

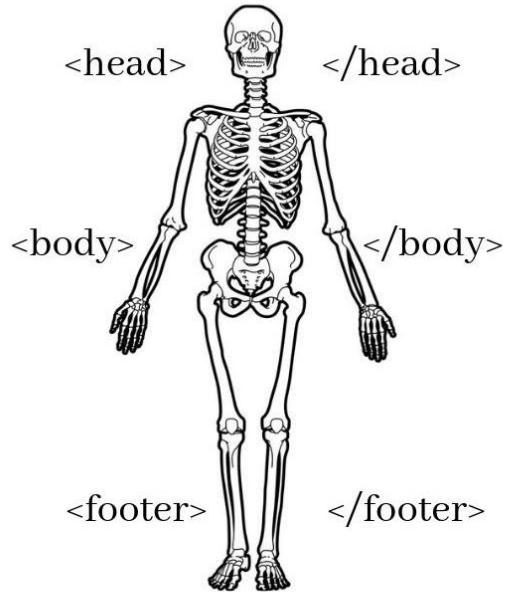
Front-End Essentials

Session 1: CSS Website Layout



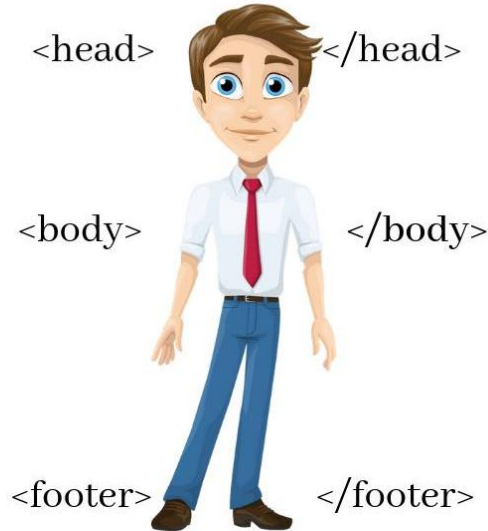
Mối quan hệ giữa HTML & CSS

HTML



Structural Layer

HTML + CSS



Presentational Layer

HTML, CSS và Js

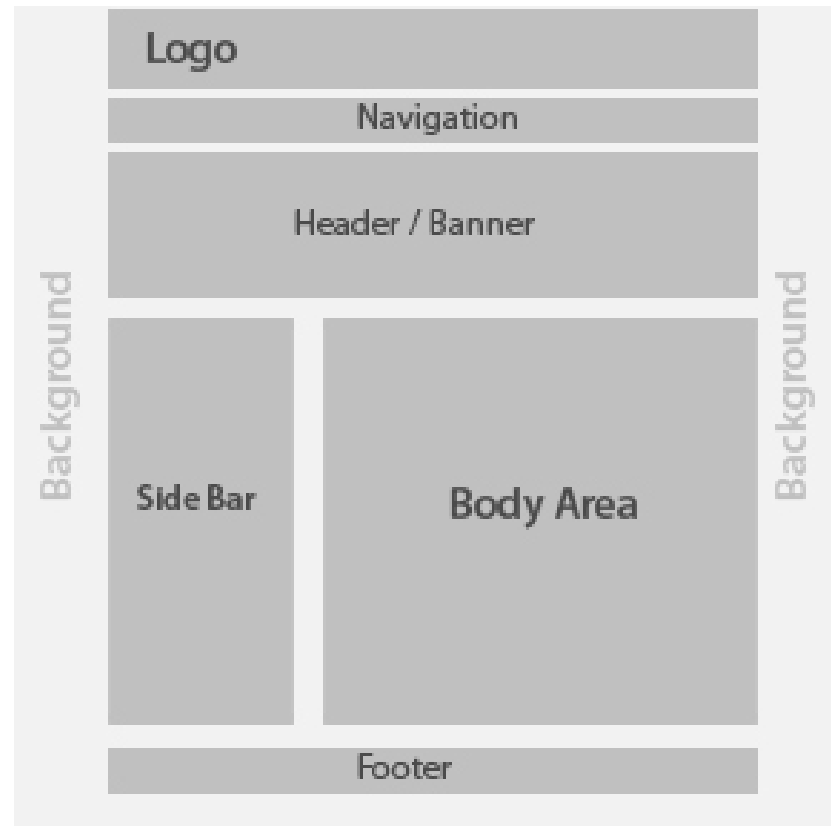
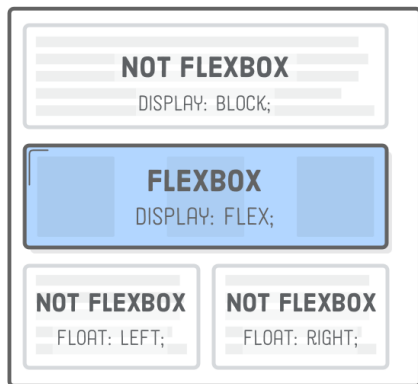


- Phân tích cấu trúc và các phần tử có trong Layout
- Xây dựng cấu trúc page sử dụng thẻ `<div>` và CSS
- Đưa các phần tử HTML vào page theo cấu trúc layout
- Sử dụng các kỹ thuật CSS định dạng các phần tử trong Layout

Case Study 1

■ Cho cấu trúc page như sau:

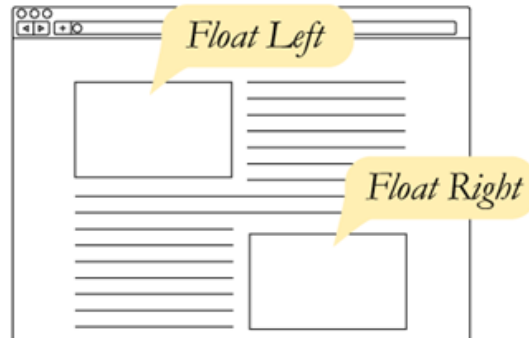
- ✓ Cấu trúc thẻ <div>?
- ✓ Bao nhiêu thẻ <div>?
- ✓ Sử dụng kỹ thuật nào để sắp xếp các vùng?
 - Float
 - Flex
 - Grid



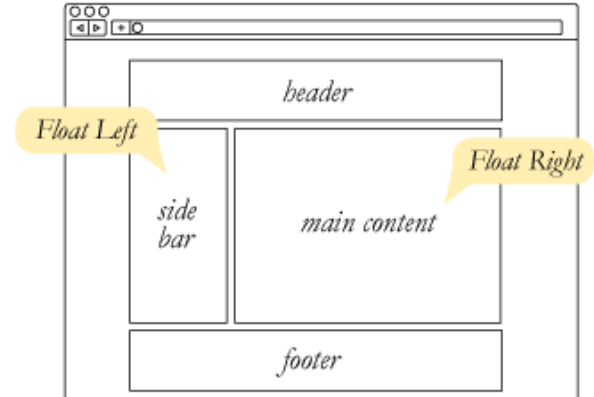
- Float là một kỹ thuật sơ khai dùng để định vị các phần tử, vùng chứa các phần tử web
- Cùng xem xét hình minh họa dưới đây: In ấn và cấu trúc layout Web



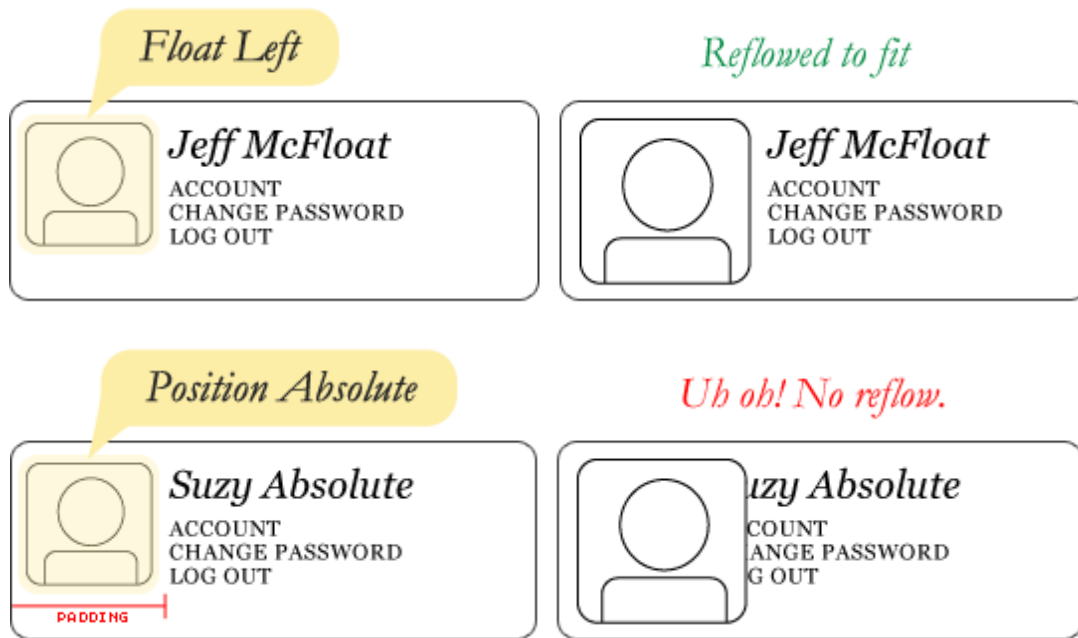
Print Layout



Web Layout

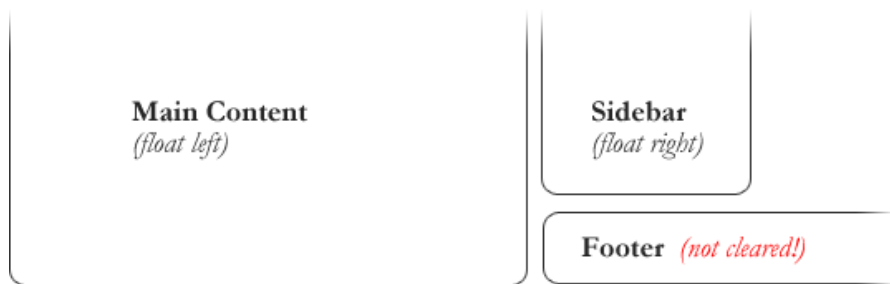


- Float rất hữu dụng trong layout hoặc vùng nhỏ

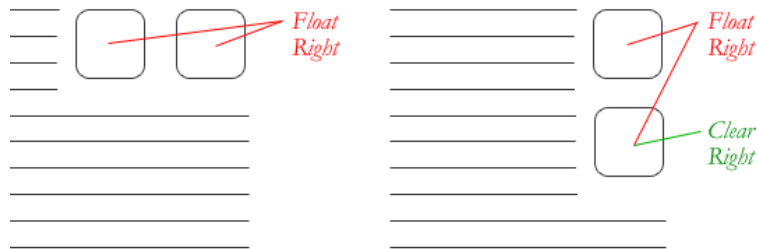


Clearing the Float

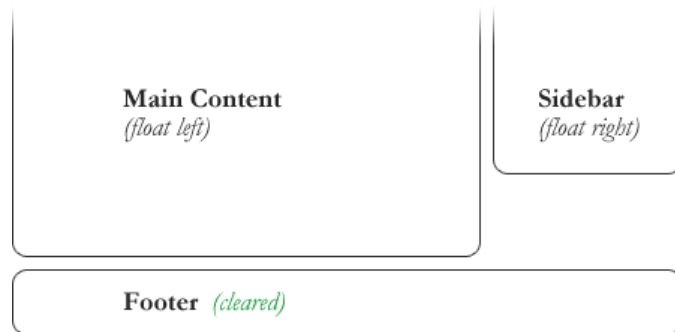
Footer không nằm độc lập trên một hàng



Clear đơn hướng

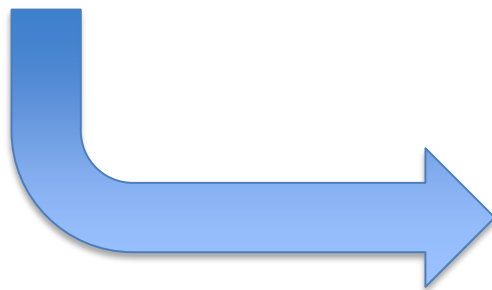
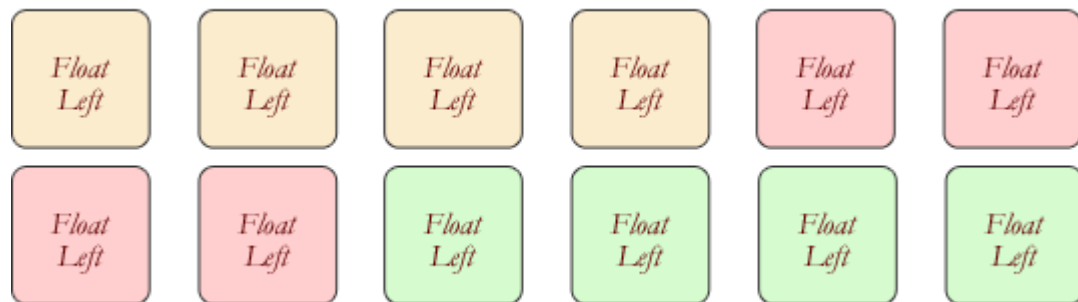


```
#footer { clear: both; }
```

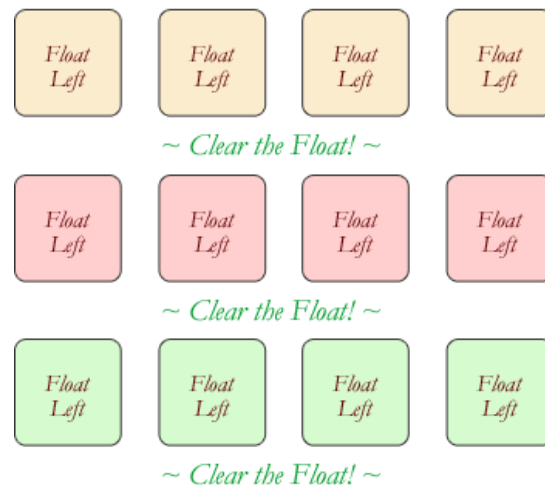


Footer sau khi sử dụng thuộc tính clear

Clearing the Float



Các phân vùng sau khi clear



- Thường bị lỗi ở trình duyệt IE6 trở xuống. Một số trường hợp sau:
 - ✓ **Bị đẩy xuống dưới (pushed down)** khi hình ảnh nhô ra (size lớn hơn) nội dung trong cùng vùng → Cách khắc phục
 - Cách 1: **Tạo các vùng riêng để chứa** các đối tượng (tùy vào cấu trúc layout) và float để sắp xếp chúng theo phương nằm ngang
 - Cách 2: Sử dụng thuộc tính **overflow: hidden** cho đối tượng nhô ra (thừa)
 - ✓ **Hai lần Margin (Double Margin Bug)**: Khi sử dụng margin cho các đối tượng hoặc vùng có sử dụng Float thì xảy ra hiện tượng bị margin 2 lần → Cách khắc phục:
 - Cách 1: Không sử dụng margin cho các vùng float (cùng tên thuộc tính class) mà **tạo thêm class cho vùng float và lập trình margin cho class này**
 - Cách 2: sử dụng **display: inline** cho đối tượng trong vùng
 - ✓ **Bottom Margin Bug** → Cách khắc phục: Padding Bottom cho vùng cha thay vì Margin các vùng con

Flexbox Layout - Flexbox

Section Title

First Article

Lorem ipsum dolor sit amet consectetur adipisicing elit. Modi impedit nulla doloribus dignissimos? Culpa cumque odio itaque incidunt facilis nulla praesentium alias atque voluptatem. Saepe, incidunt magni! Sed, quam dignissimos. Lorem ipsum dolor sit amet consectetur adipisicing elit. Libero blanditiis corporis dolorem unde. Quia quis tempore reprehenderit, nesciunt molestias repudiandae placeat cum, dolores ipsum consequatur esse, expedita nemo dolorem facere.

Second Article

These are fine and they work, but in some ways they are also rather limiting and frustrating. The following simple layout requirements are either difficult or impossible to achieve with such tools, in any kind of convenient, flexible way. These are fine and they work, but in some ways they are also rather limiting and frustrating.

Third Article

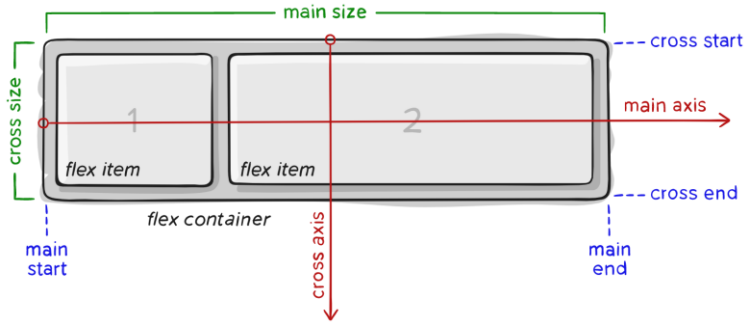
Lorem ipsum dolor sit amet consectetur adipisicing elit. Accusamus, ad animi. Dicta tempore quo eum, rem recusandae ipsa veritatis perspiciatis suscipit mollitia, saepe, ducimus reiciendis rerum corporis neque iure impedit.

```
<div class="heading">
  <h2>Section Title</h2>
</div>
<div class="main-section">
  <div class="column">
    <h3>First Article</h3>
    <p>Lorem ipsum dolor sit amet consectetur adipisicing
elit..</p>
  </div>
  <div class="column diff">
    <h3>Second Article</h3>
    <p>These are fine and they work, but in some ways the
y are also .</p>
  </div>
  <div class="column">
    <h3>Third Article</h3>
    <p>Lorem ipsum dolor sit amet consectetur adipisicing
elit. Accusamus.</p>
  </div>
</div>
```

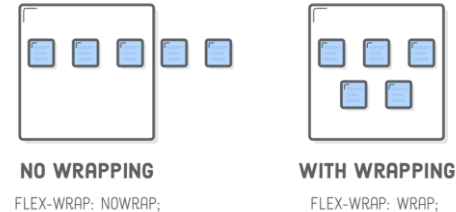
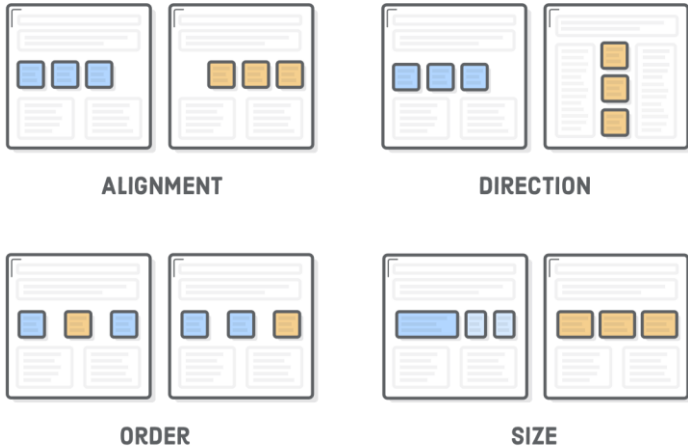
- Flexbox là phương pháp sắp xếp các vùng chứa phần tử (items) theo hàng (row) hoặc cột (column);
- Các phần tử được phân bố đều theo chiều cao và khoảng cách giữa các item theo hàng hoặc cột cho dù chưa biết trước kích thước của vùng chứa
- Canh lề giữa các vùng chứa các phần tử trong vùng cha khi được bố trí theo hàng
- Đảo thứ tự các vùng phần tử theo hàng hoặc cột

```
.main-section{
  display: flex;
  flex-direction: row;
  justify-content: space-around;
}
.column{
  background-color: aqua;
  width: 100%;
  text-align: justify;
  box-sizing: border-box;
  padding: 0.3rem 1rem;
}
.diff{
  margin: 0 0.5rem;
}
```

Flexbox Layout – Flexbox (tt)



- ❖ Có 02 dạng Flexbox chúng ta có thể sử dụng:
 - Flex Container
 - Flex Items
- ❖ Cùng với các thuộc tính điều khiển các Item trong Layout như hình bên

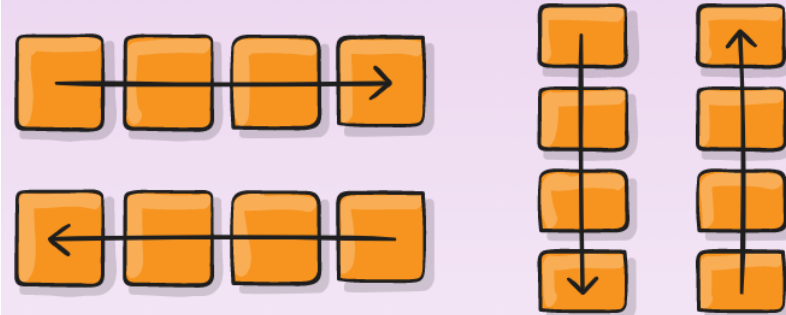


Flexbox: flex-direction

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

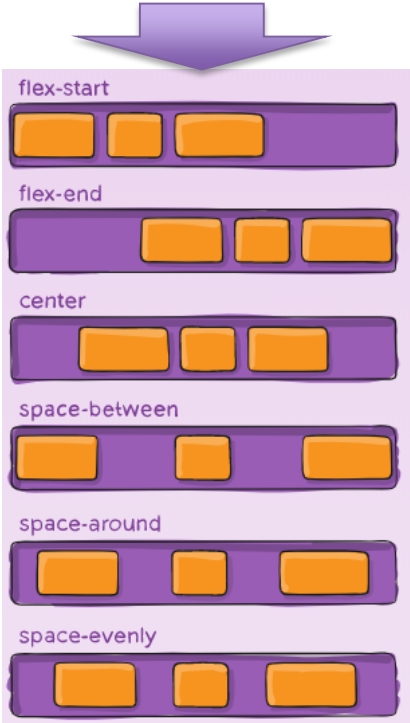
- **row** (default): left to right in ltr; right to left in rtl
- **row-reverse**: right to left in ltr; left to right in rtl
- **column**: same as row but top to bottom
- **column-reverse**: same as row-reverse but bottom to top

flex-direction

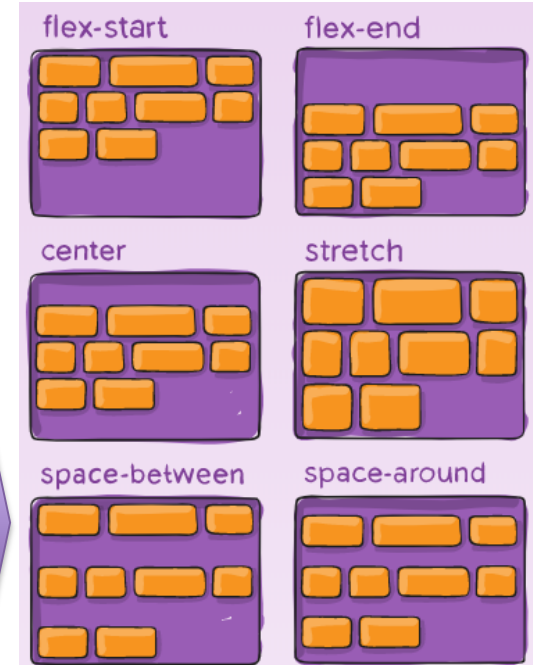


Flexbox: justify-content & align-content

```
.container { justify-content: flex-start | flex-end | center | space-between | space-around | space-evenly | start | end | left | right ... + safe | unsafe; }
```

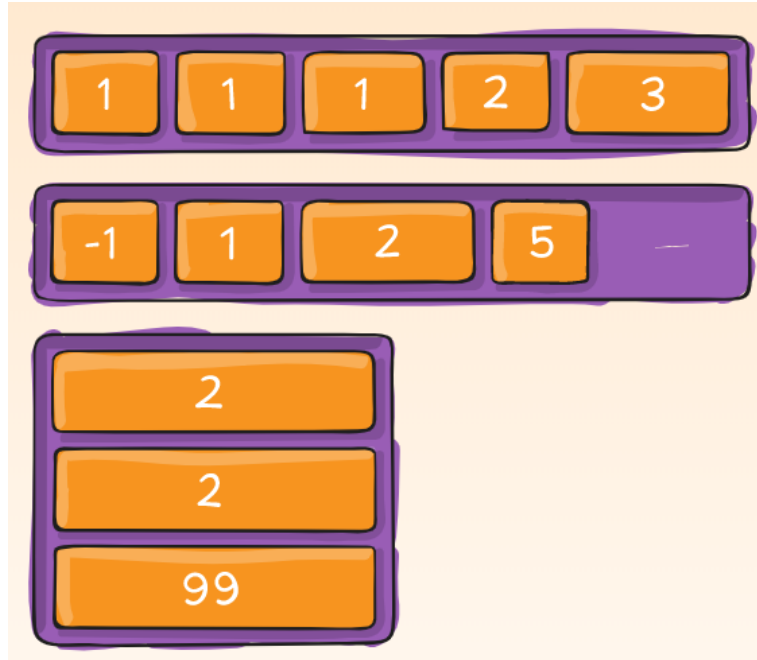


```
.container { align-content: flex-start | flex-end | center | space-between | space-around | space-evenly | stretch | start | end | baseline | first baseline | last baseline + ... safe | unsafe; }
```

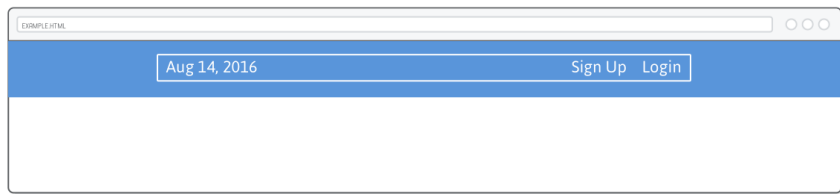


Flexbox: Order

```
.item { order: <integer>; /* default is 0 */ }
```



Flexbox: Grouping flex items



```
<div class='menu'>
  <div class='date'>Aug 14, 2016</div>
  <div class='links'>
    <div class='signup'>Sign Up</div> <!-- This is nested now -->
    <div class='login'>Login</div> <!-- This one too! -->
  </div>
</div>
```



NO GROUPING

(3 FLEX ITEMS)

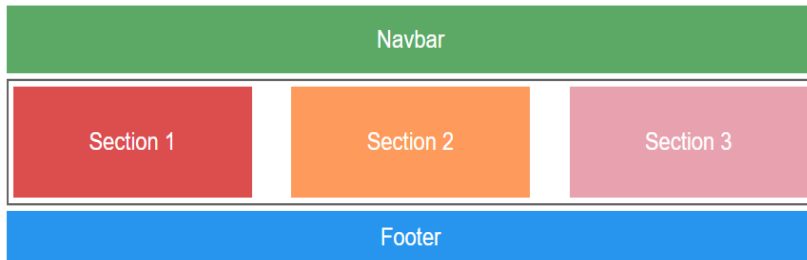


GROUPED ITEMS

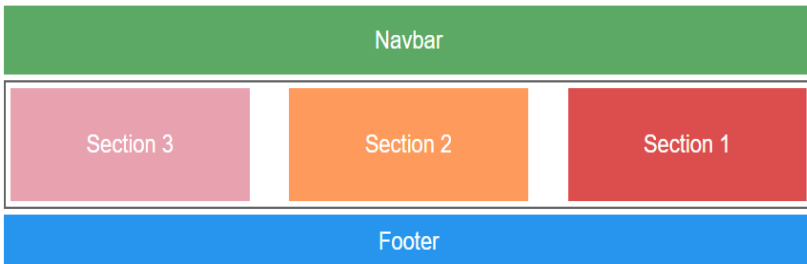
(2 FLEX ITEMS)

```
.links {
  border: 1px solid #fff; /* For debugging */
  display: flex;
  justify-content: flex-end;
}
.login {
  margin-left: 20px;
}
```


Flexbox Example



Thay đổi Order trong Flexbox



```
<div class="wrapper">
  <!-- Menu -->
  <nav>Navbar</nav>
  <!-- Vung chua noi dung -->
  <main class="container">
    <section class="item left">Section 1</section>
    <section class="item center">Section 2</section>
    <section class="item right">Section 3</section>
  </main>
  <!-- Footer -->
  <footer>Footer</footer>
</div>
```

```
* {  
  color: white;  
  text-align: center;  
  font-family: sans-serif;  
  font-size: 22px;  
}
```






```
nav {  
  width: 100%;  
  padding: 0.8rem 0;  
  background-color: #5ca966;  
}
```

```
footer {  
  width: 100%;  
  padding: 0.5rem 0;  
  background-color: #2795ee;  
}
```

```
.container{  
  border: 2px solid #656363;  
  padding: 5px;  
  margin: 5px 0;  
  display: flex;  
  justify-content: space-between;  
}  
section.item {  
  width: 30%;  
  height: 100px;  
  line-height: 100px;  
}
```

```
.left {  
  background-color: #dc4d4d;  
  order: 3;  
}  
.center {  
  background-color: #FD9A5C;  
  order: 1;  
}  
.right {  
  background-color: #e8a2af;  
}
```

- Cung cấp một hệ thống bố cục dựa trên lưới (Grid) với các hàng và cột
- Dễ dàng trong việc thiết kế cấu trúc trang mà không cần sử dụng Float và Position;
- Một cấu trúc lưới bao gồm phần tử cha và một hoặc nhiều phần tử con;
- Có thể tùy chỉnh vị trí và kích thước của từng khối
- Được hỗ trợ trên nhiều nền tảng trình duyệt

				
57.0	16.0	52.0	10	44

Grid Layout

1	2	3
4	5	6
7	8	9

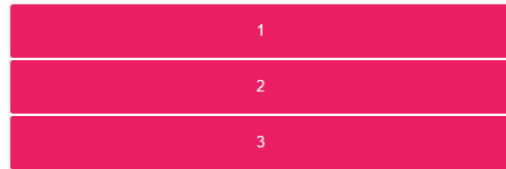
```
<div class="grid-container">  
  <div class="grid-item">1</div>  
  <div class="grid-item">2</div>  
  <div class="grid-item">3</div>  
  <div class="grid-item">4</div>  
  <div class="grid-item">5</div>  
  <div class="grid-item">6</div>  
  <div class="grid-item">7</div>  
  <div class="grid-item">8</div>  
  <div class="grid-item">9</div>  
</div>
```

```
.grid-container {  
  display: grid;  
  grid-template-columns: auto auto auto;  
  background-color: #2196F3;  
  padding: 10px;  
}
```

```
.grid-item {  
  background-color: rgba(255, 255, 255, 0.8);  
  border: 1px solid rgba(0, 0, 0, 0.8);  
  padding: 20px;  
  font-size: 30px;  
  text-align: center;  
}
```

- Các phần tử con bên trong container của một grid phải được thiết lập 1 trong 2 thuộc tính display:
 - ✓ **Grid**: Tạo một lưới dưới dạng **block-level**
 - ✓ **Grid-inline**: Tạo một lưới dưới dạng **inline-level**
- Tất cả các phần tử bên trong grid container mặc định được gọi là grid items

```
.container {  
  display: grid | inline-grid;  
}
```

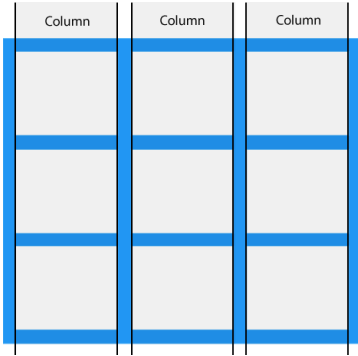


display: grid

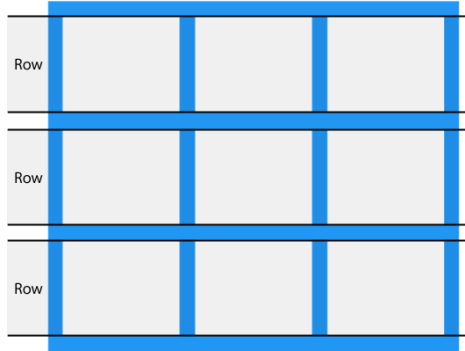


display: inline-grid

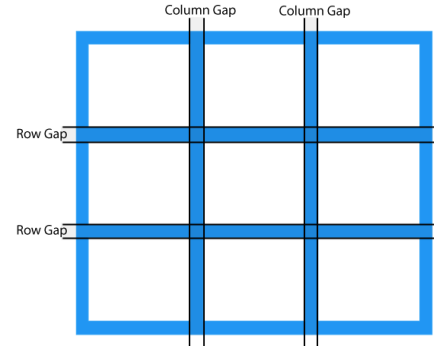
Grid Gap (Gutters)



Grid Columns



Grid Rows



Grid Gaps

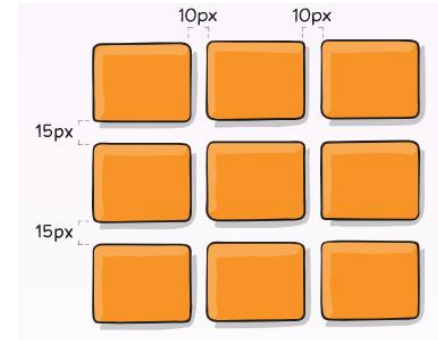
Khoảng hở giữa các hàng và cột gọi là gap

❑ Thiết lập độ lớn của gap chúng ta có thể sử dụng 03 thuộc tính:

`grid-column-gap` → gap giữa các cột

`grid-row-gap` → gap giữa các hàng

`grid-gap` → gap giữa các hàng và cột



`grid-gap: 15px 10px;`

- Một container chứa nhiều items được để trong các cột và hàng
- Thiết lập số lượng cột và hàng trong lưới của container
- Thiết lập kích thước cột và hàng trong lưới của container
- Sử dụng:

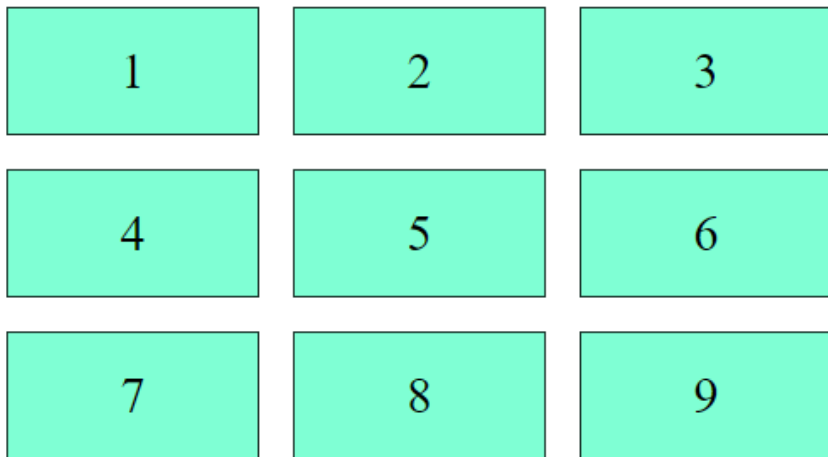
nếu không setup thì mặc định là chia đều nhau

 - ✓ **grid-template-columns** → Xác định số lượng cột và kích thước cột (để giá trị auto → các cột có cùng kích thước width)
 - ✓ **grid-template-rows** → Xác định chiều cao (height) của mỗi row

→ Sử dụng đơn vị fr để tạo ra các độ lớn các cột linh hoạt (Tính toán dựa trên không gian sẵn có trong container):

$$1fr = ((\text{width of grid}) - (3\text{rem}) - (25\% \text{ of width of grid})) / 3$$

Grid Layout – Ví dụ



```
<div class="grid-container">  
  <div class="grid-item">1</div>  
  <div class="grid-item">2</div>  
  <div class="grid-item">3</div>  
  <div class="grid-item">4</div>  
  <div class="grid-item">5</div>  
  <div class="grid-item">6</div>  
  <div class="grid-item">7</div>  
  <div class="grid-item">8</div>  
  <div class="grid-item">9</div>  
</div>
```

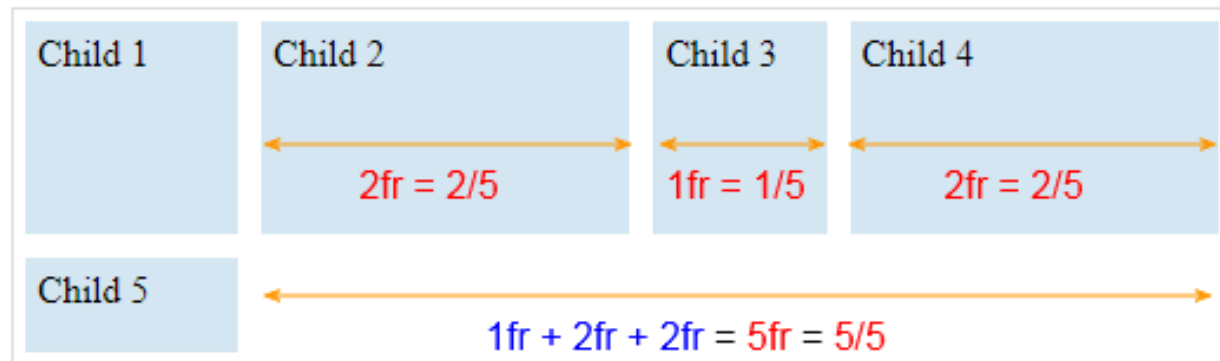
```
.grid-item {  
  background-color: aquamarine;  
  border: 1px solid rgba(0, 0, 0, 0.8);  
  padding: 20px;  
  font-size: 30px;  
  text-align: center;  
}
```

```
.grid-container{  
  display: grid;  
  grid-template-columns: 150px 150px 150px;  
  grid-template-rows: auto auto auto;  
  grid-gap: 20px;  
}
```


CSS grid-template-rows, grid-template-columns

grid-template-rows: 100px 50px;

grid-template-columns: 100px 2fr 1fr 2fr;

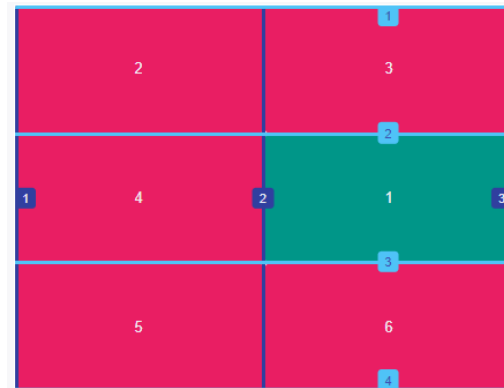


- Định vị các Item thông qua số lượng Line trong Grid
- Các đường Line trong grid là các đường biểu diễn điểm bắt đầu, điểm kết thúc hoặc khoảng cách giữa các hàng và cột
- Mỗi Line bắt đầu từ điểm bắt đầu (start) và theo hướng của lưới, được đánh số thứ tự tăng dần từ 1

```
.item1{  
  grid-row: 2;  
  grid-column: 2/2;  
}
```

Or

```
.item1{  
  grid-row-start: 2;  
  grid-row-end: 3;  
  grid-column-start: 2;  
  grid-column-end: 3;  
}
```



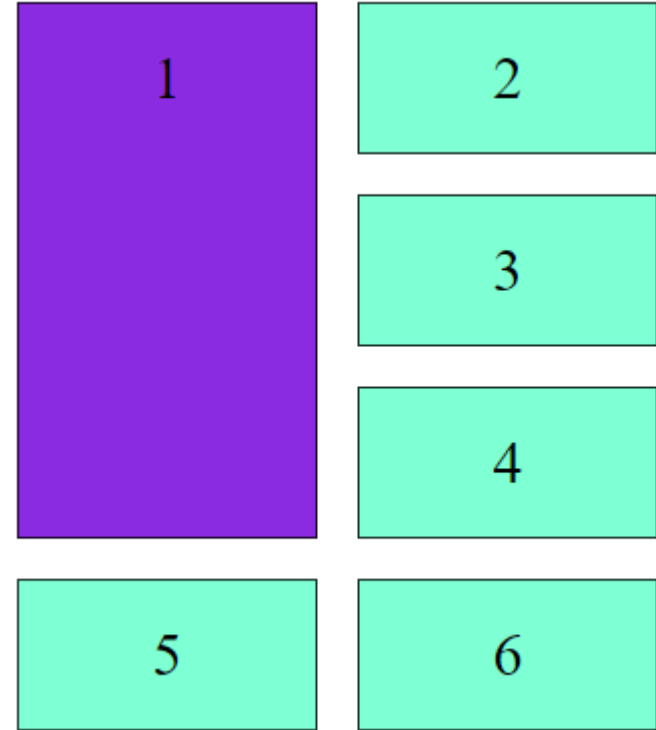
Spanning Items Across Rows and Columns

```
.item1{  
  grid-column-start: 1;  
  grid-column-end: 4;  
}
```

Tham khảo thêm:

<https://learncssgrid.com/>

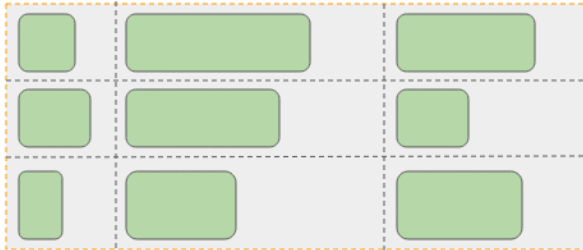
https://www.w3schools.com/css/css_grid_item.asp



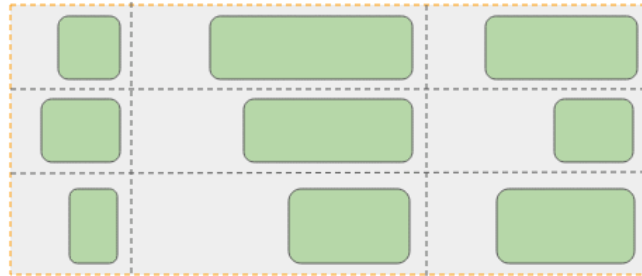
CSS justify-items

```
.container {  
  justify-items: start | end | center | stretch;  
}
```

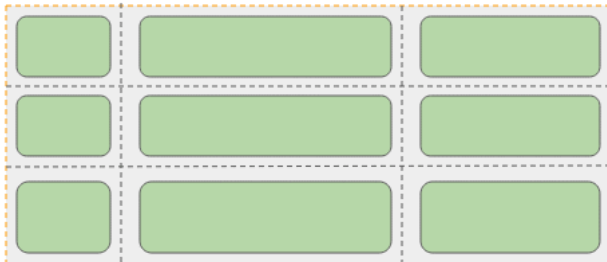
CSS {justify-items: start}



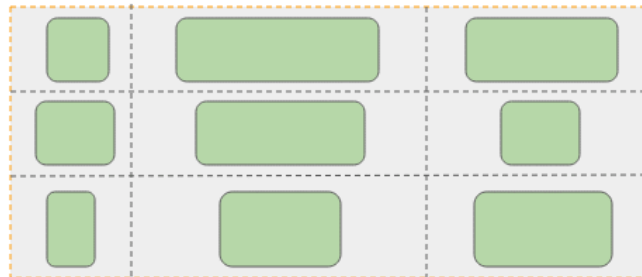
CSS {justify-items: end}



CSS {justify-items: stretch}



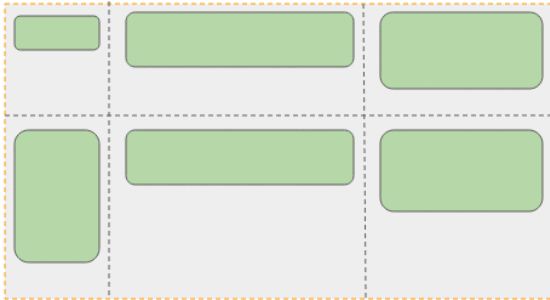
CSS {justify-items: center}



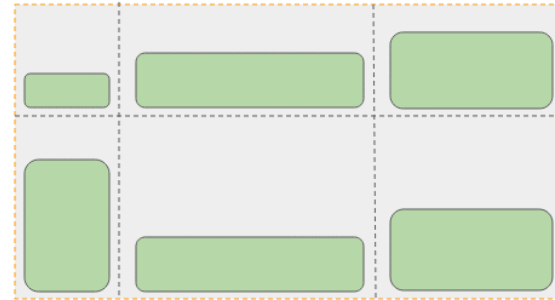
CSS align-items

```
.container {  
  align-items: start | end | center | stretch;  
}
```

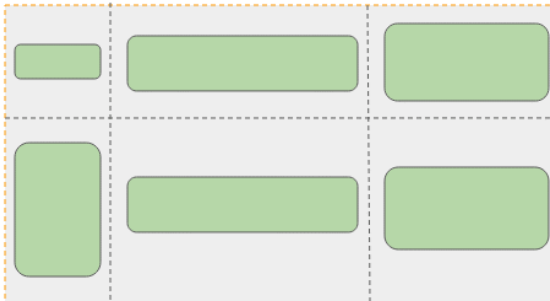
CSS {align-items: start}



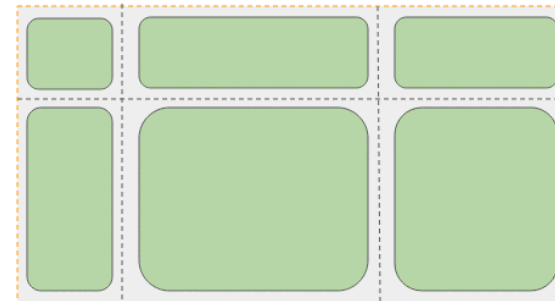
CSS {align-items: end}



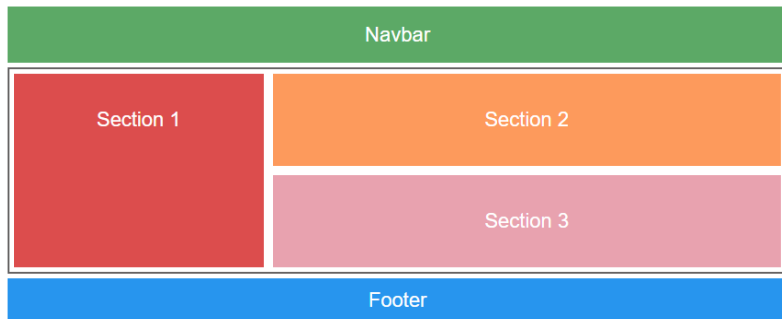
CSS {align-items: center}



CSS {align-items: stretch}



Grid Layout



```
.container {  
  border: 2px solid #656363;  
  padding: 5px;  
  margin: 5px 0;  
  
  display: grid;  
  grid-template-columns: repeat(3, 1fr);  
  grid-gap: 10px;  
  /*Thay doi cau truc*/  
  grid-template-areas:  
    "r c c"  
    "r 1 1"  
}
```









```
<div class="wrapper">  
  <!-- Menu -->  
  <nav>Navbar</nav>  
  <!-- Vung chua noi dung -->  
  <main class="container">  
    <section class="item left">Section 1</section>  
    <section class="item center">Section 2</section>  
    <section class="item right">Section 3</section>  
  </main>  
  <!-- Footer -->  
  <footer>Footer</footer>  
</div>
```

```
.left {  
  background-color: #dc4d4d;  
  grid-area: r;  
}  
.center {  
  background-color: #FD9A5C;  
  grid-area: c;  
}  
.right {  
  background-color: #e8a2af;  
  grid-area: 1;  
}
```

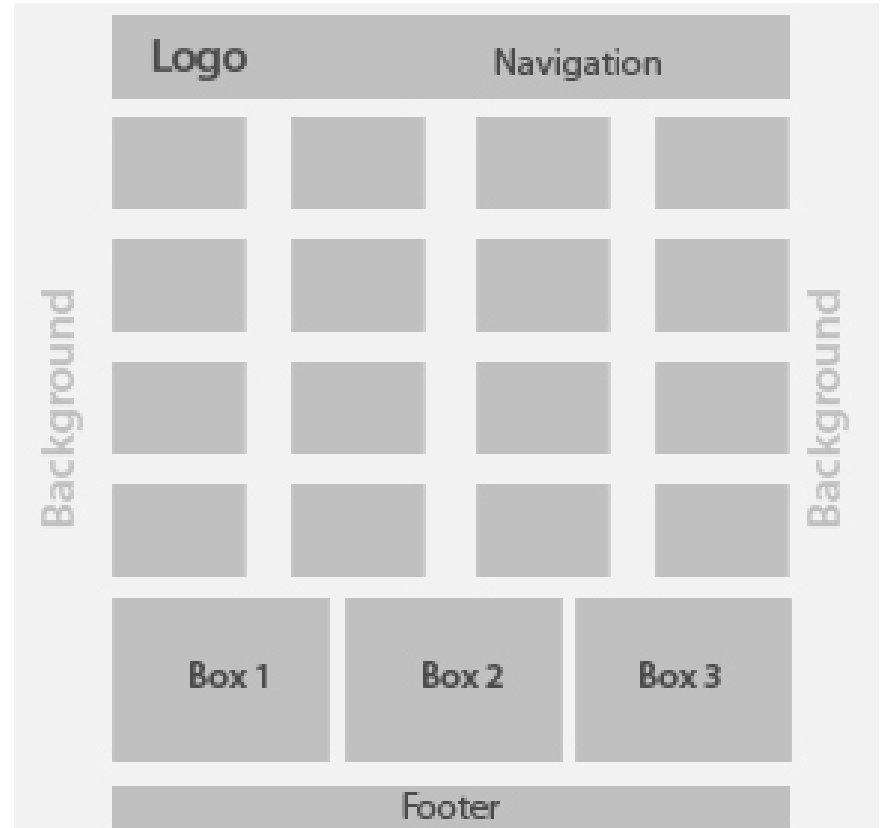
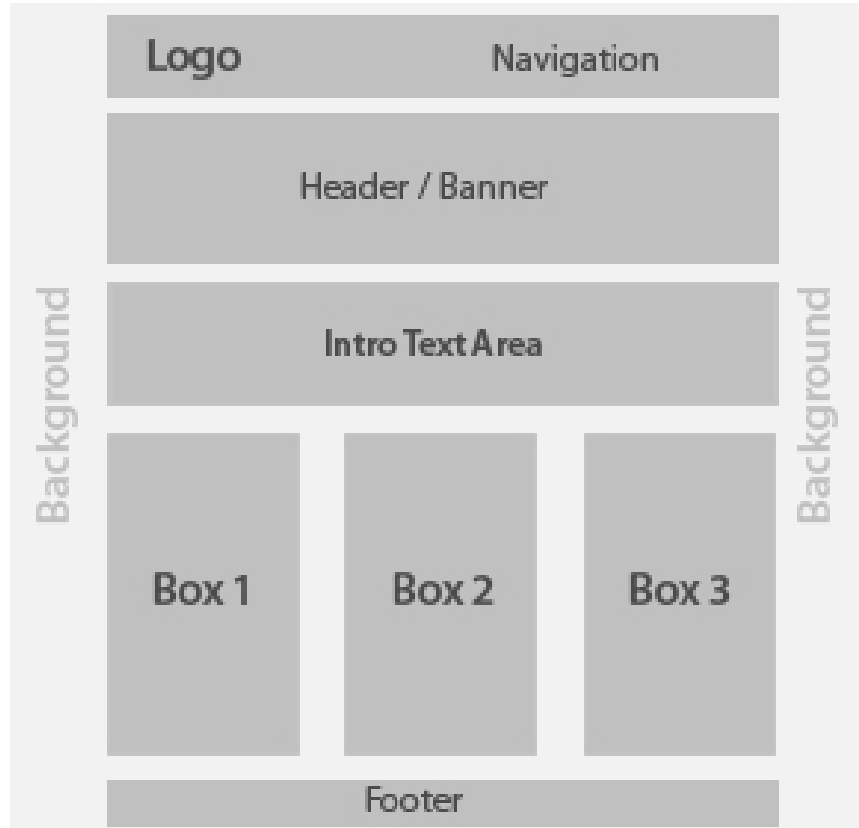
Flexbox: Browser Support

Broken up by "version" of flexbox:

- (new) means the recent syntax from the specification (e.g. `display: flex;`)
- (tweener) means an odd unofficial syntax from 2011 (e.g. `display: flexbox;`)
- (old) means the old syntax from 2009 (e.g. `display: box;`)

							
Chrome	Safari	Firefox	Opera	IE	Edge	Android	iOS
20- (old) 21+ (new)	3.1+ (old) 6.1+ (new)	2-21 (old) 22+ (new)	12.1+ (new)	10 (tweener) 11+ (new)	17+ (new)	2.1+ (old) 4.4+ (new)	3.2+ (old) 7.1+ (new)

Các Case Study tương tự Case study 1



Front-End Essentials

Session 2: Web Responsive



(a) Mobile-first approach



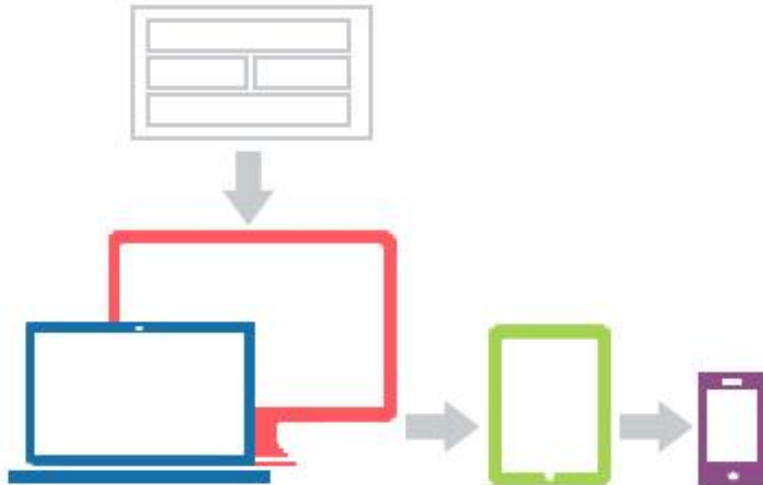
(b) Traditional approach



Sự khác biệt giữa Responsive và Adaptive

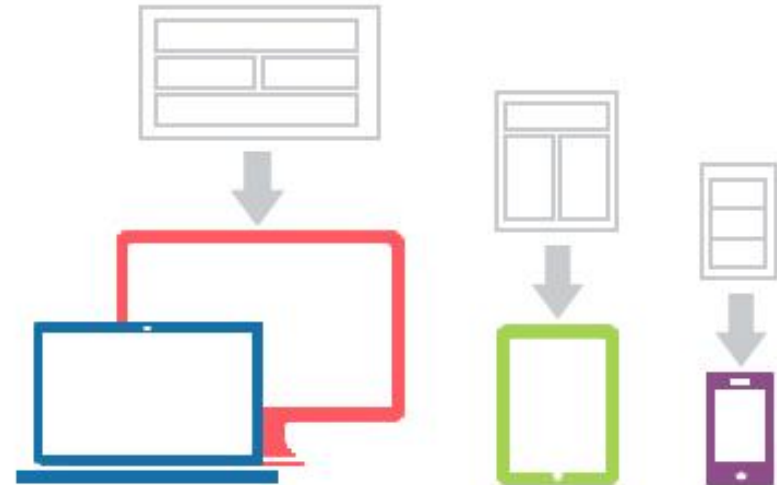
RESPONSIVE

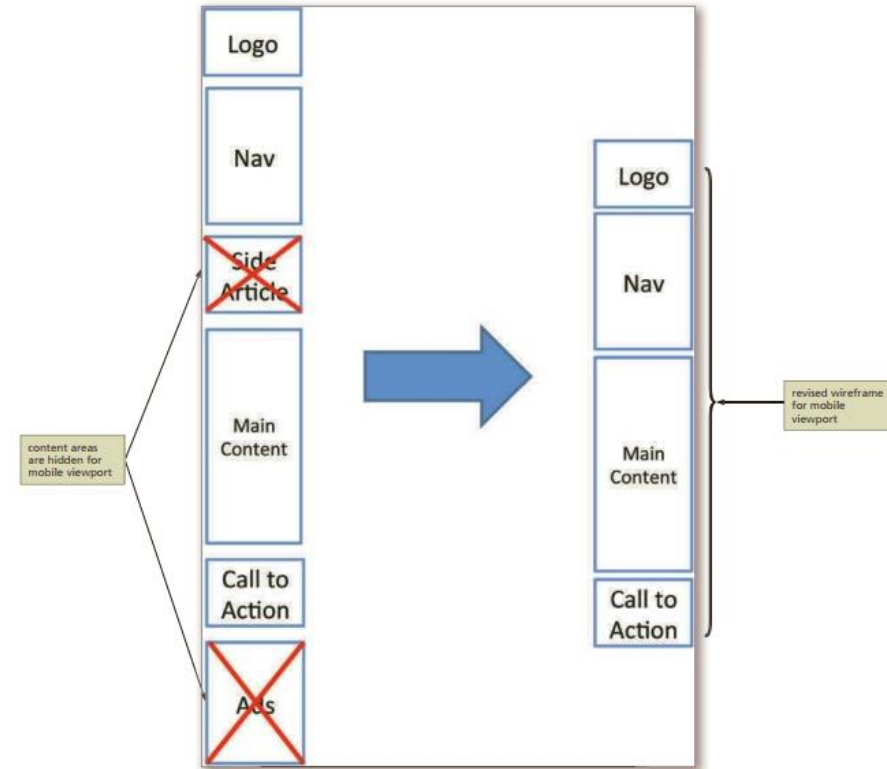
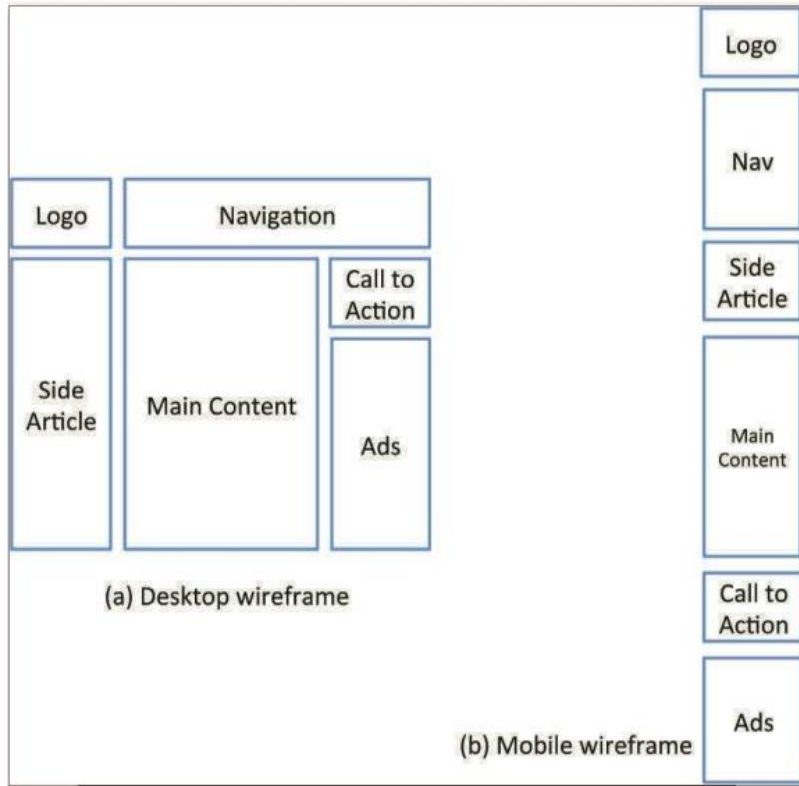
Universal design that reflows across displays



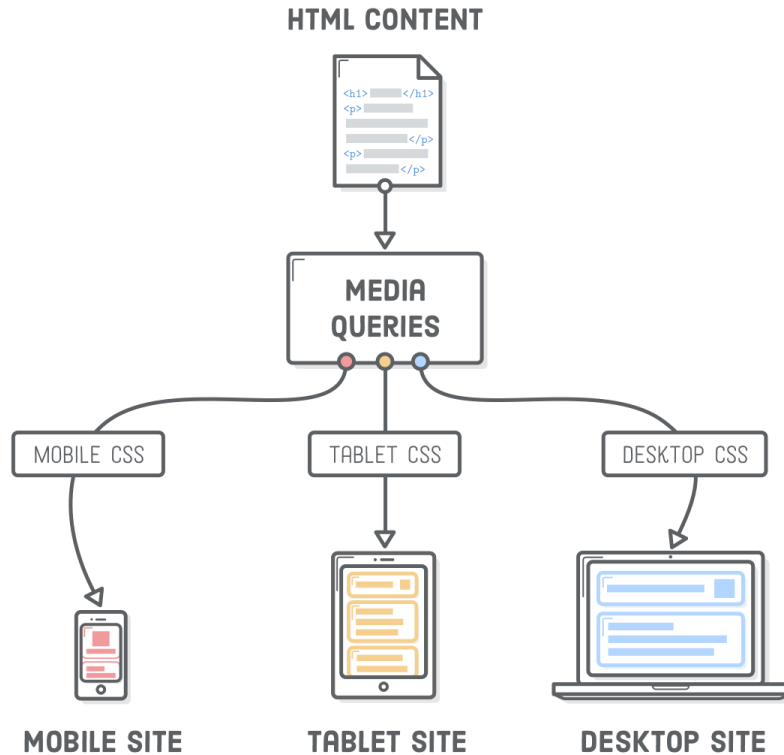
ADAPTIVE

Generates templates which are optimized and unique for each device class

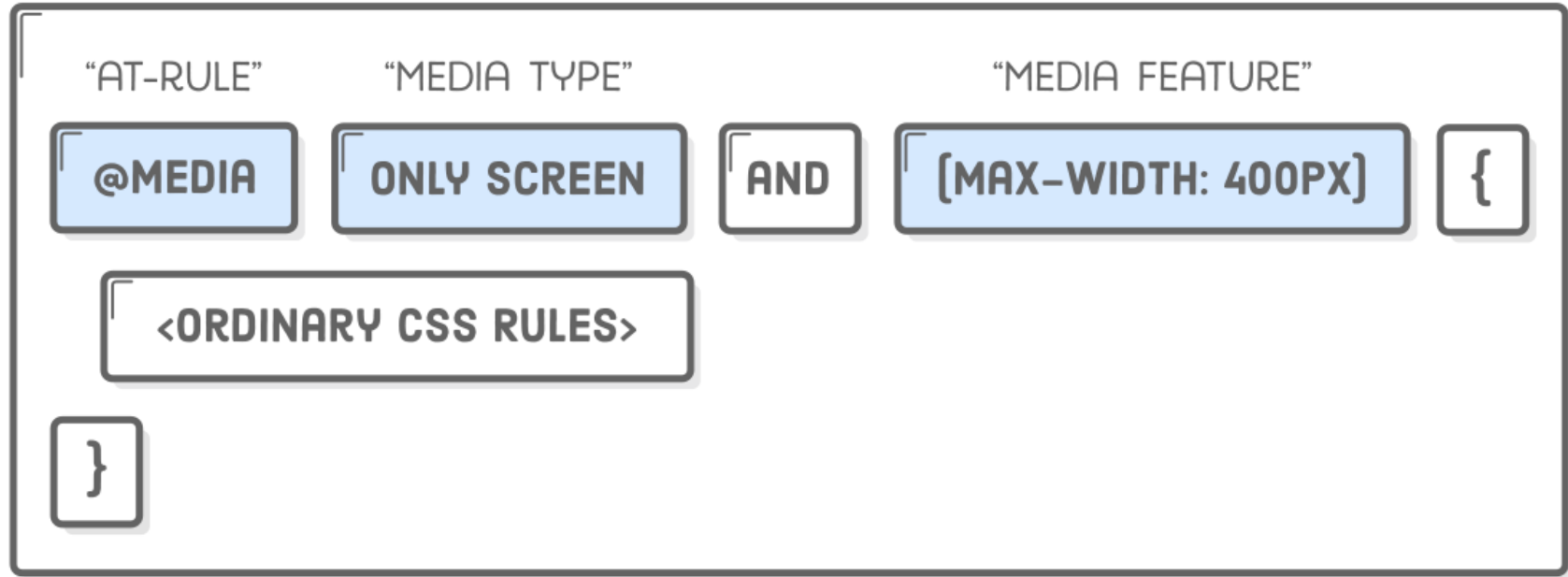




Responsive Design



“MEDIA QUERY”



Media Query

```
/* Mobile Styles */
@media only screen and (max-width: 400px) {
  body {
    background-color: #F09A9D; /* Red */
  }
}

/* Tablet Styles */
@media only screen and (min-width: 401px) and (max-width: 960px) {
  body {
    background-color: #F5CF8E; /* Yellow */
  }
}

/* Desktop Styles */
@media only screen and (min-width: 961px) {
  body {
    background-color: #B2D6FF; /* Blue */
  }
}
```



MOBILE

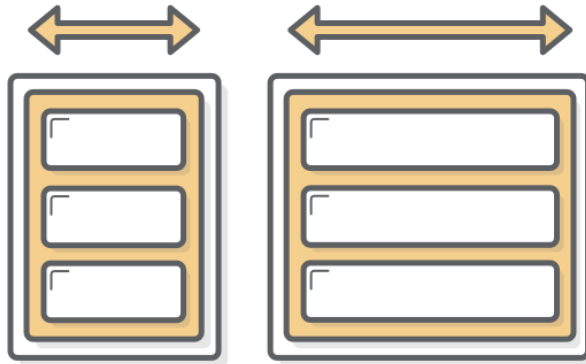


TABLET

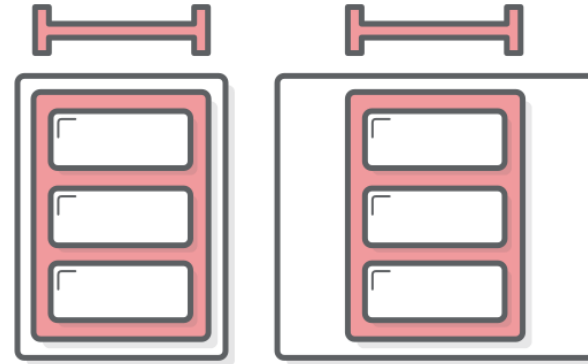


DESKTOP

Fluid and fixed-width

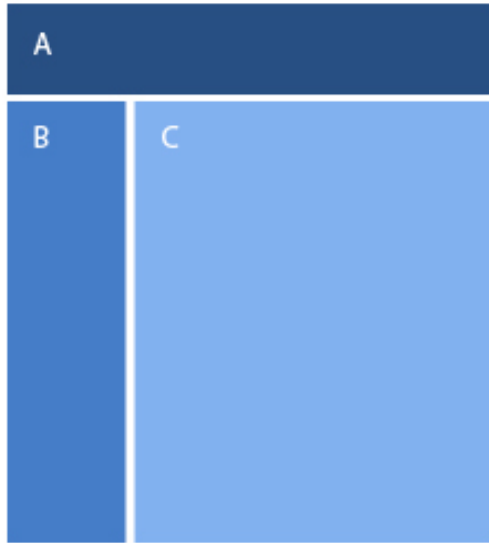


FLUID LAYOUT

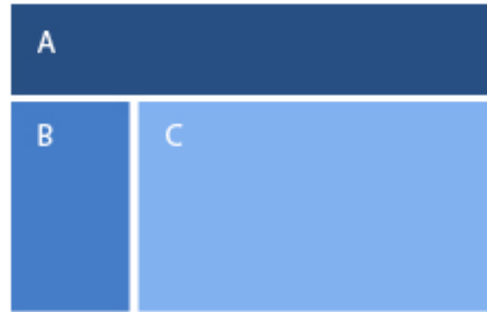


FIXED-WIDTH LAYOUT

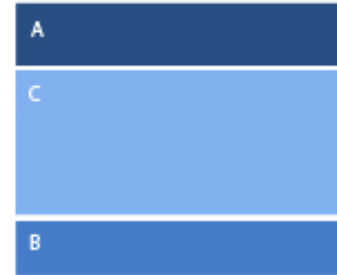
Responsive - Exercise



Desktop



Tablet Landscape

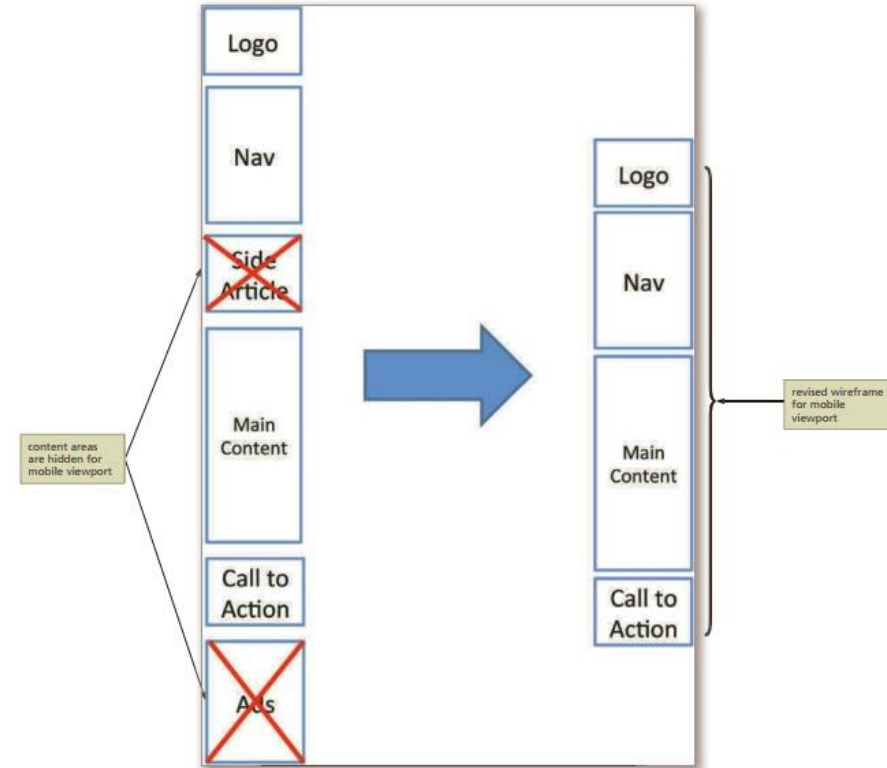
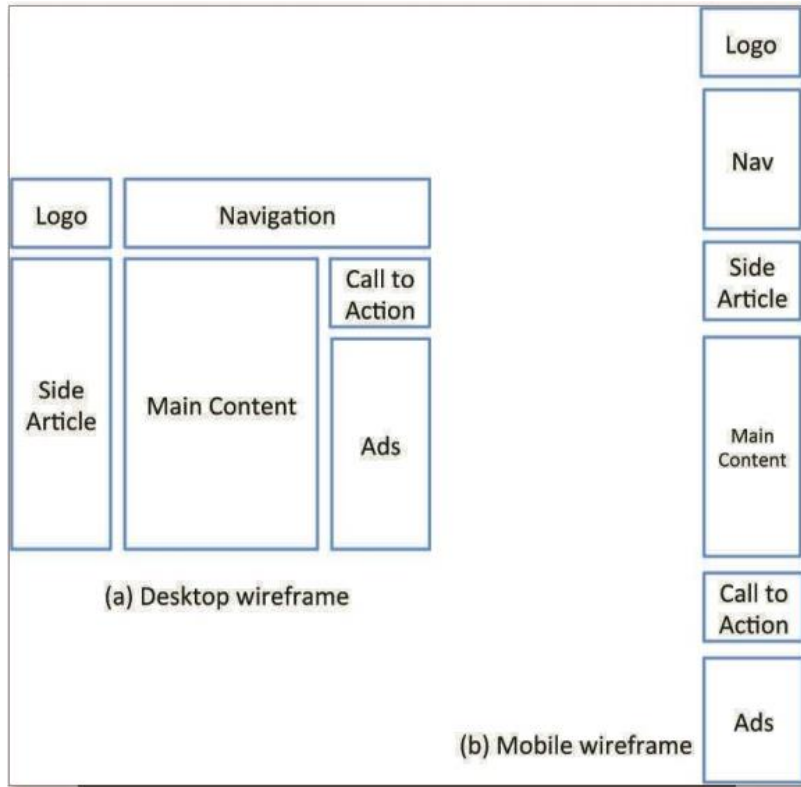


Tablet Portrait



Mobile

Responsive: Ẩn vùng



Good Job

Thank you

