



THIẾT KẾ WEB

Giảng viên: Ths. Nguyễn Đức Thiện
Email: ducthien84@gmail.com
Phone: 0974913448

BÀI 4:

RESPONSIVE WEB DESIGN

Giới thiệu Responsive

Các bước thực hiện

Mobile first

Desktop first

4.1. RESPONSIVE

RESPONSIVE WEB DESIGN



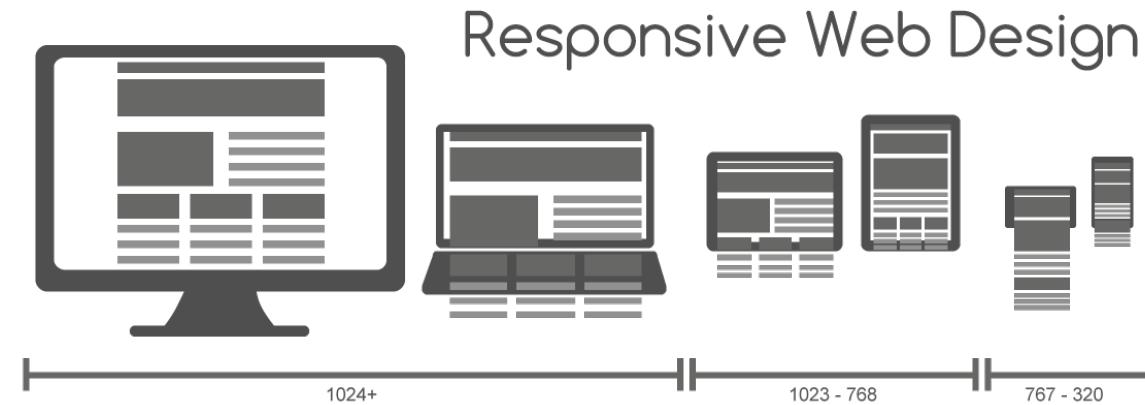
Tham khảo: https://www.w3schools.com/css/css_rwd_intro.asp



- Tương thích trên các kích thước trình duyệt khác nhau
- Sử dụng toạ độ tương đối % ?
- Responsive CSS ?
- Viết các đoạn CSS khác nhau đối với các kích thước màn hình khác nhau ?
- Responsive Web Design làm cho trang web hiển thị tốt trên mọi thiết bị
- RWD chỉ sử dụng HTML và CSS
- RWD không phải là một chương trình hoặc sử dụng Javascript



- Responsive Design là cách thiết kế đi từ bản thiết kế ban đầu có thể là trên mobile hoặc desktop
- RWD phản hồi cho sự thay đổi của kích cỡ trình duyệt bằng cách điều chỉnh vị trí các phần tử trên trang web cho phù hợp
- RWD hiển thị nội dung trang web dựa trên khoảng không gian mà trình duyệt cung cấp cho nó

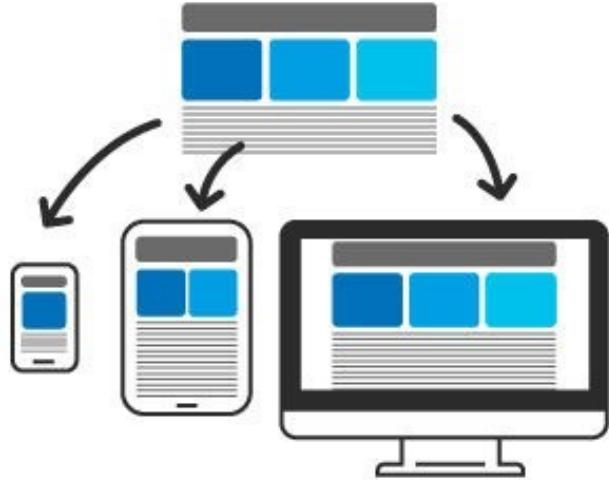


- **Adaptive Design** là cách thiết kế trong đó có các bản thiết kế tương thích với một số lượng nhất định cố định các kích cỡ của màn hình
- Khi trang **web** được hiển thị nó sẽ dựa trên không gian mà thiết bị cung cấp để tìm thấy bản thiết kế phù hợp nhất cho màn hình đó
- Ví dụ khi hiển thị trên máy tính trang web sẽ tìm bản thiết kế layout trên máy tính và hiển thị, việc co giãn kích cỡ trình duyệt trên máy tính không có ảnh hưởng đến trang web

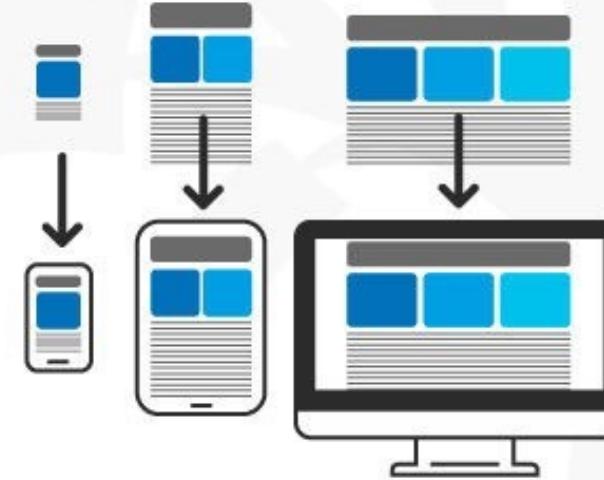


ADAPTIVE VS RESPONSIVE

Responsive Web Design



Adaptive Web Design



4.2. CÁC BƯỚC

- Thiết lập Viewport
- Viết CSS cho các chiều rộng thiết bị khác nhau
- Áp dụng nguyên tắc Mobile First
- Kiểm thử trên mobile

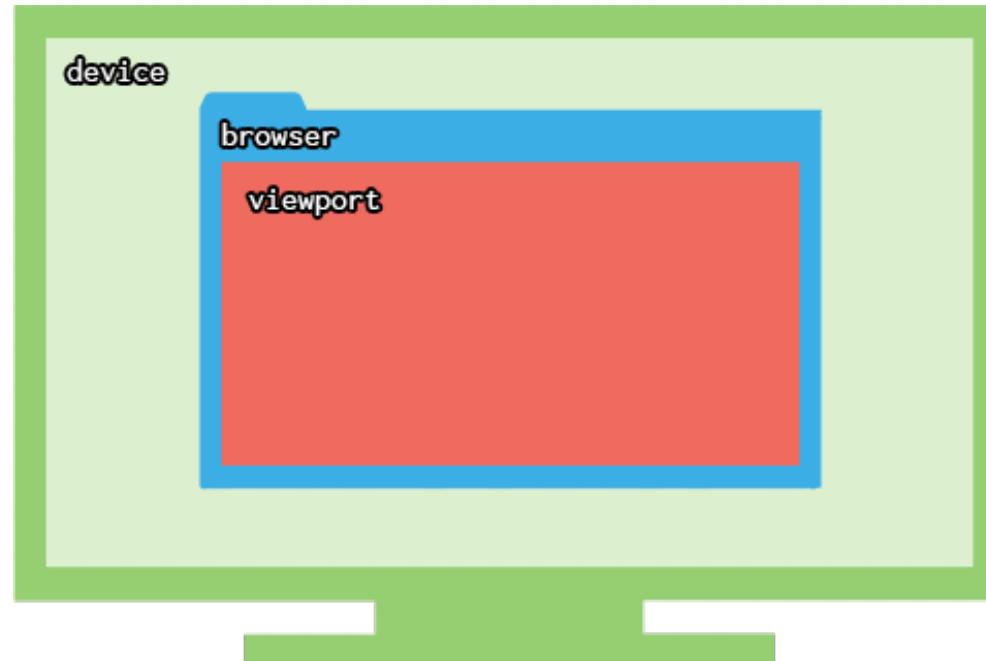
(Có thể cài đặt extension “Mobile Simulator” trên Chrome để kiểm thử)



VIEWPORT

RESPONSIVE VIEWPORT

- Khung nhìn là phần nhìn thấy của người dùng trên một trang web
- Khung nhìn là khác nhau trên các thiết bị khác nhau
- Khung nhìn sẽ bé hơn trên các thiết bị di động



THIẾT LẬP VIEWPORT

SETTING THE VIEWPORT

- HTML5 cho phép thiết lập khung nhìn dựa vào thẻ <meta>
`<meta name="viewport" content="width=device-width, initial-scale=1.0">`
- `width=device-width`: thiết lập chiều rộng của trang web bằng chiều rộng màn hình thiết bị
- `initial-scale=1.0`: thiết lập độ phóng to màn hình khi trang web hiển thị lần đầu bởi trình duyệt



THIẾT LẬP VIEWPORT

SETTING THE VIEWPORT

Có sử dụng “viewport”

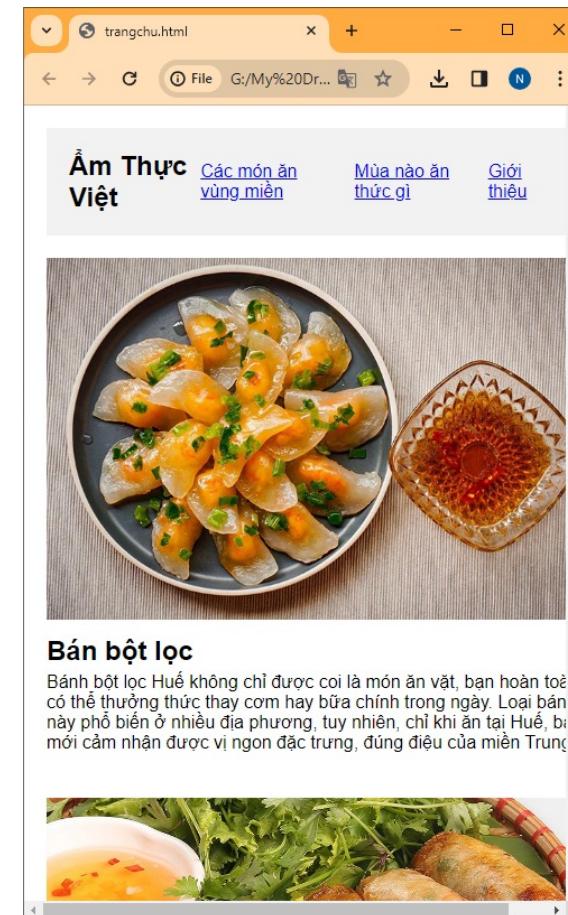
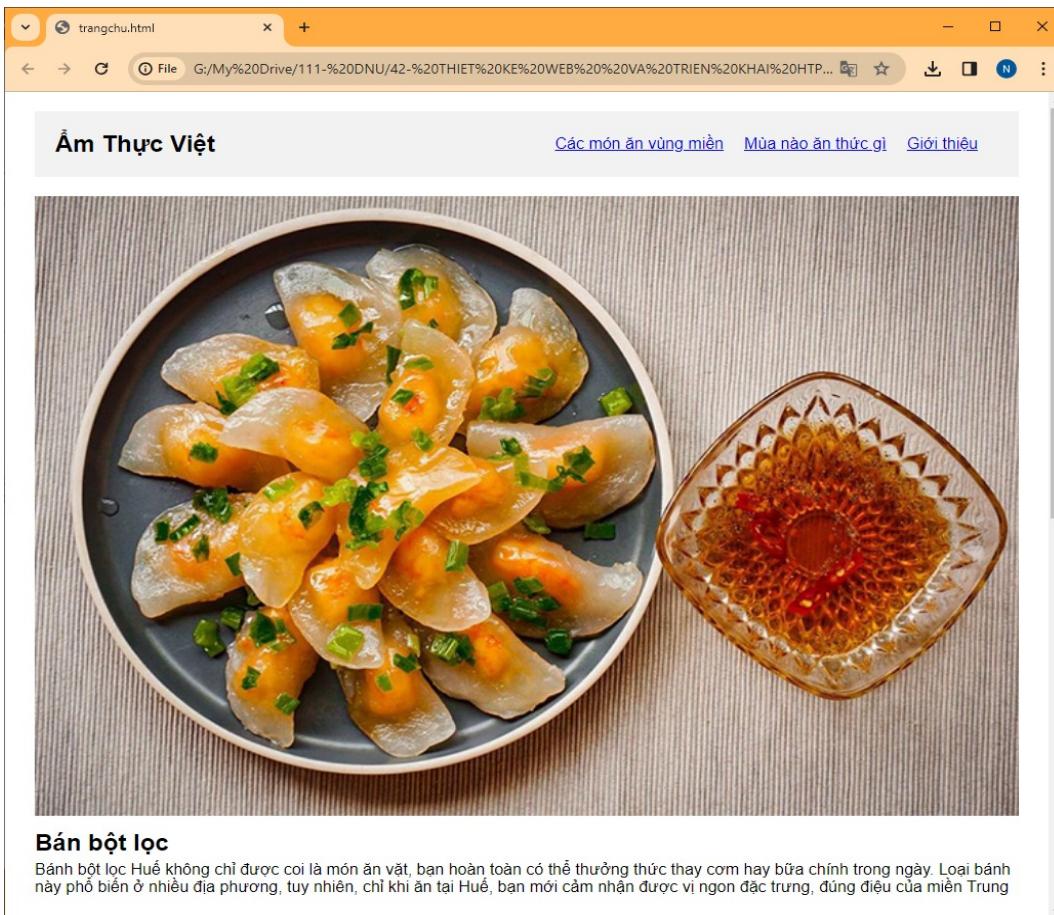


Không sử dụng “viewport”



THIẾT LẬP VIEWPORT

SETTING THE VIEWPORT



MEDIA QUERIES

DEVICE WIDTH

- Kỹ thuật được đưa vào trong **CSS3**
- Sử dụng các luật **@media** để áp dụng các style khác nhau cho các loại thiết bị khác nhau
- Các truy vấn **media** có thể được sử dụng để kiểm tra nhiều thứ:
 - Chiều rộng, chiều cao của khung nhìn
 - Chiều rộng, chiều cao của thiết bị
 - Hướng của thiết bị (ngang, dọc)
 - Độ phân giải
- Sử dụng truy vấn **media** là cách thức phổ biến để thực hiện RWD cho các thiết bị để bàn, máy tính xách tay, máy tính bảng, điện thoại
- Có thể sử dụng **media** để chỉ rõ một kiểu style nào đó chỉ được áp dụng cho các tài liệu dùng để in hoặc để đọc (print, screen, speech)
- Cú pháp: **@media not|only mediatype and (mediafeature and|or|not mediafeature) {}**



- Từ khoá **not**: phủ định toàn bộ câu truy vấn **media**
- Từ khoá **only**: từ khoá **only** ngăn chặn các trình duyệt cũ không hỗ trợ truy vấn **media** áp dụng một style cụ thể và không có tác dụng với các trình duyệt mới hiện nay
- Từ khoá **and**: dùng để kết hợp các tính năng **media** với một kiểu **media** hoặc của nhiều tính năng **media**
- Cũng có thể sử dụng nhiều tệp CSS khác nhau cho các kiểu media khác nhau:
`<link rel="stylesheet" media="screen and (min-width: 900px)" href="widescreen.css">`



- Ví dụ đơn giản thiết lập **style** hiển thị cho màn hình điện thoại di động
- Màu sắc mặc định của trang web là màu trắng, khi chuyển sang các thiết bị có chiều rộng màn hình bé hơn 320px thì màu nền chuyển sang màu cam

```
body {  
    background-color: white;  
}
```

```
@media screen and (max-width: 320px) {  
    body {  
        background-color: orange;  
    }  
}
```



- CSS hỗ trợ việc chỉ định cách trình bày khác nhau trên các thiết bị khác nhau thông qua **mediatype**

Kiểu	Ý nghĩa
all	Tương thích với mọi thiết bị
braille	Dành cho các thiết bị chữ nổi
embossed	Dành cho máy in chữ nổi
handheld	Dành cho các thiết bị cầm tay (màn hình nhỏ, băng thông hạn chế)
print	Dành cho hiển thị preview trên máy in
projection	Dành cho các thiết bị trình chiếu như máy chiếu
screen	Dành cho các thiết bị màn hình màu của máy tính
speech	Dành cho các máy đọc màn hình sẽ đọc màn hình thành giọng nói



KÍCH THƯỚC MÀN HÌNH

DEVICE WIDTH

- Sử dụng truy vấn **media** để tương thích các kích thước màn hình khác nhau

Giá trị	Màn hình
max-width: 320px;	Điện thoại di động hiển thị chiều dọc
max-width: 480px;	Điện thoại di động hiển thị chiều ngang
max-width: 600px;	Máy tính bảng, hiển thị chiều dọc
max-width: 800px;	Máy tính bảng, hiển thị chiều ngang
max-width: 768px;	Máy tính bảng loại to, hiển thị chiều dọc
max-width: 1024px;	Máy tính bảng loại to, hiển thị chiều ngang
max-width: 1025px;	Từ kích cỡ này là kích cỡ hiển thị trên máy tính



- Sử dụng thuộc tính **width**: đơn vị % để ảnh tự động co dãn theo kích cỡ màn hình
- Sử dụng thuộc tính **max-width**: đơn vị % để ảnh tự động co dãn và không vượt quá kích cỡ chiều rộng cho trước
- Ảnh nền: sử dụng thuộc tính **background-size** và các giá trị có thể nhận của nó: contain, cover, %x %y để căn chỉnh ảnh nền cho tương thích nhiều thiết bị và không làm giảm chất lượng ảnh
 - **contain**: ảnh nền sẽ co dãn để khớp với phần nội dung và giữ tỉ lệ ảnh
 - **%x %y**: ảnh nền sẽ kéo dãn %x chiều rộng ảnh gốc và %y chiều cao ảnh gốc
 - **cover**: ảnh nền sẽ co dãn để phủ hoàn toàn phần nội dung, nếu ảnh tràn ra ngoài phần nội dung sẽ bị cắt phần thừa và ảnh vẫn giữ nguyên tỉ lệ



- Sử dụng thuộc tính **width**: đơn vị % để video tự động co dãn theo kích cỡ màn hình
- Sử dụng thuộc tính **max-width**: đơn vị % để video tự động co dãn và không vượt quá kích cỡ chiều rộng cho trước

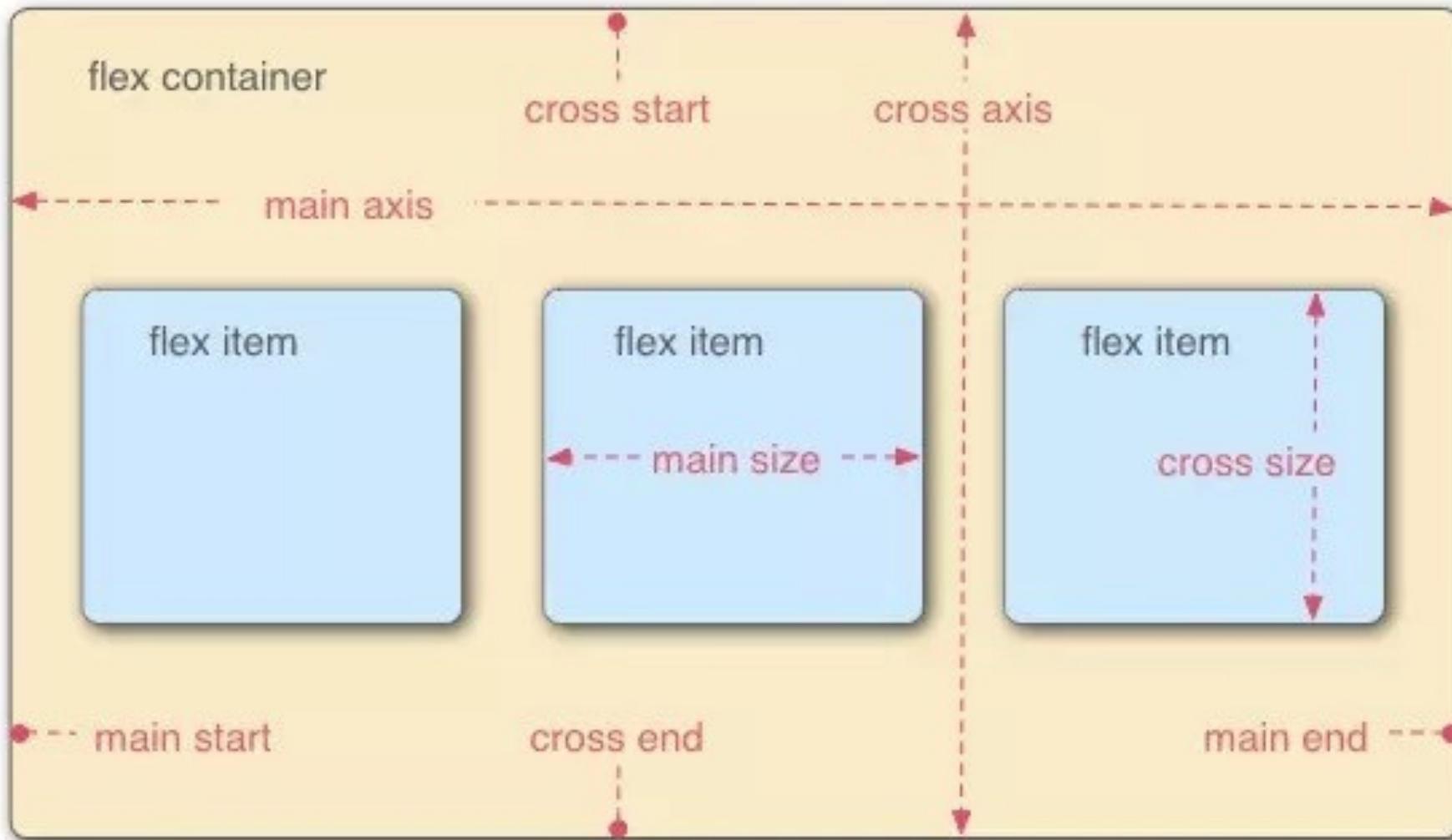


- CSS flexbox là một mẫu layout dùng để thiết kế các layout một cách linh hoạt và hiệu quả
- Flex Box bao gồm một vật chứa container và các phần tử chứa trong nó gọi là items
- Việc phân chia vị trí hiển thị của các items được kiểm soát rất dễ dàng và linh hoạt



FLEXBOX

FLEX BOX LAYOUT



- Một số thuộc tính áp dụng cho flex
 - **display**: nếu giá trị là flex thì phần tử chuyển thành flex container
 - **flex-direction**: row, row-reverse, column, column-reverse
 - **flex-wrap**: quyết định việc bọc các items trong flex container có các giá trị nowrap, wrap, wrap-reverse
 - **flex-flow**: thuộc tính gọn của flex-direction flex-wrap
 - **justified-content**: căn chỉnh items theo đường trực chính main axis có thể nhận các giá trị flex-start, flex-end, center, space-between, space-around, space-evenly
 - **align-items**: xác định việc đặt các items theo chiều cross axis nhận các giá trị stretch, flex-start, flex-end, center, baseline



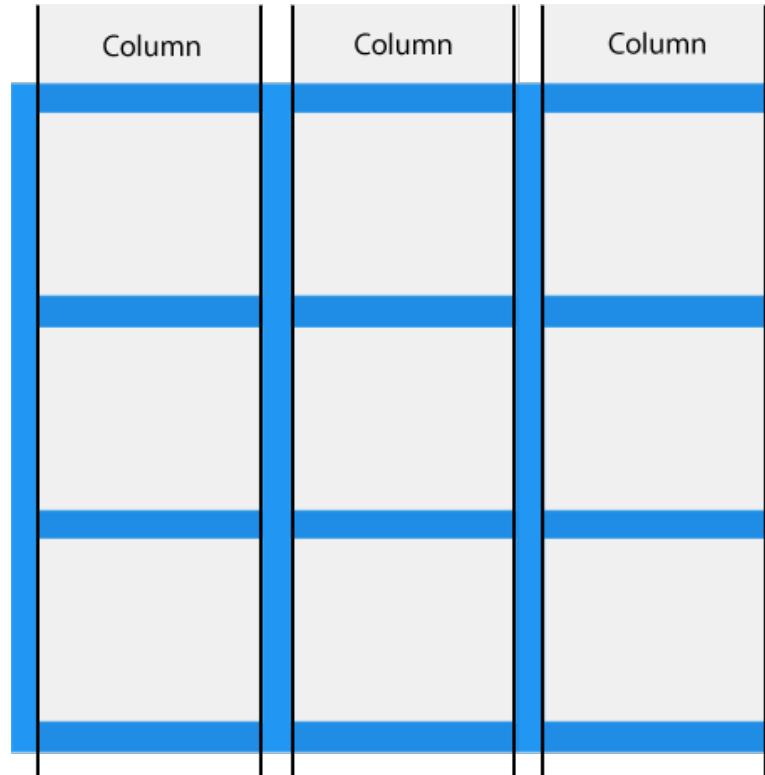
- CSS Grid Layout cung cấp hệ thống layout dạng lưới bao gồm dòng và cột
- Grid Layout trợ giúp việc thiết kế bố cục trang web mà không phải dùng đến **float** và **position**
- Một lưới bao gồm một phần tử cha và các phần tử con trong phần tử cha
- Phần tử cha được định nghĩa bằng cách để thuộc tính **display** là **grid** hoặc **inline-grid**
- Các phần tử nằm trong phần tử có **display: grid** mặc định là các phần tử con



GRID COLUMNS

LAYOUT LƯỚI

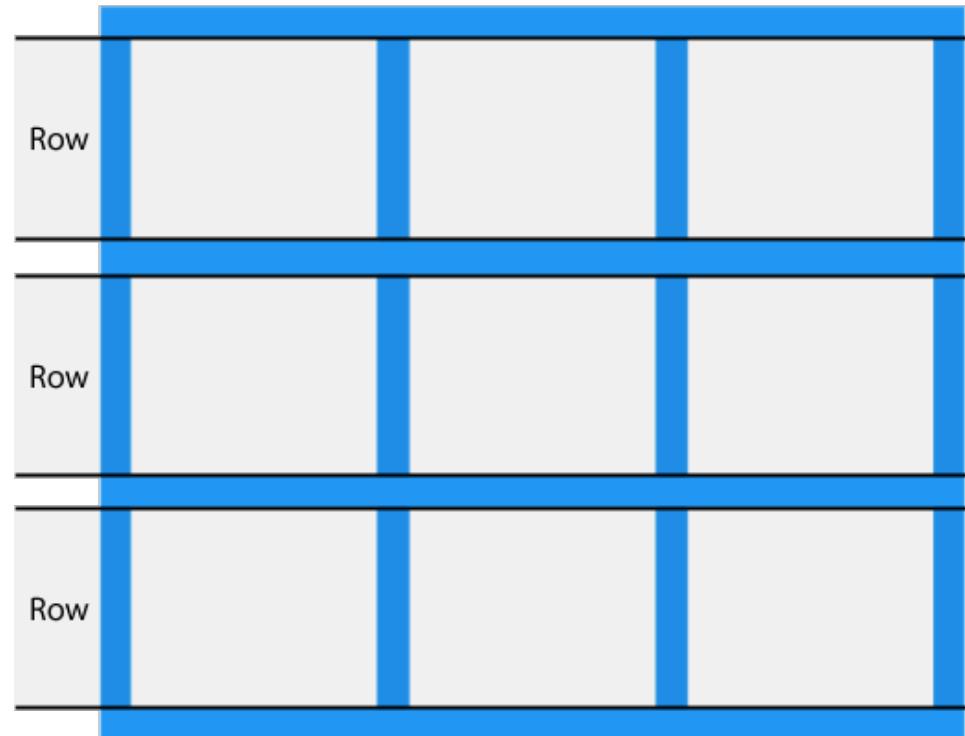
- Các phần tử nằm trên cùng một đường thẳng theo chiều dọc gọi là cột trong lưới



GRID ROWS

LAYOUT LƯỚI

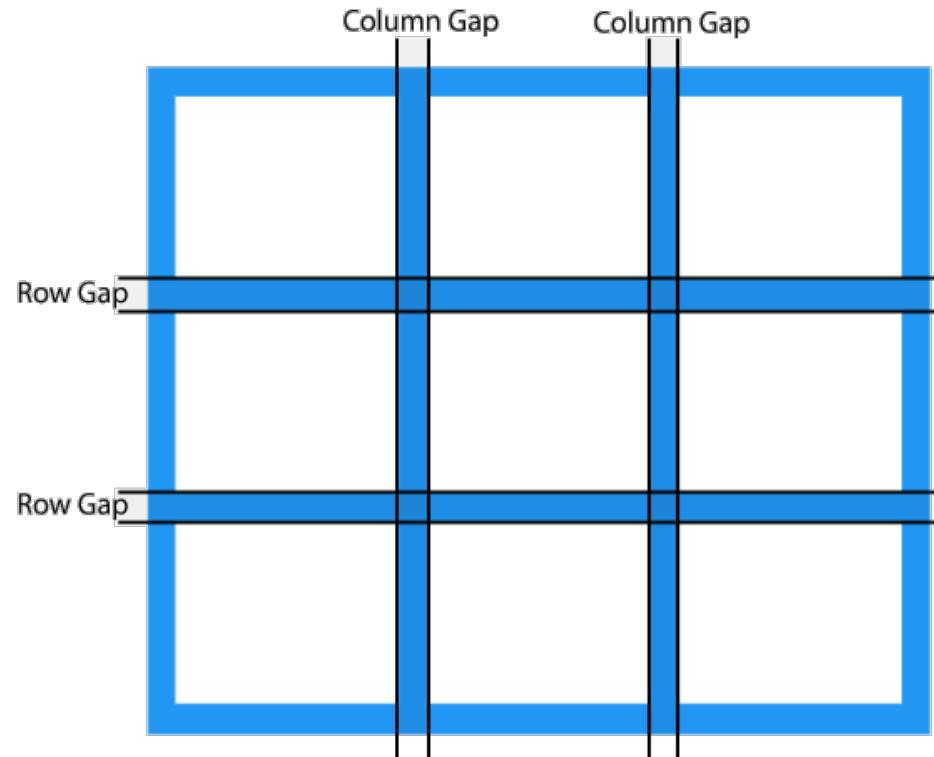
- Các phần tử nằm trên cùng một đường thẳng theo chiều ngang gọi là dòng trong lưới



GRID GAPS

LAYOUT LƯỚI

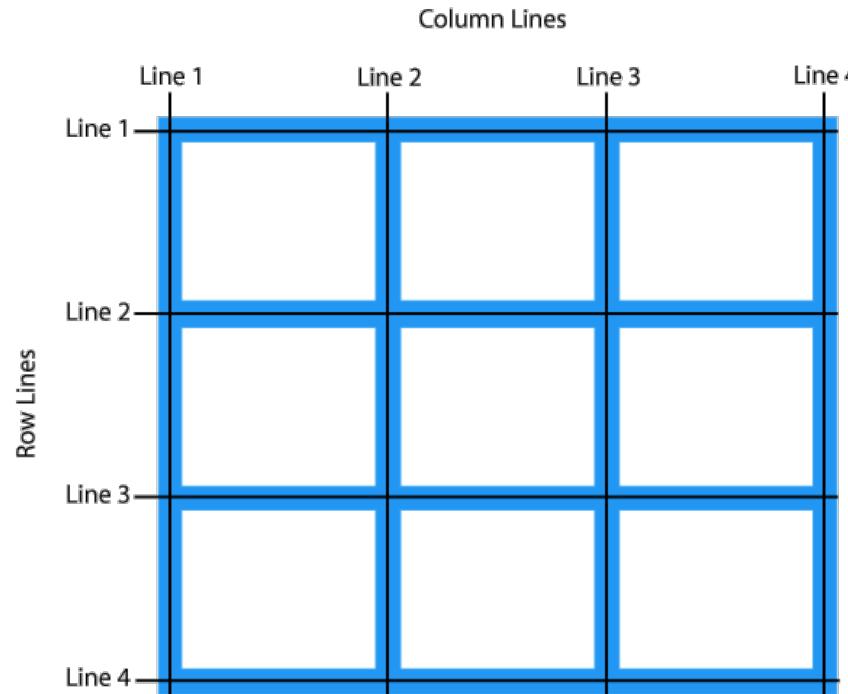
- Khoảng cách giữa mỗi dòng/cột trong lưới gọi là **gaps**
- Có thể điều chỉnh các khoảng cách (gaps) dựa vào các thuộc tính: column-gap, row-gap, gap



GRID LINES

LAYOUT LUỐI

- Đường thẳng giữa các cột gọi là **columns line**
- Đường thẳng giữa các dòng gọi là **rows line**



GRID ITEM

LAYOUT LƯỚI

- Có thể đặt phần tử tại một vị trí cụ thể trong lưới bằng cách chỉ rõ giá trị **column**, **row**
- Ví dụ:

```
.item {  
    grid-column-start: 1;  
    grid-column-end: 3;  
}
```

Đặt phần tử ở cột 1 và kéo dài đến cột 3



4.3.MOBILE FIRST

RESPONSIVE WEB DESIGN

MOBILE LAST (DEGRADED, SHOE-HORNED, SHORT-SIGHTED, CRAPPY)



MOBILE FIRST (PROGRESSIVELY ENHANCED, FUTURE-FRIENDLY, AWESOME)



Slide: 30



- Luôn luôn thiết kế hướng tới các thiết bị di động trước
- Giúp cho trang web hiển thị nhanh hơn trên các thiết bị màn hình nhỏ
- Bản thiết kế đầu tiên hướng tới thiết bị màn hình nhỏ sau đó thực hiện responsive cho các thiết bị màn hình lớn hơn
- Sử dụng truy vấn **media**: min-width



MOBILE FIRST

DESIGN FOR MOBILE FIRST

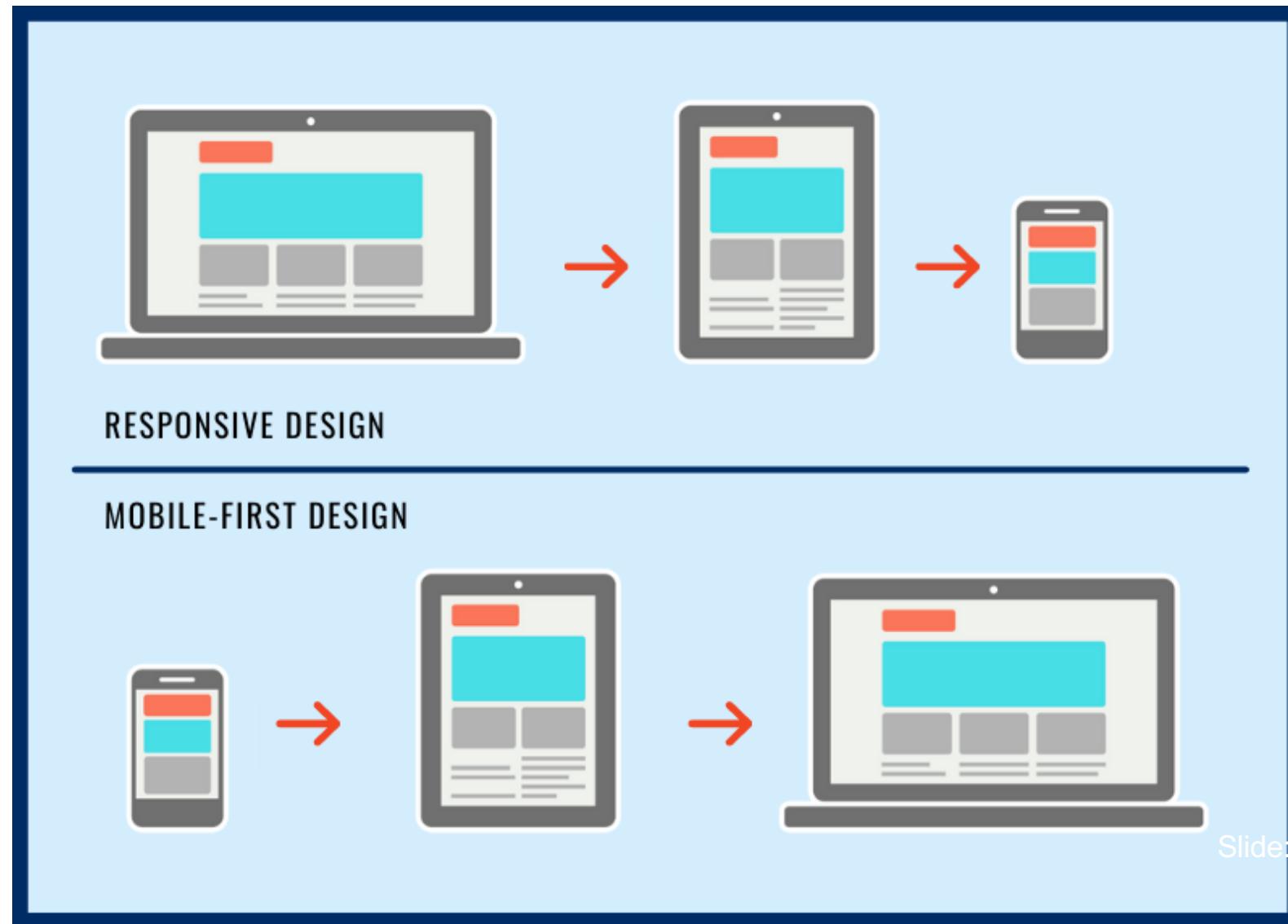
```
/*Smart phone nhỏ*/  
@media screen and (min-width:  
240px){ }  
/*Iphone(480 x 640)*/  
@media screen and (min-width:  
320px){ }  
/*Tablet nhỏ(480 x 640)*/  
@media screen and (min-width:  
480px){ }
```

```
/*Ipad dọc(768 x 1024)*/  
@media screen and (min-width:  
768px){ }  
/*Ipad ngang(1024 x 768)*/  
@media screen and (min-width:  
1024px){ }
```



4.4. DESKTOP FIRST

RESPONSIVE WEB DESIGN



- Ưu tiên thiết kế từ màn hình to rồi đến màn hình nhỏ hơn
- Bản thiết kế đầu tiên là cho thiết bị màn hình rộng
- Sử dụng truy vấn **media**: max-width



DESKTOP FIRST

DESIGN FOR DESKTOP FIRST

- /*Ipad ngang(1024 x 768)*/
- @media screen and (max-width: 1024px){ }
- /*Ipad dọc(768 x 1024)*/
@media screen and (max-width: 768px){ }
- /*Tablet nhỏ(480 x 640)*/
@media screen and (max-width: 480px){ }

/*Iphone(480 x 640)*/
@media screen and (max-width: 320px){ }
/*Smart phone nhỏ*/
@media screen and (max-width: 240px){ }



4.5. RWD FRAMEWORK

CÁC THƯ VIỆN HỖ TRỢ RESPONSIVE

- UIkit
- Foundation
- Sekeleton
- Bootstrap
- Bulma
- Semantic UI
- Materialize

