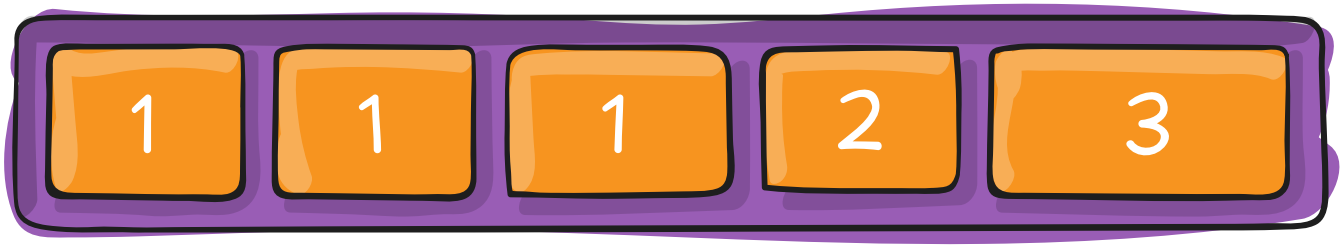
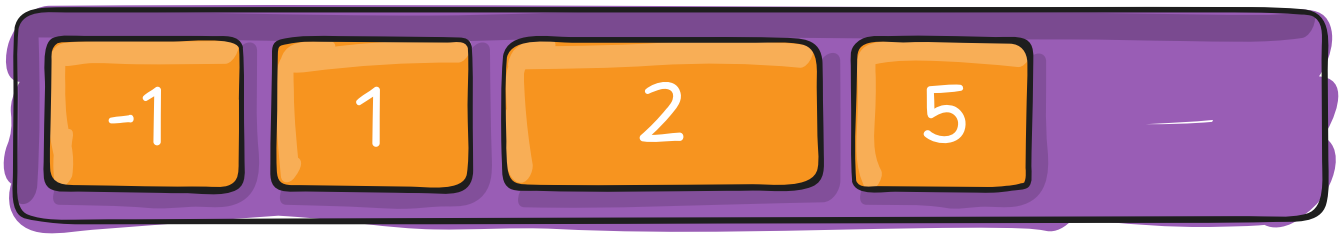
Thuộc tính Order

Thuộc tính `order` xác định thứ tự của các Flex items trong một Flex container. Mặc định, tất cả các Flex items có order là 0. Các phần tử có order nhỏ hơn sẽ xuất hiện trước các phần tử có order lớn hơn.

Sử dụng: Khi bạn muốn thay đổi thứ tự hiển thị của các phần tử mà không thay đổi thứ tự của chúng trong HTML.

> **Lưu ý:** order cũng nhận cả giá trị nguyên âm.





Ví dụ:

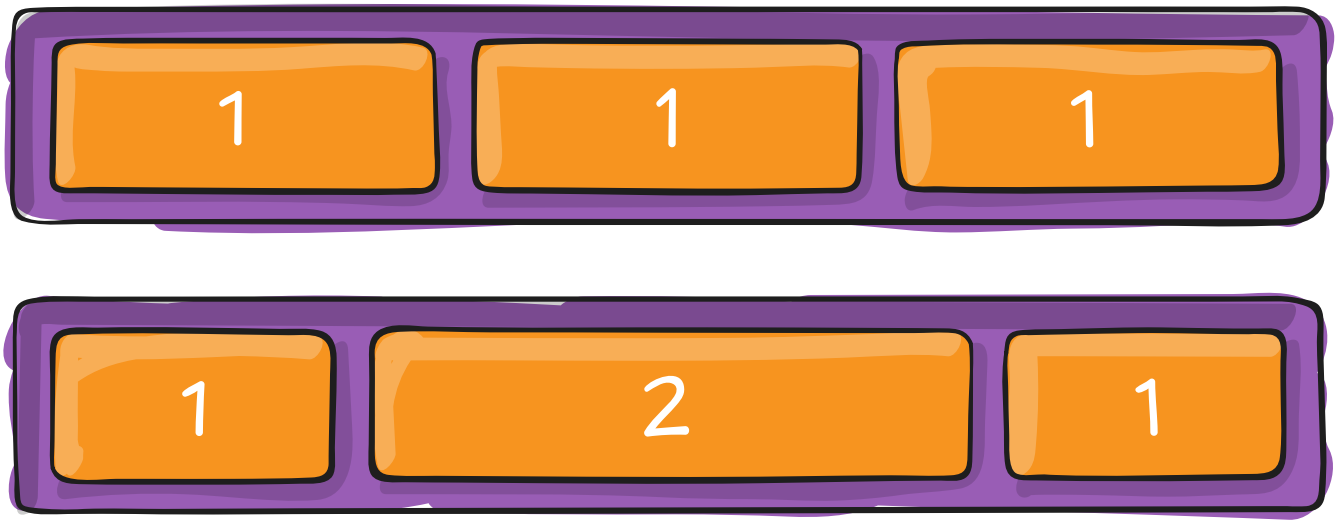
```
<style>
    .container {
        display: flex;
        background-color: #f0f0f0;
        padding: 10px;
    }
```

```
    .item {
        background-color: #4caf50;
        color: white;
        padding: 20px;
        margin: 5px;
        font-size: 20px;
    }

    .item-1 {
        order: 3;
    }

    .item-2 {
        order: 1;
    }

    .item-3 {
        order: 2;
    }
</style>

<div class="container">
    <div class="item item-1">Item 1</div>
    <div class="item item-2">Item 2</div>
    <div class="item item-3">Item 3</div>
</div>
```

Thuộc tính `flex-grow`

`flex-grow` xác định khả năng một flex-item mở rộng để chiếm không gian còn lại trong flex container. Giá trị mặc định là 0, nghĩa là phần tử sẽ không mở rộng.

Sử dụng: Khi bạn muốn một hoặc nhiều flex-items chiếm nhiều không gian hơn so với các phần tử khác.



Ví dụ:

```
<style>
    .container {
        display: flex;
        background-color: #f0f0f0;
        padding: 10px;
    }

    .item {
        background-color: #4caf50;
        color: white;
        padding: 20px;
        margin: 5px;
        font-size: 20px;
    }

    .item-1 {
        flex-grow: 1; /* Chiếm nhiều không gian hơn */
    }

    .item-2 {
        flex-grow: 2; /* Chiếm gấp đôi không gian của item-1 */
    }

    .item-3 {
        flex-grow: 1; /* Chiếm cùng không gian như item-1 */
    }
</style>

<div class="container">
    <div class="item item-1">Item 1</div>
    <div class="item item-2">Item 2</div>
    <div class="item item-3">Item 3</div>
</div>
```

Tuy nhiên, nếu như 1 phần tử có phần nội dung chiếm chỗ nhiều hơn, rất có thể flex-grow sẽ không hoạt động như mong muốn. Trong trường hợp này, chúng ta cần sử dụng `flex-basis`(sẽ học ở phần sau) để xác định kích thước ban đầu của phần tử.

Thuộc tính `flex-shrink`

`flex-shrink` Xác định khả năng một Flex item co lại nếu Flex container nhỏ hơn tổng kích thước các Flex items. Giá trị mặc định là 1, nghĩa là phần tử sẽ co lại nếu cần. Nếu giá trị là 0, phần tử sẽ không co lại.

Sử dụng: Khi bạn muốn ngăn một phần tử bị co lại hoặc bạn muốn một phần tử co lại nhiều hơn so với các phần tử khác.

Ví dụ:

```
<style>
    .container {
        display: flex;
        width: 400px;
        background-color: #f0f0f0;
        padding: 10px;
    }

    .item {
        background-color: #4caf50;
        color: white;
        padding: 20px;
        margin: 5px;
        font-size: 20px;
        width: 200px;
    }

    .item-1 {
        flex-shrink: 1; /* Co lại khi cần */
    }

    .item-2 {
        flex-shrink: 2; /* Co lại nhiều hơn */
    }

    .item-3 {
        flex-shrink: 0; /* Không co lại */
    }
</style>

<div class="container">
    <div class="item item-1">Item 1</div>
    <div class="item item-2">Item 2</div>
    <div class="item item-3">Item 3</div>
</div>
```

Thuộc tính `flex-basis`

`flex-basis` xác định kích thước ban đầu của một flex-item trước khi co lại hoặc mở rộng. Giá trị mặc định là `auto`, nghĩa là phần tử sẽ co lại hoặc mở rộng dựa trên kích thước nội dung. Giá trị này có thể là một đơn vị đo như `px`, `em`, `%` hoặc `auto` (mặc định).

Sử dụng: Khi bạn muốn xác định kích thước cơ bản của một phần tử trước khi nó bị ảnh hưởng bởi `flex-grow` hoặc `flex-shrink`. `flex-basis` cũng dùng thay thế cho `width` hoặc `height` trong một số trường hợp.

Ví dụ:

```
<style>
    .container {
        display: flex;
        background-color: #f0f0f0;
        padding: 10px;
    }

    .item {
        background-color: #4caf50;
        color: white;
        padding: 20px;
        margin: 5px;
        font-size: 20px;
    }

    .item-1 {
        flex-basis: 100px;
    }

    .item-2 {
        flex-basis: 200px;
    }

    .item-3 {
        flex-basis: 50px;
    }
</style>

<div class="container">
    <div class="item item-1">Item 1</div>
    <div class="item item-2">Item 2</div>
    <div class="item item-3">Item 3</div>
</div>
```

Thuộc tính `flex`

`flex` là một thuộc tính rút gọn để thiết lập `flex-grow`, `flex-shrink`, và `flex-basis` cùng một dòng.

Cú pháp:

```
.item {
    flex: [flex-grow] [flex-shrink] [flex-basis];
}
```

Giá trị mặc định là `flex: 1 1 auto`.

Ví dụ:

```
<style>
    .container {
        display: flex;
        background-color: #f0f0f0;
        padding: 10px;
    }

    .item {
        background-color: #4caf50;
        color: white;
        padding: 20px;
        margin: 5px;
        font-size: 20px;
        flex: 1; /* Mặc định flex-grow: 1, flex-shrink: 1, flex-basis: 0%
*/
    }
</style>

<div class="container">
    <div class="item">Item 1</div>
    <div class="item">Item 2</div>
    <div class="item">Item 3</div>
</div>
```

Khi muốn tuỳ chỉnh nhiều hơn các thuộc tính `flex-grow`, `flex-shrink`, và `flex-basis`, chúng ta có thể sử dụng `flex` như sau:

```
<style>
    .container {
        display: flex;
        background-color: #f0f0f0;
        padding: 10px;
    }

    .item-1 {
        background-color: #4caf50;
```

```
        color: white;
        padding: 20px;
        margin: 5px;
        font-size: 20px;
        flex: 2 1 150px; /* flex-grow: 2, flex-shrink: 1, flex-basis:
150px */
    }

    .item-2 {
        background-color: #ff5722;
        color: white;
        padding: 20px;
        margin: 5px;
        font-size: 20px;
        flex: 1 1 100px; /* flex-grow: 1, flex-shrink: 1, flex-basis:
100px */
    }

    .item-3 {
        background-color: #3f51b5;
        color: white;
        padding: 20px;
        margin: 5px;
        font-size: 20px;
        flex: 1 2 200px; /* flex-grow: 1, flex-shrink: 2, flex-basis:
200px */
    }
</style>

<div class="container">
    <div class="item-1">Item 1</div>
    <div class="item-2">Item 2</div>
    <div class="item-3">Item 3</div>
</div>
```

Thuộc tính `align-self`

`align-self` cho phép căn chỉnh một Flex item dọc theo chiều phụ (cross axis), ghi đè giá trị `align-items` của flex-container. Giá trị mặc định là auto, nghĩa là kế thừa từ align-items.

Sử dụng: Khi bạn muốn căn chỉnh một phần tử con khác với các phần tử còn lại trong Flex container.

```
<style>
    .container {
        display: flex;
        height: 200px;
        background-color: #f0f0f0;
        padding: 10px;
        align-items: center; /* Tất cả các phần tử con sẽ được căn giữa */
    }
```

```
    .item {
        background-color: #4caf50;
        color: white;
        padding: 20px;
        margin: 5px;
        font-size: 20px;
    }

    .item-1 {
        align-self: flex-start; /* Căn về đầu container */
    }

    .item-2 {
        align-self: flex-end; /* Căn về cuối container */
    }

    .item-3 {
        align-self: center; /* Căn giữa (không ghi đè vì align-items đã là
center) */
    }
</style>

<div class="container">
    <div class="item item-1">Item 1</div>
    <div class="item item-2">Item 2</div>
    <div class="item item-3">Item 3</div>
</div>
```

💻 **Bài tập nhỏ:**

Thực hành hoàn thiện giao diện sau với flexbox: Umea website