



**FPT POLYTECHNIC**

# **THIẾT KẾ WEB ĐÁP ỨNG MỌI THIẾT BỊ**

**BÀI 4:**

**THIẾT KẾ WEB ĐÁP ỨNG SỬ  
DỤNG GRID CSS**

[www.poly.edu.vn](http://www.poly.edu.vn)

- ◎ Kết thúc bài học này bạn có khả năng
  - Hiểu được kỹ thuật chia bố cục bằng Grid
  - Ứng dụng kết hợp Grid và Media Query để thiết kế web đáp ứng trên các thiết bị
  - Nắm được ưu nhược điểm của Grid





## Phần I: Dàn bố cục sử dụng Grid CSS

- ❖ Các thành phần chính của Grid
- ❖ Các thuộc tính của grid-container



## Phần II: Dàn bố cục sử dụng Grid CSS (tt)

- ❖ Các thuộc tính của grid-item
- ❖ Kết hợp Grid với Media Query

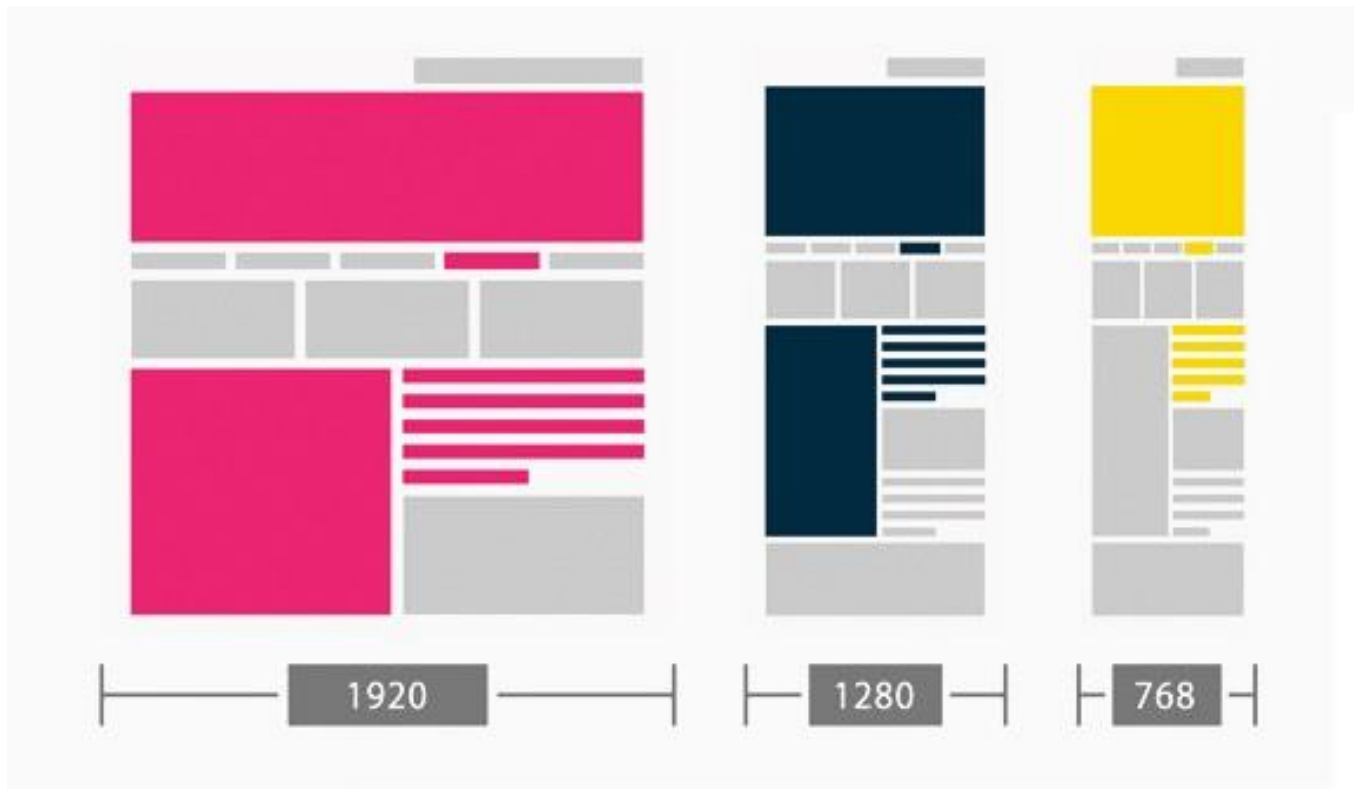


## BÀI 4: **THIẾT KẾ WEB ĐÁP ỨNG SỬ DỤNG GRID CSS**

### **PHẦN I: DÀN BỐ CỤC SỬ DỤNG GRID CSS**

- ❑ Các ứng dụng web ngày nay giao diện ngày càng phức tạp nên việc chỉ sử dụng Flexbox thực sự vẫn chưa đáp ứng tốt nhất. Hơn nữa, Flexbox chỉ hỗ trợ tạo bố cục 1 chiều (ngang hoặc dọc). Việc bố trí trang web 2 chiều vẫn gặp khó khăn.
- ❑ CSS Grid ra đời giúp làm được nhiều layout phức tạp một cách dễ dàng, nhanh chóng tuy nhiên việc sử dụng vẫn còn chưa rộng rãi.

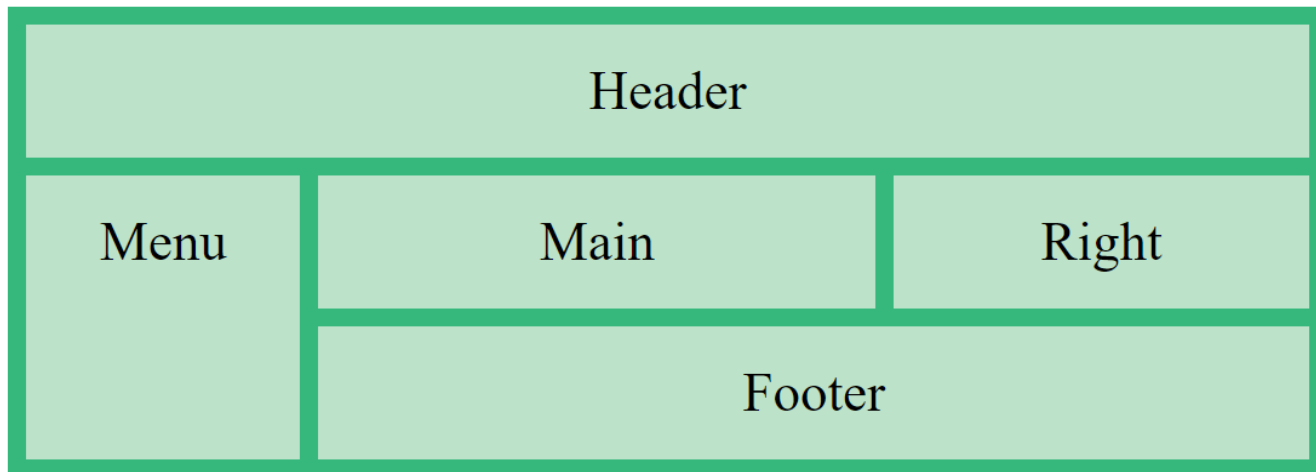
# TỔNG QUAN VỀ CSS GRID



Bố cục Grid là một hệ thống bố cục 2 chiều (x,y) được dùng trong thiết kế UI.

Theo định nghĩa từ Mozilla, grid (lưới) là một tổ hợp của những đường ngang và dọc cắt nhau – một nhóm xác định các cột và nhóm kia xác định các hàng.

Các phần tử có thể được đặt lên lưới, dựa vào các đường hàng và cột này.



Hầu hết các trình duyệt hiện đại ngày nay đều có hỗ trợ Grid. Vì vậy việc áp dụng Grid khi thiết kế sẽ sớm trở thành xu thế chung.

## Browser Support for CSS Grid

### | Desktop



### | Mobile/Tablet





CSS Grid được cấu thành từ hai thành phần chính là **grid-container** và **grid-item**. Grid-container bao gồm các grid-item, được đặt bên trong các cột và hàng.

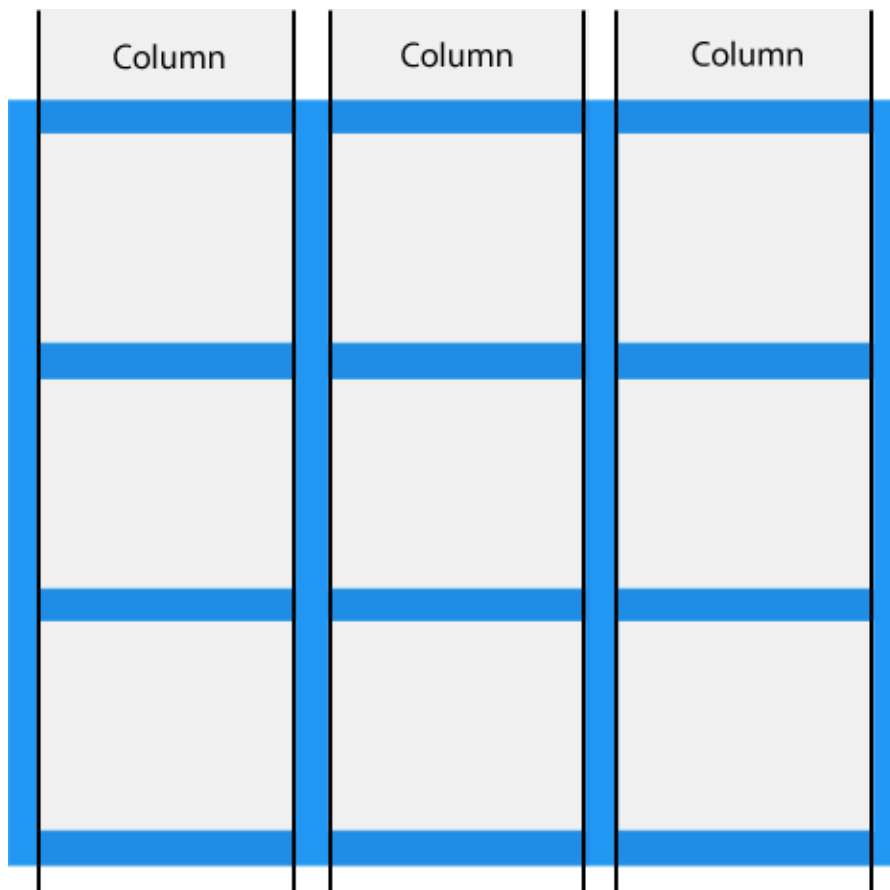
Ví dụ:

```
<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>
```

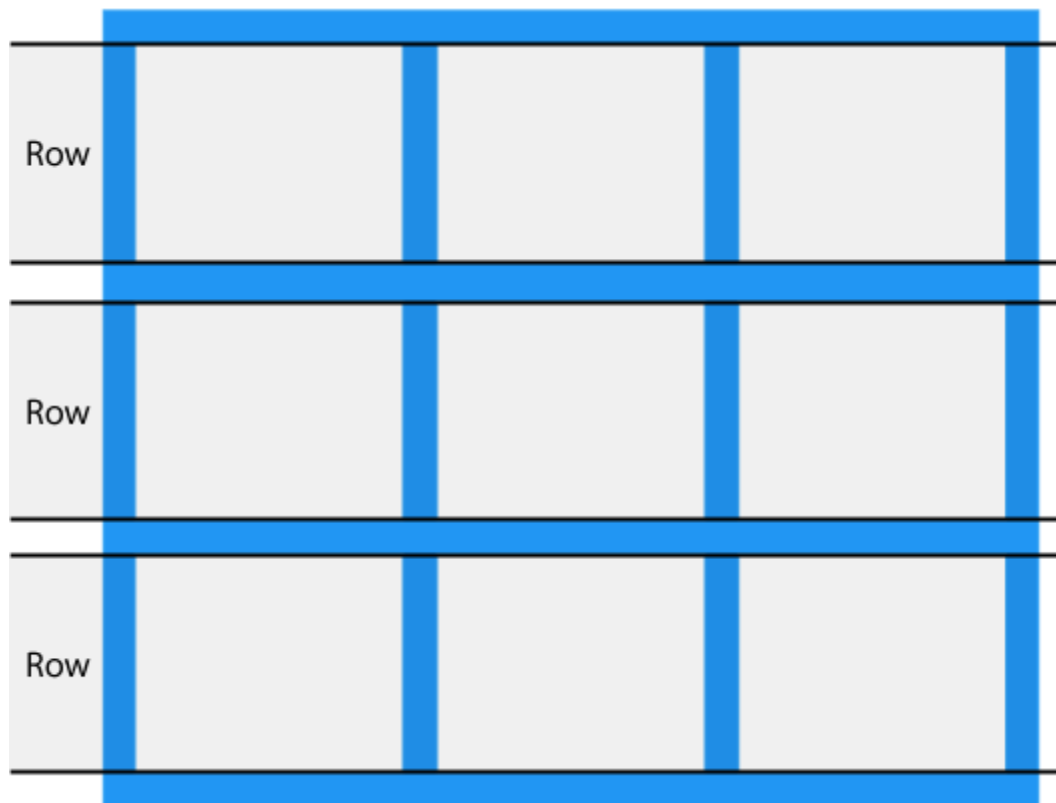
Trước khi đi chi tiết hơn về **grid-container** và **grid-item**, cần làm quen trước một số thuật ngữ quan trọng thường được sử dụng trong CSS Grid:

- ☐ Grid Column
- ☐ Grid Row
- ☐ Grid Gap
- ☐ Grid Line

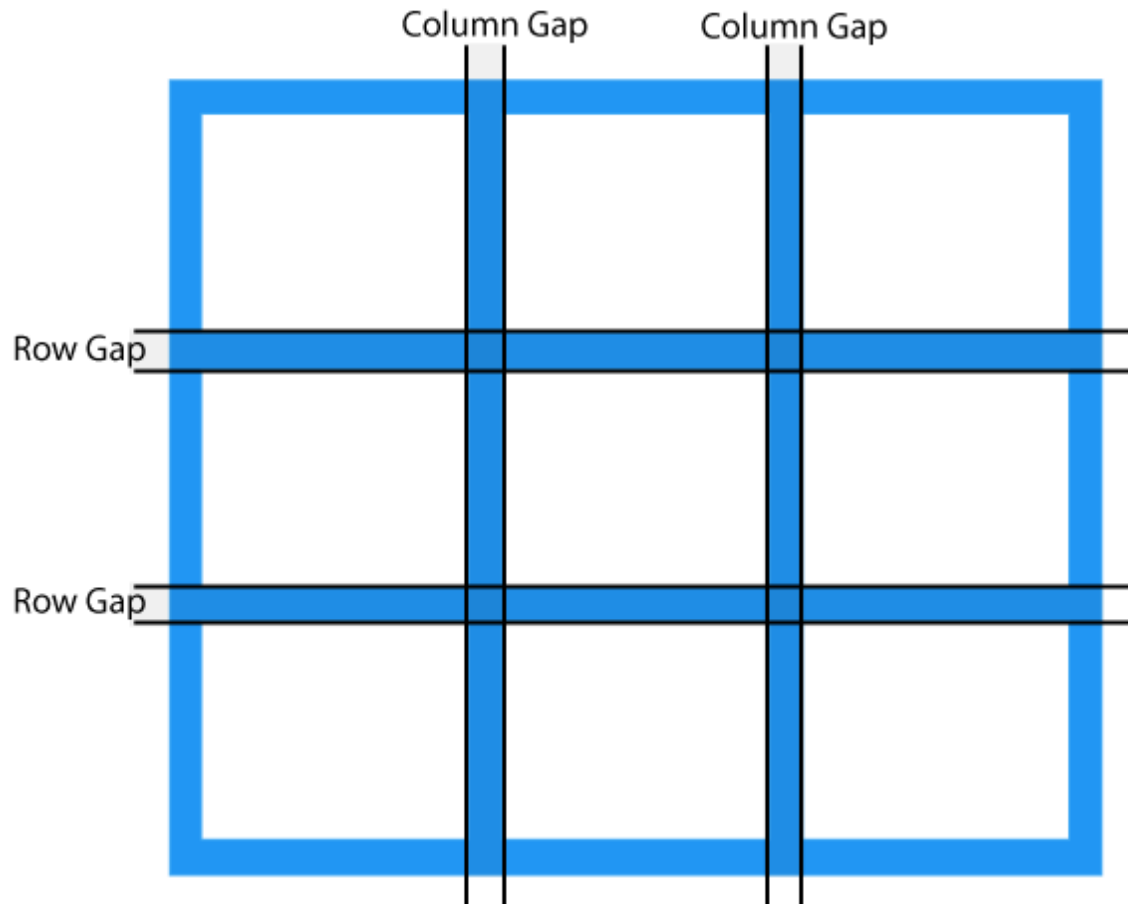
Các đường thẳng đứng của một bố cục gọi là **Grid Column**



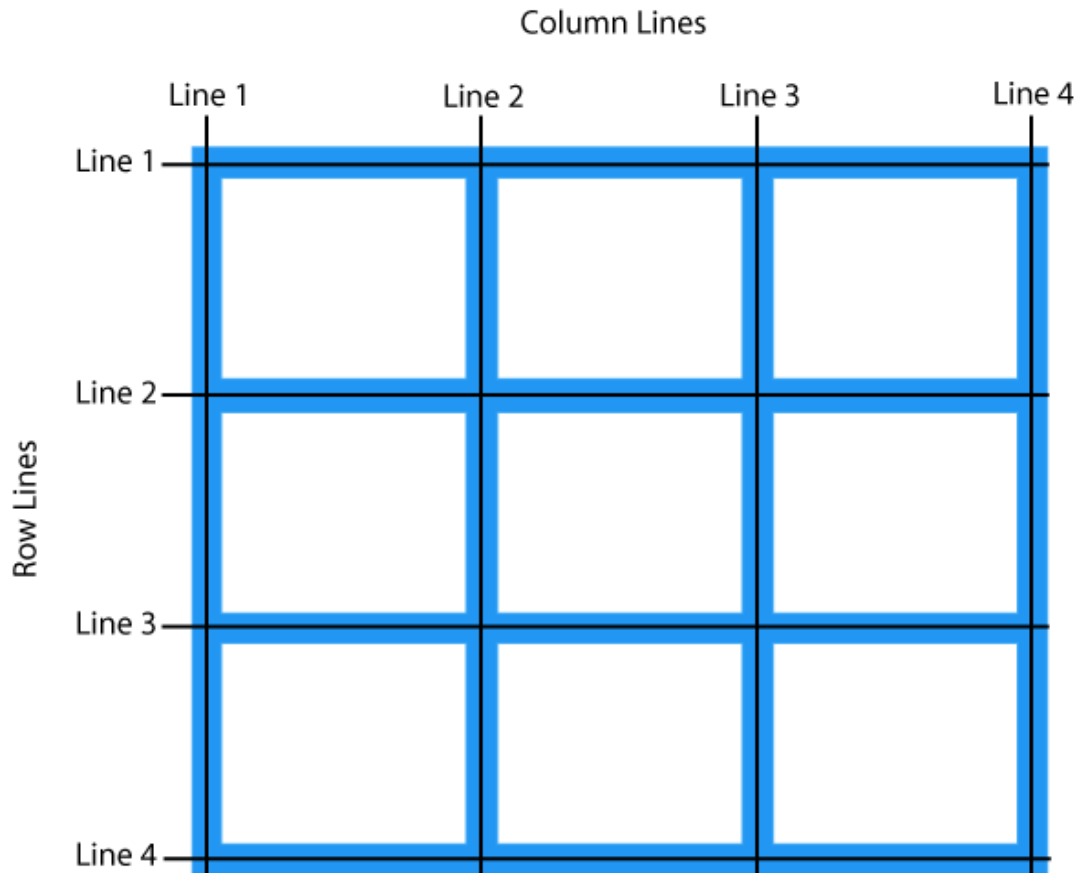
Các đường ngang của một bố cục gọi là **Grid Row**



Khoảng trống giữa mỗi cột / hàng gọi là **Grid Gap**



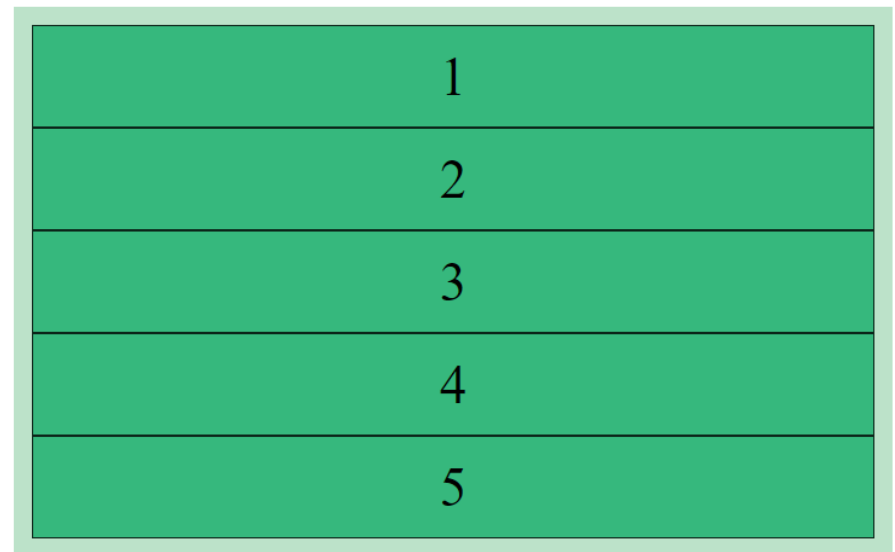
Các dòng kẻ ngang và dọc, phân chia layout thành các hàng và cột được gọi là **Grid Line**.



Để bắt đầu sử dụng với CSS Grid, cần khai báo thông qua thuộc tính **display**

Ví dụ:

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



*Lúc này chỉ mới định dạng cho **grid-container** chứ chưa định dạng cho các item bên trong nên các **grid-item** sẽ xếp chồng lên nhau.*

Để các item bên trong sắp xếp mượt mà linh hoạt hơn, cần xác định số hàng và số cột của grid-container thông qua hai thuộc tính:

- ❑ `grid-template-row`
- ❑ `grid-template-column`



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>
```

```
.grid-container {
  display: grid;
  grid-template-columns: 200px 50px 100px;
  grid-template-rows: 100px 50px;
  background-color: #BCE2C9;
  padding: 10px;
}
```

3 giá trị của grid-template-columns tương ứng với **độ rộng** của 3 cột  
2 giá trị của grid-template-rows tương ứng với **chiều cao** của 2 hàng  
(Nếu muốn chỉ định 3 cột có độ rộng bằng nhau thì sử dụng giá trị **auto**)

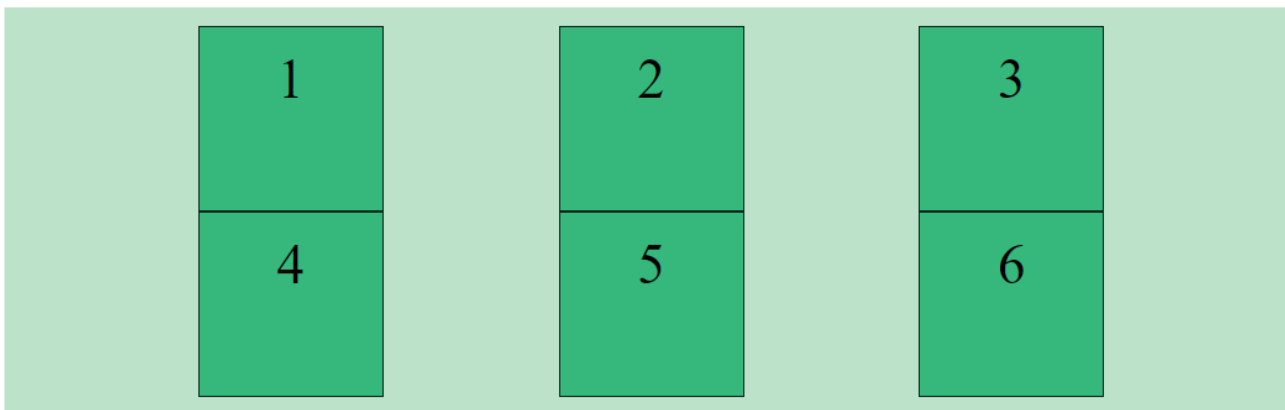
1	2	3
4	5	6

Tương tự Flexbox, CSS Grid cũng có hai thuộc tính là **justify-content** và.

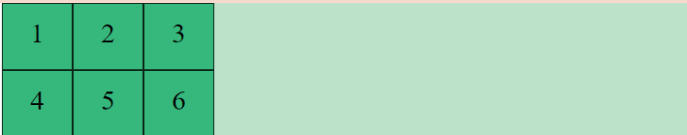
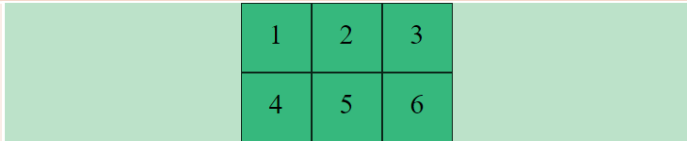

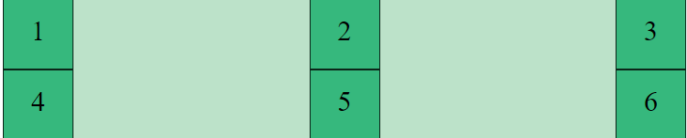
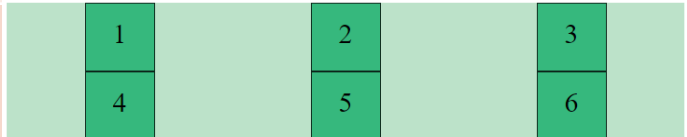
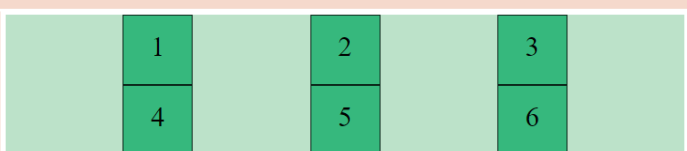
Thuộc tính justify-content được **align-content** sử dụng để căn chỉnh toàn bộ nội dung item bên trong vùng chứa container.

Ví dụ:

```
justify-content: space-evenly;
```



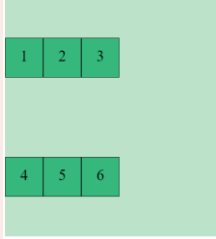
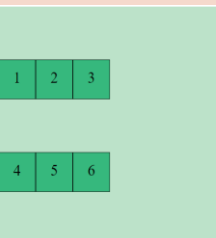
Các giá trị của thuộc tính **justify-content**:

Giá trị	Mô tả
start	
center	
end	
space-between	
space-around	
space-evenly	

**Align-content** được dùng để căn chỉnh theo chiều dọc toàn bộ nội dung bên trong vùng chứa.

Các giá trị của thuộc tính **align-content**:

Giá trị	Mô tả
start	
center	
end	

Giá trị	Mô tả
space-between	
space-around	
space-evenly	

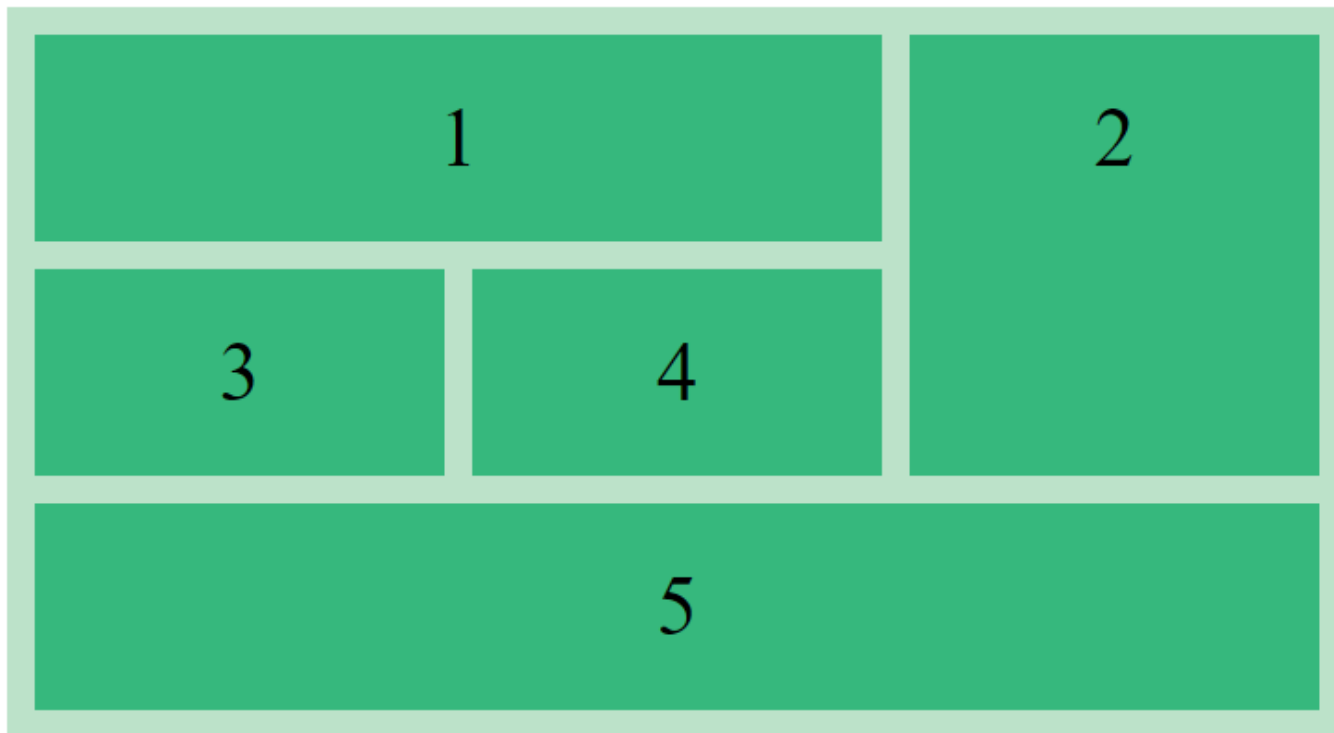
demo

## BÀI 4: **THIẾT KẾ WEB ĐÁP ỨNG SỬ DỤNG GRID CSS**

### **PHẦN II: DÀN BỐ CỤC SỬ DỤNG GRID CSS (TT)**

# **PHẦN II: GRID-ITEM**

Để xác định vị trí và thay đổi kích thước của item, chúng ta sẽ phải sử dụng thêm thuộc tính **grid-column** và **grid-row**





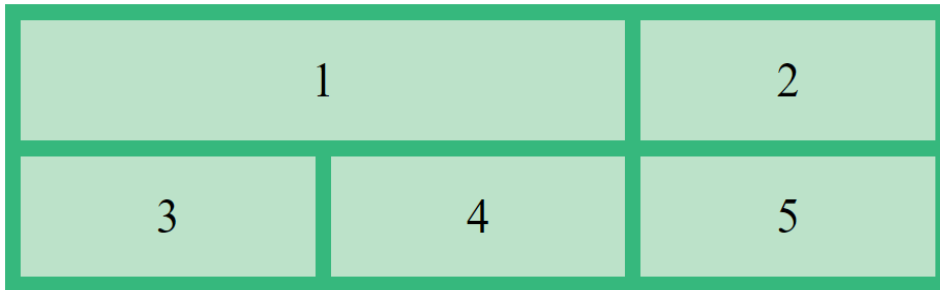
Grid-column dùng để xác định vị trí bắt đầu và kết thúc của một item bất kỳ.

Ví dụ:

```
.item1 {  
  grid-column: 1 / 3;  
}
```



```
.item1 {  
  grid-column-start: 1;  
  grid-column-end: 3;  
}
```



Lưu ý: Thuộc tính **grid-column** là một thuộc tính viết tắt cho các thuộc tính **grid-column-start** và **grid-column-end**.

Với ví dụ trên, **.item1** bắt đầu từ grid line thứ 1 đến grid line thứ 3 (tương ứng 2 cột)

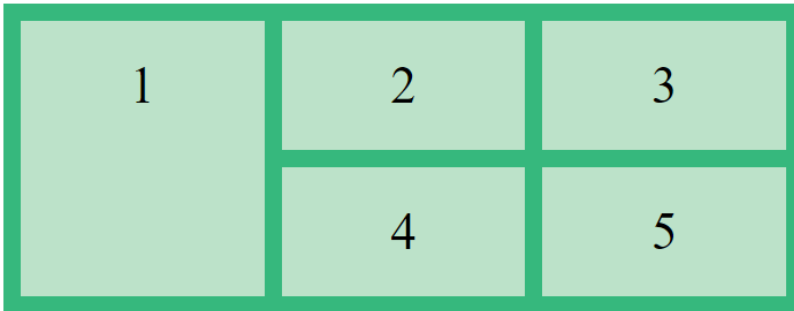
**Grid-row** dùng để xác định vị trí bắt đầu và kết thúc của một item bất kỳ theo hàng dọc.

Ví dụ:

```
.item1 {  
  grid-row: 1 / 3  
}
```



```
.item1 {  
  grid-row-start: 1;  
  grid-row-end: 3;  
}
```



Lưu ý: Với ví dụ trên, **.item1** bắt đầu từ grid line thứ 1 đến grid line thứ 3 (tương ứng 2 hàng)

**Grid-area** có thể được sử dụng như một thuộc tính viết tắt cho thuộc tính **grid-row-start**, **grid-column-start**, **grid-row-end** và **grid-column-end**

Ví dụ:

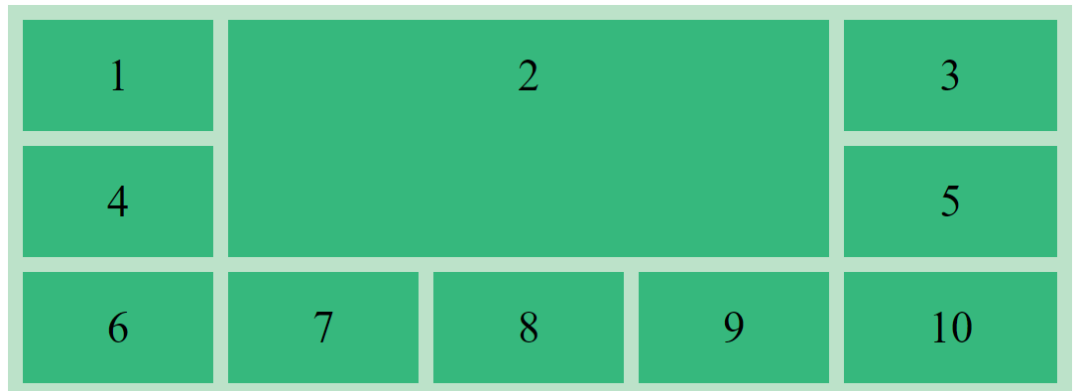
```
.item1 {  
    grid-area: 1 / 3 / 2 / 5;  
}
```

.item1 bắt đầu trên dòng 1 và cột 3 - kết thúc ở dòng 2 và cột 5

Ví dụ xây dựng grid layout như hình dưới đây:

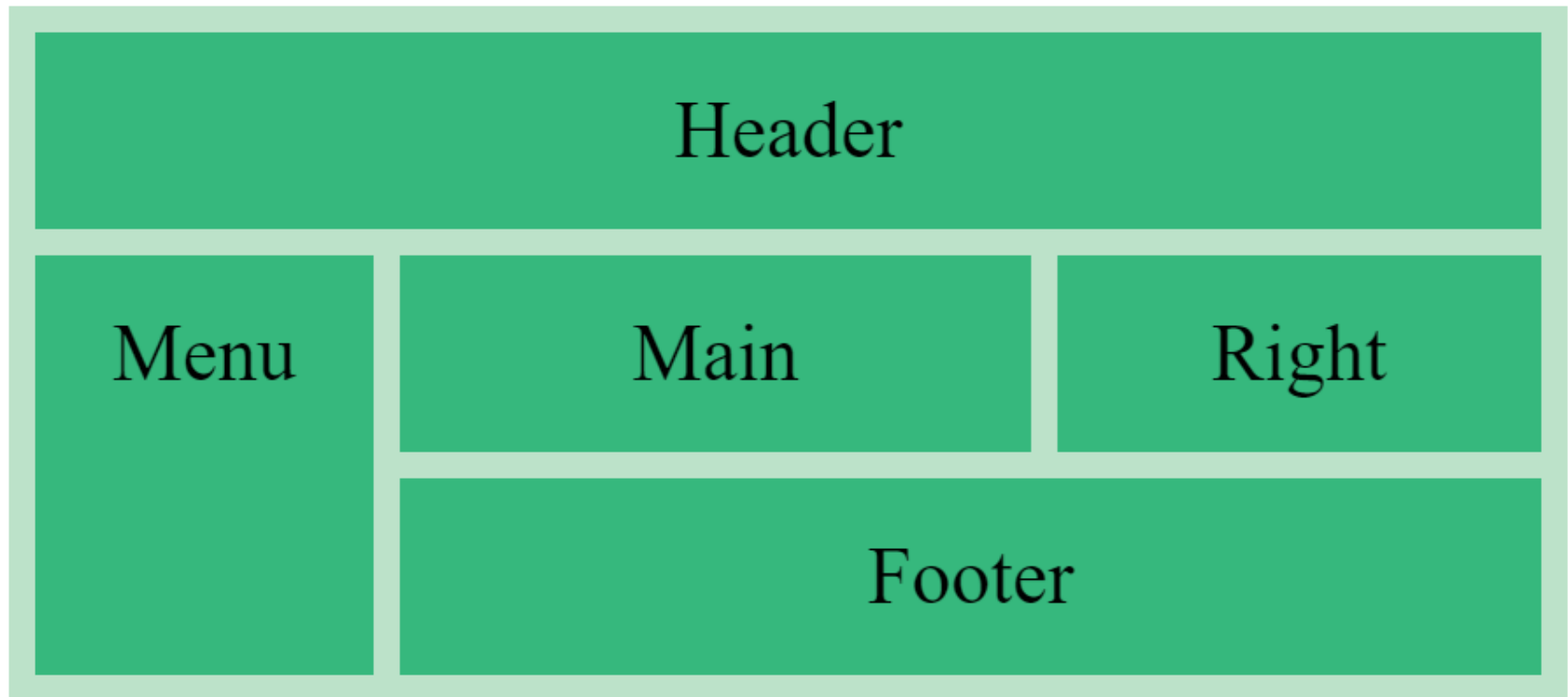
```
<div class="grid-container">  
  <div class="item1">1</div>  
  <div class="item2">2</div>  
  <div class="item3">3</div>  
  <div class="item4">4</div>  
  <div class="item5">5</div>  
  <div class="item6">6</div>  
  <div class="item7">7</div>  
  <div class="item8">8</div>  
  <div class="item9">9</div>  
  <div class="item9">10</div>  
</div>
```

```
.item2 {  
  grid-area: 1 / 2 / 3 / 5;  
}
```



# ĐẶT TÊN CHO CÁC ITEM TRONG GRID

Thuộc tính **grid-area** cũng có thể được sử dụng để gán tên cho các item.



# ĐẶT TÊN CHO CÁC ITEM TRONG GRID

Ví dụ:

```
<div class="grid-container">
  <div class="item1">Header</div>
  <div class="item2">Menu</div>
  <div class="item3">Main</div>
  <div class="item4">Right</div>
  <div class="item5">Footer</div>
</div>
```

```
.item1 { grid-area: header; }
.item2 { grid-area: menu; }
.item3 { grid-area: main; }
.item4 { grid-area: right; }
.item5 { grid-area: footer; }

.grid-container {
  display: grid;
  grid-template-areas:
    'header header header header header header'
    'menu main main main main right right'
    'menu footer footer footer footer footer';
  gap: 10px;
  background-color: #BCE2C9;
  padding: 10px;
}

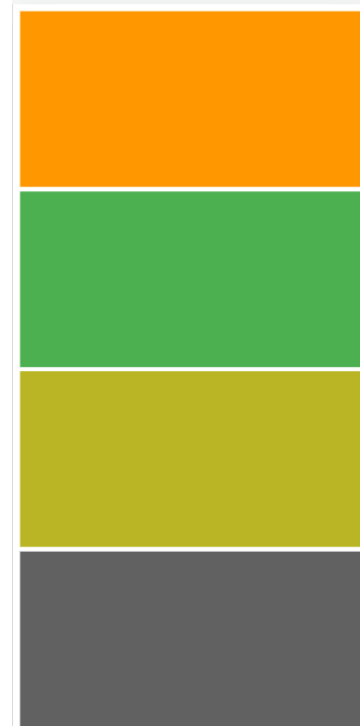
.grid-container > div {
  background-color: #36B87D;
  text-align: center;
  padding: 20px 0;
  font-size: 30px;
}
```

# KẾT HỢP SỬ DỤNG CSS GRID VÀ MEDIA QUERY

Ví dụ mẫu xây dựng bố cục trên các thiết bị như hình dưới đây:



*Giao diện trên kích thước > 600px*



*Giao diện trên kích thước <= 600px*



```
<div class="grid-container">
  <div class="grid-item1"></div>
  <div class="grid-item2"></div>
  <div class="grid-item3"></div>
  <div class="grid-item4"></div>
</div>
```

```
.grid-container{
  display: grid;
  grid-template-columns: auto auto auto;
  grid-template-rows: 200px 200px;
  gap: 5px;
}

.grid-item1{
  background-color: #FF9800;
  grid-row: 1/3;
  grid-column: 1/2;
}

.grid-item2{
  background-color: #4CAF50;
  grid-column: 2/4;
}

.grid-item3{
  background-color: #bab524;
}

.grid-item4{
  background-color: #616161;
}
```

Thực hiện Responsive lên các thiết bị, ví dụ:

```
@media screen and (max-width: 600px) {  
  .grid-container{  
    grid-template-columns: auto;  
    grid-template-rows: 200px 200px 200px 200px;  
  }  
  .grid-item1{  
    grid-column: 1/2;  
    grid-row: 1/2;  
  }  
  .grid-item2{  
    grid-column: 1/2;  
    grid-row: 2/3;  
  }  
  .grid-item3{  
    grid-column: 1/2;  
    grid-row: 3/4;  
  }  
  .grid-item4{  
    grid-column: 1/2;  
    grid-row: 4/5;  
  }  
}
```

## 1. Ưu điểm

- ❖ Dàn layout 2 chiều cả ngang lẫn dọc
- ❖ Dễ kiểm soát được khoảng cách giữa các item
- ❖ Tùy biến linh hoạt bố cục với responsive

## 2. Nhược điểm

- ❖ Nhiều trình duyệt còn chưa hỗ trợ
- ❖ Chưa được nhiều thư viện css tích hợp
- ❖ Mất nhiều thời gian để tìm hiểu

- ❑ Bố cục Grid là một hệ thống bố cục 2 chiều (x,y) được dùng trong thiết kế UI.
- ❑ CSS Grid được cấu thành từ hai thành phần chính là grid-container và grid-item.
- ❑ Kết hợp Media Query và Grid





**Cảm ơn**