

Phát triển ứng dụng web

CSS (Cascading Style Sheets)
Layout

Nội dung

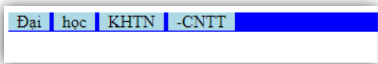
- Display
- Flexible box layout
- Grid layout
- CSS – Page Layout

Nội dung

- **Display**
- Flexible box layout
- Grid layout
- CSS – Page Layout

Block – Inline (default)

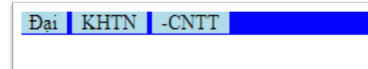
```
<html>
  <head>
    <style>
      div {
        background-color: blue;
      }
      span {
        background-color: lightblue;
        padding: 0 0.5rem;
      }
    </style>
  </head>
  <body>
    <div>
      <span>Đại</span>
      <span>học</span>
      <span>KHTN</span>
      <span>-CNTT</span>
    </div>
  </body>
</html>
```



Đại học KHTN -CNTT

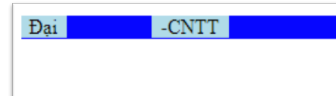
Display: none

```
<head>
  <style>
    div {
      background-color: blue;
    }
    span {
      background-color: lightblue;
      padding: 0 0.5rem;
    }
    .dnone {
      display: none;
    }
  </style>
</head>
<body>
  <div>
    <span>Đại</span>
    <span class="dnone">học</span>
    <span>KHTN</span>
    <span>-CNTT</span>
  </div>
</body>
```



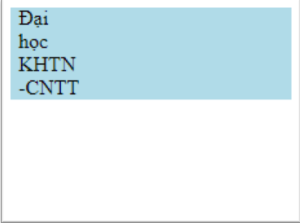
Visibility: hidden

```
<head>
  <style>
    div {
      background-color: blue;
    }
    span {
      background-color: lightblue;
      padding: 0 0.5rem;
    }
    .dnone {
      display: none;
    }
    .vhidden {
      visibility: hidden;
    }
  </style>
</head>
<body>
  <div>
    <span>Đại</span>
    <span class="dnone">học</span>
    <span class="vhidden">KHTN</span>
    <span>-CNTT</span>
  </div>
</body>
```



Display: block

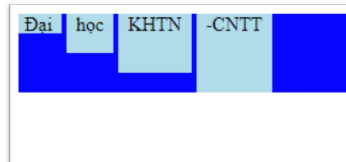
```
<head>
  <style>
    div {
      background-color: blue;
    }
    span {
      background-color: lightblue;
      padding: 0 0.5rem;
      display: block;
    }
  </style>
</head>
<body>
  <div>
    <span>Đại</span>
    <span>học</span>
    <span>KHTN</span>
    <span>-CNTT</span>
  </div>
</body>
```



Đại
học
KHTN
-CNTT

Display: inline-block

```
<style>
  div {
    background-color: blue;
  }
  span {
    background-color: lightblue;
    padding: 0 0.5rem;
    display: inline-block;
  }
  span:nth-child(1) {
    height: 1rem;
  }
  span:nth-child(2) {
    height: 2rem;
  }
  span:nth-child(3) {
    height: 3rem;
  }
  span:nth-child(4) {
    height: 4rem;
  }
</style>
```



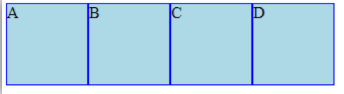
Nội dung

- *Display*
- **Flexible box layout**
- Grid layout
- CSS – Page Layout

Flexible box layout (flexbox)

- Giúp xây dựng bố cục linh hoạt, đáp ứng với nhiều độ phân giải. Nó cung cấp tổng thể khả năng điều chỉnh kích thước của các thành phần container để hiển thị theo cách hiệu quả và dễ đọc nhất có thể.

```
<head>
<style>
.container {
  display: flex;
}
.item {
  width: 5rem;
  height: 5rem;
  background-color: lightblue;
  border: solid 1px blue;
}
</style>
</head>
<body>
<div class="container">
  <div class="item">A</div>
  <div class="item">B</div>
  <div class="item">C</div>
  <div class="item">D</div>
</div>
</body>
```



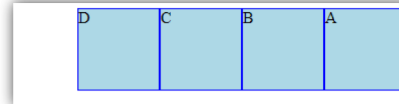
The diagram illustrates the result of the CSS code. A horizontal container (flex) contains four items (flex items) labeled A, B, C, and D. Each item is a light blue square with a solid blue border, representing the styles defined in the CSS code.

Flex Direction

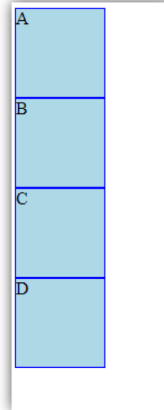
- **Flex direction** xác định chiều nào và hướng nào các thành phần bên trong sẽ được sắp đặt. Có 2 tùy chọn cho chiều là theo hàng hay cột và 2 tùy chọn cho hướng là bình thường hoặc đảo ngược.
 - row
 - row-reverse
 - column
 - column-reverse

Flex Direction

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



```
.container {  
  display: flex;  
  flex-direction: column;  
}
```

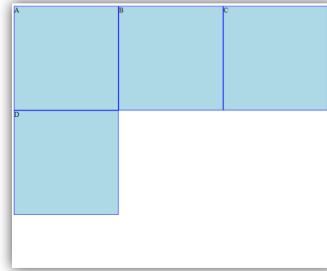


Flex Wrap

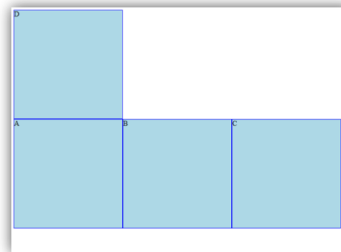
- Mặc định thì container sẽ cố gắng sắp xếp các thành phần con trên cùng hàng. Để thay đổi thì có thể sử dụng **flex-wrap** với các giá trị phù hợp:
 - wrap
 - no-wrap
 - wrap-reverse

Flex wrap

```
.container {  
  display: flex;  
  flex-wrap: wrap;  
}
```

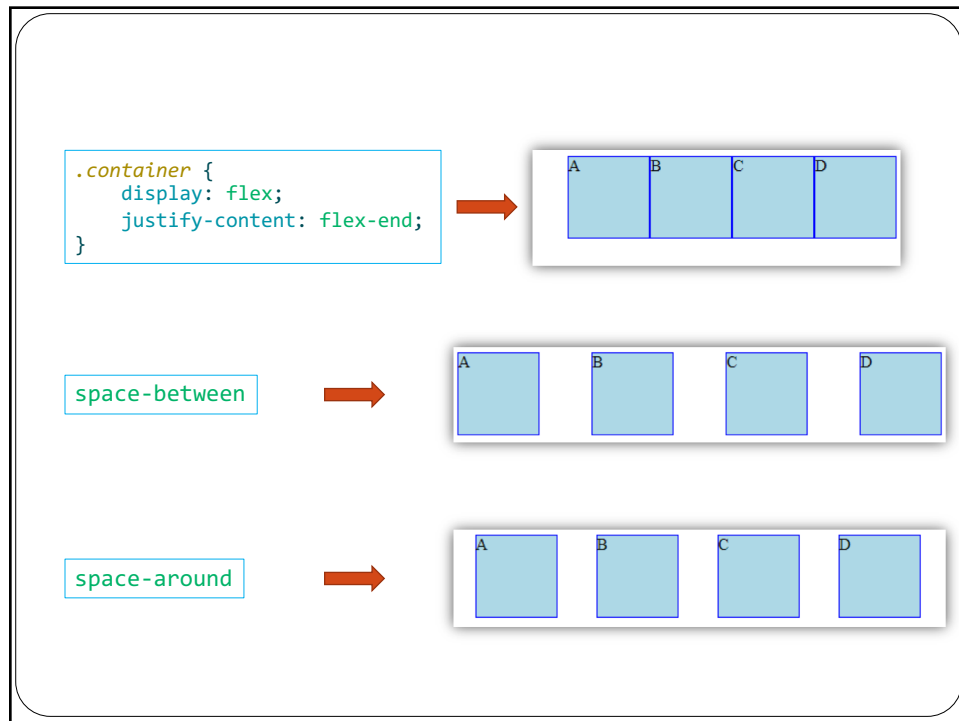


```
.container {  
  display: flex;  
  flex-wrap: wrap-reverse;  
}
```



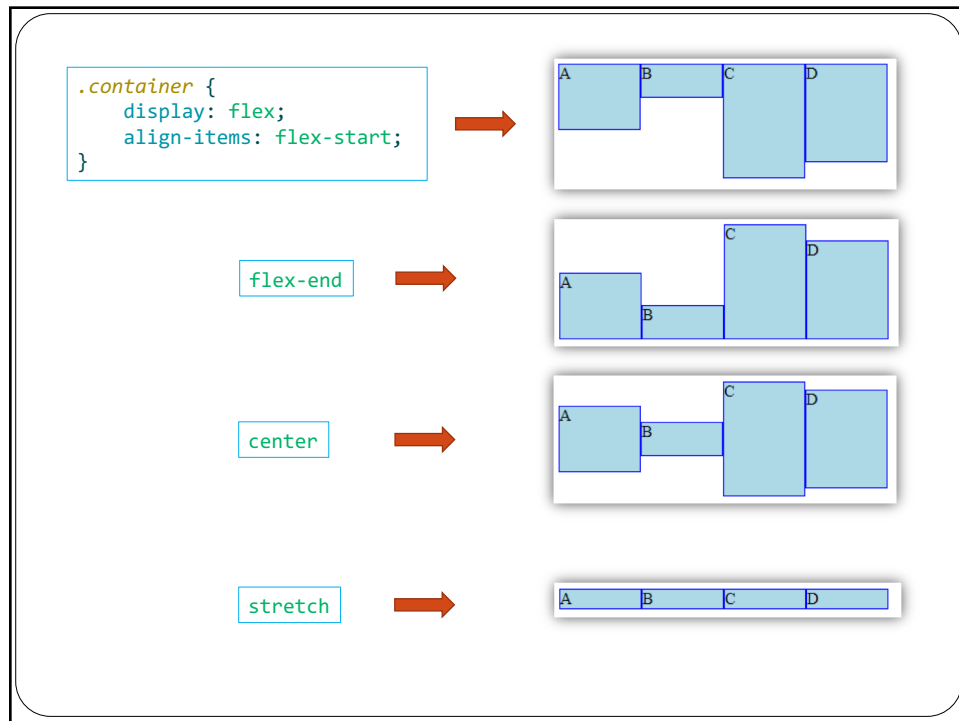
Justify Content

- ***justify-content*** định nghĩa khoảng cách xung quanh các item bên trong vùng cha. Có 6 cách thiết lập khác nhau:
 - flex-start
 - flex-end
 - center
 - space-between
 - space-around
 - space-evenly



Align Items

- ***align-items*** xác định cách các ***item*** được định vị trong hàng hoặc cột. Nó tương tự như ***justify-content*** nhưng trực giao với dòng nó được căn chỉnh. Các giá trị có thể thiết lập:
 - flex-start
 - flex-end
 - center
 - stretch
 - baseline



Align Content

- ***align-content*** xác định khoảng cách giữa các line nếu chúng được wrap. Các giá trị có thể thiết lập:
 - flex-start
 - flex-end
 - center
 - stretch
 - space-between
 - space-around

```
.container {  
  display: flex;  
  align-content: flex-end;  
  height: 20rem;  
  flex-wrap: wrap;  
}
```

center

space-between

The diagram illustrates three different flex container visualizations. The first visualization shows a 2x6 grid of items labeled A-F and G-L. The second visualization shows the same grid with 'center' alignment. The third visualization shows the same grid with 'space-between' alignment.

Order

```
.a {  
  order: 1;  
}  
.b {  
  order: -2;  
}
```

```
<div class="item a">A</div>  
<div class="item b">B</div>  
<div class="item">C</div>  
<div class="item">D</div>
```

The diagram shows the result of the CSS order property. The items are displayed in the order B, C, D, A.

Flex Grow

```
.item {  
  width: 5rem;  
  height: 5rem;  
  background-color: lightskyblue;  
  border: solid 1px orange;  
  display: flex;  
  align-items: center;  
  justify-content: center;  
  font-size: 3rem;  
  flex-grow: 1;  
}
```

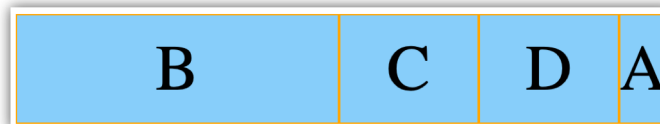
```
.a {  
  order: 1;  
  flex-grow: 2;  
}  
.b {  
  order: -2;  
  flex-grow: 0.5;  
}
```



Flex Shrink

```
.item {  
  width: 15rem;  
  height: 5rem;  
  background-color: lightskyblue;  
  border: solid 1px orange;  
  display: flex;  
  align-items: center;  
  justify-content: center;  
  font-size: 3rem;  
}
```

```
.a {  
  order: 1;  
  flex-shrink: 2;  
}  
.b {  
  order: -2;  
  flex-shrink: 0;  
}
```



Flex Basis

```
.item {  
  width: 10rem;  
  height: 5rem;  
  background-color: lightskyblue;  
  border: solid 1px orange;  
  display: flex;  
  align-items: center;  
  justify-content: center;  
  font-size: 3rem;  
}
```

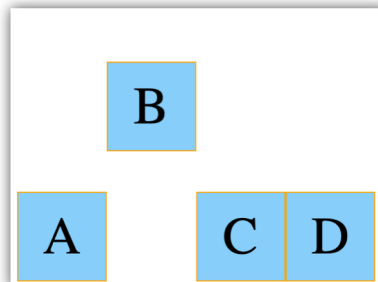
```
.a {  
  order: -1;  
  flex-shrink: 2;  
}  
.b {  
  order: 0;  
  flex-basis: 20rem;  
}
```



Align Self

```
.container {  
  display: flex;  
  align-items: flex-end;  
  height: 20rem;  
}
```

```
.container {  
  display: flex;  
  align-items: flex-end;  
  height: 20rem;  
}
```



Nội dung

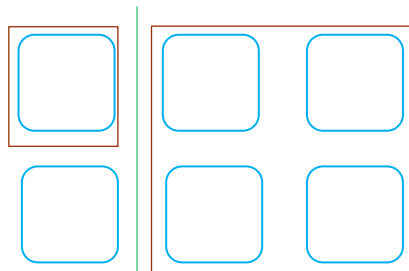
- *Display*
- *Flexible box layout*
- **Grid layout**
- CSS – Page Layout

Grid layout

- Bố cục (*layout*) giao diện dựa vào lưới (*grid*) là hướng tiếp cận khả quan nhất. Tuy nhiên với CSS trước giờ thì việc bố cục theo lưới rất khó khăn và phức tạp.
- Hiện nay **CSS Grid** với mục tiêu cung cấp hệ thống bố cục theo lưới từ gốc giúp cho việc bố cục dễ dàng, đẹp mắt mà không cần sử dụng nhiều *markup* và *style*.

Thuật ngữ

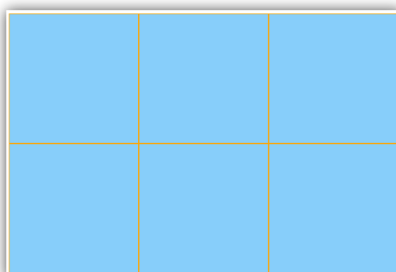
- Grid Lines
- Grid Cell
- Grid Area
- Grid Row
- Grid Column
- Grid Gap



Khai báo grid

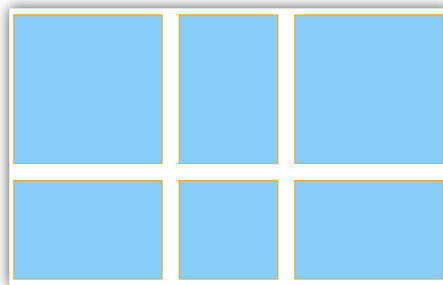
```
.container {  
  display: grid;  
  grid-template-columns: 150px 150px 150px;  
  grid-template-rows: 150px 150px;  
}  
.cell {  
  background-color: lightskyblue;  
  border: solid 1px orange;  
}
```

```
<div class="container">  
  <div class="cell"></div>  
  <div class="cell"></div>  
  <div class="cell"></div>  
  <div class="cell"></div>  
  <div class="cell"></div>  
  <div class="cell"></div>  
</div>
```



Sử dụng gap

```
.container {  
  display: grid;  
  grid-template-columns: 150px 100px 150px;  
  grid-template-rows: 150px 100px;  
  grid-gap: 1rem;  
}
```



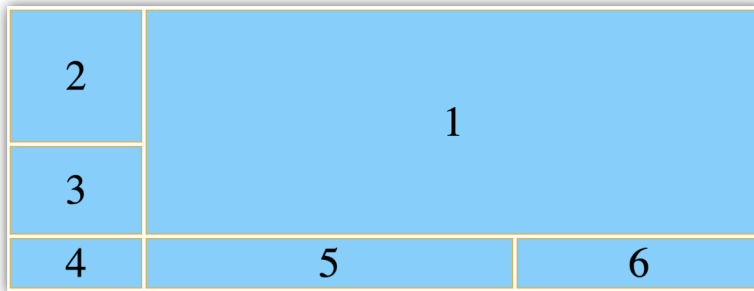
FR (Fraction) Unit

```
.container {  
  display: grid;  
  grid-template-columns: 150px 1.5fr 1fr;  
  grid-template-rows: 150px 100px;  
  grid-gap: 0.25rem;  
}
```



Grid Start - End

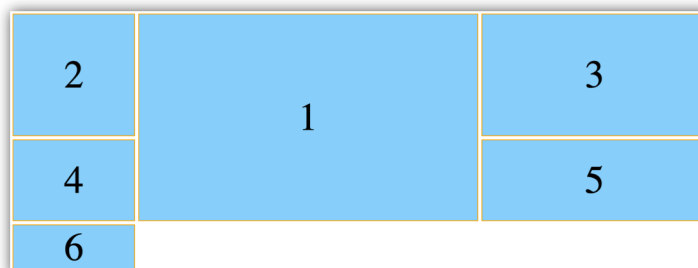
```
.s {
  grid-row: 1 / 3;
  grid-column: 2 / 4;
}
```



Named Grid Lines

```
.container {
  display: grid;
  grid-template-columns:
    [column-1] 150px
    [column-2] 1.5fr
    [column-3] 1fr;
  grid-template-rows: 150px 100px;
  grid-gap: 0.25rem;
}
```

```
.s {
  grid-row: 1 / 3;
  grid-column: column-2 / column-3;
}
```



Template Areas

```
.container {  
  display: grid;  
  grid-gap: 0.5rem;  
  grid-template-areas:  
    "top top"  
    "middle-1 middle-2 "  
    "bottom bottom";  
}
```

```
.top {  
  grid-area: top;  
}  
.middle-1 {  
  grid-area: middle-1;  
}  
.middle-2 {  
  grid-area: middle-2;  
}  
.bottom {  
  grid-area: bottom;  
}
```



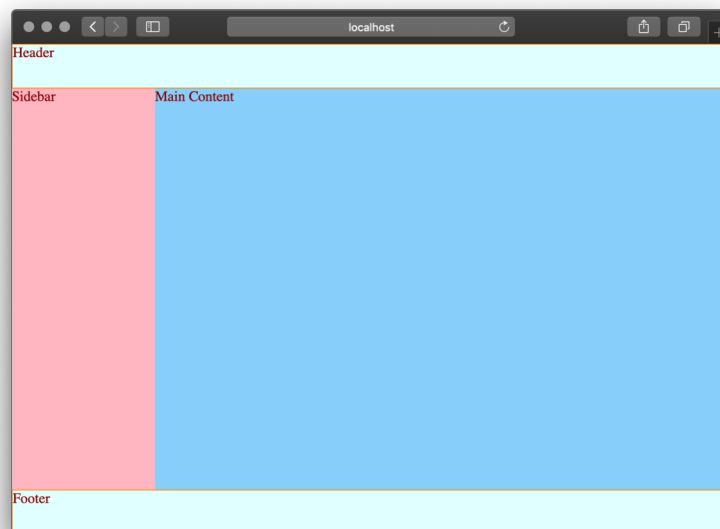
Nội dung

- *Display*
- *Flexible box layout*
- *Grid layout*
- **CSS – Page Layout**

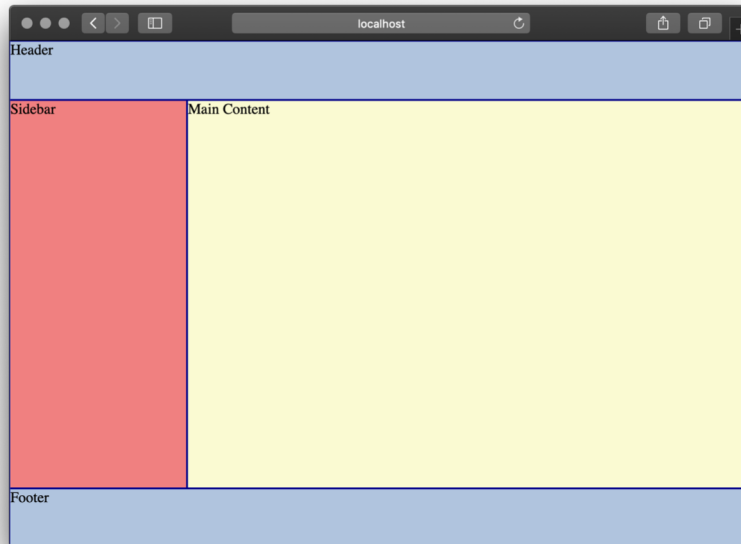
CSS - Page Layout

- Trong việc sử dụng CSS thì điều khó khăn nhất là làm sao layout được chính xác các thành phần lên page.
- Các loại Page layout thông dụng trong CSS:
 - Two-column Page Layout (sử dụng float)
 - Sidebar Navigation Layout with Position
 - Flexbox Layout
 - Grid Layout

Flexbox Layout



Grid Layout



Nội dung

- *Display*
- *Flexible box layout*
- *Grid layout*
- *CSS – Page Layout*
- **Bài tập**

Bài tập

- Làm 2 ví dụ phần layout (Flexbox và Grid)
- Làm lại Bài tập tuần 02 với Flexbox và Grid.

