Css.

Ul,li: list-style-type:disc;circle;square;decimal;

css unit.

absolut:

px: 1 điểm ảnh trên mà hình;

relative:

%: chiếm tỉ lệ màn hình và thẻ chứ nó.

Rem: phụ thuộc font-size vào thẻ html.

Em: phụ thuộc vào thẻ gần nhất chứ nó. 1em=16px; font-size: 100%=16px;

Vw: viewport-width: chiều rộng của màn hình.1vw=1% của chiều ngang. Nó bị ảnh hưởng thẻ chứa nó.

Vh: chiều cao.

Padding: top-bottom-right-left

(Border: lớp viền: Dotted: nét đứt, solid: nét liền, inherit: thừa kế, initial: giá trị ban đầu, none: để hủy kh xét border.

Border-width: độ dày của viền.

Border-style: kiểu đường viền.

Border-color: màu đường viền.

(Margin:top-bottom-right-left

//nằm trên cùng 1 hàng.

Display:flex.

Justify-content: center; thuộc tính cho các phần tử nằm ngang.

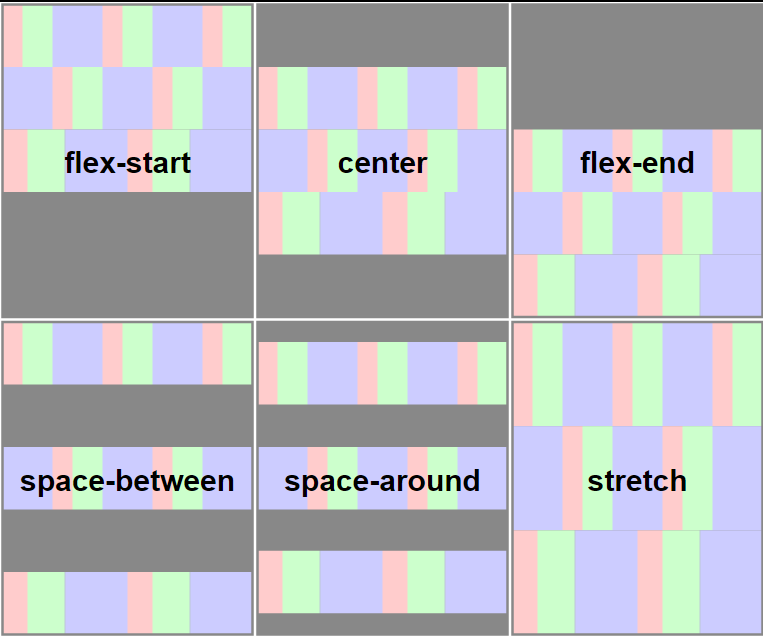
// flex-start/end; center; space-between;

space-around: 2 đầu và đuôi bị thay đổi khi style.

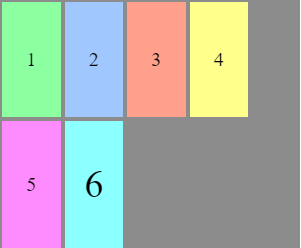
space-evenly: khoảng cách bằng nhau.

Stretch: Giá trị này sẽ kéo giãn các phần tử cho lấp đầy flex, tuy nhiên nếu độ dài của các phần tử quá ngắn thì các phần tử sẽ float: left;//

align-items: center;



Normal: phụ thuộc vào giá trị thẻ div.

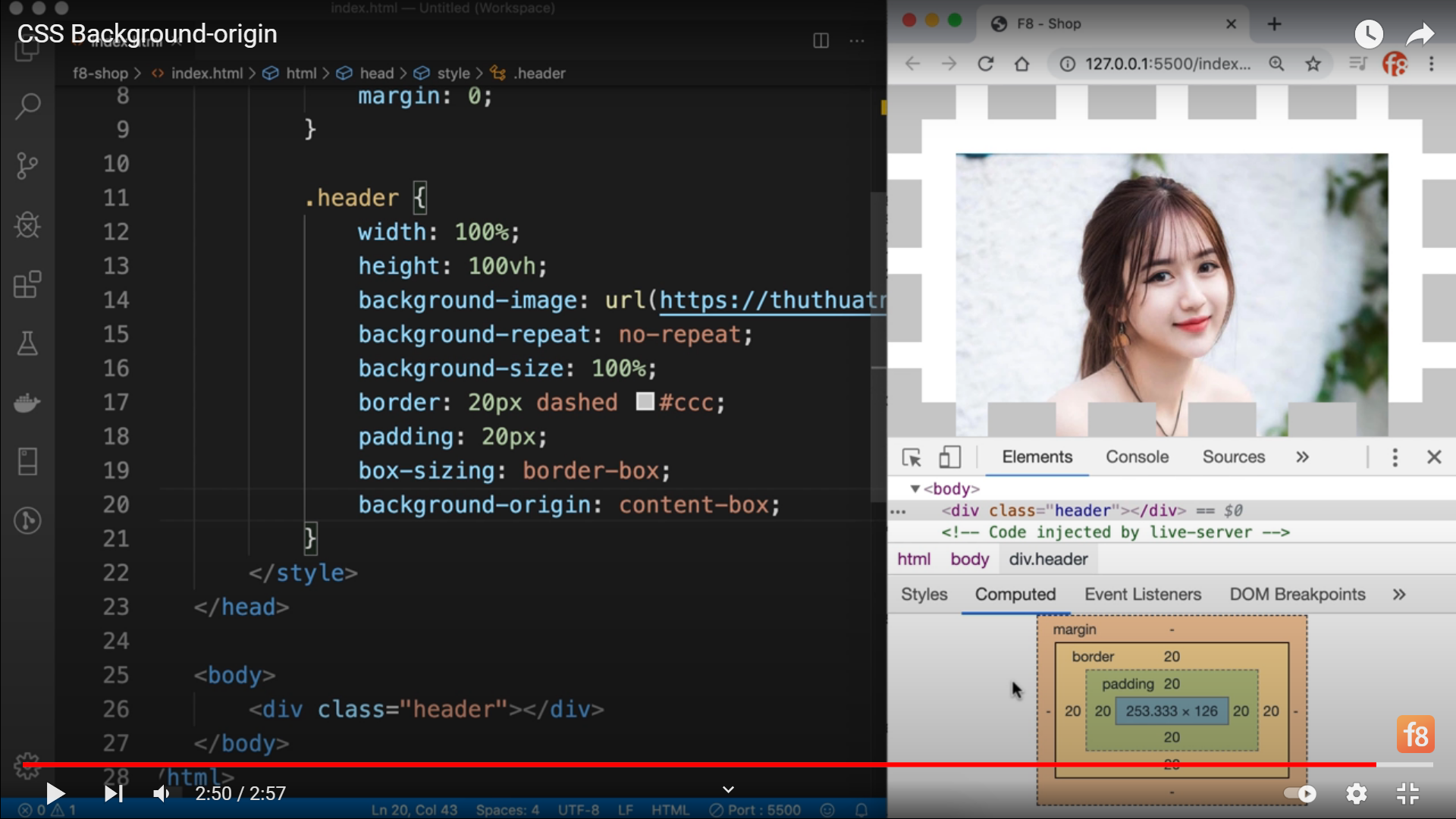
Stretch: kích thước bằng nhau. 

float:left đẩy hết các thuộc tính nằm trên 1 hàng.//)

(box-sizing: border-box/unset(hủy đi): chiều ngang=border=conten của màn hình; lúc đó sẽ tự động tính toán)

///(bacgroup-clip:đc đổ từ cái danh giới nào; border, padding,content, margin, đổi màu nền cho content or border…)

///backgroup-origin: tương tự như backgroup-cilp, nhưng nó đi kèm với bacgroup-image. Đổ từ content-box ra.



/// màu chèn thêm bức ảnh. Thêm bức ảnh.

//backgroup-image: đặt ảnh nền. url(),url()

🡺 linear-gradient(180deg,#333,#ccc), url tạo ra 1 cái dải màu chuyển đều đặn. linear: sự đều đặn, gradient:dải màu.(độ, hướng màu).

+) 90 trai-phai, o từ tren xuong duoi, 180 duoi len tren, url().

+) rgba(đỏ, xanh lam, xanh lục , trong suốt) hex mã màu lục phân.

/// size bức ảnh.

//Backgroup-size:

+) contain:nó lấy chiều bức ảnh dài nhất (nhưng kh bị che khuất của hình ảnh, hở ra khoảng trắng)

+) cover: cạnh dài bức ảnh (chấp nhận che khuất bức ảnh; full chiều ngang vừa màn hình)

+) 100px,chiều ngang bức ảnh(tuyệt đối);100%(tương đối chiếm hết toàn bộ chiều ngang).

/// thuộc tính lập lại bức ảnh.

//Backgroup-repeat: no-repeat(x-y).

///vị trí bức ảnh

//backgroud-position:

+) top-bottom-right-left + right-left-center: bên góc trái. or 10px.

+) top 20px right 20px.

///cú pháp ngắn gọn của backgroup.

* Backgrund: url() no-repeat(kh lập lại ảnh) center( căn giữa) / cover(size).

/// hàm: functions

+) var(), rgba(), lineaer-gradient(), rgb(), calc(), attr().

+) rgb( sử dụng màu này ra một cái màu đặc)

+) calc(tính toán, tính toán đc, vd: width: cacl(120 px + 50px)

+) attr(attribiu):

Vd: html <a href=”html.index”>hoc lập trình<a/> muốn lấy hiển thị cả 2.

Css: a::after{ content: ” ( “attr(href)” ) ”; }

” ( trong trường hợp có tag mới cần dấu này.

Vd: tạo ra 1 cái chữ trong 1 thằng box:

Html

Class:box date-title=”đây là box màu hồng”><

html

css: lớp giả

box::after{ content: attr(date-title); }

Css: pseudo-elements. Phần tử giả

🡺::before ; đằng trước

🡺::after; đằng sau

🡺::first-letter; kí tự đầu tiên

🡺::first-line: cái dòng đầu tiên

🡺::seclection:chịu sự ảnh hưởng

Css: position: thiết lập các cái vị trí element.

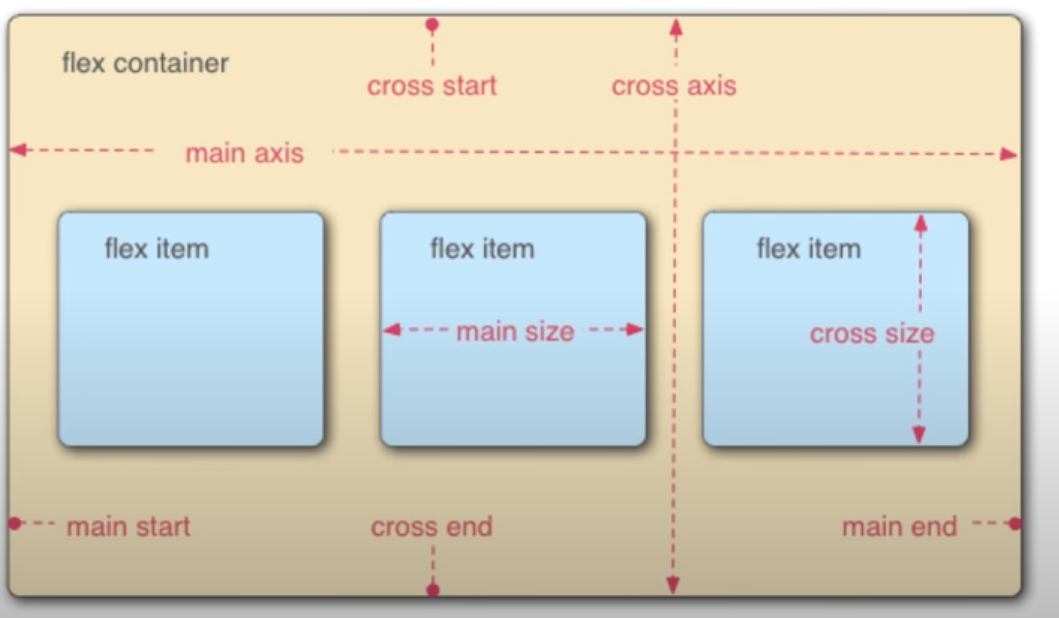
🡺 Relative: tương đối kh bị phụ thuộc nào khác cả. lấy chính mình làm gốc tạo độ.

🡺Absolute: tuyệt đối: phụ thuộc thẻ cha gần nhất.

* Fixed: đứng yên kh di chuyển, cố định. Top;0; left/right;0;
* Sticky: y chang thuộc tính fixed, safari: -webkit-sticky, top:0;

// Kỹ thuật xây dựng lay our.

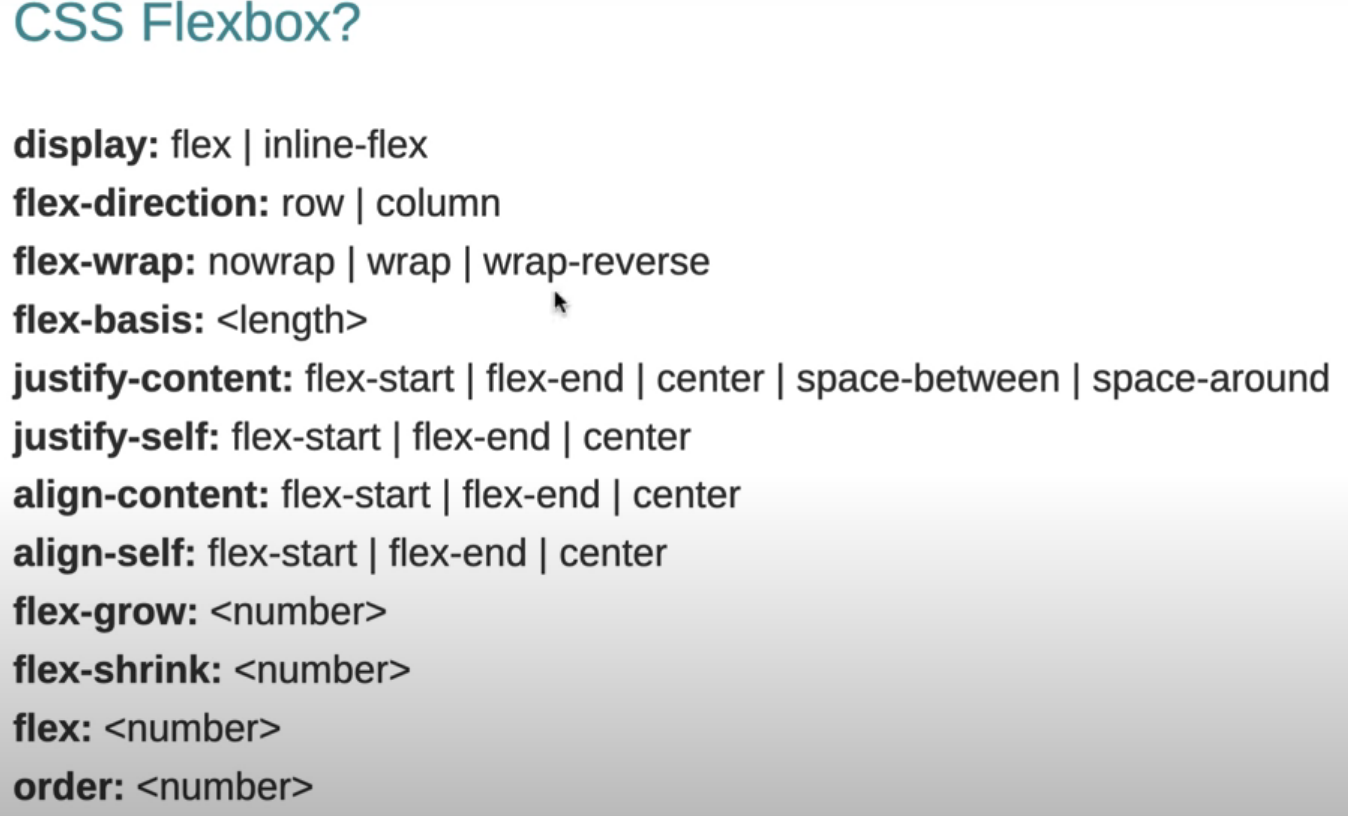
/// Flex Box.



🡺thằng thứ nhất flex container.

+) thằng container là cái thẻ chứa.

🡺flex-item: các thẻ con ở trong thằng container.



* Display: quyết định sử dụng các lay our flex box. Có nghĩa là stlye cho thằng container và item.
* Display: flex (nằm ngang)/ inline-flex(trên cùng 1 hàng)

🡺flex-direction: thay đổi đc cái phương hướng thằng main-xit. Khi dùng display: flex mặc định nó là row(hàng ngang).

* (Flex-direction:row/column. Hàng ngang/dọc

🡺flex-wrap: nowrap/wrap/wrap-reverse. Xuống hàng)

* Viết tắt direction vs wrap: flex-flow: row wrap;

🡺justify-conten: (chiều ngang) căn đc các cái flex item theo phương .hướng.

🡺 justify-conten nếu xét container cha

=> Justify-self: con item.

* Justify-conten: flex-start/ flex-end/
* center/ space-between/ spece-aroud.

🡺Justify-self: flex-start/ flex-end/ center/.

* Thực tế là nếu mà xét thuộc tín justify-conten cho thằng cha = với cái việc xét thuộc tính justify-self. Thì thằng con có tương tự cái thuộc tính đó.

🡺align-content: flex-start/ flex-end/ center.

* align-content: giống thằng justify-conten. Chiều dọc.

🡺 align-sefl: flex-start/ flex-end/ center.

🡺flex-basis: xét kích thước main-size; item.

🡺flex-grow: nở ra, lớn lên .thằng để tăng đc flex-item.

🡺flex-shrink: thu nhỏ lại.

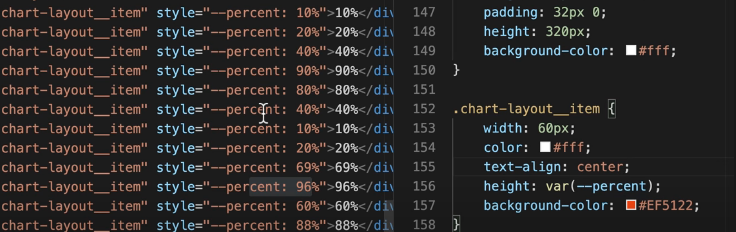
🡺flex: viết tắt của ba thằng(basis/ grow/ shrink)

🡺 flex: (basis/ grow/ shrink);

🡺order : sắp xếp các item.

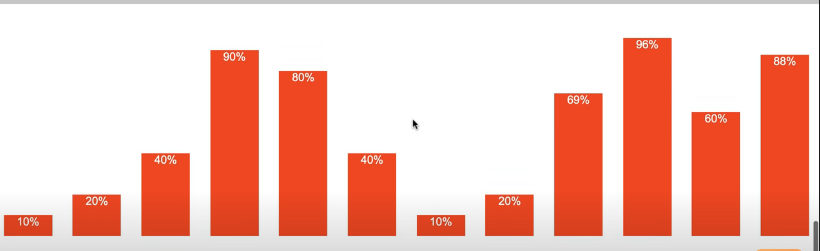
Thực hành flex-box.

Biến local.



Justify-conten: space-aroup: dãn đều ra. (ngang)

Align-item/ align-self: cho các Item theo phương hướng. (dọc)



Hiệu ứng:

Liner: đều.

Ease-in-our: chậm => nhanh

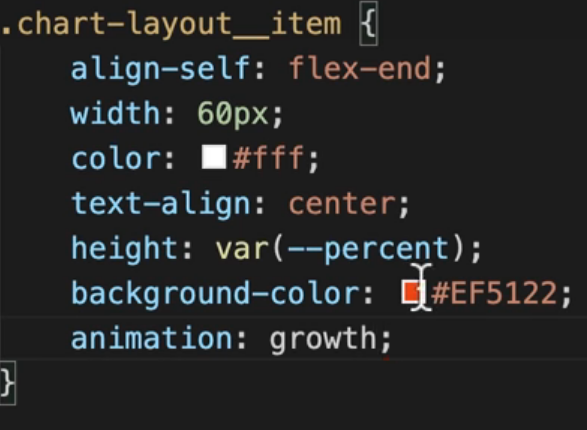
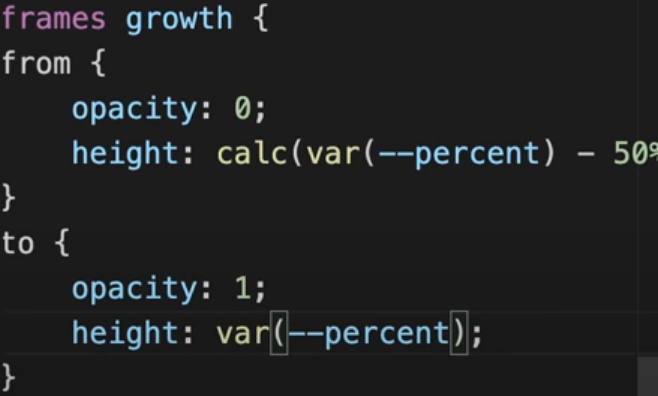
+) Sử dụng enimation . chuyển động.

- Đặt cho nó 1 thằng @keyframes

* @keyframes

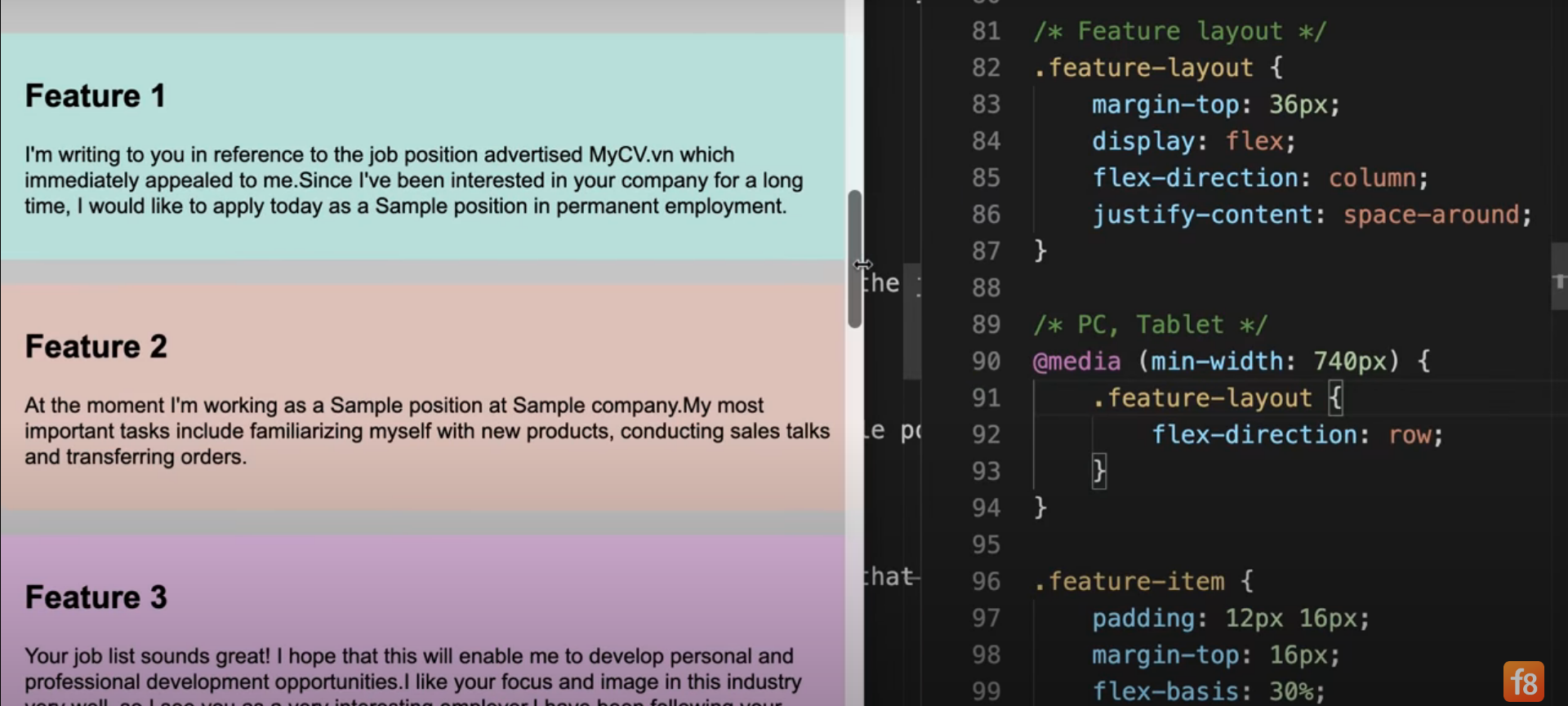
🡺from: điểm bắt đầu nó sẽ làm gì.

🡺 to: end.

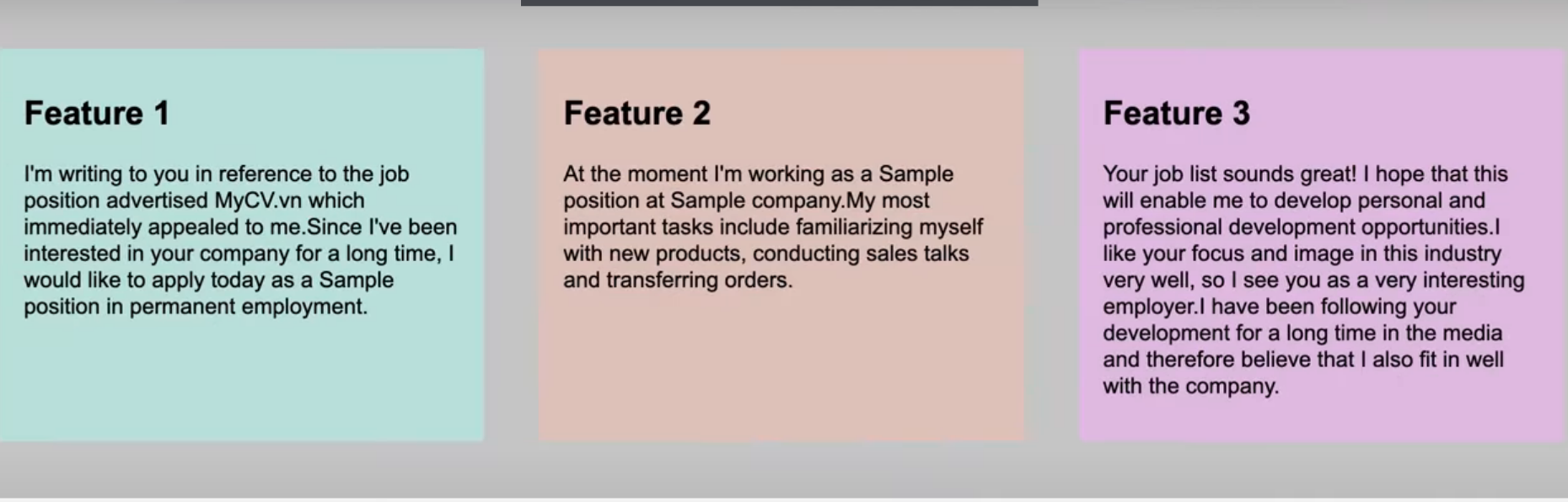
 

Ví dụ: với cái chiều dài màn hình 740 px thì offline cái thuộc tính feature layout

Mo bi



Pc



[https://codepen.io/enxaneta/full/adLPwv/](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbGVsNmxxVnNWalAzSU4xZG1Uc3VFWjhPWldqd3xBQ3Jtc0trWWI3aUk0b3d1emJjbUViYlN6OHQ3OGNoenIxVEkxRzVIclZiV3RUczA2bjdtRkZtZW1IajhGczdwODdsOG5fSnExTTk4YUZORThQLXh5UmJWM0hDYVpCU0ZteXhIWUpPTjRPdURjMUVvYnBoVUR2WQ&q=https%3A%2F%2Fcodepen.io%2Fenxaneta%2Ffull%2FadLPwv%2F)

Responsive là kỹ thuật giúp web hiển thị tương thích với nhiều kích thước màn hình khác nhau(mobi,tab,pc)

* Tối ưu trải nghiệm người dùng:
* 1. Hiển thị rỡ ràng các thành phần (hình ảnh, cỡ chữ, nút bấm,..)
* 2.ẩn/ hiện các thành phần phù hợp theo kích thước màn hình.

Chúng ta sẽ làm gì?

1. Dùng css thay đổi kích thước phù hợp cho các thành phần hiển thị trên web (hình ảnh, cỡ chữ, nút bấm…)
2. Dùng css để ẩn / hiện các thành phần phù hợp theo kích thước màn hình.
3. Để ẩn đi dùng disply : none.
4. Để hiển thị lên display: block.
5. độ phân giải iphone 6/7
6. vd: 414 x 736 kích thước màn hình
7. 76% là tỉ lệ màn hình
8. xoay màn hình

view port: google: meta view port parameters.

view port:

thẻ meta giúp mình khai báo nhiều thông tin website của mình, trong trường hợp này chúng ta đang tìm hiều với cái meta có cái name là view port.

view port được hiểu là cái khung nhìn.

content: là cái phần định nghĩa, là cái phần tùy chọn. phần tham số của thẻ view port.

# ////Media query (@media)

@media 1.not|only 2.mediatype and( 3.mediafeature and|or|not 3.mediafeature){

Css:code}

1.keywords:

- not: loại trừ, ví dụ A,B,C, mà mình nói not A là loại trừ thằng A.

- only: chỉ với, ví dụ only A chỉ áp dụng cho thằng A.

- and: Và.

- or: Hoặc.

2.MediaTypes: đứng sau vị trí Not|Only.

🡺 muốn sử dụng 1 trường hợp css thôi.

- Print: chế độ in.

- Screen: chế độ màn hình. Vd: điện thoại, máy tính bảng…

- Speech: màn hình nói được. thường được sử dụng cho người khuyết tật.

- All-default: xác định cả 3 đối tượng trên.

3.Media Features: giá trị: px

🡺 kích thước màn hình.

- min-width: bé nhất. lớn hơn 1024 px thì mới offline.

-max-width: lớn nhất. vd: bé hơn 1024 px thì mới offline.

- …

4.Prolyfill:

6. Breakpoints: là những điểm những vị trí web chúng ta sẽ thay đổi để thích ứng tạo nên giao diện responsive.

Gg: media query common breackpionts.

có 3 thẻ div

mobile

tablet

pc

display: none; để ẩn đi

mobile: width < 740px

@media only screen and (max-width: 739px){

.mobile{

display: block; hiển thị dạng khối

}

}

tablet: width >= 740px and width < 1024px

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

.tablet{

display: block; hiển thị dạng khối

}

}

pc : width >= 1024 px

@media only screen and (min-width: 1024px){

.pc{

display: block; hiển thị dạng khối

}

}

media query: **px, rem or em**?

Trong 3 cai này chúng ta sử dụng:

**Em**: 16px 🡺 1em.

Gg: px to em đổi đơn vị.

Học responsive qua ví dụ:

Test\_reponsive.

Responsive: navigation

Navbar

Gg:

Trang: fort awesome

bars icon

10.hệ thống lưới. gird system

1.tạo nên các khung nền, hỗ trợ việc sắp xếp bố cục theo

🡺 trật tự/ thống nhất/ cân bằng.

2. hệ thống lưới thường gặp:

🡺lưới nhiều cột (mutilcolum)

🡺lưới một cột(modular gird)

🡺lưới module(modular gird)

🡺lưới đường cơ sở(baseline gird)

3.vai trò:

-tổ chức có các đường giong tiện lợi, dễ dàng sắp xếp các thành phần ngăn nắp.

-cân bằng: dù là đối xứng/bất xứng. offset đối xứng.

-tách biệt thành phần: phân chia nội dung, tạo khoảng cách các thành phần hiệu quả.

4.reponsove.

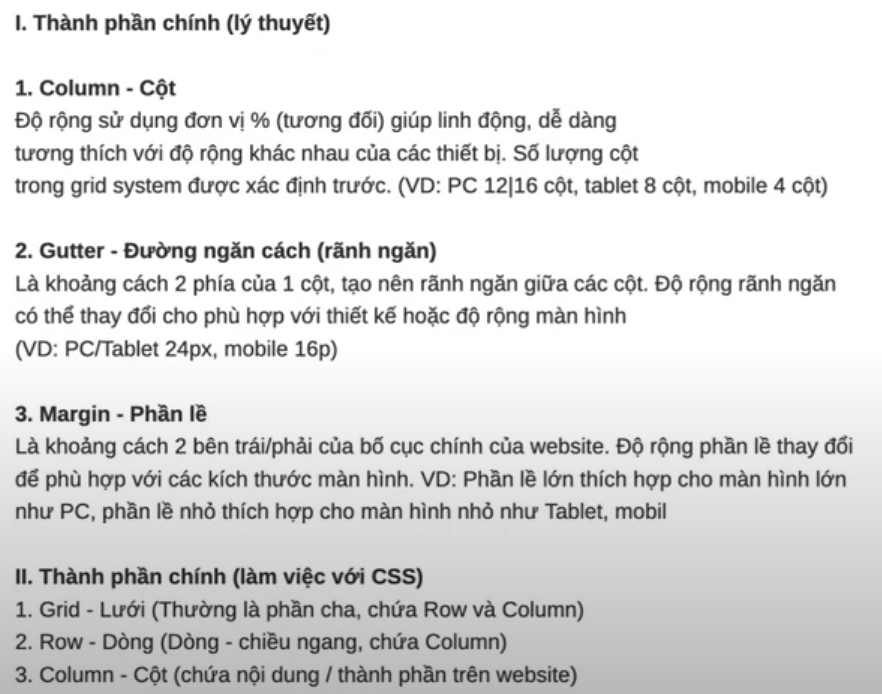
- gird: thành phần cha <> toàn bộ trang wed tong cong 12 cột

- row: dòng

- column: cột

- getter: khoảng cách 2 phía của column

10.hệ thống lưới: gird system: p2.



1:**cột: %**

2: khoảng cách giữa các colum: **pc: 24px, mobile: 16px**

**3. thay đổi độ rộng 2 bên lề. cho phù hợp với pc. Tablet/mobile**

**II.thành phần chính.**

1. **Gird: lưới( thường là thành phần cha chứa các row và column)**
2. **Row: hàng ngang(thường chứa các column trên 1 hàng ngang)**
3. **Column: hàng dọc/ cột(thường chứa các nội dung website)**

* **Colum thường đc sử dụng pc: 12/16 cột, tablet: 8 cột, mobile 4 cột.**

12.tạo thư viện.

1. Tự tau xây sựng thư viện css.
2. Biết cách ứng dụng GIRD vào layOut
3. Tìm hiểu về GIRD layout của thư viện boottrap

I.Tạo đối tượng chính: GIRD, ROW, COLUMN.

1.tạo class.

- gird: full-width, chiếm hết chiều ngang đối tượng (cha)

- wide: chiều ngang tối đa 1200px

Tạo thư viện 1.

2.đặt lại chiều rộng trên các thiết bị.

/\* 6 tablet - pc low resolution màn hình pc độ phân giải thấp \*/

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

    .wide {

        width: 644px;

    }

}

Pc độ phân giải thấp

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

    .wide {

        width: 984px;

    }

}

Tạo đối tượng ROW

* Thằng gird để chứa dối tượng row.
* 1. Chứ các column, giúp các column nằm theo chiều ngang.
* 2. Khi tổng chiều ngang của column vượt quá kích thước row, thì column sẽ tự xuống hàng.
* 3. Loại bỏ khoảng thừa do gutters tạo ra giữa 2 bên column/ tạo 2 ở 2 phía.
* Cái gutter là khoảng trắng ở 2 bên được tạo ra bởi 2 cái padding của 2 column. 
* Row này có tác dụng tạo ra để giúp các bạn không bị lừa 2 bên phải và trái của cột đầu tiên và cuối cùng.

Css:

@media (min-width: 740px){

.row{

Margin-left: -8px;

Margin-right: -8px;

}

}

@media (min-width: 1113px){

.row{

Margin-left: -12px;

Margin-right: -12px;

}

}

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

.row{

Margin-left: -12px;

Margin-right: -12px;

}

}