**I. Làm quen với HTML**

**1. Cấu trúc của 1 file HTML**

- Tạo 1 folder, trong folder tạo 1 file **index.html**

// ! enter =>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

</body>

</html>

- Thẻ html: <html> </html>

- Thẻ head: <head> </head> // Phần đầu

- Thẻ body: <body> </body> // Phần thân

- Thẻ title: <title> </title > // Thẻ tiêu đề

- Thẻ meta có thuộc tính charset : <meta charset=”utf-8”>

**2. Comment trong HTML**

- Cú pháp: <!—enter

<body>

    <!--Phần body-->

</body>

* Windows: Ctrl + /
* MacOS: Cmd + /

**3. Các thẻ HTML thông dụng**

- Khi viết tiêu đề, sử dụng thẻ h (VD: h1,h2,…)

- Khi viết đoạn chữ, text sử dụng thẻ p: <p></p>

- Hiển thị hình ảnh, sử dụng thẻ image: <img src=”địa chỉ ảnh” alt=”Dùng khi địa chỉ ảnh lỗi”>

- Đặt liên kết vào thẻ, sử dụng thẻ a:

<a href=”https://www.facebook.com”>Facebook</a>

- Thẻ ul, li: Thể hiện danh sách, dùng 1 thẻ ul và n thẻ li (li\*4 => tạo 4 thẻ li)

<ul>

        <li></li>

        <li></li>

        <li></li>

        <li></li>

</ul>

- Thẻ table: Tạo bảng

<table>

        <thead>

           <th>STT</th>

           <th>Tên</th>

           <th>Địa chỉ</th>

        </thead>

        <tbody>

            <tr>

                <td>1</td>

                <td>CudHnyuH</td>

                <td>HN</td>

            </tr>

        </tbody>

    </table>

- Thẻ input:

<input type="checkbox">

<input type="text">

<input type="radio">

- Thẻ button:

<button>Đăng ký</button>

- Thẻ div: Tạo khối bao quanh phần tử khác

<div> </div>

**4. Attribute trong HTML là gì?**

- Attribute: thuộc tính

- title=””;…

**II. Làm quen với CSS**

**1. Sử dụng CSS trong HTML**

- CSS in HTML?

- 3 cách:

* Internal: Bên trong, viết trong file HTML
* <!DOCTYPE html>
* <html lang="en">
* <head>
* <meta charset="UTF-8">
* <meta http-equiv="X-UA-Compatible" content="IE=edge">
* <meta name="viewport" content="width=device-width, initial-scale=1.0">
* <title>Shop - Test</title>
* <style>
* h1 {
* color: red;
* }
* </style>
* </head>
* <body>
* <div>
* <h1>CSS in HTML?</h1>
* </div>
* </body>
* </html>
* External:

- Tạo file main.css trong folder.

- Link file main.css với index.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Shop - Test</title>

    <link rel="stylesheet" href="/main.css">

</head>

<body>

    <div>

        <h1>CSS in HTML?</h1>

    </div>

</body>

</html>

* Inline

<h1 style="color: blue;">CSS in HTML?</h1>

**2. ID và Class**

- CSS selector: cách gọi đến các phần tử (element) để CSS

// HTML

<h1 id="first-heading">CSS in HTML?</h1>

//CSS

#first-heading {

    color: aqua;

}

// HTML

<h1 class="second-heading">CSS in HTML?</h1>

// CSS

.second-heading {

    color: brown;

}

**3. CSS selectors cơ bản**

**Giới thiệu**

Trong CSS, CSS selectors là các cách chúng ta sử dụng để chọn ra các phần tử (elements) mà chúng ta muốn "style" cho chúng. Có 2 CSS selectors thân thuộc với chúng ta nhất là id và class.

**Sử dụng CSS selector ID**

<h2 id="primary-heading">Tiêu đề 1</h2>

<h2 id="secondary-heading">Tiêu đề 2</h2>

...

<style>

#primary-heading {

color: red;

}

</style>

Kết quả, thẻ có id="primary-heading" thành màu đỏ.

**Tiêu đề 1**

**Tiêu đề 2**

Trong cùng 1 page HTML, mỗi ID cần là duy nhất. Đây là nguyên tắc của việc sử dụng ID, nếu vi phạm sẽ gặp cảnh báo trong tab Console của dev tool, và Javascript sẽ không lấy ra được cả 2 phần tử có dùng ID (Javascript các bạn sẽ học ở [khóa sau](https://fullstack.edu.vn/courses/javascript-co-ban)).

**Sử dụng CSS selector class**

**Ví dụ 1: Lựa chọn qua 1 class**

<p class="paragraph">Đoạn văn 1</p>

<p class="paragraph">Đoạn văn 2</p>

<p class="other-paragraph">Đoạn văn 3</p>

<style>

.paragraph {

color: green;

}

</style>

Kết quả, tất cả các class paragraph đều có màu xanh lục.

Đoạn văn 1

Đoạn văn 2

Đoạn văn 3

**Ví dụ 2: Lựa chọn qua nhiều class**

Một thẻ có thể có nhiều class, mỗi class cách nhau bằng 1 khoảng trắng (dấu space).

<p class="p1-normal">Đoạn văn 1</p>

<p class="p1-normal">Đoạn văn 2</p>

<p class="p1-normal p2-red">Đoạn văn 3</p>

<p class="p1-normal p2-red">Đoạn văn 4</p>

<style>

.p1-normal {

color: green;

}

/\* Lưu ý, các tên class viết liền nhau, không có khoảng trắng \*/

.p1-normal.p2-red {

color: red;

}

</style>

Kết quả, tất cả các thẻ đồng thời có cả class p1-normal và p2-red đều có màu đỏ.

Đoạn văn 1

Đoạn văn 2

Đoạn văn 3

Đoạn văn 4

Tương tự như vậy, nếu thẻ có nhiều class hơn bạn chỉ cần nối thêm tên class vào khi CSS, ví dụ: .class1.class2.class3. Trong trường hợp này, thuộc tính CSS nằm trong selector được nối bởi nhiều class hơn (chỉ ra chi tiết hơn) sẽ được ưu tiên hơn.

**Ví dụ 3: CSS cho các thẻ con**

<p class="children-1">Đoạn văn 1 (không parent)</p>

<p class="children-2">Đoạn văn 2 (không parent)</p>

<div class="parent">

<p>Đoạn văn 1 (parent 1)</p>

<p class="children-1">Đoạn văn 1 (con của .parent)</p>

<p class="children-2">Đoạn văn 2 (con của .parent)</p>

</div>

<style>

/\* Chọn tất cả thẻ "p" là con của thẻ có class "parent" \*/

.parent p {

color: green;

}

/\* Chọn tất cả thẻ có class "children-1" là con của thẻ có class "parent" \*/

.parent .children-1 {

color: red;

}

</style>

Đoạn văn 1 (không parent)

Đoạn văn 2 (không parent)

Đoạn văn 1 (parent 1)

Đoạn văn 1 (con của .parent)

Đoạn văn 2 (con của .parent)

Chúng ta có thể tùy ý kết hợp các CSS selectors đã học để lựa chọn ra đúng thẻ chúng ta muốn CSS. Ở ví dụ trên, để lựa chọn thẻ con nằm trong 1 thẻ khác ta sẽ sử dụng selector có cú pháp selector-cha selector-con (lưu ý, giữa 2 selectors có một khoảng trắng - dấu cách).

**Bảng tra cứu CSS selectors cơ bản**

Trong bài này mình chỉ đưa ra những CSS selectors cơ bản và thường xuyên được sử dụng nhất.

| **Selector** | **Ví dụ** | **Mô tả** |
| --- | --- | --- |
| .class | .intro | Chọn tất cả các thẻ có class="intro" |
| .class1.class2 | .name1.name2 | Chọn tất cả các thẻ có cả name1 và name2 được đặt trong thuộc tính class của nó |
| .class1 .class2 | .name1 .name2 | Chọn tất cả các thẻ có class="name2" là con của một phần tử có class="name1" |
| \* | \* | Chọn tất cả các thẻ |
| element | h2 | Chọn tất cả các thẻ h2 |
| element.class | div.box | Chọn tất cả thẻ div có class="box" |
| element, element | div, h2 | Chọn tất cả thẻ div và h2 |
| element element | div p | Chọn tất cả thẻ p trong thẻ div |
| element > element | div > p | Chọn tất cả thẻ p là con trực tiếp của thẻ div |
| element + element | div + p | Chọn thẻ p đứng liền kề sau thẻ div |
| element ~ element | div ~ p | Chọn tất cả thẻ p đứng sau thẻ div |

Tham khảo thêm tại: [https://www.w3schools.com/cssref/css\_selectors.asp](https://fullstack.edu.vn/external-url?continue=https%3A%2F%2Fwww.w3schools.com%2Fcssref%2Fcss_selectors.asp)

CSS selectors là công cụ vô cùng mạnh mẽ. Toàn bộ các selectors trong CSS đều có thể linh động kết hợp với nhau khi sử dụng giúp lập trình viên có thể chọn được bất cứ phần tử HTML nào trong website.

**4. Độ ưu tiên trong CSS**

- Giữa Internal và External sẽ không có sự ưu tiên CSS, chỉ ưu tiên CSS được viết mới hơn.

- Độ ưu tiên giảm dần:

Inline => #id => .class => tag (h1,…) => Universal( \*{})

- Equal specificity? Sự giống nhau khi gọi đến 1 element

- Universal selector and inherited? Những cái chung

\*{

}

**5. Sử dụng biến trong CSS**

:root {

    --text-color: red;

}

h1 {

    color: var(--text-color);

}

- Biến toàn cục :root {}

**6. Các đơn vị trong CSS**

*a. Absolute units? Đơn vị tuyệt đối*

* px
* pt
* cm
* mm
* inch
* pc

- Kích thước không bị thay đổi bởi các yếu tố bên ngoài.

*b. Relative units? Đơn vị tương đối*

* % // Phụ thuộc vào thẻ chứa nó
* rem // Phụ thuộc vào thẻ html
* em // Phụ thuộc vào thẻ gần nhất chứa nó có thuộc tính font-size.
* vw // Chiều ngang trình duyệt
* vh // Chiều dọc trình duyệt
* vmin
* vmax
* ex
* ch

**-** Kích thước thay đổi theo đối tượng phụ thuộc.

**7. Một số hàm trong CSS**

* Var()
* Liner-gradient()
* Rgba()
* Rgb()
* Calc
* Attr()

Background: rgb(0,0,0); //mã màu rgb

Width: calc(120px +- … 20px); // tính toán kích thước

// HTML

<a href="http://facebook.com">facebook</a>

// CSS

a::after {

    content: "(" attr(href)")";

}

**8. Pseudo classes trong CSS**

- Lớp giả

* :root // Tham chiếu tới phần tử gốc
* :hover // Kích hoạt khi di chuột vào
* :active // Bấm chuột vào giữ
* :first-child //
* :last-child

**9. CSS Pseudo-elements**

* ::before
* ::after
* ::first-letter
* ::first-line
* ::selection

p::before {  
  content: "Read this -";  
  background-color: yellow;  
  color: red;  
  font-weight: bold;  
}

**III. Đệm, viền và khoảng lề**

**1. Thuộc tính padding (đệm)**

*Đệm CSS*

Các thuộc tính CSS padding được sử dụng để tạo không gian xung quanh nội dung của phần tử, bên trong bất kỳ đường viền đã xác định nào.

Với CSS, bạn có toàn quyền kiểm soát phần đệm. Có các thuộc tính để đặt phần đệm cho mỗi bên của một phần tử (trên, phải, dưới và trái).

*Đệm - Các mặt riêng lẻ*

CSS có các thuộc tính để chỉ định phần đệm cho mỗi bên của một phần tử:

* padding-top
* padding-right
* padding-bottom
* padding-left

Tất cả các thuộc tính đệm có thể có các giá trị sau:

* *chiều dài* - chỉ định phần đệm theo px, pt, cm, v.v.
* *%* - chỉ định phần đệm theo % chiều rộng của phần tử chứa
* kế thừa - chỉ định rằng phần đệm phải được kế thừa từ phần tử cha

**Lưu ý:** Giá trị âm không được phép.

**2. Thuộc tính border (đường viền)**

Thuộc tính đường viền CSS cho phép bạn chỉ định kiểu, chiều rộng và màu của đường viền của phần tử.

*Kiểu viền CSS*

Thuộc tính border-style chỉ định loại đường viền sẽ hiển thị.

Các giá trị sau đây được cho phép:

* dotted- Xác định một đường viền chấm
* dashed- Xác định đường viền nét đứt
* solid- Xác định một đường viền vững chắc
* double- Xác định đường viền kép
* groove- Xác định đường viền có rãnh 3D. Hiệu ứng phụ thuộc vào giá trị màu viền
* ridge- Xác định đường viền có gờ 3D. Hiệu ứng phụ thuộc vào giá trị màu viền
* inset- Xác định đường viền chèn 3D. Hiệu ứng phụ thuộc vào giá trị màu viền
* outset- Xác định đường viền đầu 3D. Hiệu ứng phụ thuộc vào giá trị màu viền
* none- Xác định không có biên giới
* hidden- Xác định một đường viền ẩn

Thuộc tính border-style có thể có từ một đến bốn giá trị (đối với đường viền trên cùng, đường viền bên phải, đường viền dưới cùng và đường viền bên trái).

**3. Thuộc tính margin (khoảng cách lề)**

Lề được sử dụng để tạo không gian xung quanh các phần tử, bên ngoài bất kỳ đường viền đã xác định nào.

*Lề CSS*

Các thuộc tính CSS marginđược sử dụng để tạo không gian xung quanh các phần tử, bên ngoài bất kỳ đường viền đã xác định nào.

Với CSS, bạn có toàn quyền kiểm soát các lề. Có các thuộc tính để đặt lề cho mỗi bên của một phần tử (trên, phải, dưới và trái).

*Ký quỹ - Các mặt riêng lẻ*

CSS có các thuộc tính để chỉ định lề cho mỗi bên của một phần tử:

* margin-top
* margin-right
* margin-bottom
* margin-left

Tất cả các thuộc tính lề có thể có các giá trị sau:

* tự động - trình duyệt tính toán ký quỹ
* *chiều dài* - chỉ định lề tính bằng px, pt, cm, v.v.
* *%* - chỉ định lề tính bằng % chiều rộng của phần tử chứa
* kế thừa - chỉ định rằng lề phải được kế thừa từ phần tử cha

**Mẹo:** Giá trị âm được cho phép.

**4. Thuộc tính box-sizing**

*Kích thước hộp CSS*

Thuộc tính CSS box-sizing cho phép chúng ta bao gồm phần đệm và đường viền trong tổng chiều rộng và chiều cao của phần tử.

*Không có Thuộc tính kích thước hộp CSS*

Theo mặc định, chiều rộng và chiều cao của phần tử được tính như sau:

chiều rộng + phần đệm + đường viền = chiều rộng thực tế của phần tử  
chiều cao + phần đệm + đường viền = chiều cao thực tế của phần tử

Điều này có nghĩa là: Khi bạn đặt chiều rộng/chiều cao của một phần tử, phần tử đó thường xuất hiện lớn hơn mức bạn đã đặt (vì đường viền và phần đệm của phần tử được thêm vào chiều rộng/chiều cao đã chỉ định của phần tử).

Box-sizing = border-box;

**IV. Thuộc tính tạo nền**

**1. Thuộc tính background-image**

- Đặt 1 hoặc nhiều hình nền cho 1 phần tử.

Theo mặc định, hình nền được đặt ở góc trên cùng bên trái của phần tử và được lặp lại theo cả chiều dọc và chiều ngang.

## *CSS Syntax*

background-image: *url*|none|initial|inherit;

## *Property Values*

|  |  |
| --- | --- |
| **Value** | **Description** |
| url('*URL*') | The URL to the image. To specify more than one image, separate the URLs with a comma |
| none | No background image will be displayed. This is default |
| [conic-gradient()](https://www.w3schools.com/cssref/func_conic-gradient.php) | Sets a conic gradient as the background image. Define at least two colors |
| [linear-gradient()](https://www.w3schools.com/cssref/func_linear-gradient.php) | Sets a linear gradient as the background image. Define at least two colors (top to bottom) |
| [radial-gradient()](https://www.w3schools.com/cssref/func_radial-gradient.php) | Sets a radial gradient as the background image. Define at least two colors (center to edges) |
| [repeating-conic-gradient()](https://www.w3schools.com/cssref/func_repeating-conic-gradient.php) | Repeats a conic gradient |
| [repeating-linear-gradient()](https://www.w3schools.com/cssref/func_repeating-linear-gradient.php) | Repeats a linear gradient |
| [repeating-radial-gradient()](https://www.w3schools.com/cssref/func_repeating-radial-gradient.php) | Repeats a radial gradient |
| initial | Sets this property to its default value. [Read about initial](https://www.w3schools.com/cssref/css_initial.php) |
| inherit | Inherits this property from its parent element. [Read about inherit](https://www.w3schools.com/cssref/css_inherit.php) |

.header {

Width: 100%;

Height: 100vh;

Background-image: url(…);

Background-size: 100%;

Background-repeat: no-repeat;

}

**V. Thuộc tính vị trí**

**1. CSS position relative**

- Vị trí tương đối phụ thuộc và chính nó, lấy chính mình làm gốc tọa độ.

Position: relative;

Top: …;

Bottom: …;

Left: …;

Right: …;

**2. CSS position absolute**

- Vị trí tuyệt đối, bị phụ thuộc vào thẻ cha gần nhất có thuộc tính position.

Position: absolute;

Top: …;

Bottom: …;

Left: …;

Right: …;

**3. CSS position fixed**

- Không thay đổi header khi kéo chuột.

Position: fixed;

**4. CSS position sticky**

- Làm menu, header bám dính.

Position: sticky;

**VI. Một số tricks và tips**

**1. Các cách căn giữa trong CSS**

Text-align: center; // căn giữa theo chiều ngang

Line-height: …px; // căn giữa theo chiều dọc

Display: flex;

Margin: auto;

Display: flex;

Align-items: center;

Justify-content: center;

Position: absolute;

Top: 50%;

Left: 50%

Transform: translateY(-50%) translateX(-50%);

**2. Hiển thị ảnh dự phòng khi ảnh chính lỗi**

<img onerror=”this.src=’địa chỉ lưu ảnh’” src=”…”>

**VII. Xây dựng web The band**

1. Phân tích

2. Dựng base (xây móng)

3. Xây dựng từng phần theo phân tích

4. Hoàn thiện

[https://www.w3schools.com/w3css/tryw3css\_templates\_band.htm](https://fullstack.edu.vn/external-url?continue=https%3A%2F%2Fwww.w3schools.com%2Fw3css%2Ftryw3css_templates_band.htm)

Những thành phần thường gặp trên giao diện Website

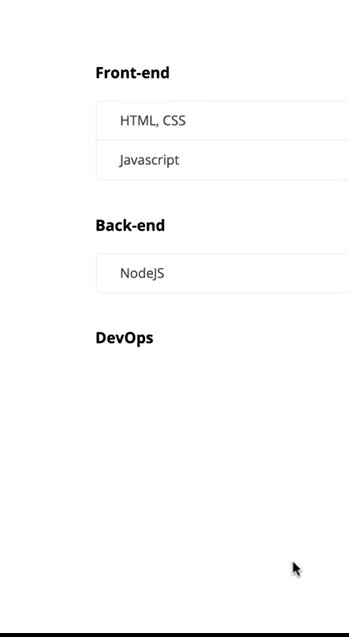
- Header (Phần đầu trang)

- Navigation (Phần điều hướng) – Nằm trong hoặc ngoài header

- Breadcrumb

C:\Users\Administrator\Downloads\1 (1).jpg

- Sidebar : 1 thanh thường được đặt ở 1 phía của màn hình



- Slider: Nội dung trượt, thông thường là hình ảnh

- Banner: Chứa nội dung ảnh, thường là ảnh quảng cáo

- Content: Nội dung chính trang web, trung tâm website

- Footer (Chân trang):



**1. Phân tích dự án**

- Header

- Slider

- Content: About (Giới thiệu), Tour, Contact, Image map

- Footer

**2. Web Resposive**

- Media queries ?

@media (min-width: …px) and (max-width: …px) {

}

- PC >= 1024px

@media (min-width: 1024px) {

}

- 740px <= Ipad <=1023

@media (min-width: 740px) and (max-width: 1023px) {

}

- Mobile < 740px

@media (max-width: 739px) {

}

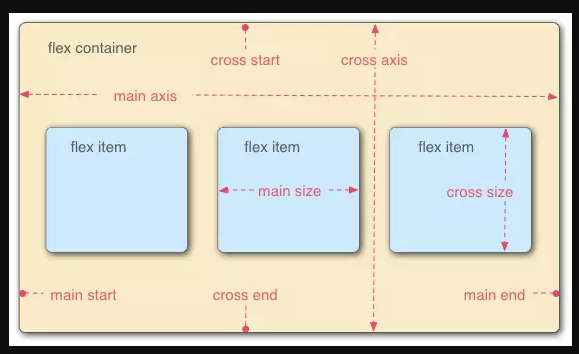
**3. Dựng bố cục web với Flexbox**

CSS flexbox là một one-dimensional(hay còn gọi là 1D) layout pattern, một trong những pattern giúp bạn dễ dàng thiết kế layout một cách linh hoạt và hiệu quả. Phân chia không gian giữa các items và kiểm soát căn chỉnh chúng trong container flex layout(vùng chứa). Với flexbox, chúng ta có thể dễ dàng sắp xếp các items từ trái sang phải, từ trên xuống dưới, đồng thời kiểm soát khoảng cách và thứ tự của các items trong container.

- Gồm 2 thành phần chính:

+ flex container

+ flex items



Trong flexbox thì chủ yếu có hai thành phần chính là: thùng chứa cha (flex container) và các phần tử con nằm bên trong (flex items).  
Ngoài ra chúng ta cũng cần quan tâm tới một số thuộc tính sau:

* **main start, main end, cross start, cross end:** Điểu bắt đầu của container theo main axis được gọi là main start, điểm kết thúc của container theo main axis gọi là main end, với cross start và cross cũng tương tự nhưng dựa theo cross axis.
* **main axis:** Trục này chính là hướng của các item hiển thị, mặc định thì sẽ chạy từ trái qua phải.
* **cross axis:** Trục này vuông góc với main axis, chạy từ trên xuống dưới.
* **main size:** Là kích thước của mỗi item dựa theo trục main axis.
* **cross size:** Là kích thước của mỗi item dựa theo trục cross axis.

**Các thuộc tính của flex container**

Dưới đây là một số thuộc tính có thể sử dụng đối với flex container:

* display
* flex-direction
* flex-wrap
* flex-flow
* justified-content
* align-items
* align-content

**4. Quy ước đặt tên class BEM**

*a. BEM là gì?*

- Là tiêu chuẩn đặt tên class khi viết CSS

- Tên viết tắt của Block Element Modifier

- Block: Khối

- Element: Thành phần trong khối

- Modifier: Bổ sung ý nghĩa cho ‘Block’ hoặc ‘Element’

+ Tại sao phải dùng BEM?

- Mỗi người đặt 1 kiểu

- Member đặt class trùng nhau, CSS đè lên nhau

+ Cú pháp:

.block

.block\_\_element

.block—modifier

.block\_\_element—modifier

+ Tính ứng dụng:

- Xây dựng layout website

- Xây dựng thành phần trên website

+ Ưu điểm

- Tính rõ ràng

- Tái sử dụng dễ dàng

- Giúp cả team làm việc với nhau dễ dàng

- Tính modul, không lo CSS của class này ảnh hưởng tới class khác

+ Nhược điểm

- Tên class dài

- Một số người cho là xấu

+ Khi nào dùng BEM là phù hợp?

- Dự án nhiều members

- Dự án lớn, số lượng pages nhiều hoặc số lượng các thành phần trên giao diện nhiều

+ Thực hành

- Làm button

- Làm message

- Làm 1 thành phần trên websites

**1. Tạo đối tượng Grid**

Grid.css

.grid {

Width: 100%;

Display: block;

Padding: 0;

}

.grid.wide {

Max-width: 1200px;

Margin: 0 auto;

}

// Tablet - PC

@media (min-width: 740px) and (max-width: 1023px) {

.wide {

Width: 644px;

}

}

// PC

@media (min-width: 1024px) and (max-width: 1239px) {

.wide {

Width: 644px;

}

}

**2. Tạo đối tượng Row**

Vai trò:

- Chứa các columns, giúp các columns nằm theo hàng ngang.

- Khi tổng chiều ngang columns vượt quá kích thước Row, cho columns xuống dòng.

- Loại bỏ khoảng thừa do gutters tạo ra ở 2 phía.

.row {

Display: flex;

Flex-wrap: wrap;

Margin-left: -4px;

Magin-right: -4px;

}

// Tablet

@media (min-width:740px) {

.row {

Margin-left: -8px;

Magin-right: -8px;

}

}

// PC

@media (min-width:1113px) {

.row {

Margin-left: -12px;

Magin-right: -12px;

}

}

// PC

@media (min-width:1024px) and (max-width:1239px) {

.wide .row {

Margin-left: -12px;

Magin-right: -12px;

}

}

**3. Tạo đối tượng Column**

Vai trò: Chứa các thành phần trong website.

.col {

Padding-left: 4px;

Padding-right: 4px;

}

.c-0 {

Display: none;

}

.c-1 {

Flex: 0 0 8.33333%;

Max-width: 8.33333%;

}

.c-2 {

Flex: 0 0 16.66667%;

Max-width: 16.66667%;

}

.c-3 {

Flex: 0 0 25%;

Max-width: 25%;

}

.c-4 {

Flex: 0 0 33.33333%;

Max-width: 33.33333%;

}

.c-5 {

Flex: 0 0 41.66667%;

Max-width: 41.66667%;

}

.c-6 {

Flex: 0 0 50%;

Max-width: 50%;

}

.c-7 {

Flex: 0 0 58.33333%;

Max-width: 58.33333%;

}

.c-8 {

Flex: 0 0 66.66667%;

Max-width: 66.66667%;

}

.c-9 {

Flex: 0 0 75%;

Max-width: 75%;

}

.c-10 {

Flex: 0 0 83.33333%;

Max-width: 83.33333%;

}

.c-11 {

Flex: 0 0 91.66667%;

Max-width: 91.66667%;

}

.c-12 {

Flex: 0 0 100%;

Max-width: 100%;

}

RESPONSIVE WEB DESIGN using Grid

Grid.css

.grid {

width: 100%;

display: block;

padding: 0;

}

.grid.wide {

max-width: 1200px;

margin: 0 auto;

}

.row {

display: flex;

flex-wrap: wrap;

margin-left: -4px;

margin-right: -4px;

}

.row.no-gutters {

margin-left: 0;

margin-right: 0;

}

.col {

padding-left: 4px;

padding-right: 4px;

}

.row.no-gutters .col {

padding-left: 0;

padding-right: 0;

}

.c-0 {

display: none;

}

.c-1 {

flex: 0 0 8.33333%;

max-width: 8.33333%;

}

.c-2 {

flex: 0 0 16.66667%;

max-width: 16.66667%;

}

.c-3 {

flex: 0 0 25%;

max-width: 25%;

}

.c-4 {

flex: 0 0 33.33333%;

max-width: 33.33333%;

}

.c-5 {

flex: 0 0 41.66667%;

max-width: 41.66667%;

}

.c-6 {

flex: 0 0 50%;

max-width: 50%;

}

.c-7 {

flex: 0 0 58.33333%;

max-width: 58.33333%;

}

.c-8 {

flex: 0 0 66.66667%;

max-width: 66.66667%;

}

.c-9 {

flex: 0 0 75%;

max-width: 75%;

}

.c-10 {

flex: 0 0 83.33333%;

max-width: 83.33333%;

}

.c-11 {

flex: 0 0 91.66667%;

max-width: 91.66667%;

}

.c-12 {

flex: 0 0 100%;

max-width: 100%;

}

.c-o-1 {

margin-left: 8.33333%;

}

.c-o-2 {

margin-left: 16.66667%;

}

.c-o-3 {

margin-left: 25%;

}

.c-o-4 {

margin-left: 33.33333%;

}

.c-o-5 {

margin-left: 41.66667%;

}

.c-o-6 {

margin-left: 50%;

}

.c-o-7 {

margin-left: 58.33333%;

}

.c-o-8 {

margin-left: 66.66667%;

}

.c-o-9 {

margin-left: 75%;

}

.c-o-10 {

margin-left: 83.33333%;

}

.c-o-11 {

margin-left: 91.66667%;

}

/\* >= Tablet \*/

@media (min-width: 740px) {

.row {

margin-left: -8px;

margin-right: -8px;

}

.col {

padding-left: 8px;

padding-right: 8px;

}

.m-0 {

display: none;

}

.m-1,

.m-2,

.m-3,

.m-4,

.m-5,

.m-6,

.m-7,

.m-8,

.m-9,

.m-10,

.m-11,

.m-12 {

display: block;

}

.m-1 {

flex: 0 0 8.33333%;

max-width: 8.33333%;

}

.m-2 {

flex: 0 0 16.66667%;

max-width: 16.66667%;

}

.m-3 {

flex: 0 0 25%;

max-width: 25%;

}

.m-4 {

flex: 0 0 33.33333%;

max-width: 33.33333%;

}

.m-5 {

flex: 0 0 41.66667%;

max-width: 41.66667%;

}

.m-6 {

flex: 0 0 50%;

max-width: 50%;

}

.m-7 {

flex: 0 0 58.33333%;

max-width: 58.33333%;

}

.m-8 {

flex: 0 0 66.66667%;

max-width: 66.66667%;

}

.m-9 {

flex: 0 0 75%;

max-width: 75%;

}

.m-10 {

flex: 0 0 83.33333%;

max-width: 83.33333%;

}

.m-11 {

flex: 0 0 91.66667%;

max-width: 91.66667%;

}

.m-12 {

flex: 0 0 100%;

max-width: 100%;

}

.m-o-1 {

margin-left: 8.33333%;

}

.m-o-2 {

margin-left: 16.66667%;

}

.m-o-3 {

margin-left: 25%;

}

.m-o-4 {

margin-left: 33.33333%;

}

.m-o-5 {

margin-left: 41.66667%;

}

.m-o-6 {

margin-left: 50%;

}

.m-o-7 {

margin-left: 58.33333%;

}

.m-o-8 {

margin-left: 66.66667%;

}

.m-o-9 {

margin-left: 75%;

}

.m-o-10 {

margin-left: 83.33333%;

}

.m-o-11 {

margin-left: 91.66667%;

}

}

/\* PC medium resolution > \*/

@media (min-width: 1113px) {

.row {

margin-left: -12px;

margin-right: -12px;

}

.row.sm-gutter {

margin-left: -5px;

margin-right: -5px;

}

.col {

padding-left: 12px;

padding-right: 12px;

}

.row.sm-gutter .col {

padding-left: 5px;

padding-right: 5px;

}

.l-0 {

display: none;

}

.l-1,

.l-2,

.l-2-4,

.l-3,

.l-4,

.l-5,

.l-6,

.l-7,

.l-8,

.l-9,

.l-10,

.l-11,

.l-12 {

display: block;

}

.l-1 {

flex: 0 0 8.33333%;

max-width: 8.33333%;

}

.l-2 {

flex: 0 0 16.66667%;

max-width: 16.66667%;

}

.l-2-4 {

flex: 0 0 20%;

max-width: 20%;

}

.l-3 {

flex: 0 0 25%;

max-width: 25%;

}

.l-4 {

flex: 0 0 33.33333%;

max-width: 33.33333%;

}

.l-5 {

flex: 0 0 41.66667%;

max-width: 41.66667%;

}

.l-6 {

flex: 0 0 50%;

max-width: 50%;

}

.l-7 {

flex: 0 0 58.33333%;

max-width: 58.33333%;

}

.l-8 {

flex: 0 0 66.66667%;

max-width: 66.66667%;

}

.l-9 {

flex: 0 0 75%;

max-width: 75%;

}

.l-10 {

flex: 0 0 83.33333%;

max-width: 83.33333%;

}

.l-11 {

flex: 0 0 91.66667%;

max-width: 91.66667%;

}

.l-12 {

flex: 0 0 100%;

max-width: 100%;

}

.l-o-1 {

margin-left: 8.33333%;

}

.l-o-2 {

margin-left: 16.66667%;

}

.l-o-3 {

margin-left: 25%;

}

.l-o-4 {

margin-left: 33.33333%;

}

.l-o-5 {

margin-left: 41.66667%;

}

.l-o-6 {

margin-left: 50%;

}

.l-o-7 {

margin-left: 58.33333%;

}

.l-o-8 {

margin-left: 66.66667%;

}

.l-o-9 {

margin-left: 75%;

}

.l-o-10 {

margin-left: 83.33333%;

}

.l-o-11 {

margin-left: 91.66667%;

}

}

/\* Tablet - PC low resolution \*/

@media (min-width: 740px) and (max-width: 1023px) {

.wide {

width: 644px;

}

}

/\* > PC low resolution \*/

@media (min-width: 1024px) and (max-width: 1239px) {

.wide {

width: 984px;

}

.wide .row {

margin-left: -12px;

margin-right: -12px;

}

.wide .row.sm-gutter {

margin-left: -5px;

margin-right: -5px;

}

.wide .col {

padding-left: 12px;

padding-right: 12px;

}

.wide .row.sm-gutter .col {

padding-left: 5px;

padding-right: 5px;

}

.wide .l-0 {

display: none;

}

.wide .l-1,

.wide .l-2,

.wide .l-2-4,

.wide .l-3,

.wide .l-4,

.wide .l-5,

.wide .l-6,

.wide .l-7,

.wide .l-8,

.wide .l-9,

.wide .l-10,

.wide .l-11,

.wide .l-12 {

display: block;

}

.wide .l-1 {

flex: 0 0 8.33333%;

max-width: 8.33333%;

}

.wide .l-2 {

flex: 0 0 16.66667%;

max-width: 16.66667%;

}

.wide .l-2-4 {

flex: 0 0 20%;

max-width: 20%;

}

.wide .l-3 {

flex: 0 0 25%;

max-width: 25%;

}

.wide .l-4 {

flex: 0 0 33.33333%;

max-width: 33.33333%;

}

.wide .l-5 {

flex: 0 0 41.66667%;

max-width: 41.66667%;

}

.wide .l-6 {

flex: 0 0 50%;

max-width: 50%;

}

.wide .l-7 {

flex: 0 0 58.33333%;

max-width: 58.33333%;

}

.wide .l-8 {

flex: 0 0 66.66667%;

max-width: 66.66667%;

}

.wide .l-9 {

flex: 0 0 75%;

max-width: 75%;

}

.wide .l-10 {

flex: 0 0 83.33333%;

max-width: 83.33333%;

}

.wide .l-11 {

flex: 0 0 91.66667%;

max-width: 91.66667%;

}

.wide .l-12 {

flex: 0 0 100%;

max-width: 100%;

}

.wide .l-o-1 {

margin-left: 8.33333%;

}

.wide .l-o-2 {

margin-left: 16.66667%;

}

.wide .l-o-3 {

margin-left: 25%;

}

.wide .l-o-4 {

margin-left: 33.33333%;

}

.wide .l-o-5 {

margin-left: 41.66667%;

}

.wide .l-o-6 {

margin-left: 50%;

}

.wide .l-o-7 {

margin-left: 58.33333%;

}

.wide .l-o-8 {

margin-left: 66.66667%;

}

.wide .l-o-9 {

margin-left: 75%;

}

.wide .l-o-10 {

margin-left: 83.33333%;

}

.wide .l-o-11 {

margin-left: 91.66667%;

}

}