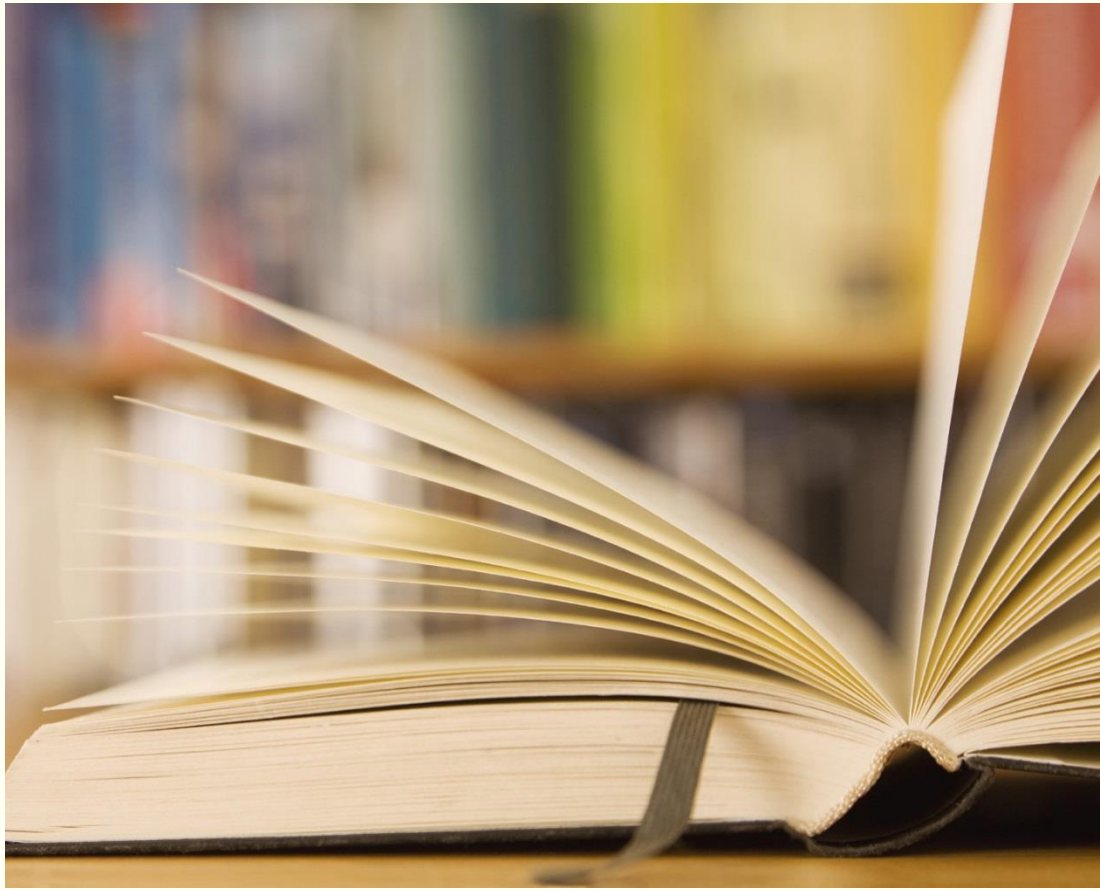
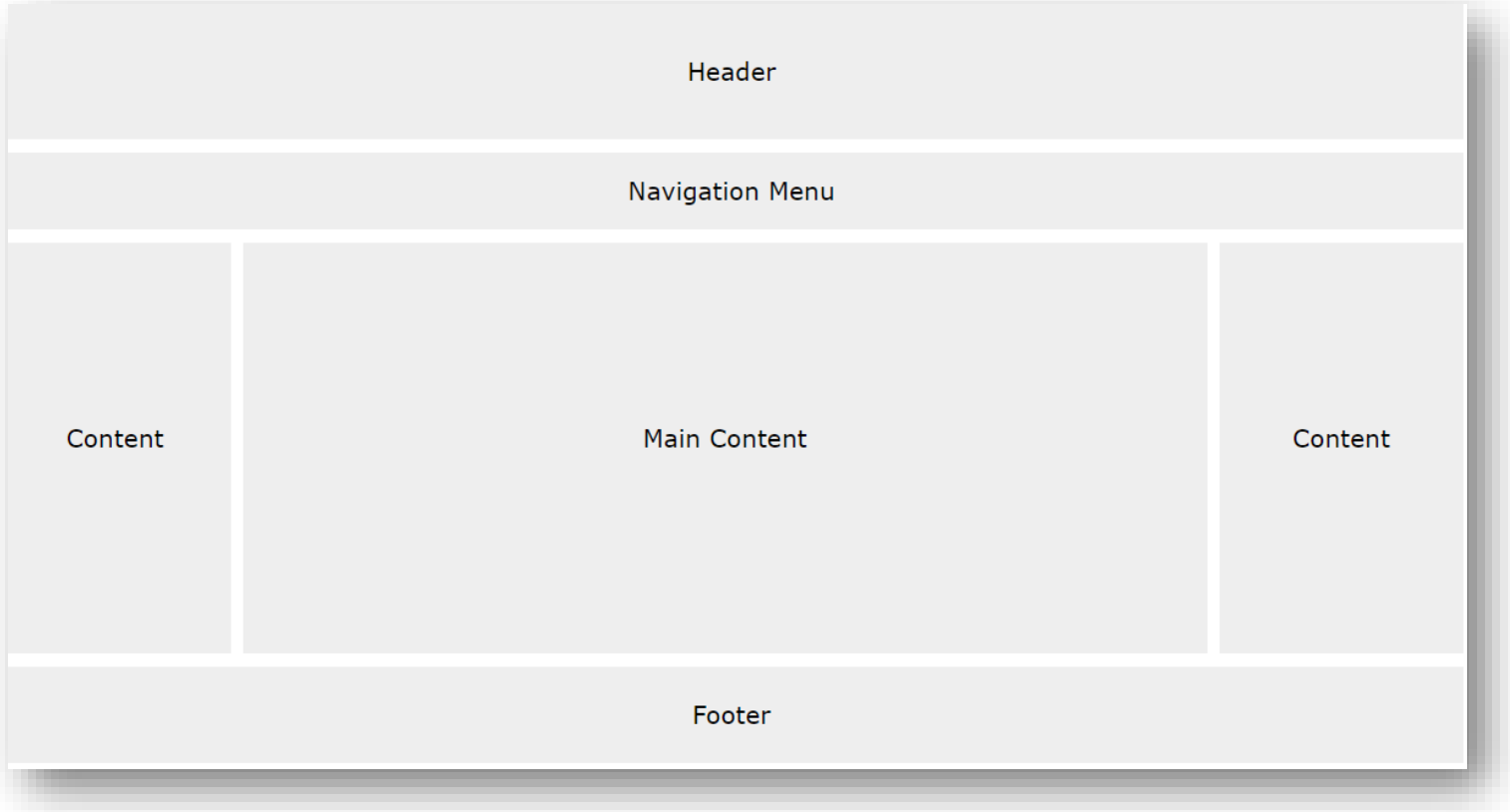
HTML LAYOUT



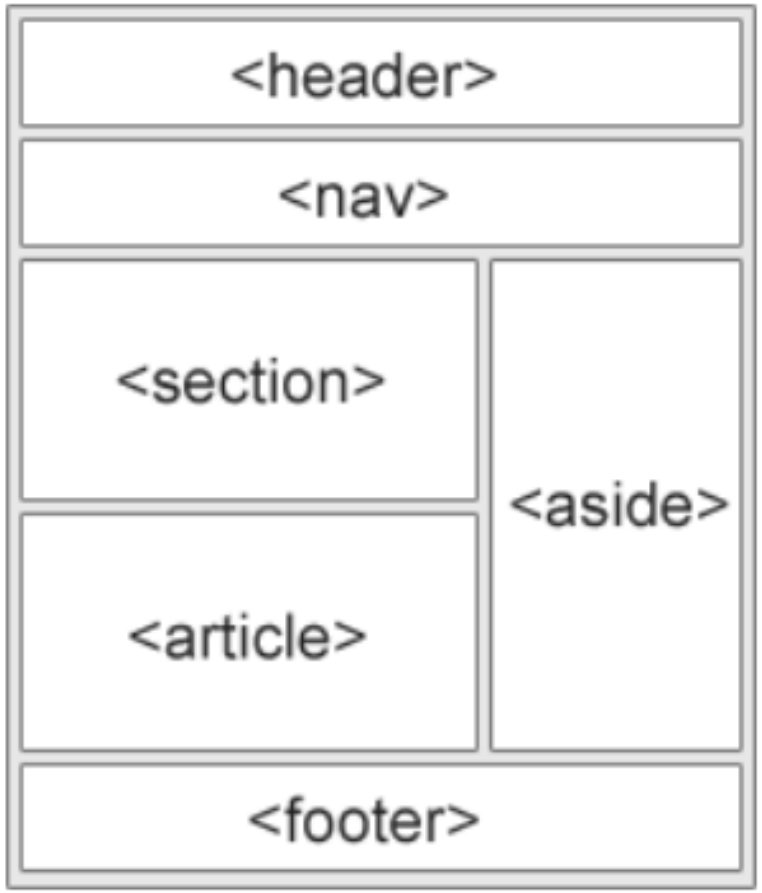
Layout trong HTML

- Trang web thường hiển thị nội dung dưới dạng các cột (như trên báo hoặc tạp chí). Một trang web thường được chia thành các phần headers, menus, content và footer



Layout trong HTML

- **HTML5** cung cấp các phần tử mới để định nghĩa các phần trong trang web
  - **<head>** tiêu đề tài liệu hoặc một phần
  - **<nav>** bộ chứa các đường dẫn điều hướng
  - **<section>** một phần trong tài liệu
  - **<article>** một bài viết độc lập, tự chứa trong chính nó
  - **<aside>** nội dung bên cạnh một nội dung khác (sidebar)
  - **<footer>** chân trang
  - **<details>** chi tiết bổ sung thêm
  - **<summary>** tiêu đề cho phần tử

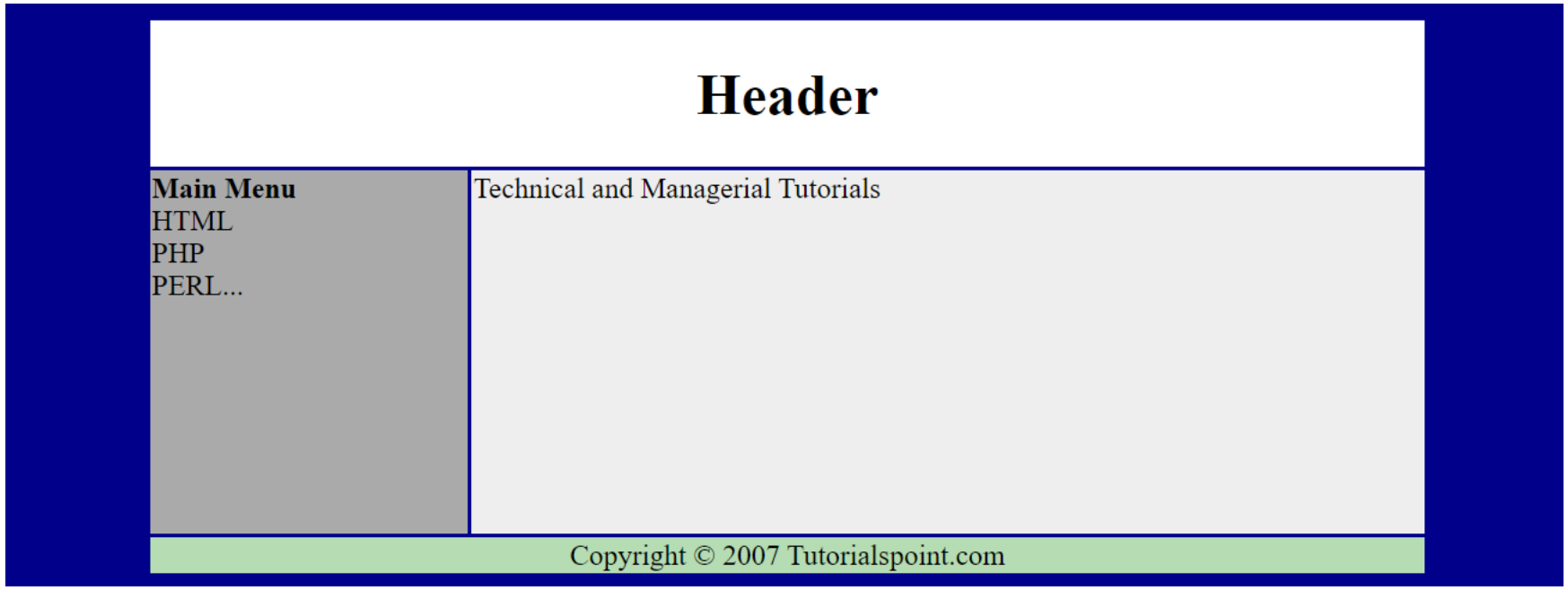


Layout trong HTML

- **Kỹ thuật bố cục trang HTML**: Có bốn kỹ thuật khác nhau để tạo bố cục trang, mỗi kỹ thuật đều có ưu và nhược điểm của nó:
  - **CSS framework**: sử dụng các framework có sẵn như **Bootstrap**
  - **CSS float:** sử dụng thuộc tính float của CSS**.**
  - **CSS flexbox**: hoạt động của các phần tử có thể đoán trước được, bố cục trang phù hợp với các kích thước màn hình và các thiết bị khác nhau.
  - **CSS grid:** bố cục dựa trên lưới, với các hàng và cột

Tạo layout sử dụng Table

- Cách đơn giản và phổ biến nhất để tạo bố cục là sử dụng thẻ **HTML <table>.**



CSS Float Layout

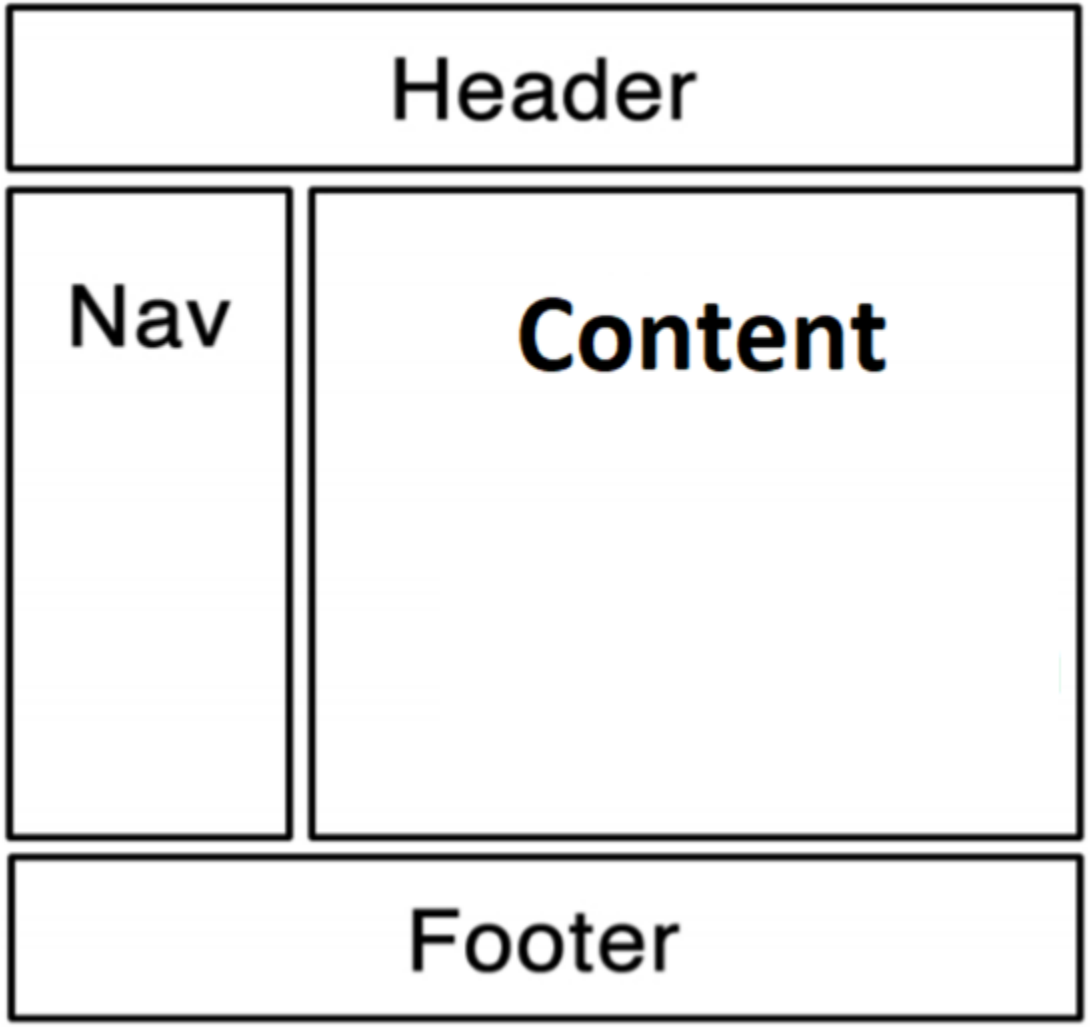
- **Tạo bộ bố cục web sử dụng thuộc tính float của CSS:**
  - **Thuộc tính Float**: để chuyển một phần tử sang góc trái hoặc phải của không gian bao quanh nó, mặc định, thuộc tính float của các phần tử HTML là none.
  - Thuộc tính Float gồm các giá trị sau:
    - **Left:** Cố định phần tử về bên trái.
    - **Right:** Cố định phần tử về bên phải.
    - **None:** Nằm tại chính vị trí được tạo
    - **Inherit** (kế thừa): Phần tử kế thừa giá trị từ float cha.

CSS Float Layout

- **Cú pháp**: **tagName { float: giá trị; }**
  - ***Lưu ý:*** khi một phần tử có thuộc tinh float=left hoặc float=right thì tất cả các thẻ cùng cấp phía sau nó sẽ được tràn lên phía trên và lấp đầy chỗ trống.

Sử dụng thẻ div

- **Thẻ div**: xác định nghĩa một phân vùng hoặc vùng trong một tài liệu HTML, thẻ div thường được dùng để làm vùng chứa các phần tử HTML khác hay dùng để định nghĩa bố cục giao diện của 1 trang đơn giản.



Sử dụng thẻ div

- **Cách thiết kế layout với thẻ div**

```
<body>
    <div id="header">
      <h1>Tiêu đề của trang</h1>
    </div>
    <div id="nav">
      Menu 1<br>
      Menu 2<br>
    </div>
    <div id="section">
      Nội dung chính
    </div>
    <div id="footer">
      thông tin liên hệ
    </div>
</body>
```

Sử dụng thẻ div

- **Cách thiết kế layout với thẻ div**

```
<style>
#header {
    background-color:black;
    color:white;
    text-align:center;
    padding:5px;
}
#nav {
    line-height:30px;
    background-color:#eeeeee;
    height:300px;
    width:100px;
    float:left;
    padding:5px;
}
#section {
    width:350px;
    float:left;
    padding:10px;
}
#footer {
    background-color:black;
    color:white;
    clear:both;
    text-align:center;
    padding:5px;
}
</style>
```

Thiết kế Layout với HTML5

- **Sử dụng các thẻ <header>, <nav>, <section>, and <footer>**

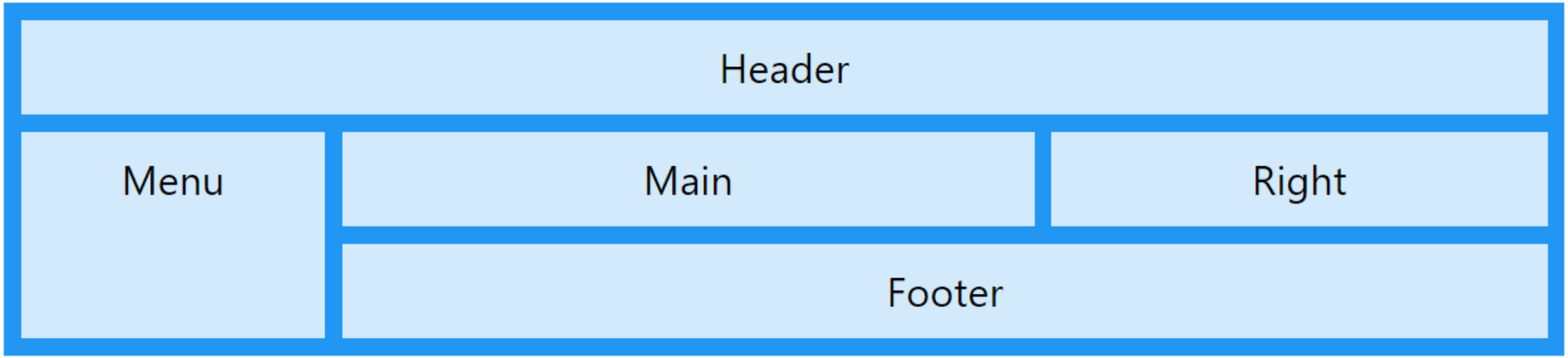
```
<body>
    <header>
        <h1>City Gallery</h1>
    </header>
    <nav>
        Menu 1<br>
        Menu 1 <br>
    </nav>
    <section>
        Nội dung chính
    </section>
    <footer>
        Copyright © abc.com
    </footer>
</body>
```

Thiết kế Layout với HTML5

```
header {
    background-color:black;
    color:white;
    text-align:center;
    padding:5px;
}
nav {
    line-height:30px;
    background-color:#eeeeee;
    height:300px;
    width:100px;
    float:left;
    padding:5px;
}
section {
    width:350px;
    float:left;
    padding:10px;
}
footer {
    background-color:black;
    color:white;
    clear:both;
    text-align:center;
    padding:5px;
}
```

CSS Grid Layout

- **CSS Grid Layout**: cung cấp một hệ thống bố cục dựa trên lưới, với các hàng và cột, giúp thiết kế các trang web dễ dàng hơn mà không cần phải sử dụng float và định vị.



Phần tử lưới - grid

- Bố cục lưới bao gồm một phần tử **div** bọc ngoài cha, (phần tử cha) và các **item** bên trong (phần tử con).

```
<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>
```

1	2	3
4	5	6
7	8	9

Thuộc tính Display

- Phần tử HTML trở thành vùng chứa lưới khi thuộc tính **display** được thiết lập với giá trị **grid** hoặc **inline-grid**.
- **Cú pháp**:

```
htmlItem { display: grid; }
```

```
htmlItem { display: inline-grid; }
```

  - **display : grid** tạo lưới cấp khối (block level).
  - **display : inline-grid** tạo ra một lưới cấp độ dòng (inline level).

Các thuộc tính của phần tử container

- **Xác định số hàng và số cột của lưới:** sử dụng thuộc tính
  - **grid-template-row:** chỉ định số dòng
  - **grid-template-column**: chỉ định số cột
- Ví dụ: tạo một lưới 3 cột 2 dòng

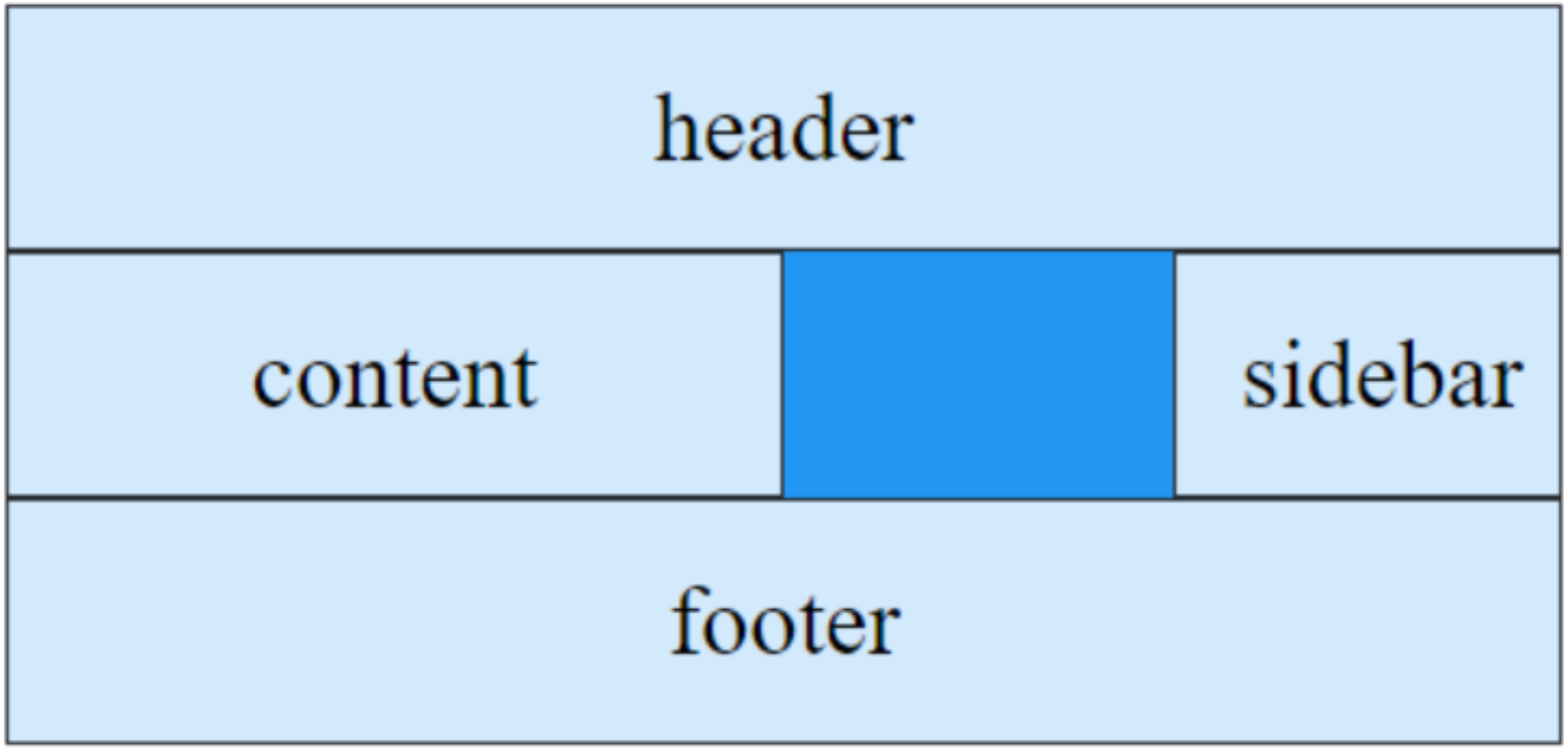
```
.grid-container
 {
    display: grid;
    grid-template-columns: 100px 100px 100px;
    grid-template-rows: 50px 50px;
    background-color: #2196F3;
  }
```

Các thuộc tính của phần tử container

- **grid-template-areas** : tạo layout
- **Ví dụ**: Tạo layout 4 cột và 3 hàng :
  - **Header**: hàng 1
  - **Content**: 2 phần, **sidebar** 1 phần của hàng 2
  - **Footer**: hàng cuối, còn lại 1 phần nằm ở giữa content và sidebar.
  - Vậy layout có 4 khối : header, content, sidebar và footer.

Các thuộc tính của phần tử container

- **grid-template-areas**
- Ví dụ (tt)



Các thuộc tính của phần tử container

- **grid-template-areas**
- **Ví dụ (tt)**

```
<div class="grid-container">
      <div class="item-header">header</div>
      <div class="item-content">content</div>
      <div class="item-sidebar">sidebar</div>
      <div class="item-footer">footer</div>
</div>
```

Các thuộc tính của phần tử container

- **Ví dụ (tt)**
  - Tạo class cho **item-header** với thuộc tính **grid-area : header;**
  - Tạo class cho **item-content** với thuộc tính **grid-area : content**;
  - Tạo class cho **item-sidebar** với thuộc tính **grid-area : sidebar**;
  - Tạo class cho **item-footer** với thuộc tính **grid-area : footer**;
- Sau đó thiết lập thuộc tính g**rid-template-areas** cho **grid-container** như sau

```
grid-template-areas : "header header header header"
                      "content content . sidebar"
                      "footer footer footer footer"
```

Các thuộc tính của grid item

- **Xác định vị trí bắt đầu và kết thúc của item trong grid**
  - grid-column-start
  - grid-column-end
  - grid-row-start
  - grid-row-end
- Ví dụ:
  - item1 bắt đầu cột 1 và kết thúc ở cột 4,
  - item2 bắt đầu ở cột 1 và kết thúc ở cột 2,
  - item3 bắt đầu ở cột 1 và kết thúc ở cột 3 :

Các thuộc tính của grid item

- **Xác định vị trí bắt đầu và kết thúc của item trong grid**
- Ví dụ:

```
item1 {
        grid-column-start: 1;
        grid-column-end: 4;
}
item2 {
        grid-column-start: 1;
        grid-column-end: 2;
}
item3 {
        grid-column-start: 1;
         grid-column-end: 3;
}
```

Các thuộc tính của grid item

- **Xác định vị trí bắt đầu và kết thúc của item trong grid**
- Ví dụ: