# HTML

1. Làm quen với HTML

* Cấu trúc 1 file html

<!DOCTYPE html>

<html>

<head>

<title> Tiêu đề tab trình duyệt </title>

<meta charset=”utf-8”>

</head>

<body>

</body>

</html>

## <meta charset=”utf-8”>: hỗ trợ hiển thị tiếng việt trên website

* Attribute trong HTML
* Là các thuộc tính cho các thẻ
* Vị trí bên trong thẻ mở

### <h1 onclick=””>Attribute?</h1>

1. Làm quen với CSS

* Cách link CSS:

Có 3 cách link CSS

## Internal

### <style>

### H1 {

### Color: red;

### }

### </style>

## External (cách sử dụng)

### <head>

### <link rel=stylesheet” href=”style.css”>

### </head>

## Inline (cách mạnh nhất)

### <div style=”color:red;”>

### CSS in HTML?

### </div>

* Css selectors

Là chọn ra các đối tượng để css

.class-a: lựa chọn toàn bộ class có tên là class-a

#id-a: lựa chọn t oàn bộ id có tên id là id-a

<div id="main">

        <div class="content">

            <p>Đoạn văn 1</p>

            <p>Đoạn văn 2</p>

            <p>Đoạn văn 3</p>

            <div class="description">

                <p>Đoạn văn 4</p>

                <p>Đoạn văn 5</p>

            </div>

        </div>

        <p>Đoạn văn 6</p>

        <p>Đoạn văn 7</p>

    </div>

Dấu cách: lựa chọn toàn bộ đối tượng con trong đối tượng cha

Dấu lớn hơn: lựa chọn đối tượng con trực tiếp

Dấu cộng: lựa chọn đối tượng duy nhất ngay sau

Dẫu ngã: lựa chọn toàn bộ đối tượng ngay sau

## Chọn toàn bộ thẻ p:

### #main p

## Chọn 123

### #main .content > p

## Chọn 45

### #main .content .description

## Chọn 6

### #main .content+p

## Chọn 67

### #main .content~p

## Khi để .content.description thì nó sẽ chọn đối tượng nào vừa có class là content vừa có class là description

* Độ ưu tiên trong CSS

Inline: 1000 điểm

#id: 100 điểm

.class: 10 điểm

tag: 1 điểm

### h1#heading-id

## h1 chính là tag ứng với 1 điểm -> lựa chọn toàn bộ h1 có id là heading-id.

universal: 0 điểm

## biến toàn cục

### \* {

### Color: red;

### }

Inherited: 0 điểm

## Kế thừa

### Html{

### Color: blue;

### }

* Đặt biến trong CSS

Bao gồm biến toàn cục và biến cục bộ

### :root {

### --text-color: red;

### }

### h1{

### Color: var(--text-color);

### }

### h2{

### --my-color: blue;

### Color: var(--my-color);

### }

## :root là lớp giả pseudo elements

## Biến toàn cục sẽ để trong :root

## Biến cục bộ sẽ để trực tiếp bên trong các thẻ được CSS và chỉ tác động bên trong thẻ

* Đơn vị trong CSS

Bao gồm đơn vị tuyệt đối – Absolute units, đơn vị tương đối – Relative units

Đơn vị tuyệt đối hay dùng: px

## px: điểm ảnh – độ phân giải càng cao thì 1 px chứa càng nhiều điểm ảnh

Đơn vị tương đối hay dùng: %, rem, em , vw, vh

## %: tỷ lệ % theo chiều dài/rộng

## Rem: phụ thuộc vào đối tượng đã được gán thuộc tính

### Html{

### Font-size: 100%;

### }

### h1 {

### Font-size: 1rem;

### }

## Font-size: 100% mặc định sẽ tương đương với 16px

## Khi thay đổi font-size trong html thì toàn bộ các thẻ có kích thước đơn vị rem sẽ tự động thay đổi theo -> responsive moblie/ ipad ...

## Em: phụ thuộc vào thẻ được định nghĩa gần nhất

Sử dụng giống như rem nhưng nó sẽ phụ thuộc vào thẻ được gán thuộc tính gần nhất

## Vw: viewport width

### 50vw

Chiếm 50% màn hình, khác với % vì % phụ thuộc vào kích thước thẻ chứa nó

## Vh: viewport height

* Pseudo classes trong CSS – Lớp giả

Một số lớp giả hay sử dụng:

:root

:hover

:active

:first-child

:last-child

::before

::after

## :root – tham chiếu đến phần tử gốc, chính là cặp thẻ đóng mở <html></html>

:hover – khi di chuột qua đối tượng sẽ kích hoạt css

### h1:hover{

### Color: red;

### }

:active – khi bấm và giữ chuột thì sẽ kích hoạt css

:first-child và :last-child – lựa chọn đối tượng đầu tiên và cuối cùng

### <ul>

### <li>đoạn 1</li>

### <li>đoạn 2</li>

### <li>đoạn 3</li>

### <li>đoạn 4</li>

### </ul>

### Li:first-child{

### Color:blue;

### }

### Li:last-child{

### Color: blue;

### }

## ::before và ::after được sử dụng khi muốn tạo thông tin trong thẻ tại vị trí đầu tiên/ cuối cùng, **bắt buộc phải có thuộc tính content**, có thể để thuộc tính content:”” .

Trong Css:

### Div::before {

### Content:”before”;

### }

### Div::after{

### Content:”after”;

### }

Trong html:

### <div>

before

after

### </div>

* Background Css

### Background: url() no-repeat center / contain;

Viết tắt của: background-image, background-repeat, background-position, background-size

## Background size có 2 loại cover và contain

* Contain: giữ nguyên không cắt ảnh, lấy kích thước ảnh theo chiều ngắn hơn.
* Cover: có thể cắt một phần ảnh sao cho có thể che phủ toàn bộ block chứa nó
* Thuộc tính vị trí – CSS position

Gồm 4 loại:

* Relative
* Absolute
* Fixed
* Sticky

## Cách căn chỉnh vị trí sử dụng: top, left, right, bottom + px

### .title {

### Position:relative;

### Top: 20px;

### }

## Absolute: thẻ được gán position bằng absolute sẽ căn chỉnh vị trí so với **thẻ** **cha** gần nhất với nó được gán thuộc tính position

## Fixed: Cố định vị trí thẻ

### .title {

### Position:fixed;

### Top: 0;

### }

## Sticky: bám dính (ít sử dụng vì 1 số trình duyệt chưa hỗ trợ)

* Một số tricks and tips

## Căn giữa CSS

1. Sử dụng position

### .box{

### background-color: red;

### height: 100px;

### position: relative;

### }

### h3{

### position: absolute;

### top: 50%;

### transform: translate(-50%,-50%);

### }

1. Sử dụng line-height

## Sử dụng chiều cao đối tượng bằng đúng chiều cao thẻ chứa nó

### .box{

### background-color: red;

### height: 100px;

### text-align: center;

### line-height: 100px;

### }

1. Sử dụng flex

### .box{

### background-color: red;

### height: 100px;

### display: flex;

### align-items:center;

### justify-content: center;

### }

## Thường sử dụng khi phải căn giữa nhiều đối tượng

## Hiển thị hình ảnh khi bị lỗi

### <img onerror=”this.src=’/img/200x200.png’” src = “link ảnh”>

Tải ảnh thông báo lỗi về src code -> khi đường link ảnh bị lỗi thì sẽ chạy onerror

## Chỉnh chiều cao bằng 50% chiều rộng của chính nó

### padding-top: 50%;

## Khoảng cách giữa các chữ

### Letter-spacing: 4px;

## Chỉnh độ cao một dòng (khoảng 1.4 đối với tiếng anh, 1.6 đối với tiếng việt)

### Line-height: 1.4;

## Căn chữ thẳng hàng nhau hai bên

### Text-aligh: justify;

## Thẻ inline khi thêm float vào nó sẽ có thuộc tính là block

## Margin âm chỉnh có tác dụng khi sử dụng float

### Margin-top : -3px;

## Cách để background color là hình tròn – sử dụng border

### Border-radius: 50%

## Mức độ ưu tiên cao !important, nó sẽ ghi đè lên TẤT CẢ các styte đã được css trước đó.

## Sử dụng khi css 1 thuộc tính duy nhất.

### .text-white {

### Color: #fff;

### }

## Khi đặt thẻ a href=”#” thì khi click vào nó sẽ ở nguyên trang đó

## Khi sử dụng float mà khích thước thẻ con to hơn thẻ cha thì sẽ bị tràn ra ngoài thẻ cha.

## Để sử lý vấn đề đó sử dụng overflow:auto

### .clear-fix {

### Overflow:auto

### }

Hoặc

### .clear-fix::after {

### Content:””;

### Clear: both;

### Display:table;

### }

## Thẻ img mặc định có tính chất là inline

## Kỹ thuật margin âm để loại bỏ khoảng thừa 2 đầu ngoài. Thẻ con bên trong sẽ được tăng kích thước để bù cho khoảng âm đó. Nhờ vậy loại bỏ khoảng trắng thừa.

### .place-list {

### Margin-left:-8px;

### Margin-right:-8px;

### }

## Giao diện nổi lên khi click vào gọi là giao diện Modal, thường dùng thuộc tính position là fixed. Khi mở lên thì phần background phía sau bị mờ đi – gọi là lớp phủ overlay.

## Khi hover vào chuyển mũi tên thành bàn tay con trỏ

### Cursor: pointer;

## Khi sử dụng thẻ Label và Input. Khi click vào text ở trong phần Label cũng trỏ ra ô input. Lưu ý label chỉ sử dụng với **ID** đặt trong thẻ input

### <label for=”**quantity**” class=”modal-label”>

### Tickets, $15 per person

### </lable>

### <input id=”**quantity**” type=”text” class=”modal-input” placeholder=”How many?”>

## Extension tự động sửa trên tag khi phần mở tag thay đổi tên

### Auto Rename Tag

## Làm hoạt ảnh chuyển động

### @keyframes **modalFadeIn** {

### From {

### Opacity:0

### Transform: translateY(-100px);

### }

### To {

### Opacity: 1

### Transform: translateY(0);

### }

### }

Copy phần tên keyframe lên container cần áp dụng

CSS:

### .modal-container {

### Animation: **modalFadeIn** Ease 0.5s

### }

Ease: timing function (nhanh -> chậm -> nhanh)