

BÀI 3

ID VÀ CLASS

KHÁI NIỆM

- Dùng để phân biệt các thẻ giống nhau. (cùng 1 loại thẻ có định dạng khác nhau)
- Dùng để tạo ra style mới theo ý muốn

CÁCH DÙNG

- ID= "tenid" vd: div id="container">

(id không tuyệt đối không được dùng 2 lần)

- Class="tenclass" vd: <div class="khung">

Lưu ý: tên class không được bắt đầu bằng số (vd: 1cot, 2vien)

CÁCH DÙNG

- Với id: tên thẻ#tenid -> phím tab
- Với class: tên thẻ.tenclass -> phím tab

Ví dụ minh họa

Bước 1: code file html

Bước 2: code css

LÀM VIỆC VỚI CÁC THẺ INPUT

- <input type="loại input" name ="" value ="" css>

- Các loại input:

- Text

- bình thường

- password

- Button

- Thường

- submit

- Reset

- file

- Radio button

- Checkbox

CÁC THUỘC TÍNH THÔNG DỤNG

- **value** chỉ định một giá trị khởi tạo cho phần tử input

Value="giá trị khởi tạo"

- **readonly** chỉ định phần tử input chỉ có thể đọc (không thể thay đổi dữ liệu trên phần tử đó)

- **size** chỉ định kích thước của trường input (số ký tự)

- **height và width** chỉ định chiều rộng và cao của phần tử <input>

- **maxlength** chỉ định độ dài tối đa cho phép của trường input

- **autofocus** là một thuộc tính boolean.

- **placeholder** xác định một gợi ý mà miêu tả giá trị mong đợi cho trường input, sẽ được thay thế khi nhập mới

CÁC THUỘC TÍNH THÔNG DỤNG

- **required** là một thuộc tính kiểu boolean. Khi xuất hiện nó xác định rằng trường input không được để trống trước khi submit form

```
<form action="#">
```

```
    Username: <input type="text" name="username" required>
```

```
    <input type="submit">
```

```
</form>
```

2. SET FONT: Tích hợp font vào website

- Tích hợp font không cần cài cách làm:

1. chọn 1 font (down trên mạng) ->

cho vào cùng thư mục với html

2. vào css nhập nội dung

```
@font-face{
```

```
    font-family: myfont;//tên do tự đặt
```

```
    src: url(KingLionel-PersonalUse.ttf);
```

```
        //tên font tải về
```

```
}
```

2. SET FONT: Tích hợp font vào website

cách sử dụng:


Như các thuộc tính bình thường

Vd

```
h1{  
    font-family: long;  
}
```

**- Nếu muốn sử dụng nhiều font thì cách làm tương tự
(nhiều @font-face)**

3. ICON:

- Dùng hình ảnh (đã được học)
 - Dùng font icon: (awesome : <https://fontawesome.com/>)
 - * tải font (để cùng thư mục web)
 - * khai báo link css cho font icon (thường chọn all)
 - * sử dụng: vào trang fontawesome chọn icon cần dùng,
- > start user icon -> copy mã lệnh dán vào file html
- Ví dụ: `<i class="fas fa-users"></i>` 
- * Định dạng: thêm class và định dạng như kiểu text

4. CHIA LAYER:

4.1: Chia theo khối:

- thuộc tính **position**: *xác định vị trí cho thành phần*

giá trị	Ví dụ	Mô tả
static	position: static;	Thành phần sẽ nằm theo thứ tự trong văn bản, đây là dạng mặc định.
<u>relative</u>	position: relative;	Định vị trí tuyệt đối cho thành phần.
<u>absolute</u>	position: absolute;	Định vị trí tuyệt đối cho thành phần theo thành phần bao ngoài (thành phần định vị trí tương đối position: relative;) hoặc theo cửa sổ trình duyệt.
<u>fixed</u>	position: fixed;	Định vị trí tương đối cho thành phần theo cửa sổ trình duyệt.
inherit	position: inherit;	Xác định thừa hưởng thuộc tính từ thành phần cha (thành phần bao ngoài).

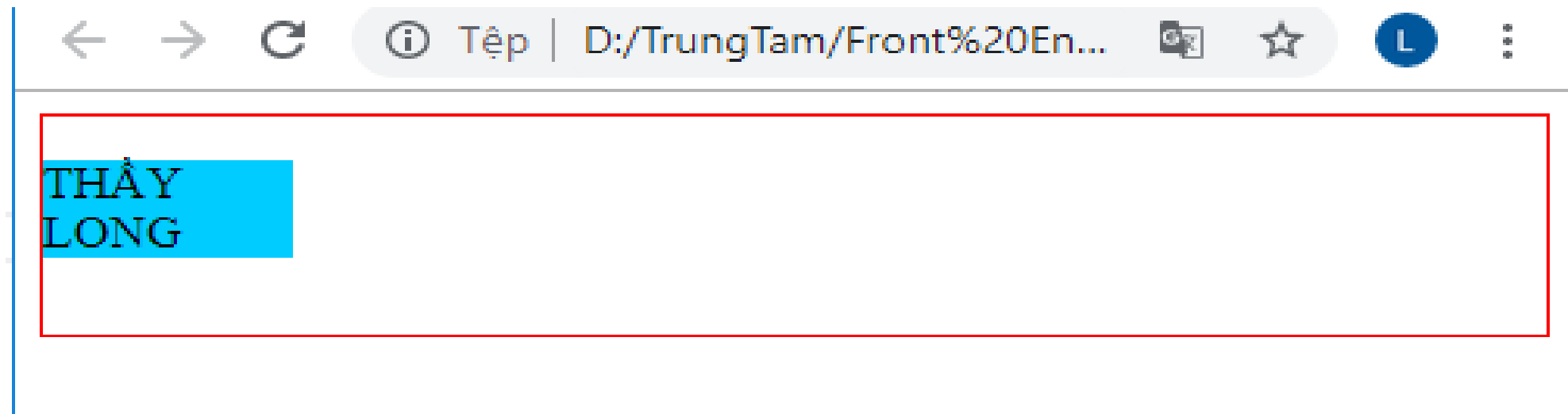
Ví dụ:

```
<html>
  <head></head>
  <body>
    <div>
      <p>THẦY LONG</p>
    </div>
  </body>
</html>
```

File css

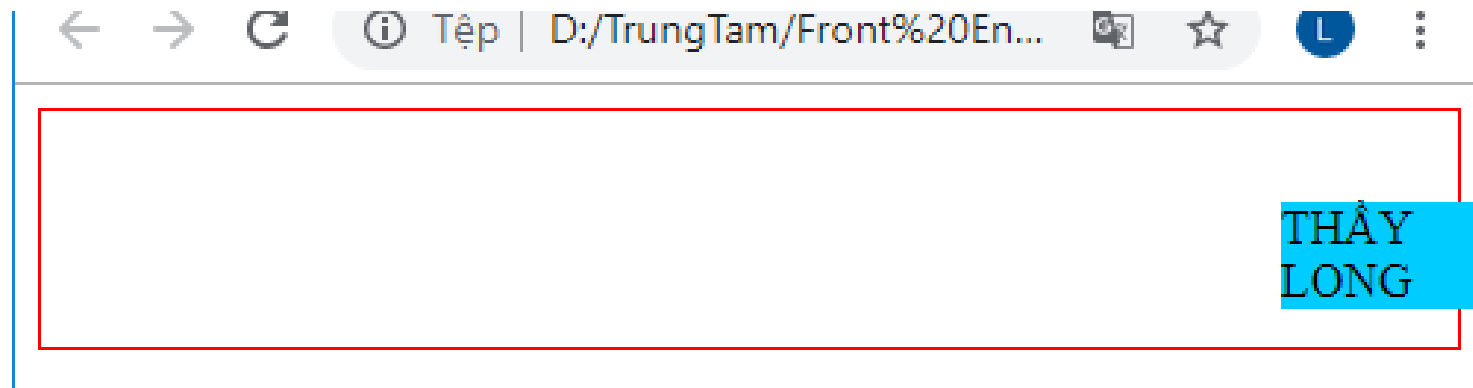
```
div {  border: 1px solid red;
        height: 80px;
      }
div p {
        background: #00CCFF;
        width: 80px;
      }
```

Kết quả hiển thị khi chưa có **position**:

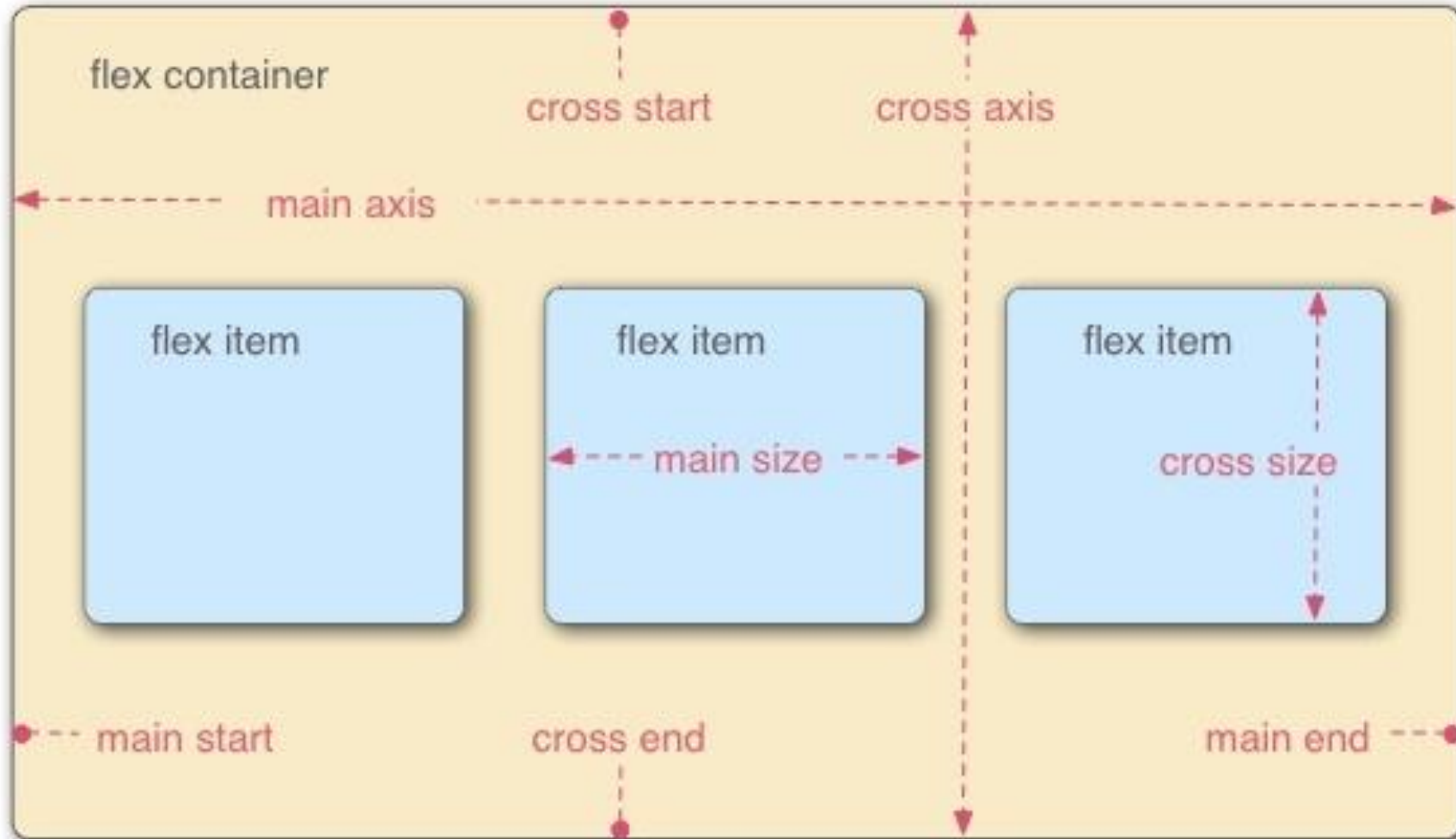


THÊM THUỘC TÍNH POSITION VÀO CSS

```
div {  
  position: relative  
}  
div p {  
  /*thêm vào */  
  position: absolute;  
  right: -20px;  
  top: 15px;  
}
```



- thuộc tính **Flexbox**: *dàn trang*, tự cân đối kích thước của các phần tử bên trong để hiển thị trên mọi thiết bị



Hai thành phần quan trọng nhất trong một bố cục Flexbox là gồm container và item:

- **container**: là thành phần lớn bao quanh các phần tử bên trong, thiết lập kiểu hiển thị inline (sắp xếp theo chiều ngang) hoặc kiểu sắp xếp theo chiều dọc. Khi đó, các item bên trong sẽ hiển thị dựa trên thiết lập của container này.
- **item**: Các phần tử con của container được gọi là item, ở item ta có thể thiết lập nó sẽ sử dụng bao nhiêu cột trong một container, hoặc thiết lập thứ tự hiển thị của nó.

Ngoài ra, ta còn có những thành phần sau:

- **main start, main end:** Khi thiết lập flexbox, điểm bắt đầu của container gọi là main start và điểm kết thúc được gọi là main end. Điều này có nghĩa, các item bên trong sẽ hiển thị từ main start đến main end (hoặc là được phép hiển thị đến main end). Và chiều vuông góc của nó là **cross start, cross end** cũng có ý nghĩa tương tự nhưng luôn vuông góc với main start, main end.
- **main axis:** Trục này là trục chính để điều khiển hướng mà các item sẽ hiển thị. Như bạn thấy ở trên hình main axis là trục dọc nên các item sẽ hiển thị theo chiều dọc, tuy nhiên ta có thể sử dụng thuộc tính flex-direction để thay đổi trục của main axis và lúc đó các item sẽ hiển thị theo nó. Và cross axis luôn là trục vuông góc của main axis.
- **main size:** Bạn có thể hiểu đơn giản là kích thước (chiều rộng hoặc dọc) của mỗi item dựa theo trục main axis.
- **cross size:** Là kích thước (chiều rộng hoặc dọc) của mỗi item dựa theo trục cross axis

- Tóm lại: muốn sử dụng thuộc tính flexbox ta phải có 2 thẻ
 - Thẻ ngoài: chứa thuộc tính display: flex (hoặc inline-flex)