
    <style>
        #box {width: 500px;height: 300px;background: #FF0000;margin-top:
20px;}
        .floatleft {float: left;}
        .floatright {float: right;}
        .width30 {width: 30%}
        .width70 {width: 69%}
    </style>
    <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"
type="text/javascript"></script>
    <script type="text/javascript">
        $(document).ready(function() {
$('a#restore').click(function() {
        $('#box').css({'background':'#FF0000',
'width':'500px','height':'300px'});});
$('a#changebg').click(function() {
        $('#box').css('background','#CCCCCC')});
$('a#changebgandsize').click(function() {
        $('#box').css({'background':'#00FF00',
'width':'100px','height':'100px'});});
$('a#slidedown').click(function() {
        $('#box').slideDown();});
$('a#slideup').click(function() {
        $('#box').slideUp();});
$('a#fadeout').click(function() {
        $('#box').fadeOut();});
$('a#fadein').click(function() {
        $('#box').fadeIn();});
$('a#hide').click(function() {
        $('#box').hide();});
$('a#show').click(function() {
        $('#box').show();});
$('a#hide4s').click(function() {
        $('#box').hide('4000');});
$('a#show4s').click(function() {
        $('#box').show('4000');});
        });
    </script>
</head>
<body>
   <div class="floatleft width30">
     <ul class="list">
        <li><a id="restore" href="#">Khoi phuc trang thai</a></li>
        <li><a id="changebg" href="#">Doi mau</a></li>
        <li><a id="changebgandsize" href="#">Doi mau, doi kich
thuoc</a></li>
        <li><a id="slideup" href="#">Su dung slideUp</a></li>
```

```
        <li><a id="slidedown" href="#">Su dung slideDown</a></li>
        <li><a id="fadeout" href="#">Su dung fadeOut</a></li>
        <li><a id="fadein" href="#">Su dung fadeIn</a></li>
        <li><a id="hide" href="#">Su dung hide</a></li>
        <li><a id="show" href="#">Su dung show</a></li>
        <li><a id="hide4s" href="#">Su dung hide trong 4 giay</a></li>
        <li><a id="show4s" href="#">Su dung show trong 4 giay</a></li>
      </ul>
    </div>
    <div class="floatright width70">
       <div id="box"></div>
    </div>
</body>
</html>
```

Kết quả:



Hình 8.19. Ví dụ về hiệu ứng trong jQuery

B. BÀI TẬP

2.1. Câu hỏi ôn tập

Câu 1. jQuery là một thư viện:

P. Client script

Q. Server script

R. Server/client script

S. Không ở dạng nào

Câu 2. Câu lệnh jQuery nào đặt màu nền của tất cả các thẻ p thành màu đỏ?

K. $("p").css("background-color","red");

L. $("p").layout("background-color","red");

M. $("p").manipulate("background-color","red");

N. $("p").style("background-color","red");

Câu 3. Trong jQuery, phần tử $("div.intro") là phần tử nào?

P. Tất cả các thẻ div có class là intro

Q. Thẻ đầu tiên có class intro

R. Tất cả các thẻ div có id là intro

S. Thẻ đầu tiên có id là intro

Câu 4. Phương thức jQuery nào cho phép ẩn phần tử?

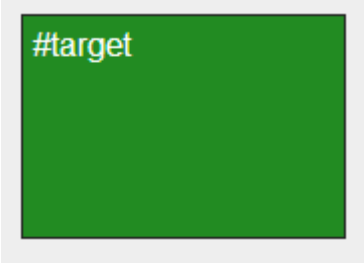
K. hide()
L. hidden()
M. display(none)
N. visible(false)

Câu 5: Phương thức jQuery nào cho phép thay đổi thuộc tính style của phần tử được lựa chọn?

T. style()
U. css()
V. html()
W. properties()

2.2. Bài tập

Bài 1. Sử dụng jQuery, viết chương trình thực hiện các yêu cầu sau đây:

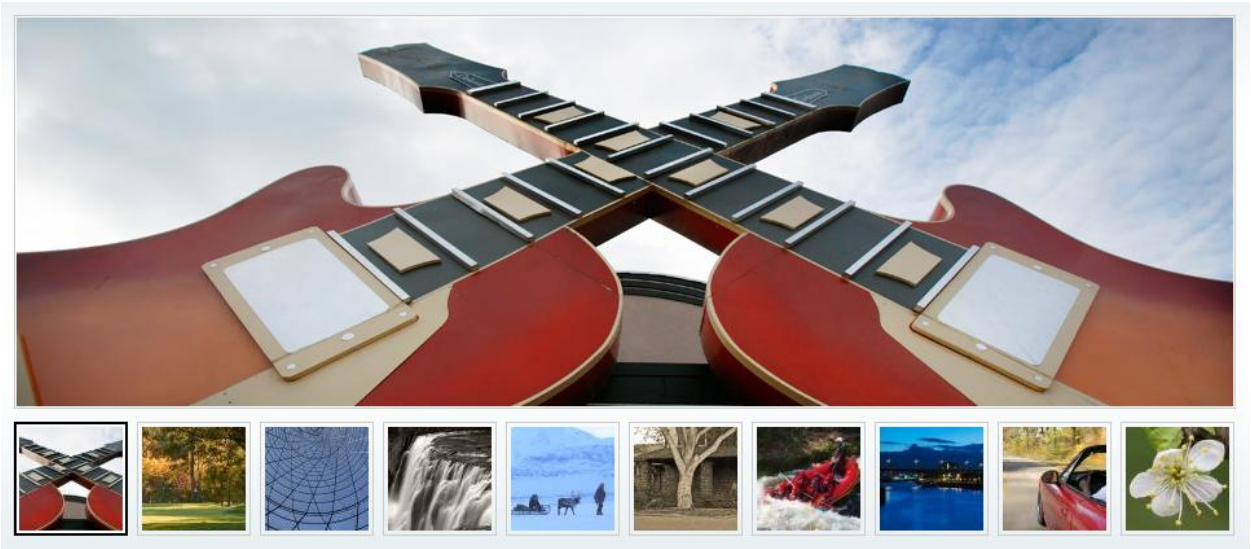


- Thay đổi màu sắc, kích thước của box có id là #target
- Thay đổi font chữ, cỡ chữ của text nằm trong box có id là #target

Bài 2. Viết chương trình tự động tính giá trị của cột thành tiền trong bảng sau:

STT	Tên sách	Đơn giá	Số lượng	Thành tiền
1	Giáo trình HTML & CSS	50.000	2	
2	Thủ thuật trong Javascript	45.000	3	
3	Jquery toàn tập	100.000	5	
4	PHP & MySQL	150.000	2	

Bài 3. Xây dựng chương trình sử dụng jQuery tạo slideshow đơn giản sau:



Bài 4. Xây dựng chương trình sử dụng jQuery tạo menu đa cấp đơn giản sau:

Home
About
Clients
Contact
FAQs
Testimonials
Home
About »
Clients
Contact Us

BÀI 9. PHOTOSHOP TRONG THIẾT KẾ WEB

Thiết kế giao diện trong Photoshop là phần không thể thiếu được trong quá trình thiết kế trang web. Cho dù là trang web nhỏ, đơn giản hay những trang phức tạp, đều yêu cầu phải được thiết kế trước trong Photoshop hoặc những trình đồ họa tương đương. Tuy nhiên, trong phạm vi giới hạn chương trình, chúng ta sẽ không đi sâu vào tìm hiểu việc thiết kế giao diện trong photoshop như thế nào (việc này dành cho Desinger) mà tập trung vào việc tìm hiểu các công cụ trong photoshop để phục vụ tốt nhất cho việc chuyển đổi giao diện từ file thiết kế (PSD) sang mã HTML.

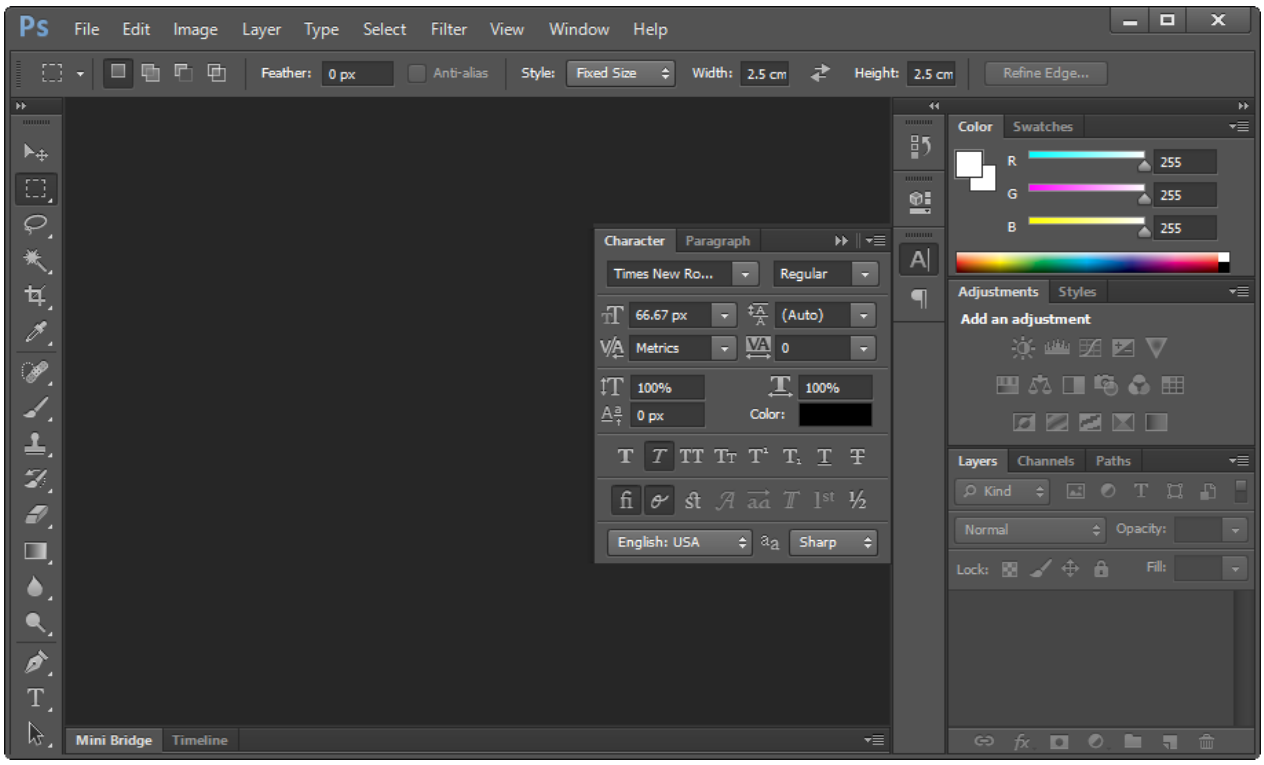
A. NỘI DUNG

1.1. Làm quen với photoshop

Photoshop là một trong những chương trình chỉnh sửa ảnh mạnh với mọi trình độ, kỹ năng nhưng lại là chương trình *"đáng sợ nhất"* đối với những người mới bắt đầu. Chúng ta sẽ cùng tìm hiểu các công cụ và những thao tác cơ bản nhất của Photoshop.

1.1.1. Giới thiệu chung photoshop

Adobe Photoshop (PS) là một chương trình phần mềm đồ họa của hãng Adobe System, ra đời vào năm 1988 trên hệ máy Macintosh. PS hiện nay là sản phẩm đứng đầu thị trường phần mềm chỉnh sửa ảnh và được coi là tiêu chuẩn của các nhà đồ họa chuyên nghiệp. Ngoài chức năng chính là chỉnh sửa ảnh, PS còn được sử dụng rộng rãi trong các lĩnh vực thiết kế đồ họa, thiết kế web, vẽ tranh và vẽ texture cho các ứng dụng 3D.

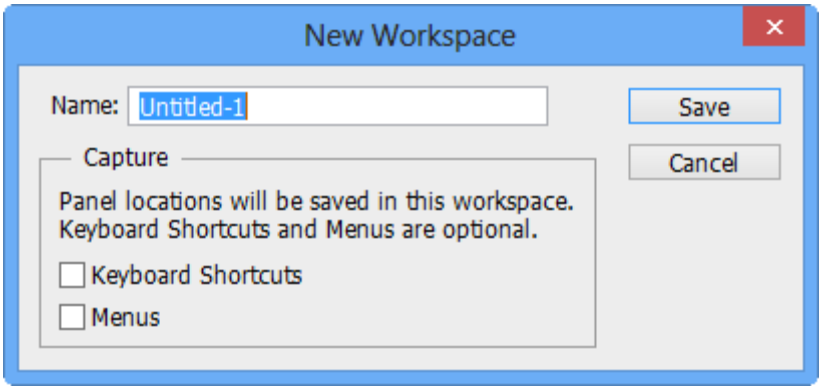


Hình 9.1. Giao diện của photoshop CS6

1.1.2. Một số thao tác cơ bản

- **Tùy chỉnh không gian làm việc**: Bạn sẽ muốn thử xem qua các panel và palette sẵn có xem cái nào phù hợp với công việc của mình. Trong những lựa chọn này, bạn sẽ thấy có sự sắp xếp khác nhau về phần bảng màu và panel phù hợp với từng mục đích công việc. Điều này khá thuận tiện khi bạn muốn có những chỉnh sửa khác nhau cho bức ảnh ở chế độ Design hay Painting.

- CS6 có chế độ lưu những tùy chỉnh không gian làm việc theo ý thích của bạn lại để có thể sử dụng sau này.



Hình 9.2. Tạo và lưu lại một không gian làm việc mới

- **Mở file ảnh trong PS**: Ở hàng trên cùng của cửa sổ, vào File -> Open rồi chọn đường dẫn tới file ảnh. Các bạn có thể giữ phím Ctrl để chọn mở nhiều file cùng một lúc.

- **Lưu file**: File -> Save

- **Lưu file dưới định dạng khác**: File -> Save As
- **Lưu file cho web**: Save for Web

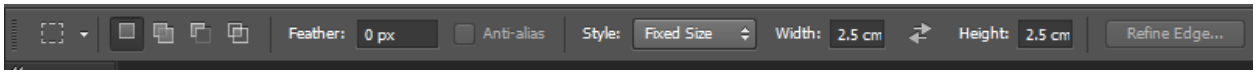
1.1.3. Toolbox & Panel

a. Toolbox

Toolbox là nơi bạn có được tất cả các thao tác xử lý ảnh từ trỏ chuột. Theo mặc định, nó bị khóa ở phía cạnh trái của màn hình làm việc. Bằng cách nhấp vào phím mũi tên << bạn có thể mang toolbox này đặt tại bất kỳ vị trí nào trên phần không gian làm việc.

Toolbox là nơi bạn có được tất cả các thao tác xử lý ảnh từ trò chuột. Theo mặc định, nó bị khóa ở phía cạnh trái của màn hình làm việc. Bằng cách nhấp vào phím mũi tên << bạn có thể mang toolbox này đặt tại bất kỳ vị trí nào trên phần không gian làm việc.

Tùy chọn Palette: Ở phía trên màn hình làm việc, ngay dưới thanh menu, bạn sẽ thấy các bảng các tùy chọn. Khi lựa chọn các công cụ khác nhau trên toolbox, bạn sẽ thấy các tùy chọn này thay đổi tùy vào từng công cụ.



Hình 9.3. Tùy chọn Palette

Các công cụ trên Toolbox:

Biểu tượng	Tên công cụ	Mô tả
	Rectangular Marquee Tool (phím tắt: M)	Đây là công cụ lựa chọn dạng cơ bản, lựa chọn vùng theo dạng hình khối cơ bản (có thể là hình chữ nhật, elip, 1 dòng đơn hoặc 1 cột đơn. Nếu muốn lựa chọn theo dạng hình vuông hoặc hình tròn cân đối thì giữ Shift trong quá trình khoanh vùng. Nếu muốn thay đổi khung chọn giữa hình chữ nhật và hình ellip thì nhấn Shift + M (hoặc chuột phải vào biểu tượng công cụ trên thanh Toolbox và chọn hình như ý).
	Move Tool (phím tắt: V)	Đây là công cụ di chuyển. Nếu bạn đang chọn một vùng, sử dụng công cụ này để di chuyển vùng chọn đó. Bạn cũng có thể di chuyển nhiều layer một lúc sau khi đã chọn layer cần di chuyển.
	Lasso Tool (phím tắt: L)	Đây là một công cụ lựa chọn khác, công cụ này cho phép bạn vẽ nhanh một vùng chọn. Vùng chọn này có thể là một hình dạng đặc biệt tùy theo từng phần trên bức ảnh của bạn. Giữ Shift + L để thay

Biểu tượng	Tên công cụ	Mô tả
		đổi chế độ chọn khoanh vùng (Lasso) hay đa giác (Polygonal hoặc Magnetic).
	Quick Selection Tool (phím tắt: W)	Đây là công cụ chọn vùng gần đúng. Kích biểu tượng bút vẽ vào một vùng của bức ảnh, Photoshop sẽ đọc và cố gắng lấy vùng chọn có màu sắc tương đồng. Nhấn Shift + W để thay đổi giữa công cụ Quick Selection Tool và Magic Wand Tool.
	Crop Tool (phím tắt: C)	Vẽ thành một vùng chọn hình chữ nhật, sau đó cắt lấy bức ảnh nằm trong vùng chọn đã vẽ. Công cụ này rất hữu ích thường được sử dụng để cắt xén một bức ảnh có không gian hơi "thừa". Nhấn Shift + C để thay đổi giữa công cụ Slice và Slice Select, một công cụ hữu ích để tạo ra nhiều hình ảnh từ một hình duy nhất, thường sử dụng trong việc lên hình ảnh cho một website.
	Eyedropper Tool (phím tắt: I)	Chọn một màu bất kỳ từ tài liệu mà bạn đã mở. Shift + I để thay đổi giữa các công cụ: Color Sampler, Ruler và Note Tool.
	Spot Healing Brush Tool (phím tắt: J)	Rất hữu ích trong việc xóa các vết ố, trầy xước không mong muốn trên ảnh. Nhấn Shift + J để thay đổi giữa Healing Brush, Patch Tool và công cụ Red Eye (dùng để xử lý mắt đỏ)
	Brush Tool (phím tắt: B)	Đây là công cụ phứt tạp duy nhất trên Toolbox. Rất nhiều bài trên Quản Trị Mạng đã hướng dẫn sử dụng công cụ này để vẽ các hình ảnh lặp đi lặp lại. Nhấn Shift + B để lựa chọn lần lượt công cụ Pencil, Color Replacement Tool, và Mixer Brushes.
	Clone Stamp Tool (phím tắt: S)	Đây cũng là một công cụ brush sửa ảnh khác, giữ Alt và kích chuột vào vùng nền "nguồn" và sau đó kích chuột trái chọn vùng nền muốn "phủ" vùng "nguồn" lên. Nhấn Shift + S để chọn các công cụ đóng dấu khác nhau.
	History Brush Tool (phím tắt: Y)	Làm việc song song với History Palette, bạn có thể “lấy lại màu gốc” với công cụ này. Sử dụng như một công cụ lọc, bạn chọn phần muốn lấy lại màu nền như ảnh gốc (phần được chọn phải nằm trong vùng đã bị thay đổi nền). Nhấn Shift + Y để thay đổi giữa History Brush Tool và Art History Brush Tool.

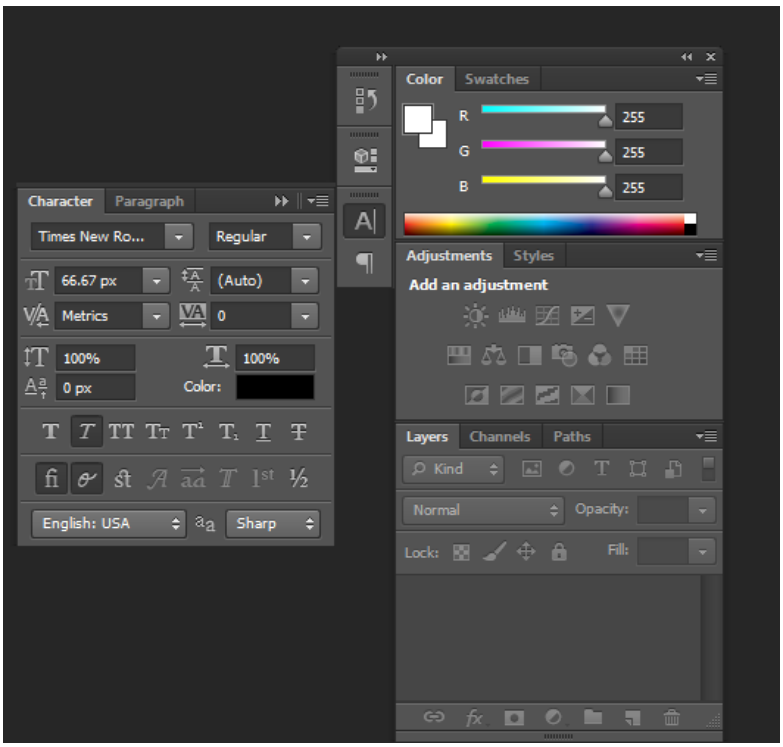
Biểu tượng	Tên công cụ	Mô tả
	Eraser Tool (phím tắt: E)	Đây là công cụ dùng để xóa một vùng ảnh. Nếu ảnh là layer Background hoặc bị khóa thì vùng bị xóa sẽ lấy theo màu Background Color. Nhấn Shift + E để chuyển giữa chế độ xóa Eraser, Background Eraser, hay Magic Eraser.
	Gradient Tool (phím tắt: G)	Kích và kéo gradient màu phủ đầy lên layer hoặc vùng mà bạn chọn bằng màu foreground và background trên thanh công cụ. Có rất nhiều tùy chọn gradient màu bạn có thể sử dụng. Mở rộng trong công cụ này còn có Paint Bucket Tool - dùng để phủ một màu đơn cho 1 vùng hoặc 1 layer đang chọn. Nhấn Shift + G để chuyển đổi giữa 2 chế độ phủ màu.
	Blur, Sharpen, and Smudge Tools	Mặc định, công cụ này không có phím tắt. Có ba công cụ chỉnh sửa ảnh nằm trong nút công cụ này. Smudge là thanh công cụ đặc biệt có thể tạo hiệu ứng vuốt màu tuyệt vời trên hình ảnh của bạn. Chuột phải vào nút công cụ để chọn những công cụ khác nhau: Blur và Sharpen.
	Dodge and Burn Tools (phím tắt: O)	Dodge và Burn là công cụ chỉnh sửa giúp làm sáng hoặc tối một vùng cho bức ảnh. Nhấn Shift + O để thay đổi lựa chọn giữa các công cụ.
	Pen Tool (phím tắt: P)	Đây là một công cụ "gây ác mộng" cho người dùng mới bắt đầu. Các công cụ Pen rất khó trong việc sử dụng nhưng là một ưu điểm của Photoshop. Shift + P sẽ cho phép bạn chuyển đổi giữa các công cụ làm việc với path (đường dẫn).
	Type Tool (phím tắt: T)	Cho phép bạn nhập một đoạn nội dung, mặc định là theo chiều ngang. Nhấn Shift + T để chọn công cụ Vertical Type Tool (nhập nội dung theo chiều dọc) và Type Mask Tools.
	Path Selection and Direct Selection Tools (phím tắt: A)	Đây là một công cụ chỉnh sửa các phân đoạn trong một đoạn đường dẫn (path). Có thể bỏ qua công cụ này trừ khi bạn sử dụng Pen Tool để vẽ 1 đoạn đường dẫn. Nhấn Shift + A để chuyển đổi giữa Path Selection và Direct Selection.
	Custom Shape Tool (phím tắt: U)	Đây là công cụ để tạo các hình vector hoặc clipart từ một thư viện hình có sẵn. Nhấn Shift + U để lựa chọn giữa các công cụ vẽ hình chữ nhật, tam giác, các đường nét và hình clipart.

Biểu tượng	Tên công cụ	Mô tả
	Zoom Tool (phím tắt: Z)	Đây là công cụ dùng để zoom to và nhỏ ảnh. Phóng to với việc nhấn chuột trái và thu nhỏ bằng cách nhấn thêm Alt trong khi kích chuột. Đây là công cụ cơ bản nhất trên thanh công cụ.
	Hand Tool (phím tắt: H)	Cuộn tài liệu mà không cần sử dụng con trỏ chuột hay phím mũi tên. Nhấn và giữ phím Space bất cứ khi nào bạn cần sử dụng công cụ này, nhả phím Space khi không cần sử dụng đến nó nữa.
	Background/Foreground	Đây là bảng 2 màu sắc đang được sử dụng để chỉnh sửa ảnh. Màu trên là Foreground, màu dưới là Background. Nhấn phím X trên bàn phím để chuyển đổi màu Foreground thành Background và ngược lại. Nhấn phím D để đưa 2 màu này về mặc định là trắng và đen.
	Quick Mask Mode (phím tắt: Q)	Đây là một chế độ thay thế cho việc tạo các lựa chọn phức tạp với các công cụ Brush, Eraser và Paint Bucket. Nhấn phím Q để chuyển đổi giữa việc chọn chế độ Quick Mask Mode và chế độ thường.

b. Panel

Một trong những điều hay nhất của Photoshop cũng là điều tệ nhất: đó là quá trình tải các tùy chọn. Hãy xem lướt qua các panel mặc định để có thể nắm được những tính năng nào cần dùng trong quá trình sử dụng Photoshop để bạn có thể tìm lại khi cần.

Panel mặc định được đặt ở phía bên phải cửa sổ làm việc của Photoshop. Nó chứa mọi thiết lập hữu ích cho người mới sử dụng Photoshop, bạn có thể điều chỉnh, ẩn hoặc hiện chúng. Không gian làm việc của Photoshop có thể rộng hơn giúp bạn thoải mái làm việc khi bạn cho ẩn các thanh panel đi.



Hình 9.4. Các panel trong photoshop

Mặc định thiết lập của các panel trong phần làm việc Essentials sẽ có dạng như sau. Ngoài cửa sổ làm việc Essentials bạn còn có thể chọn các khung làm việc khác như Design, 3D… bằng việc kích vào » để lựa chọn thêm. Ngoài ra để khôi phục lại các panel trong từng khung làm việc bạn có thể lựa chọn Reset trong menu ngữ cảnh khi kích vào ».

Các panel có thể được gộp lại bằng cách kích vào ▸▸ ở phía bên phải trên của panel. Bạn có thể kéo panel đi bất kỳ vị trí nào thuận tiện cho việc sử dụng bằng cách kích và kéo panel tương ứng ra khỏi vị trí cố định của nó. Đương nhiên, panel cũng có thể được đặt lại đúng vị trí như ban đầu bằng cách kéo về sát cạnh phải của cửa sổ Photoshop, cho đến khi bạn thấy xuất hiện một thanh màu xanh phía bên phải cửa sổ.

Các panel cơ bản:

Biểu tượng	Tên panel	Mô tả
	History	Đây là một trong những tính năng quan trọng nhất trên Photoshop, History cho phép người dùng undo lại nhiều mức công việc đã làm.
	Color	Đây là một panel đơn giản, nó chính là phần màu sắc mà bạn đặt cho foreground và background. Bạn có thể điều chỉnh giá trị màu Red, Green và Blue từ 0 – 255. Giá trị càng cao màu càng sáng và ngược lại. Ngoài ra bạn cũng có thể chọn màu từ thanh màu ở phía dưới panel, nó có vẻ dễ dàng chọn hơn.

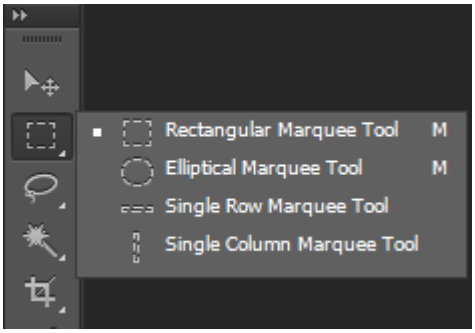
Biểu tượng	Tên panel	Mô tả
		Ngoài ra bạn cũng có thể tùy chỉnh các tùy chọn màu khác bằng cách kích vào và đặt slider cho những chế độ màu khác nhau.
	Swatches	Đây là một nguồn tài nguyên hay cho người mới bắt đầu và cả với các chuyên gia, swatches là một bảng màu sắc đã được lưu.
	Styles	Là một panel chứa các Layer Effects được lưu, Styles có thể là một cách định dạng khá thú vị cho người mới bắt đầu và cũng giúp cho những chuyên gia tiết kiệm thời gian trong việc tái sử dụng các định dạng layer phổ biến.
	Adjustments	Đây là panel điều chỉnh tạo ra các lớp cho bức ảnh bạn thêm sống động và đẹp hơn. Bạn có thể sử dụng các tính năng Hue/Saturation, Contrast hay Levels cho những bức ảnh. Tính năng này thực chất là tạo một layer mới lên trên bức ảnh của bạn và mọi điều chỉnh đều thực hiện trên layer mới này.
	Masks	Panel marks là một phương thức để loại bỏ các phần của layer, đưa chúng về dạng transparency. Đây là một cách khá hay để bỏ background từ một layer mà không cần sử dụng hiệu ứng
	Layers	Layer là một tính năng chuẩn của Photoshop, nó cho phép người dùng tạo ra những bức ảnh 2D từ nhiều phần có thể chỉnh sửa.
	Channels	Các bức ảnh số hiện thị màu sắc theo sự kết hợp của các màu cơ bản. Các màu cơ bản này sẽ hiển thị độc lập trong panel Channels của Photoshop. Người mới sử dụng có thể bỏ qua panel này.
	Paths	Đây là một thành phần khá phức tạp của Photoshop, các path dựa trên thành phần chính là vector (tương tự như Illustrator). Vector thực chất không khó hiểu nhưng công cụ Pen thì lại rất khó và gây bực bội cho người mới sử dụng.

1.1.4. Thao tác với vùng chọn

Khi bạn mở một văn bản ảnh trong PS, bạn có thể thay đổi chỉnh sửa tất cả các pixel của ảnh đó. Trong trường hợp bạn chỉ muốn chỉnh sửa một phần nhất định của ảnh, bạn cần phải tạo một vùng chọn xung quanh khu vực cần chỉnh sửa đó.

Tạo vùng chọn chính xác là một kĩ năng cực kì quan trọng. Trong PS có khá nhiều công cụ để tạo vùng chọn. chúng chia làm 3 nhóm cơ bản là Marquee tools, Lasso tools

và Quick selection tools. Bài học này chủ yếu để cho các bạn mới học PS làm quen với một số công cụ tạo vùng chọn.

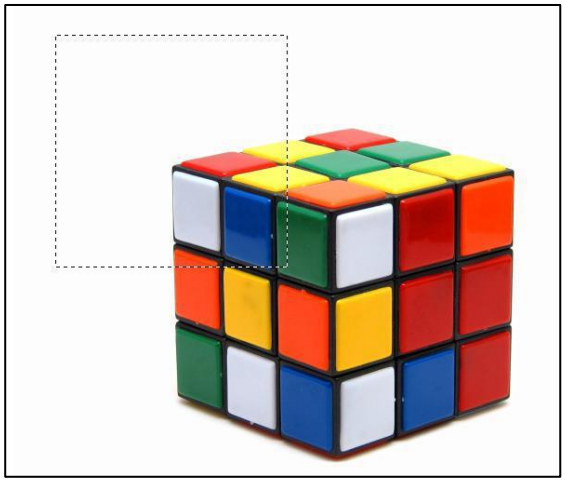


Các bạn có thể thấy là trên icon của một số công cụ trong bảng công cụ có một dấu tam giác nhỏ ở góc dưới, dấu hiệu này có nghĩa là bên trong icon đó có một vài công cụ khác nhau chứ không phải chỉ một công cụ. Mỗi khi thấy dấu tam giác đó, các bạn có thể click chuột phải vào icon để hiện ra tòan bộ các công cụ.

Ngoài ra, tất cả các công cụ trong bảng công cụ đều có phím tắt. Giữ phím Shift + phím tắt tương ứng của công cụ sẽ cho phép bạn lần lượt chọn các công cụ nằm bên dưới mà không cần click chuột phải vào. Trong bài học này chúng ta sẽ tìm hiểu các công cụ tạo vùng chọn của PS.

a. Marquee tool

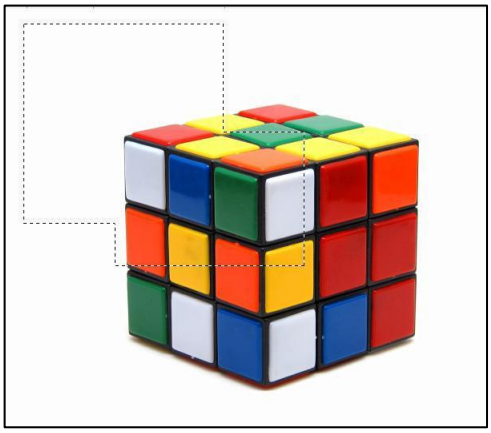
Click vào icon hoặc phím tắt M để chọn Rectangular Marquee tool. Công cụ này có chức năng tạo vùng chọn hình chữ nhật/vuông. Ấn chuột và kéo trên văn bản ảnh, bạn sẽ thấy một hình chữ nhật với đường nét đứt xuất hiện. Đường nét đứt này còn có một tên gọi hình ảnh hơn là *"đàn kiến chạy"* (marching ants) nhưng để ngắn gọn, tôi sẽ gọi nó là đường nét đứt. Khu vực nằm phía bên trong đường nét đứt là vùng chọn.



Hình 9.5. Tạo vùng chọn với Marquee tool

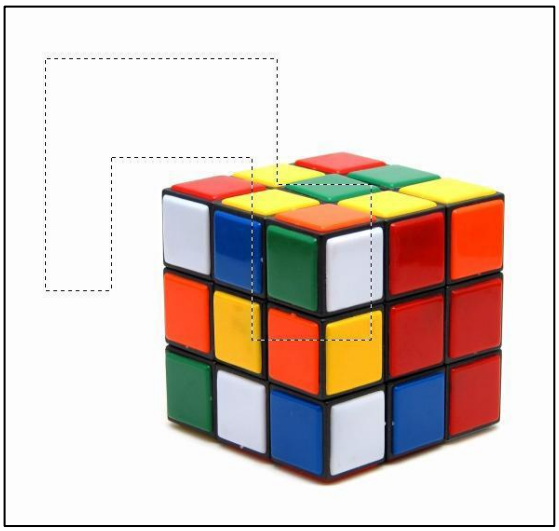
Nếu giữ phím Shift và kéo chuột, PS sẽ tự động giữ tỉ lệ vùng chọn là một hình vuông. Trong lúc đang kéo các bạn có thể giữ phím Space bar để di chuyển vùng chọn. Nếu giữ phím Alt và kéo chuột, vùng chọn sẽ được kéo từ trong ra thay vì từ góc.

Thay đổi và thêm/bớt vào vùng chọn: một khi đã có vùng chọn bất kì hình chữ nhật trên file ảnh, bạn có thể vào Select -> Transform Selection để hay đổi vùng chọn cho vừa ý. Nhớ ấn Enter sau khi chỉnh sửa vùng chọn xong. Chọn Marquee tool, giữ Phím shift, con trỏ chuột sẽ xuất hiện dấu + ở bên cạnh. Lúc này nếu bạn kéo một vùng chọn mới, vùng chọn này sẽ được thêm vào với vùng chọn vốn có.



Hình 9.6. Thêm vùng chọn

Ngược lại, giữ phím Alt, dấu – sẽ xuất hiện cạnh con trỏ chột. Lúc này kéo vùng chọn mới sẽ trừ bớt đi của vùng chọn cũ:

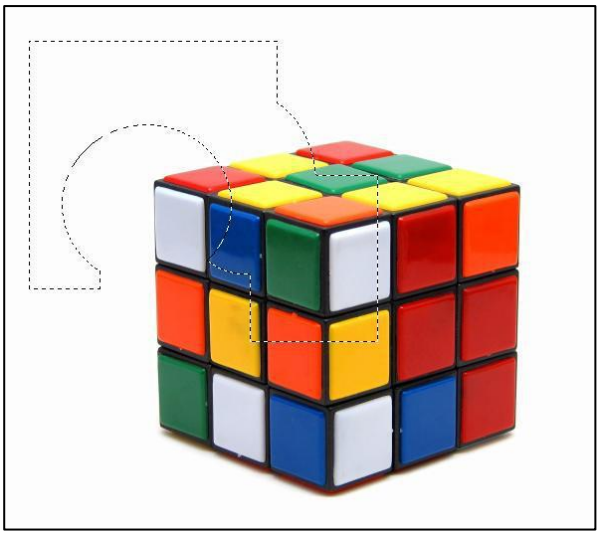


Hình 9.7. Bớt vùng chọn

Lưu ý: khi chuyển từ công cụ tạo vùng chọn này sang công cụ tạo vùng chọn khác, vùng chọn của bạn không bị mất đi, và bạn có thể thêm/bớt vào vùng chọn có sẵn với bất cứ công cụ tạo vùng chọn nào.

Chọn Elliptical Marquee Tool hoặc phím tắt M. Công cụ này có chức năng tương tự Rectangular Marquee Tool, xong nó tạo vùng chọn hình Elip thay vì hình chữ

nhật. Giữ phím Shift và kéo chuột sẽ cho ra vùng chọn hình tròn. Cũng tương tự như trên, sử dụng Elliptical Marquee tool cũng có thể thêm bớt vào vùng chọn có sẵn, cho ra kết quả là một vùng chọn khá kì dị như hình dưới đây:



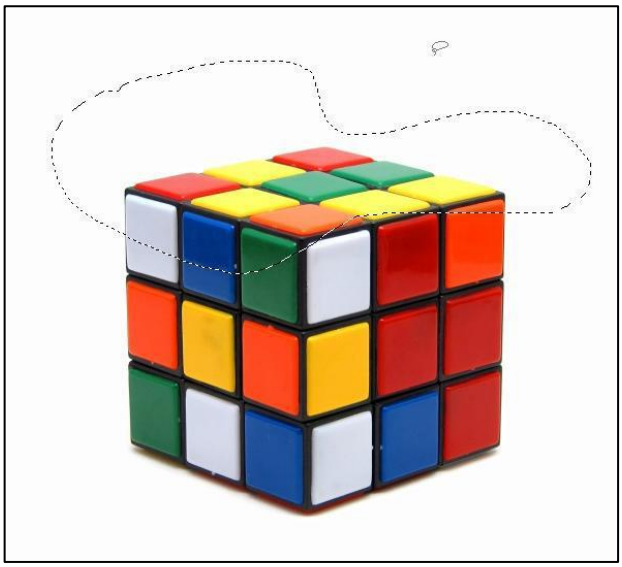
Hình 9.8. Sử dụng công cụ Elliptical Marquee Tool

Bạn có thể ấn tổ hợp phím Ctrl + D để bỏ vùng chọn hiện tại. Hãy ghi nhớ nó vì đây là một tổ hợp phím tắt bạn sẽ sử dụng rất nhiều về sau này.

b. Lasso tool

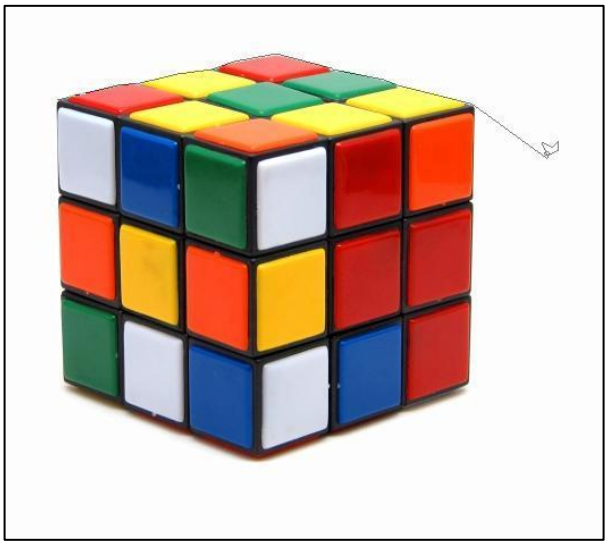
Chúng ta chuyển sang một công cụ tạo vùng chọn mới, đó là Lasso tool. Lasso tool có 3 loại khác nhau, phím tắt chung là L, để chuyển giữa các loại lasso click chuột phải vào icon trên thanh công cụ hoặc dùng Shift + L.

Tương tự như Marquee tool, giữ Shift sẽ thêm vào vùng chọn và Alt sẽ cắt bớt khỏi vùng chọn. Công cụ này cho phép bạn kéo chuột và vẽ thành một hình dạng tùy ý muốn, khi thả chuột ra hình dạng đó sẽ trở thành một vùng chọn. Khi đã thử nghiệm xong ấn Ctrl + D để bỏ vùng chọn.



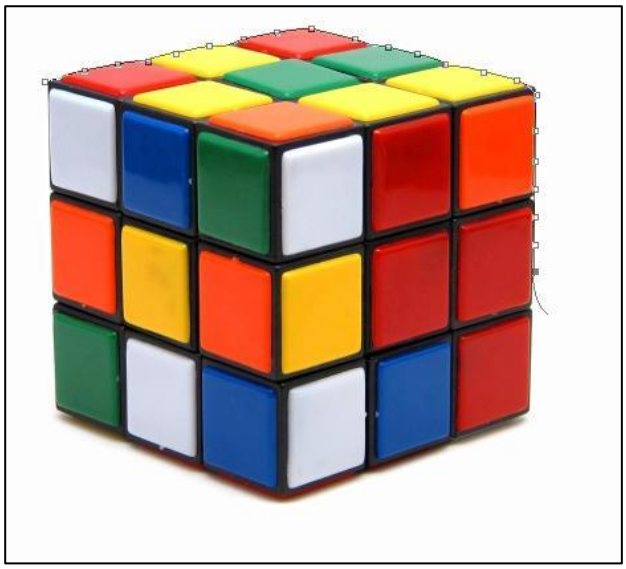
Hình 9.9. Sử dụng công cụ Lasso Tool

Đây là Polygonal Lasso Tool. Công cụ này tương tự Lasso tool nhưng cho phép bạn click và thả chuột để tạo nên vùng chọn thay vì vẽ tự do như trên. Hãy thử dùng công cụ này để chọn xung quanh khối Rubik. Khi xong hãy ấn Ctrl + D để bỏ vùng chọn.



Hình 9.10. Sử dụng Polygonal Lasso Tool

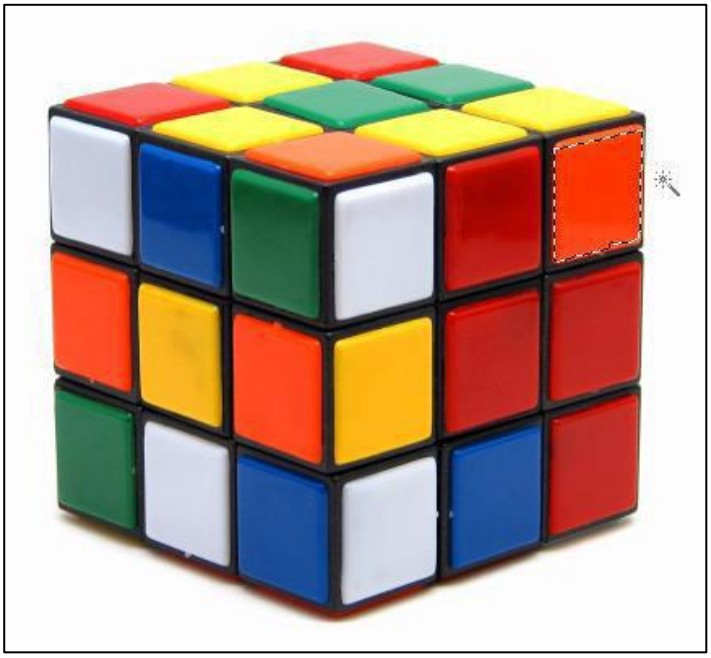
Đây là Magnetic Lasso Tool. Như tên gọi của nó, công cụ này hoạt đông như một nam châm. Chỉ cần click chuột một lần và rê chuột xung quanh khối rubik, PS sẽ tự động tìm ra các mép và tự động đặt các "điểm mốc" dọng theo đường chọn. Nếu có điểm mốc nào đặt lệch vị trí, chỉ cần ấn phím Delete để xóa điểm mốc đó và quay lại. Ở trên thanh tùy chọn các bạn có thể điều chỉnh mức độ nhạy của công cụ này. Hãy thử chọn một vùng chọn xung quanh khối Rubik. Khi xong hãy ấn Ctrl + D để bỏ vùng chọn.



Hình 9.11. Sử dụng Magnetic Lasso Tool

c. Magic wand tool

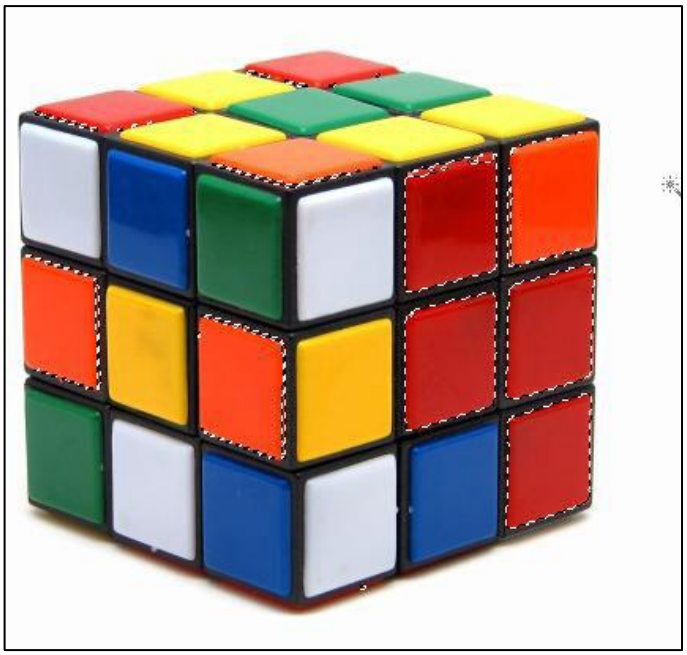
Magic wand, có nghĩa là cây đũa thần. Để chọn công cụ này click vào hoặc phím tắt W. Công cụ này chọn vùng chọn dựa theo màu sắc. Click chuột vào một vùng bất kì trên văn bản ảnh, Magic wand sẽ tự động chọn vùng chọn có màu sắc tương tự xung quanh đó:



Hình 9.12. Sử dụng Magic wand Tool

Khi chọn magic wand, các bạn sẽ thanh tùy chọn như dưới:

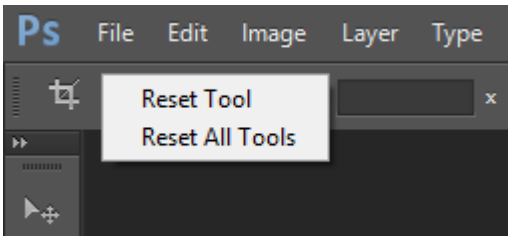
Trong đó thông số tolerance có thể coi là độ nhạy của magic wand, nếu bạn uncheck ô Contigious, magic wand sẽ chọn tất cả khu vực màu tương tự trong hình thay vì chỉ vùng được click:



Hình 9.13. Sử dụng thông số tolerance trong Magic wand tool

Tương tự như mọi công vụ tạo vùng chọn khác, giữ phím Shift sẽ thêm vào vùng chọn, giữ Alt sẽ cắt bớt đi khỏi vùng chọn. Khi đã thử nghiệm xong với magic wand Ấn Ctrl + D để bỏ vùng chọn.

Reset công cụ: Nếu như có một công cụ hoạt động không như ý muốn của bạn, rất có thể vì bạn đã thay đổi gì đó trong tùy chọn của công cụ đó. Một khi những tùy chọn này bị thay đổi, PS sẽ giữ nguyên những thay đổi đó. Nếu bạn không chắc, cách tốt nhất là reset lại công cụ. Click chuột phải vào icon của công cụ trên thanh tùy chọn:



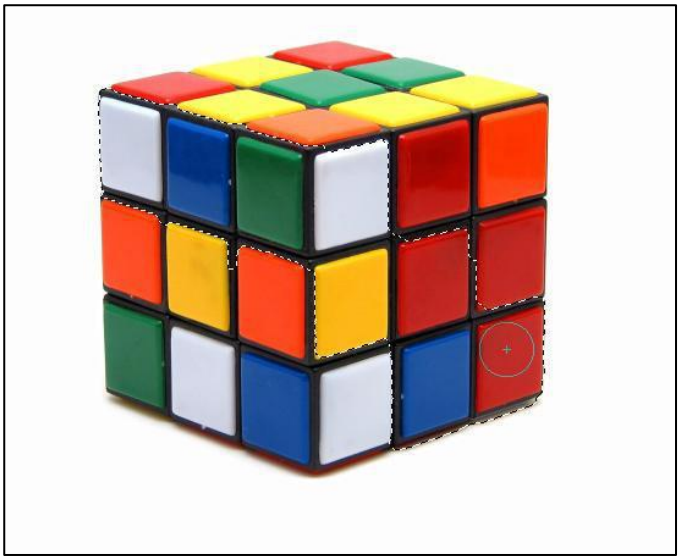
Hình 9.14. Reset công cụ

Chọn Reset tool sẽ khiến cho các tùy chọn trở về mặc định ban đầu của PS. Chọn Reset All Tools sẽ reset tòan bộ các công cụ trong tools bar.

d. Quick Selection tool

Click vào icon hoặc phím tắt W. Quick Seletion tool hoạt động giống như một brush, có nghĩa là bạn có thể chỉnh size, size lớn thì vùng chọn sẽ lớn và ngược lại. Dùng phím tắt "[" hoặc "]" để tăng/giảm size. Bạn chỉ cần click và kéo chuột trên văn bản ảnh,

công cụ sẽ tự động cho ra vùng chọn, đây là một công cụ tự động tìm nét khá thông minh và kết quả có thể sẽ làm bạn ngạc nhiên.



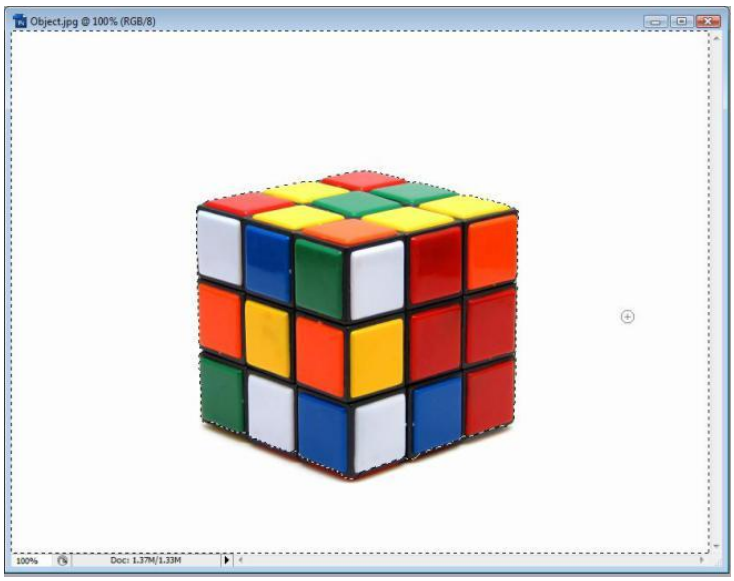
Hình 9.15. Sử dụng công cụ Quick Seletion tool

Theo mặc định, quick selection tool luôn là thêm vào vùng chọn có sẵn, nghĩa là nếu bạn đã có sẵn một vùng chọn trước đó, trừ phi bạn bỏ vùng chọn đó, sử dụng quick seletion tool sẽ thêm vào vùng chọn đó chứ không tạo vùng chọn mới. Nếu bạn giữ phím Alt, quick selection tool sẽ cắt bớt khỏi vùng chọn có sẵn.

Thử dùng quick selection tool để tạo vùng chọn quanh khối rubik. Sau khi thử nghiệm xong ấn Ctrl + D để bỏ vùng chọn.

Đảo vùng chọn: Khối rubik này là một khối hình khá đơn giản, xong trong đa phần các trường hợp chúng ta sẽ phải làm việc với những hình ảnh phức tạp hơn nhiều. Trong những trường hợp vật thể có hình dạng phức tạp nhưng nằm trên một background tương đối đồng đều, thay vì tỉ mẩn tạo vùng chọn chính xác xung quanh vật thể, chúng ta tạo vùng chọn dựa trên background của hình, rồi đảo vùng chọn.

Trong trường hợp này, chúng ta thấy khối rubik nằm trên một nền trắng rất đều, vì thế hãy dùng công cụ Magic wand hay quick selection tool để chọn tòan bộ vùng màu trắng trên văn bản ảnh:



Hình 9.16. Sử dụng Magic wand hoặc quick selection tool

Một khi đã có vùng chọn như trên, vào Select -> Inverse hoặc phím tắt Ctrl + Shift + I để đảo vùng chọn. Và chúng ta sẽ có một vùng chọn khá hòan hảo xung quanh khối rubik.

e. Các thao tác với vùng chọn

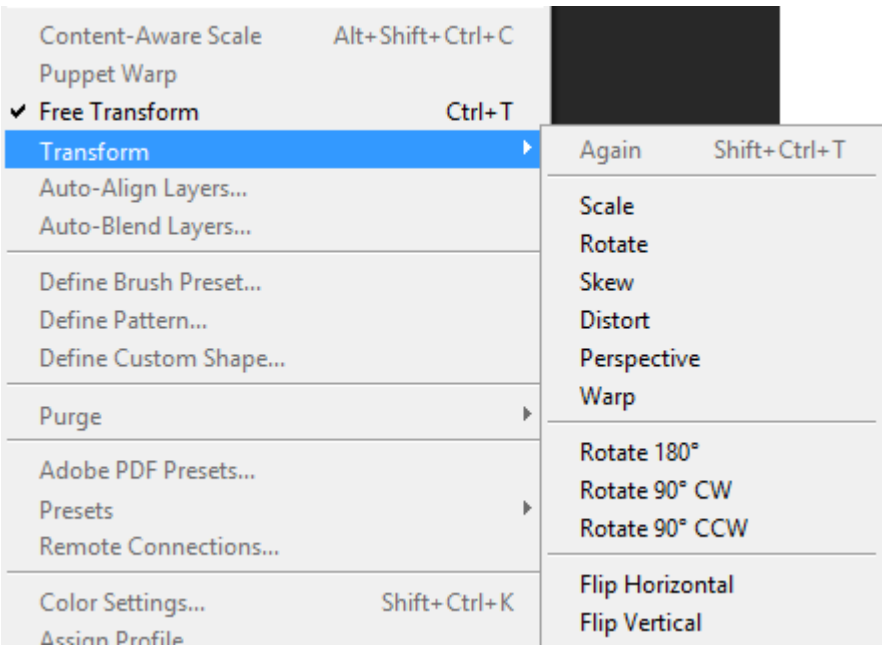
- **Hủy vùng chọn**: Select -> Deselest (Ctrl + D)

- **Di chuyển vùng chọn**: Dùng công cụ Move (V), đặt con trỏ vào giữa vùng chọn, sau đó giữ chuột trái rồi di chuyển sang vị trí mới mà bạn mong muốn.

- **Sao chép vùng chọn**: Dùng công cụ Move (V) đặt con trỏ vào phần lựa chọn, nhấn và giữ phím ALT. Khi đó con trỏ trở thành mũi tên đôi và bạn có thể kéo thả vào vị trí mong muốn. Bạn cũng có thể sử dụng cách thông thường Copy/Paste để thực hiện sao chép và dán vùng chọn sang một vị trí khác trong cùng 1 file PSD hoặc 1 file PSD khác.

- **Thêm, bớt vùng chọn**: Nhấn SHIFT và dùng chuột trái để kéo thêm vùng chọn (đã được nêu trong phần trước). Ngược lại, nếu muốn trừ bớt vùng chọn ta nhấn ALT và dùng chuột trái kéo để chọn bỏ bớt vùng chọn *(đã được nêu trong phần trước).*

- **Biến đổi vùng chọn trên Layer**: Vào menu Edit -> Transform (Ctrl + T):
    - **Scale**: Co giãn vùng ảnh chọn hoặc đối tượng ảnh trên layer
    - **Skew**: làm nghiêng vùng ảnh chọn hoặc đối tượng ảnh trên layer.
    - **Distort**: Hiệu chỉnh biến dạng hình ảnh.
    - **Perspective**: Thay đổi phối cảnh của vùng ảnh chọn
    - **Rotate**: Xoay vùng ảnh chọn hoặc đối tượng trên layer.



Hình 9.17. Biến đổi vùng chọn với Transform

- **Làm mềm biên chọn**: Đây là phương pháp làm cho biên chọn mềm hơn và không bị gãy khúc. Để làm mềm biên chọn ta click chọn menu Select -> Feather, nhập giá trị độ mờ biên tùy ý, sau đó click chọn OK.

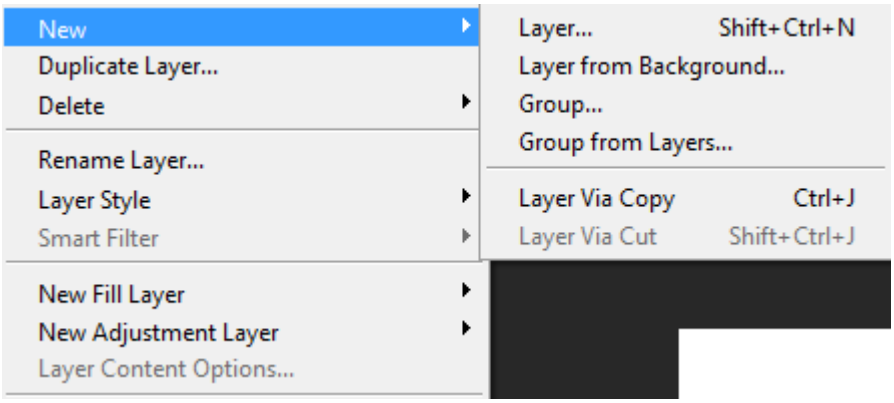
1.1.5. Thao tác với layer

Một trong những thành phần chuẩn của Photoshop là Layers, nó chính là nền tảng để bạn có thể tạo nên những layout web đẹp. Bảng Layers là một trong những thành phần quan trọng nhất của Photoshop, bất cứ khi nào sử dụng Photoshop bạn cũng mất rất nhiều thời gian vào việc thao tác trên panel này. Nếu không thấy hiển thị thành phần này trong cửa sổ làm việc, bạn có thể mở lại nó bằng cách vào menu Window -> Layers (hoặc nhấn phím F7).

Mỗi file trong Photoshop bao gồm 1 hoặc nhiều layer. Khi bạn mở một file mới, theo mặc định nó sẽ là layer background, layer này có thể chứa một màu nào đó hoặc một hình ảnh mà có thể nhìn thấy qua vùng trong suốt của layer nằm trên nó. Bạn có thể quan sát và thao tác với các layer trên Layers palette.

Những layer mới sẽ là hình trong suốt cho đến khi bạn thêm nội dung hoặc các đơn vị Pixel hình ảnh vào. Làm việc với layer cũng tương tự như bạn vẽ trên một cuốn sách nhiều trang. Mỗi trang giấy có thể được chỉnh sửa, thay đổi vị trí, xóa bỏ mà không ảnh hưởng đến những trang khác. Khi các trang giấy được sắp xếp chồng lên nhau, toàn bộ bức vẽ sẽ hiện lên.

a. Cách tạo layer ảnh và copy layer ảnh



Hình 9.18. Các thao tác với layer

- Tạo mới layer: Vào menu Layer -> New -> Layer (Shift + Ctrl + N), đặt tên layer.

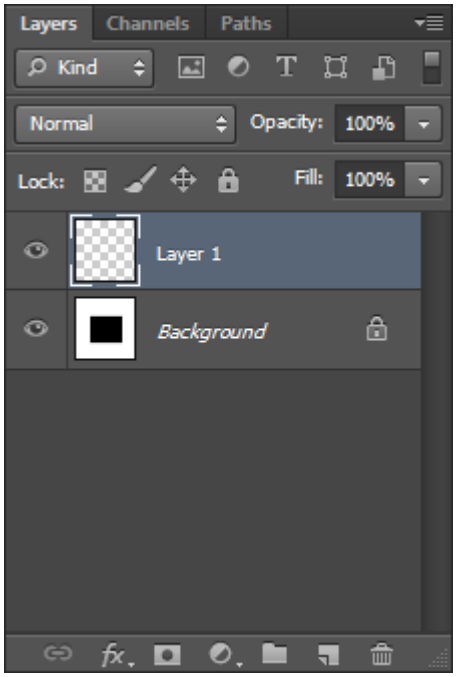
- Copy vùng ảnh chọn tạo thành layer mới: Vào menu Layer -> New -> Layer Via Copy (Ctrl + J).

- Cut vùng ảnh chọn tạo thành layer mới: Vào menu Layer -> New -> Layer Via Cut (Shift + Ctrl + J).

- Nhân đôi layer: Vào menu Layer -> Duplicate Layer…

b. Sắp xếp các layer

- Để sắp xếp các layer, ta dùng chuột đặt vào layer cần di chuyển, khi đó con trỏ chuột có hình bàn tay, nhấn chuột trái và di chuyển lên xuống dưới các layer hiện có.

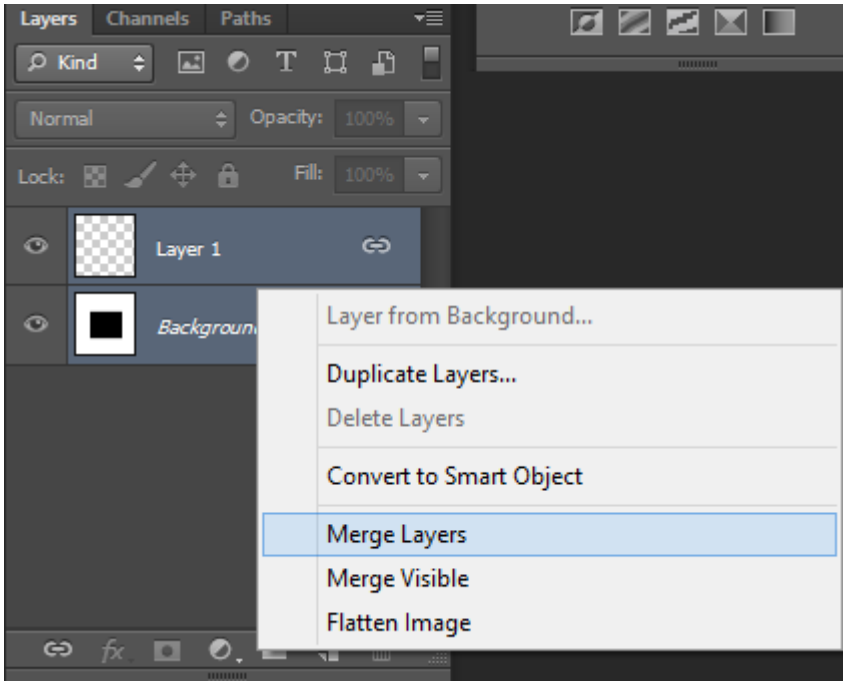


Hình 9.19. Sắp xếp các layer

c. Liên kết và gộp các layer

- Một cách rất hiệu quả và hay sử dụng là liên kết 2 hay nhiều layer lại với nhau. Với các layer đã được liên kết, bạn có thể di chuyển và biến đổi chúng một cách đồng thời để duy trì được vị trí cố định của các phần ảnh trên layer.

Để liên kết các layer lại với nhau, ta giữ Ctrl rồi dùng chuột trái chọn các layer cần liên kết. Sau đó bấm vào biểu tượng link (hình mắt xích), khi đó các layer đã được liên kết lại với nhau.



Hình 9.20. Thực hiện gộp các layer

- Cũng làm tương tự như trên, tuy nhiên ta không bấm vào biểu tượng link và thực hiện click chuột phải, sau đó chọn Merge Layer (Ctrl + E) để gộp các layer lại với nhau. Để gộp tất cả các layer lại với nhau tạo thành 1 ảnh ta bấm tổ hợp phím Shift + Ctrl + E.

1.1.6. Công cụ cắt xén ảnh Crop

Crop hình có tác dụng loại bỏ những phần dư thừa của một bức ảnh để mang lại bố cục chặt chẽ hơn và tập trung vào các chi tiết quan trọng.

Khi chọn Crop Tool trong Photoshop (thanh công cụ thứ 5 từ trên xuống hoặc phím C), ta sẽ thấy giá trị trên các thanh tùy chọn bên trên, đây là phần do bạn tự nhập giá trị.

Hình 9.21. Thanh tùy chọn của công cụ Crop

Nếu crop hình để bố cục lại hình ảnh mà không quan tâm đến kích thước và độ phân giải sau khi crop, bạn nên để trống tất cả các thông số trên thanh tùy chọn này hoặc bấm nút clear. Như vậy hình ảnh sẽ giữ nguyên giá trị kích thước gốc. Tức là chất lượng ảnh crop không thay đổi, chỉ thay đổi độ rộng (width) và chiều cao (height).

Để xác định kích thước file ảnh sau khi crop, bạn nhập giá trị chiều rộng vào ô *Width*, chiều cao vào ô *Height*, bức ảnh sẽ có kích thước là giá trị bạn đã nhập.

Ô *Resolution* bên cạnh 2 giá trị *Width* và *Height* cho phép bạn nhập độ phân giải mong muốn (ảnh thông thường là 72 pixels/inch).

Cách thực hiện: Trên file ảnh cần crop, giữ chuột trái và rê chuột trên phần ảnh cần giữ lại (không cần chọn chính xác vì có thể chỉnh lại). Khi nhả chuột, một khung bao có 8 nút điều khiển xuất hiện, phần bên trong khung bao sẽ là phần ảnh được giữ lại.

Bạn có thể cân chỉnh lại vùng chọn bằng cách, bấm và rê 1 trong 4 nút điều khiển trên mỗi cạnh khung bao để thay đổi kích thước ngang hoặc dọc. Bạn có thể giữ ảnh với tỉ lệ ban đầu khi kết hợp giữ *Shift* và thay đổi 1 trong 4 góc.

Khi chọn thanh *Perspective* trên thanh tùy chọn, các nút điều khiển ở 4 góc khung hình bao sẽ cho phép tùy chọn phối cảnh khác nhau, có thể làm méo hình và biến dạng hình ảnh.

Nếu muốn thay đổi vị trí vùng chọn, kích chuột vào bên trong khung bao và rê sang nơi khác, nút tròn ngay trung tâm khung bao là tâm đối xứng. Nếu muốn xoay vùng chọn, đặt chuột bên ngoài khung bao, một mũi tên cong xuất hiện cho phép bạn xoay vùng chọn theo ý muốn.

Khi đã ưng ý, bạn kết thúc Crop bằng cách nhấp đúp chuột vào bên trong khung bao hoặc bấm phím *Enter* hoặc click chuột vào nút *"V"* trên thanh tùy chọn.

1.2. Thiết kế giao diện website

1.2.1. Phác thảo ý tưởng thiết kế website

Bất kỳ một công việc nào, trước khi thực hiện cũng nên có một kế hoạch cụ thể. Công việc thiết kế website cũng không nằm ngoài quy luật đó. Trước khi thực hiện thiết kế website, bạn nên phác thảo trước ý tưởng, nó sẽ đem lại cho bạn rất nhiều lợi ích.

Có rất nhiều người cho rằng thiết kế web đơn giản chỉ là việc thiết kế để tạo ra giao diện website gồm nhiều hình ảnh hoặc chỉ cần đặt ở đâu thì đặt. Tuy nhiên không hẳn như thế. Thiết kế giao diện website chỉ là một khâu trong việc thiết kế để tạo ra website hay ta gọi ngắn gọn là "thiết kế website". Nếu không có sự thống nhất từ đầu việc xây dựng giao diện website thì sẽ có rất nhiều vấn đề sau này mà người tạo giao diện bắt buộc phải "đập đi" và xây dựng lại. Bạn hoàn toàn có thể hạn chế được những việc này bằng phác thảo ý tưởng thiêt kế.

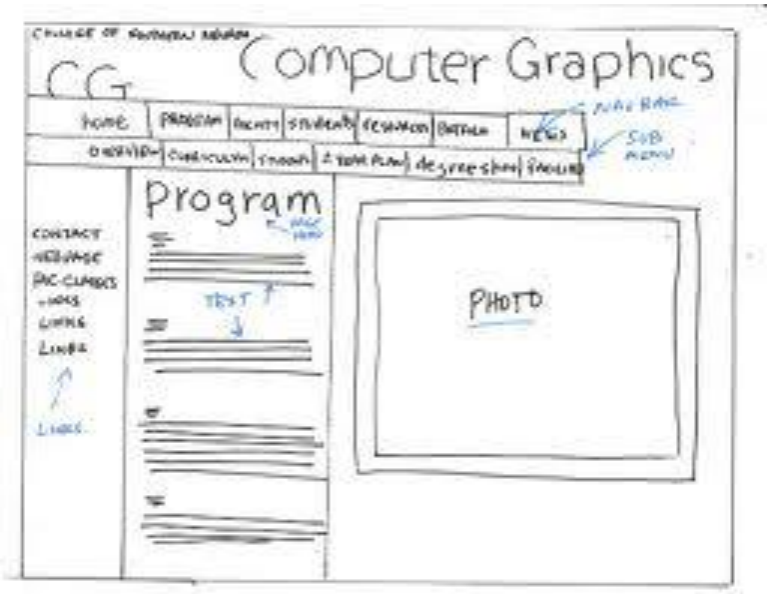
Bạn có thể sử dụng giấy thông thường hoặc giấy chuyên dụng để phác thảo ý tưởng của mình. Phác thảo ý tưởng giúp bạn định hình được những công việc cần làm, phiên bản phần mềm cần sử dụng khi xây dựng website. Từ ý tưởng sơ khai ban đầu đó bạn có thể hình dung ra các công việc cần thực hiện cho bước tiếp theo của từng phần, từng mục trong bản thảo, khi bạn đã xác định công việc chính là gì, bạn có thể dễ dàng phát triển chi tiết theo từng phần.

Nhìn chung việc phác thảo ý tưởng trước khi thiết kế giúp bạn:

- Tạo sự thống nhất giữa các bộ phận thiết kế web
- Thiết kế website một cách dễ dàng
- Sắp xếp công việc cần làm theo một trình tự khoa học
- Giảm thiểu sai sót trong quá trình thực hiện thiết kế website dần xây dựng cho bạn một phong cách riêng.

Bạn muốn có một website theo sở thích cá nhân, phục vụ cho mục đích kinh doanh hay theo yêu cầu của khách hàng... Những điều lưu ý dưới đây sẽ giúp bạn có một cái nhìn tổng thể để tạo nên trang web mà bạn mong muốn:

Chọn ý tưởng thiết kế: Ý tưởng thiết kế không phải là những ý tưởng trong đầu mà là các ý tưởng đã được phác ra giấy. Do đó, bạn hãy phác thảo ý tưởng thiết kế của mình ra giấy thông thường hoặc giấy chuyên dụng.



Hình 9.22. Phác thảo ý tưởng thiết kế website

Triển khai ý tưởng của bạn: Sau khi kiểm duyệt ý tưởng, bạn phát triển ý tưởng đó thành tổng thể hoàn chỉnh. Trước hết bạn chọn bao nhiêu thành phần sẽ có trong tổng thể và đó là những thành phần nào. Các thành phần này sẽ được định vị và chọn tỷ lệ tương ứng với nhau và tương ứng với toàn bộ tổng thể. Sau đó bạn thêm các đặc điểm phụ vào.

Thiết kế giao diện website: Giao diện là bộ phận giao tiếp giữa người sử dụng với các nội dung của website, có nhiệm vụ chính là hướng dẫn người truy cập điều khiển và sử dụng website. Nội dung của website được thể hiện trên giao diện bằng các biểu tượng, các nút và các nối kết. Để cho người sử dụng dễ tìm thông tin mà họ mong muốn, menu cần phải thể hiện cấu trúc của nội dung qua vị trí, kích thước và màu sắc của các nút.

Các biểu tượng đặc trưng sẽ giúp người truy cập dễ dàng liên tưởng đến nội dung của nó. Để có thể dễ dàng nhận ra các nút và các nối kết, bạn nên thiết kế giao diện đơn giản, không cầu kỳ và trang trí quá nhiều làm rối mắt. Đồng thời các nút và các nối kết phải thay đổi hình dạng màu sắc khi người truy cập đưa con trỏ chuột vào hoặc kéo ra. Quan trọng nhất vẫn là tên của nút và nối kết. Đặt tên nút sao cho ngắn gọn nhưng dễ hiểu và hiểu chính xác.

Làm thiết kế mẫu: Thiết kế website mẫu có thể được coi là thiết kế hoàn chỉnh nhưng chưa có phần lập trình, bao gồm lời, hình ảnh, nút và các trang trí. Ví dụ: Bạn có thể chọn hình ảnh một con mắt cho mỗi nút, con mắt mở lên khi con chuột ở bên ngoài và khép lại khi người truy cập đưa con chuột lên trên nút.

Hình 9.23. Thiết kế mẫu

Thiết kế mẫu trình bày phần thiết kế web hoàn chỉnh của từng trang với đầy đủ nội dung và cách trình bày các thành phần. Bạn cũng phải cân nhắc tính thẩm mỹ trong bước này. Kiểm tra lại các nguyên tắc về mỹ thuật và các nguyên tắc thiết kế để đảm bảo tính thống nhất, nhất quán và tập trung của thiết kế.

Kiểm tra tính khả dụng của website: Để kiểm tra tính khả dụng, bạn phải trả lời 4 câu hỏi sau đây:

- Trang web này là trang web gì?
- Tôi có thể làm gì trong trang web này?
- Tôi sẽ làm điều đó như thế nào?
- Tại sao tôi làm điều đó ở đây?

Một số yêu cầu cụ thể sau đây sẽ giúp bạn thực hiện dễ dàng hơn:

- Trang chủ phải có phần giới thiệu ngắn để cho biết đây là website gì.
- Nội dung ngắn gọn, súc tích, từ ngữ dùng chính xác, không gây nhầm lẫn.
- Kích thước file tối thiểu
- Các nút và kết nối dễ nhận biết

- Font chữ và kích cỡ dễ đọc

- Không có quá nhiều trang trí.

- Không có quá nhiều thứ mời gọi như: *"Click Me", "Free", "Hot",...* ễ làm rối mắt người dùng.

Khi bạn đã hiểu rõ những việc trên, thì việc thiết kế website của bạn sẽ trở nên dễ dàng và hiệu quả hơn, giúp bạn kinh doanh đạt hiệu quả cao và thu lại được nhiều lợi nhuận.

1.2.2. Thiết kế giao diện website

Việc thiết kế giao diện website *(layout web)* là việc làm của các designer. Do đó giáo trình này sẽ không đi sâu vào việc hướng dẫn chi tiết cách thiết kế một giao diện web như thế nào mà chỉ giới thiệu sơ lược các bước để bạn có cái nhìn tổng quát về việc thiết kế. Các bước cơ bản để thiết kế giao diện website gồm có:

a. Bước 1. Lên ý tưởng cho website

Trong mọi việc, bước đầu tiên bao giờ cũng là bước khó nhất. Điều đó càng đúng với nếu bạn chưa có kinh nghiệm và không biết gì nhiều về những việc mà mình sẽ làm. Mặc dù đã được nêu khá chi tiết trong phần trước, nhưng có lẽ từng đó chưa đủ để bạn có thể bắt tay ngay vào việc lên ý tưởng cho website của mình. Khi đó, bạn nên tìm nguồn cảm hứng và định hình ý tưởng thiết kế bằng cách tìm kiếm trên mạng những trang web có cấu trúc, bố cục, hướng thiết kế, ý tưởng tương đối giống với ý tưởng thiết kế website của bạn để tham khảo, đặc biệt là những website cùng nằm trong 1 lĩnh vực. Quá trình này nên dừng lại khi:

- Bạn đã có được ít nhất 10 – 20 thiết kế mà bạn cảm thấy đẹp và phù hợp với trang web của mình

- Có ít nhất một trang thiết kế mà nó khiến bạn chỉ muốn copy nguyên xi của nó về *(dù rằng mục đích của bạn không phải là như vậy)*.

Khi đó bạn sẽ thực hiện vẽ lại website theo ý tưởng của bạn ra giấy đã là điều không mấy khó khăn.

b. Bước 2: Thiết kế layout với photoshop

Ở bước này, bạn buộc phải sử dụng tất cả kiến thức về đồ họa cũng như các kỹ thuật photoshop đã được học để có thể thiết kế được một layout web dựa trên ý tưởng website đã được lên từ bước 1.

1.3. Chuyển đổi từ PSD sang HTML

Việc biến một giao diện được thiết kế trong Photoshop thành HTML là một phần rất quan trọng trong thiết kế web. Sau này nếu bạn muốn tự tạo giao diện cho trang web của mình cho dù nó động hay là web tĩnh, thì việc chuyển một thiết kế từ PSD sang HTML là

việc đầu tiên bạn phải làm. Giả sử bạn đã có 1 file giao diện website được thiết kế bởi designer. Khi đó, công việc tiếp theo của bạn là thực hiện tuần tự theo các bước sau:

- Sử dụng photoshop để cắt những hình ảnh cần thiết cho website.
- Viết mã HTML, sử dụng CSS để bố cục và trình bày nội dung.
- Bổ sung hiệu ứng javascript/jQuery *(nếu cần thiết)*
- Kiểm tra và đối chiếu giữa các trình duyệt.

Tuy nhiên, để có thể thành thạo các bước kể trên đòi hỏi bạn phải nắm vững kiến thức đã được trình bày trong các phần trước như:

- Photoshop: đặt đường hướng dẫn *(guides)*, lấy mã màu bất kỳ trong file thiết kế, tắt bật, dồn ghép các layer, cắt ảnh thành phần từ file thiết kế, đo kích thước các thành phần trên file thiết kế.
- HTML: viết được mã HTML tương ứng đối với các thành phần của file thiết kế. Ở đây, bạn nên thực hiện bố cục tổng thể, sau đó mới chi tiết hóa từng phần trong website.
- CSS: CSS dành cho bố cục website, CSS cho hình ảnh, CSS cho text và quan trọng nhất là CSS để tạo mối liên hệ giữa các thành phần trong website.

B. BÀI TẬP

2.1. Câu hỏi ôn tập

Câu 1. Cách nào sau đây không dùng để tô màu cho một vùng lựa chọn:

T. Edit/Fill : Foreground
U. Alt/Opt + Backspace
V. Paint Bucket (Hộp sơn)
W. Select/Inserve

Câu 2. Công cụ nào sau đây không có trong Photoshop CS toolbar?

O. Eyedroper
P. Color Range Selection
Q. Replace Color
R. Type Mask

Câu 3. Channel palette có thể làm được những gì?

T. Hiển thị các layer màu sắc.
U. Dùng để thực hiện các chức năng hiệu chỉnh (Adjustment).
V. Thay đổi màu và chế độ hình ảnh (imagine mode) của ảnh.
W. Cho biết thông tin hình ảnh của mỗi kênh màu.

Câu 4. Cái nào sau đây không phải là 1 phương pháp lựa chọn?

O. Quickmasking
P. Layermasking
Q. Sử dụng công cụ Lasso
R. Color Range

Câu 5: Chế độ QuickMask là gì?

X. Giúp bạn ẩn các Layer 1 cách nhanh nhất
Y. 1 loại Ruby mới.
Z. Là 1 alpha channel tạm thời dùng để
AA. Tồn tại "vĩnh viễn" giống

lưu trữ vùng lựa chọn. như layer mask

2.2. Bài tập

Bài 1. Sử dụng các kiến thức photoshop đã được học để thiết kế layout web sau:

Bài 2. Chuyển giao diện website trên sang mã HTML & CSS (cắt giao diện).