TRƯỜNG ĐẠI HỌC CẦN THƠ
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

BÀI GIẢNG
CT188 - NHẬP MÔN LẬP TRÌNH WEB

KHOA TRUYỀN THÔNG ĐA PHƯƠNG TIỆN

TRƯỜNG ĐẠI HỌC CẦN THƠ
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

CHƯƠNG 3
Cascading Style Sheets - CSS

Trình bày: TS. Phạm Trương Hồng Ngân (pthngan@ctu.edu.vn)

CTU ĐẠI HỌC CẦN THƠ Cộng đồng - Toàn diện - Ưu việt www.ctu.edu.vn

Nội dung

8. Bố cục trang web với CSS
   1. Giới thiệu về bố cục
   2. Bố cục 1 chiều (Flexbox)
   3. Bố cục 2 chiều (Grids)
   4. Thiết kế đáp ứng (Responsive design)
   5. Thư viện CSS

GIỚI THIỆU VỀ BỐ CỤC

Giới thiệu

- Các kỹ thuật CSS bố cục trang web cho phép chọn và kiểm soát các phần tử
- Các yếu tố liên quan đến vị trí một phần tử:
  - Vị trí mặc định trong bố cục bình thường
  - Các phần tử khác xung quanh
  - Phần tử cha (chứa phần tử hiện tại)
  - Thiết lập khung nhìn chính (viewport)

Giới thiệu (2)

- Các kỹ thuật bố cục trang web:
  - Bố cục mặc định (Normal Flow)
  - Bố cục 1 chiều (Flexbox)
  - Bố cục 2 chiều (Grids)
  - Thuộc tính float (tự học)
  - Chỉ định vị trí (Thuộc tính position –đã học)
  - Bố cục nhiều cột (Multiple-column layout –tự học)

BỐ CỤC 1 CHIỀU (FLEXBOX)

FLEXBOX

- Phương pháp hỗ trợ thiết kế cấu trúc bố cục đáp ứng linh hoạt mà không cần sử dụng float hoặc position
- Là phương pháp bố cục một chiều để sắp xếp các mục theo hàng hoặc cột
- Các phần tử con (flex) sẽ mở rộng để lấp đầy khoảng trống còn lại hoặc thu nhỏ để vừa với không gian nhỏ

FLEXBOX (3)

- Sử dụng flexbox
  - Phần tử cha (container)
    - Phải có thuộc tính display với giá trị là flex
    - Các thuộc tính khác: flex-direction, flex-wrap, flex-flow, justify-content, align-items, align-content
  - Phần tử con (flex item)
    - Các thuộc tính: align-self, flex, flex-basis, flex-grow, flex-shrink, order

FLEXBOX (2)

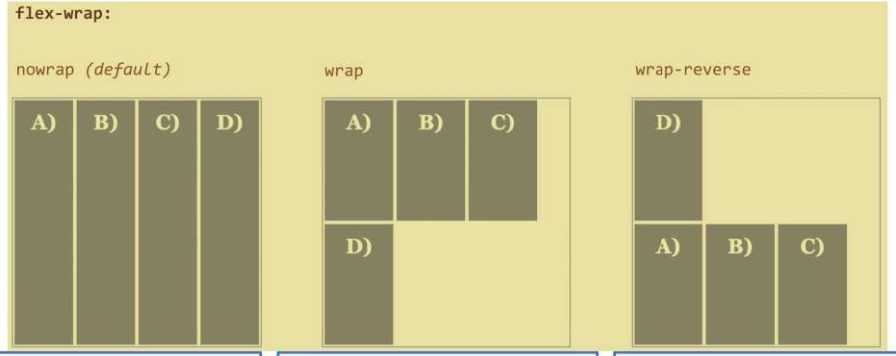
```
<html>
   <head>
      <style>
      .container { display: flex; }
      </style>
   </head>
   <body>
      <div class="container">
         <div>A)</div>
         <div>B)</div>
         <div>C)</div>
         <div>D)</div>
      </div>
   </body>
</html>
```


FLEXBOX (4)

- Các thuộc tính của phần tử cha

Thuộc Tính	Mô Tả
flex-wrap	Chỉ định các phần tử con có xuống dòng hay không khi không đủ không gian chứa các phần tử con
align-content	Canh lề các dòng flex
align-items	Canh lề theo chiều dọc các phần tử con
flex-direction	Chỉ định hướng các phần tử con
justify-content	Canh đều các phần tử con theo chiều ngang
flex-flow	Thuộc tính rút gọn của 2 thuộc tính flex-direction và flex-wrap

FLEXBOX (10)




```
.container {
display: flex;
flex-wrap: nowrap;
}
```

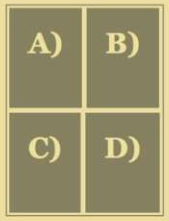
```
.container {
display: flex;
flex-wrap: wrap;
}
```

```
.container {
display: flex;
flex-wrap: wrap-reverse;
}
```

FLEXBOX (5)

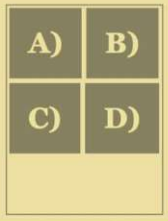
align-content:

stretch *(default)*



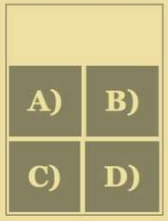

```
.container {
display: flex;
flex-wrap:wrap;
align-content: stretch;
}
```

flex-start




```
.container {
display: flex;
flex-wrap:wrap;
align-content: flex-start;
}
```

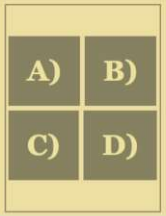
flex-end




```
.container {
display: flex;
flex-wrap:wrap;
align-content: flex-end;
}
```

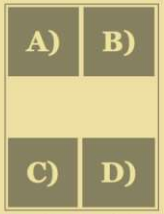

FLEXBOX (6)

center



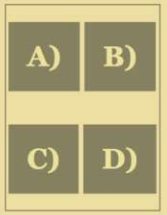

```
.container {
display: flex;
align-content:
center;
}
```

space-between




```
.container {
display: flex;
align-content:
space-between;
}
```

space-around

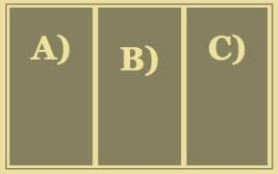



```
.container {
display: flex;
align-content:
space-around;
}
```


FLEXBOX (7)

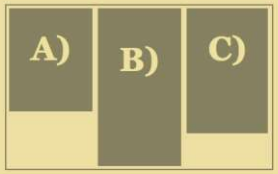
align-items:

stretch *(default)*



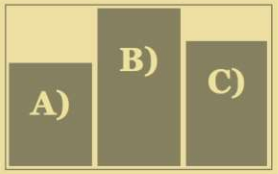

```
.container {
display: flex;
align-items:
stretch;
}
```

flex-start




```
.container {
display: flex;
align-items:
flex-start;
}
```

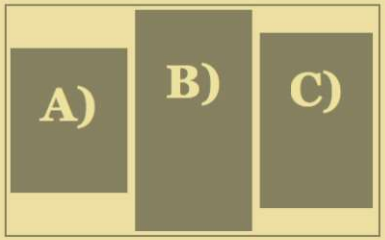
flex-end




```
.container {
display: flex;
align-items:
flex-end;
}
```

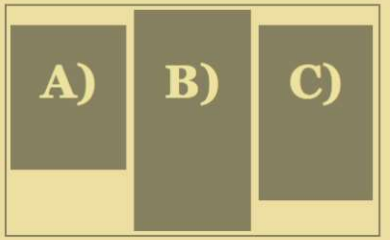

FLEXBOX (8)

center



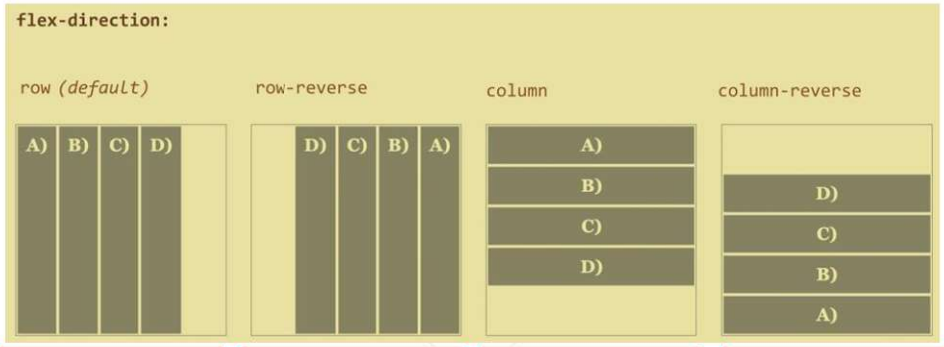

```
.container {
display: flex;
align-items: center;
}
```

baseline




```
.container {
display: flex;
align-items: baseline;
}
```

FLEXBOX (9)



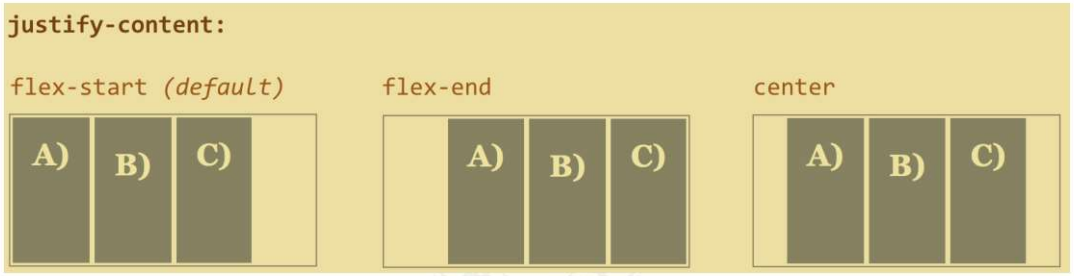

```
.container {
display: flex;
flex-direction:
row;
}
```

```
.container {
display: flex;
flex-direction:
row-reverse;
}
```

```
.container {
display: flex;
flex-direction:
column;
}
```

```
.container {
display: flex;
flex-direction:
column-reverse;
}
```


FLEXBOX (11)

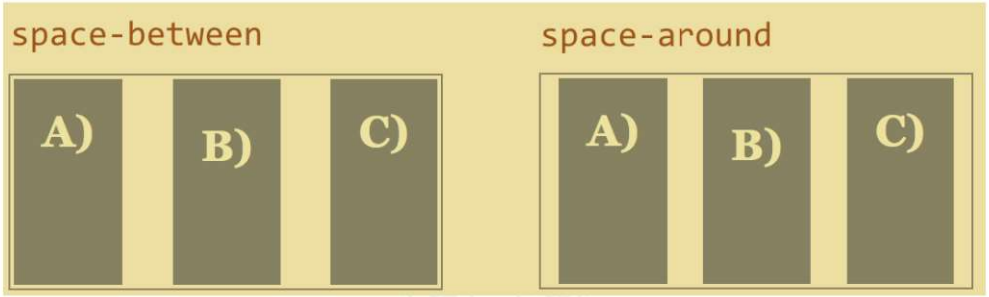



```
.container {
display: flex;
justify-content:
flex-start;
}
```

```
.container {
display: flex;
justify-content:
flex-end;
}
```

```
.container {
display: flex;
justify-content:
center;
}
```


FLEXBOX (12)




```
.container {
display: flex;
justify-content:
space-between;
}
```

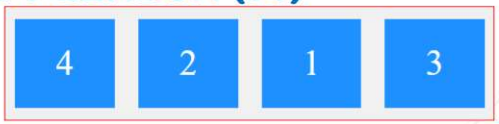
```
.container {
display: flex;
justify-content:
space-around;
}
```


FLEXBOX (13)

❖Các thuộc tính của phần tử con

Thuộc Tính	Mô Tả
order	Xác định thứ tự của các phần tử con
flex-grow	Xác định phần tử con sẽ tăng bao nhiêu so với các phần tử con khác
flex-shrink	Xác định phần tử con sẽ co lại so với các phần tử còn lại
flex-basis	Xác định độ dài ban đầu của phần tử con
flex	Thuộc tính rút gọn của 3 thuộc tính flex-grow, flex-shrink và flex-basis
align-self	Canh lề cho phần tử con (ghi đè thuộc tính align-items của phần tử cha)

FLEXBOX (14)




```
<div class="container">
  <div style="order: 3">1</div>
  <div style="order: 2">2</div>
  <div style="order: 4">3</div>
  <div style="order: 1">4</div>
</div>
```


```
<div class="container">
  <div style="flex-grow:1">1</div>
  <div style="flex-grow:1">2</div>
  <div style="flex-grow:8">3</div>
</div>
```

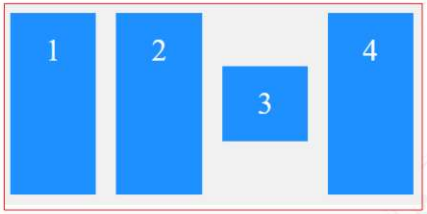

```
<div class="container">
  <div>1</div>
  <div>2</div>
  <div style="flex-shrink:0">3</div>
  <div>4</div>
  <div>5</div>
</div>
```


FLEXBOX (15)

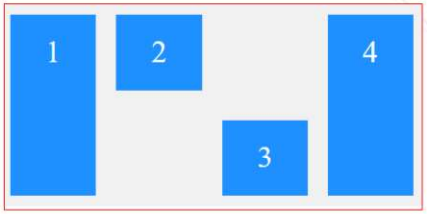

```
<div class="container">
  <div>1</div>
  <div>2</div>
  <div style="flex-basis:200px">3</div>
  <div>4</div>
</div>
```


```
<div class="container">
  <div>1</div>
  <div>2</div>
  <div style="flex: 0 0 200px">3</div>
  <div>4</div>
</div>
```


FLEXBOX (16)




```
<div class="container">
  <div>1</div>
  <div>2</div>
  <div style="align-self: center">3</div>
  <div>4</div>
</div>
```




```
<div class="container">
  <div>1</div>
  <div style="align-self: flex-start">2</div>
  <div style="align-self: flex-end">3</div>
  <div>4</div>
</div>
```


BỐ CỤC 2 CHIỀU (GRID)

BỐ CỤC 2 CHIỀU (GRID)

- Cung cấp một hệ thống bố cục dựa trên lưới (với các hàng và cột)
- Hỗ trợ thiết kế bố cục trang web không sử dụng float và position
- Sử dụng grid:
  - Phần tử cha (container)
    - Phải có thuộc tính display với giá trị là grid hoặc inline-grid
    - Các thuộc tính khác: gap, grid-template-areas,...
  - Phần tử con (item)
    - Các thuộc tính: grid-area, grid-column,

GRID (2)

- Tạo bố cục với grid-template-areas

- Phần tử con:
+ grid-area: Xác định Tên vùng hiển thị

```
.item-top{ grid-area:top;}
.item-left{grid-area:left;}
.item-center{grid-area:center;}
.item-right{grid-area:right;}
.item-bottom{grid-area:bottom;}
```

- Phần tử cha:
+ display: grid
+ grid-template-areas:
  ~ Xác định các vùng hiển thị
  ~ Mỗi dòng để trong cặp dấu “”
  ~ Mỗi cột trong dòng cách nhau một khoảng trắng

```
.container {
display: grid;
grid-template-areas:
"top top top"
"left center center"
"bottom bottom bottom";
}
```

Các phần tử con sẽ hiển thị thành 3 dòng
- Dòng 1: Hiển thị các phần tử có grid area là **top** và chiếm **3** cột
- Dòng 2: Hiển thị các phần tử có grid area là **left (1 cột)** và **center (2 cột)**
- Dòng 3: Hiển thị các phần tử có grid area là **bottom** và chiếm **3 cột**

GRID (3)

```
<head>
<style>
.item-top{ grid-area:top;
    background-color:red; }
.item-left{ grid-area:left;
    background-color:green; }
.item-center{ grid-area:center;
    background-color:yellow; }
.item-right{ grid-area:right;
    background-color:blue; }
.item-bottom{ grid-area:bottom;
    background-color:pink; }
.container { display: grid;
    grid-template-areas:
    "top top top"
    "left center center"
    "bottom bottom bottom"; }
</style>
</head>
```

```
<body>
  <div class="container">
    <div class="item-top">
      TOP</div>
    <div class="item-left">
      LEFT</div>
    <div class="item-center">
      CENTER</div>
    <div class="item-bottom">
      BOTTOM</div>
  </div>
</body>
```

TOP
LEFT CENTER
BOTTOM

GRID (3)

LEFT TOP
CENTER
BOTTOM

```
.container {
   display: grid;
   grid-template-areas:
   "left top top"
   "center center center"
   "bottom bottom bottom";
}
```

```
<body>
  <div class="container">
    <div class="item-top">
      TOP</div>
    <div class="item-left">
      LEFT</div>
    <div class="item-center">
      CENTER</div>
    <div class="item-bottom">
      BOTTOM</div>
  </div>
</body>
```

GRID (3)


```
.container {
  display: grid;
  grid-template-areas:
  "top top top top"
  "left center center right"
  "bottom bottom bottom bottom";
}
```

```
<body>
  <div class="container">
    <div class="item-top"> TOP</div>
    <div class="item-left">LEFT</div>
    <div class="item-center">CENTER</div>
    <div class="item-right">RIGHT</div>
    <div class="item-bottom">BOTTOM</div>
  </div>
</body>
```


GRID (3)


```
.container {
  display: grid;
  grid-template-areas:
   "top top top"
   ". center center"
   "bottom bottom bottom";
}
```

```
<body>
  <div class="container">
    <div class=" item-top"> TOP</div>
    <div class=" item-center ">CENTER</div>
    <div class="item-bottom">BOTTOM</div>
  </div>
</body>
```


Thiết kế đáp ứng (Responsive design)

Media Queries

- Media queries là một CSS rule hỗ trợ cho việc thiết kế giao diện thích ứng (Responsive design)
- Giúp cho việc hiển thị nội dung phù hợp với các thiết bị có kích thước màn hình khác nhau, ví dụ màn hình máy tính, màn hình điện thoại hay máy in
- Khi điều kiện thỏa mãn, các declarations sẽ áp dụng
- Cú pháp của một media query:

```
@media not|only mediatype and (mediafeature and|or|not mediafeature)
{declarations}
```

- Các truy vấn có thể được kết hợp theo nhiều cách khác nhau bằng cách sử dụng các toán tử logic

Xem thêm các features tại:
https://developer.mozilla.org/en-US/docs/Web/CSS/Media_Queries/Using_media_queries#media_features

Media Queries - Media Types

Giá trị	Mô tả
all	Mặc định. Được sử dụng cho tất cả các loại thiết bị phương tiện
print	Dùng cho máy in
screen	Được sử dụng cho màn hình máy tính, máy tính bảng, điện thoại thông minh, v.v.
speech	Được sử dụng cho trình đọc màn hình

Media Queries - Media Features

Giá trị	Mô tả
height	Chiều cao khung nhìn-viewport
width	Chiều rộng khung nhìn-viewport
max-height	chiều cao tối đa của vùng hiển thị
max-width	Chiều rộng tối đa của vùng hiển thị
min-height	Chiều cao tối thiểu của vùng hiển thị
min-width	Chiều rộng tối thiểu của vùng hiển thị

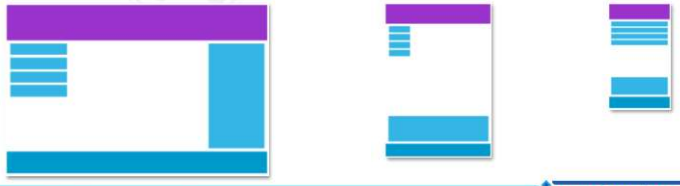
Media Queries

```
@media print {
    body {
        color: black;
    }
}
```

```
@media screen, print {/*...*/}
/*Thiết bị có trạng thái hover*/
@media (hover: hover) {/*...*/}
/*Thiết bị có chiều rộng tối đa là 12450px*/
@media (max-width: 12450px) {/* ... */}
/*Thiết bị có chiều rộng tối thiểu là 30em và màn hình thiết bị ở chế ngang*/
@media (min-width: 30em) and (orientation: landscape) {/*...*/}
/*Chỉ áp dụng cho màn hình thiết bị có chiều rộng tối đa là 600px*/
@media only screen and (max-width: 600px) {
    body {
        background-color: lightblue;
    }
}
/*Thiết bị có chiều cao tối thiểu là 680px hoặc là màn hình thiết bị ở chế độ
ngang*/
@media (min-height: 680px), screen and (orientation: landscape) {/*...*/}
```


Media Queries-Design for Mobile First

- Ưu tiên thiết kế giao diện cho thiết bị di động trước khi thiết kế cho các thiết bị khác
- Giúp trang web hiển thị nhanh hơn trên thiết bị có màn hình kích thước nhỏ
  - Ví dụ: Thay vì thay đổi kiểu khi chiều rộng nhỏ hơn 768px, nên thay đổi thiết kế khi chiều rộng lớn hơn 768px
- Các điều kiện tương ứng với các thiết bị gọi là Điểm Ngắt – Break Point



Media Queries-Design for Mobile First

```
/* For mobile phones */
[class*="col-"] {
  width: 100%;
}
@media only screen and (min-width: 768px) {
  /* For desktop */
  .col-1 {width: 8.33%;}
  .col-2 {width: 16.66%;}
  .col-3 {width: 25%;}
  .col-4 {width: 33.33%;}
  .col-5 {width: 41.66%;}
  .col-6 {width: 50%;}
  .col-7 {width: 58.33%;}
  .col-8 {width: 66.66%;}
  .col-9 {width: 75%;}
  .col-10 {width: 83.33%;}
  .col-11 {width: 91.66%;}
  .col-12 {width: 100%;}
}
```


Media Queries-Design for Mobile First

- Khó tạo Điểm Ngắt chính xác cho từng thiết bị
- Để đơn giản, có thể tạo điểm ngắt cho năm nhóm

```
/* Màn hình rất nhỏ (phones, độ rộng <= 600px) */
@media only screen and (max-width: 600px) {/*...*/}
/* Màn hình nhỏ (portrait tablets và large phones, độ rộng >= 600px)*/
@media only screen and (min-width : 600px) {/*...*/}
/*Màn hình trung bình (landscape tablets, độ rộng >= 768px)*/
@media only screen and (min-width : 768px) {/*...*/}
/* Màn hình lớn (laptops/desktops, độ rộng >= 992px) */
@media only screen and (min-width : 992px) {/*...*/}
/* Màn hình rất lớn (large laptops and desktops, độ rộng >= 1200px)*/
@media only screen and (min-width : 1200px) {/*...*/}
```


Ví dụ

Ví dụ (2)

```
.container { display: grid; }
/* Màn hình rất nhỏ (phones, độ rộng từ 600px trở xuống)*/
@media only screen and (max-width: 600px) {
      .container {
            grid-template-areas:
            "top "
            "left"
            "center"
            "right"
            "bottom"; }
      .item-right{display:block;}
}
```

Ví dụ (3)

```
/* Màn hình nhỏ (portrait tablets và large phones,
độ rộng từ 600px trở lên) */
@media only screen and (min-width : 600px) {
   .container {
      grid-template-areas:
      "top top top"
      "left center center"
      "bottom bottom bottom";
   }
   .item-right{display:none;}
}
```


Ví dụ (4)

```
/*Màn hình trung bình (landscape tablets, độ rộng từ
768px trở lên)*/
@media only screen and (min-width : 768px) {
   .container {
      grid-template-areas:
      "top top top"
      "left center right"
      "bottom bottom bottom";
   }
   .item-right{display:block;}
}
```


Thư Viện CSS

Thư viện CSS

- Khái niệm thư viện CSS
- Khai báo thư viện CSS
- Nguyên tắc sử dụng thư viện CSS
- Một số thư viện CSS thông dụng

Khái niệm thư viện CSS

- Thư viện css là tập hợp các đoạn mã css
- Sử dụng một thư viện css gồm 2 bước:
  - (1) Khai báo thư viện css
  - (2) Sử dụng thư viện theo cú pháp riêng đã được khai báo trong thư viện
- Có rất nhiều thư viện CSS hỗ trợ thiết kế giao diện website phù hợp với từng tiêu chí của người dùng
- Mỗi thư viện sẽ có những đặc trưng riêng biệt về định dạng bố cục giao diện, ví dụ như: định dạng về hiển thị văn bản, hình ảnh, cách bố trí các thành phần web

Khai báo thư viện CSS

- Các thư viện css được đóng gói thành các file css
- Sử dụng thẻ **<link>** để khai báo:

```
<head>
    <title>Sử dụng thư viện fontawesome</title>
    <link rel="stylesheet" href="css/fontawesome.css">
    <link rel="stylesheet" href="css/my_custom.css">
</head>
```

- Lưu ý : File thư viện css nếu có phải được khai báo trước file *my_custom.css*.
- Trong ví dụ trên là cách khai báo thư viện ***fontawesome***, file CSS của thư viện font awesome được khai báo trước file my_custom

Nguyên tắc sử dụng thư viện CSS

- Không chỉnh sửa file thư viện nếu thật sự không cần thiết, nên dùng các phương pháp khác thay thế
- Tuân thủ cú pháp sử dụng thư viện nếu có, nên đặt file thư viện trước
- Ví dụ: trong ví dụ sau có các files ***index.html***, ***index.css*** và ***lib.css*** :
  - File ***index.css***

```
btn_goto_shoping{
    color: yellow;
}
```


Nguyên tắc sử dụng thư viện CSS

- File thư viện ***lib.css***

```
.btn{
    padding: 7.5px 15px;
    border-radius: 2px;
    display: inline-block;
    color: #ffffff;
}
.btn_primary{
    background-color: #0066ff;
}
```

Nguyên tắc sử dụng thư viện CSS

- File *index.html*

```
<head>
    <link href="css/lib.css" rel="stylesheet" />
    <link href="css/index.css" rel="stylesheet" />
</head>
<body>
    <span class="btn btn_primary btn_goto_shoping">
    Go to my store</span>
</body>
```

- Lớp **btn** và **btn_primary** của thư viện ***lib.css*** dùng để định dạng thẻ **,** sử dụng thêm lớp **btn_goto_shoping** để tùy chỉnh thêm cho nút "Go to my store"

Nguyên tắc sử dụng thư viện CSS

- Thứ tự đặt các **class** cho thẻ **** phải theo thứ tự: **btn btn_primary btn_goto_shoping** class của thư viện đặt trước, các file người dùng thiết kế đặt sau
- Sử dụng nhiều thư viện trong một website
  - Một website sẽ gồm nhiều thư viện css để tiết kiệm công sức của lập trình viên
  - Khi sử dụng nhiều thư viện css cần tuân theo các nguyên tắc sau:
    - Hạn chế trùng lặp chức năng giữa các thư viện
    - Thư viện nào quan trọng hơn thì khai báo phía sau thư viện ít quan trọng

Nguyên tắc sử dụng thư viện CSS

- Cập nhật thư viện css
  - Các thư viện sẽ thường xuyên được cập nhật, với mục đích cập nhật công nghệ mới và sửa lỗi
  - Khi sử dụng phiên bản trực tiếp hoặc cập nhật thư viện về website, cần phải lưu ý:
    - Việc cập nhật là có thật sự cần thiết hay không?
    - Cập nhật thư viện sẽ mang đến thay đổi như thế nào cho toàn website?
    - Các thư viện khác, website đang sử dụng có tương thích với bản mới hay không?

Một số thư viện CSS thông dụng

- Thư viện Animation
- Thư viện Font Awesome tạo icon cho website
- Các thư viện hỗ trợ trình diễn hình ảnh

Thư viện Animation

- Thư viện Animation hỗ trợ tạo ra các hiệu ứng chuyển động
- Truy cập trực tiếp trang chủ của thư viện để tải file CSS và toàn bộ cho các hiệu ứng: daneden.github.io/animate.css
- Xem các hoạt ảnh chuyển động đã được demo và đọc hướng dẫn sử dụng tại: github.com/daneden/animate.css

Sử dụng thư viện CSS-Font Awesome

- Là thư viện chứa các font chữ ký hiệu hay sử dụng trong website
  - Font chữ ký hiệu là các icons thường sử dụng trong các layout website
- Hỗ trợ hầu hết các trình duyệt và hệ điều hành
- Có thể kết hợp với CSS3 để tạo các icons động
- Hướng dẫn sử dụng: https://fontawesome.com
- Đăng ký sử dụng online:

https://fontawesome.com/start

Sử dụng thư viện CSS-Font Awesome

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/li
bs/font-awesome/4.7.0/css/font-awesome.min.css">
</head>
<body>
<i class="fa fa-car"></i>
<i class="fa fa-car" style="font-size:48px;"></i>
<i class="fa fa-car" style="font-size:60px;color:red;"></i>
</body>
</html>
```

Các thư viện hỗ trợ trình diễn hình ảnh

- iHover (35 hiệu ứng)
- Imagehover (44 hiệu ứng)
- Hover Effect Ideas (30 hiệu ứng)
- Một số thư viện khác

iHover

- iHover là một bộ sưu tập các hiệu ứng chuyển động viết bằng CSS3, có 20 hiệu ứng vòng tròn và 15 hiệu ứng hover vuông, xem tại: gudh.github.io/ihover/dist/

Image Hover

- Thư viện này chứa 44 hiệu ứng làm với CSS
- Một số hiệu ứng như: fades, pushes, slides, hinges, reveals, zooms, blurs, flips, folds và shutters trong nhiều hướng, xem tại: http://imagehover.io/

Hover Effect Ideas

- Có khoảng 30 hiệu ứng trên hai bộ hướng dẫn và mã nguồn, cung cấp hiệu ứng di chuyển chuột
- Xem tại: tympanus.net/Development/HoverEffectIdeas/

Một số thư viện khác

- Tham khảo thêm một số thư viện hỗ trợ khác:
  - Hover CSS (khoảng 100 hiệu ứng)
  - Animatism (100 hiệu ứng)
  - Caption Hover Effect (7 hiệu ứng)
  - CSS Image Hover Effects (15 hiệu ứng)
  - Direction-aware 3D hover effect
  - Hover Animation
  - Tiles Animated Hover
  - SVG clip-path Hover Effect