



Lesson 05

Responsive Web Design with

Flexbox

Module 01: WEB DESIGN



Mục tiêu

- Hiểu được khái niệm bố cục Flexbox
- Hiểu được các thuộc tính của Flex Container
- Sử dụng được các thuộc tính của Flex Item
- Canh chỉnh được giao diện với Auto Margin
- Xây dựng được RWD Music App với Flexbox

Khái niệm bố cục Flexbox



Tại sao chúng ta nên sử dụng Flexbox?

- ♦ Tạo bố cục thông minh cho một trang web với CSS đã tồn tại khá lâu với việc sử dụng **một số thủ thuật** với table, thuộc tính float,...để bố cục cách hiển thị phù hợp.
- ♦ Nhưng float, table cũng **đi cùng khá nhiều lỗi phát sinh ngoài ý muốn** mặc dù chúng ta đã rất cẩn thận quản lý, bố cục của chúng -> Có cách nào tốt hơn không?
- ♦ Giải pháp của vấn đề trên sẽ **được giải quyết với** một **cú pháp hiện đại, gọn gàng hơn với** mô hình bố cục **CSS Flexbox**.

CSS FLEXBOX

Complex flexbox example

First article

Tacos actually microdosing, pour-over semiotics banjo chicharrones retro fanny pack portland everyday carry vinyl typewriter. Tacos PBR&B pork belly, everyday carry ennui pickled sriracha normcore hashtag polaroid single-origin coffee cold-pressed. PBR&B tattooed trust fund twee, leggings salvia iPhone photo booth health goth gastropub hammock.

Second article

Tacos actually microdosing, pour-over semiotics banjo chicharrones retro fanny pack portland everyday carry vinyl typewriter. Tacos PBR&B pork belly, everyday carry ennui pickled sriracha normcore hashtag polaroid single-origin coffee cold-pressed. PBR&B tattooed trust fund twee, leggings salvia iPhone photo booth health goth gastropub hammock.

Smile Laugh Wink Shrugs
Blush

Tacos actually microdosing, pour-over semiotics banjo chicharrones retro fanny pack portland everyday carry vinyl typewriter. Tacos PBR&B pork belly, everyday carry ennui pickled sriracha normcore hashtag polaroid single-origin coffee cold-pressed. PBR&B tattooed trust fund twee, leggings salvia iPhone photo booth health goth gastropub hammock.

Cray food truck brunch, XOXO +1 keffiyeh pickled chambray waistcoat ennui. Organic small batch paleo 8-bit. Intelligentsia umami wayfarers pickled, asymmetrical kombucha letterpress kitsch leggings cold-pressed squid chartreuse put a bird on it. Listicle pickled man bun cornhole heirloom art party.

Khái niệm bố cục Flexbox



☐ Flexbox là gì?

- ◆ **Bố cục Flexbox** (gọi ngắn gọn là **Flex**): Là **một phương pháp hiệu quả để bố cục, canh chỉnh và phân phối không gian giữa các phần tử trong trang web** ngay cả khi **viewport** và **size** của các phần tử của bạn **không xác định và/hoặc động**.

Complex flexbox example

First article

Tacos actually microdosing, pour-over semiotics banjo chicharrones retro fanny pack portland everyday carry vinyl typewriter. Tacos PBR&B pork belly, everyday carry ennui pickled sriracha normcore hashtag polaroid single-origin coffee cold-pressed. PBR&B tattooed trust fund twee, leggings salvia iPhone photo booth health goth gastropub hammock.

Second article

Tacos actually microdosing, pour-over semiotics banjo chicharrones retro fanny pack portland everyday carry vinyl typewriter. Tacos PBR&B pork belly, everyday carry ennui pickled sriracha normcore hashtag polaroid single-origin coffee cold-pressed. PBR&B tattooed trust fund twee, leggings salvia iPhone photo booth health goth gastropub hammock.

Smile Laugh Wink Shrugs

Blush

Tacos actually microdosing, pour-over semiotics banjo chicharrones retro fanny pack portland everyday carry vinyl typewriter. Tacos PBR&B pork belly, everyday carry ennui pickled sriracha normcore hashtag polaroid single-origin coffee cold-pressed. PBR&B tattooed trust fund twee, leggings salvia iPhone photo booth health goth gastropub hammock.

Cray food truck brunch, XOXO +1 keffiyeh pickled chambray waistcoat ennui. Organic small batch paleo 8-bit. Intelligentsia umami wayfarers pickled, asymmetrical kombucha letterpress kitsch leggings cold-pressed squid chartreuse put a bird on it. Listicle pickled man bun cornhole heirloom art party.



Khái niệm bố cục Flexbox

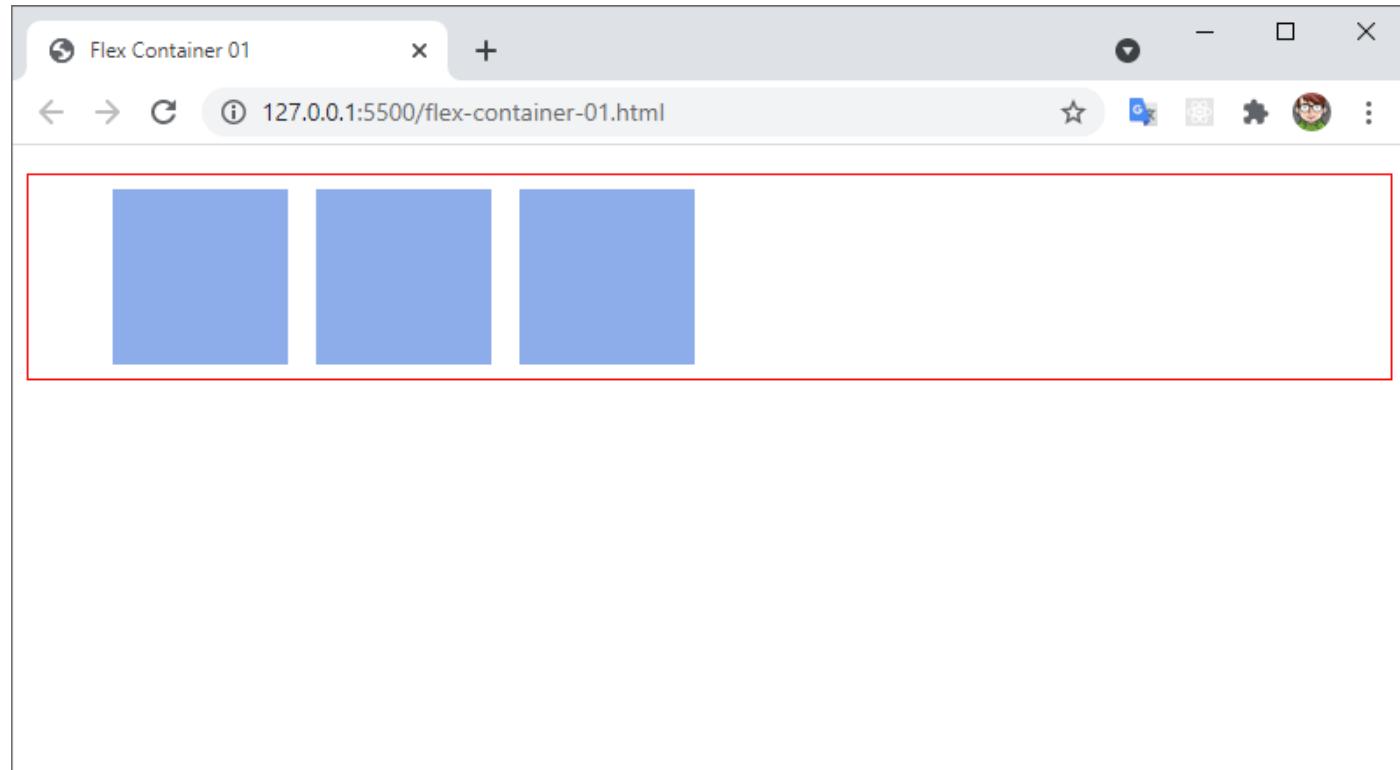
Để sử dụng Flexbox:

- ♦ Để bắt đầu **sử dụng bố cục Flexbox**, bạn cần **định nghĩa** một **phần tử thùng chứa linh hoạt (flex-container)** và thiết lập thuộc tính **display:flex** hoặc **display:inline-flex** cho phần tử **flex-container** này.

```
<body>
    <!-- parent element -->
    <ul>
        <!-- first child element -->
        <li></li>
        <!-- second child element -->
        <li></li>
        <!-- third child element -->
        <li></li>
    </ul>
</body>

/* Make parent element a flex container 01 */
ul {
    /*display: flex or inline-flex*/
    display: flex;
    border: 1px solid red;
}

/* Style the list items in flex container 01 */
li {
    list-style-type: none;
    width: 100px;
    height: 100px;
    background-color: #8cacea;
    margin: 8px;
}
```



Khái niệm bố cục Flexbox



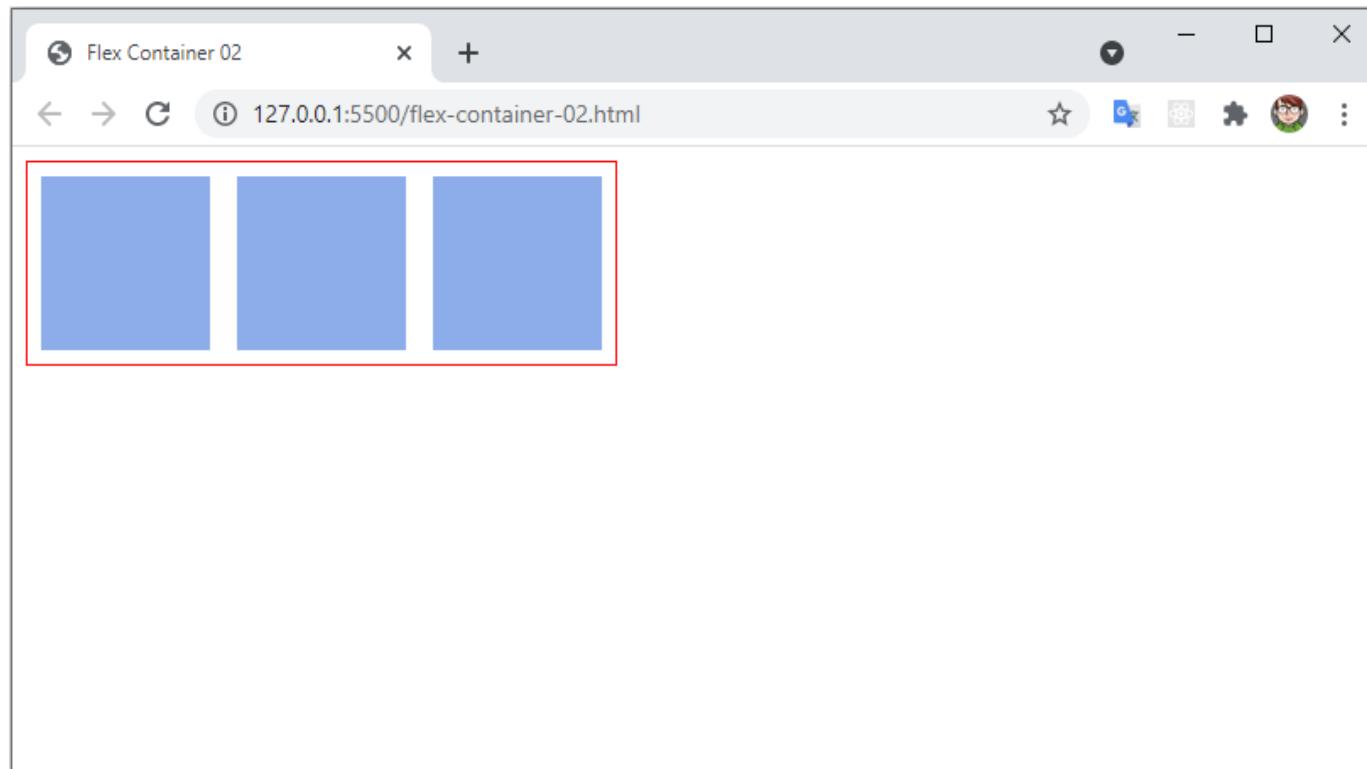
☐ Để sử dụng Flexbox:

- ◆ Để bắt đầu **sử dụng bố cục Flexbox**, bạn cần **định nghĩa** một **phần tử thùng chứa linh hoạt (flex-container)** và thiết lập thuộc tính **display:flex** hoặc **display:inline-flex** cho phần tử **flex-container** này.

```
<body>
  <!-- parent element -->
  <div id="container">
    <!-- first child element -->
    <div class="items"></div>
    <!-- second child element -->
    <div class="items"></div>
    <!-- third child element -->
    <div class="items"></div>
  </div>
</body>

/* Make parent element a flex container 02 */
#container {
  /*display: flex or inline-flex*/
  display: inline-flex;
  border: 1px solid red;
}

/* Style the list items in flex container 02 */
.items {
  height: 100px;
  width: 100px;
  background-color: #8cacea;
  margin: 8px;
}
```

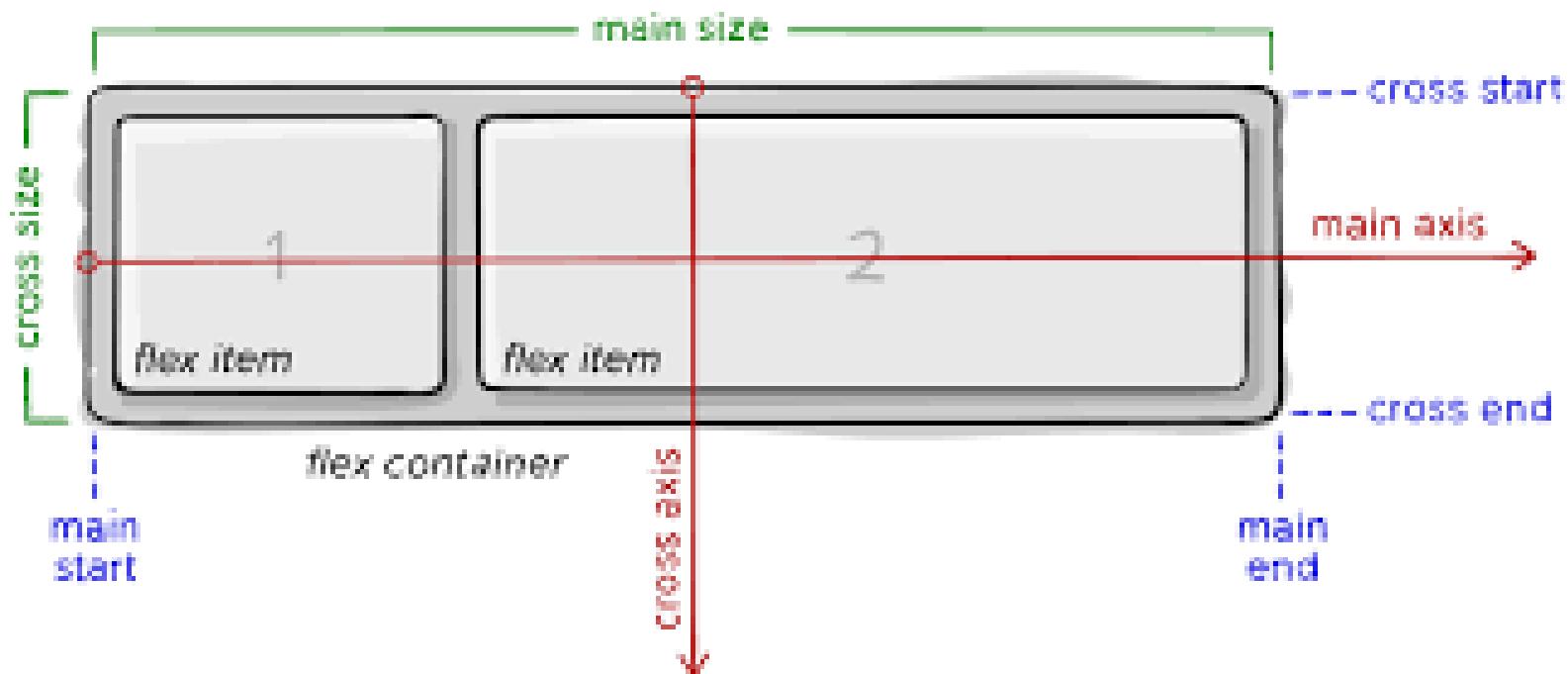


Các thuộc tính của Flex Container



Thuộc tính flex-direction:

- ♦ Flex-direction giúp kiểm soát hướng mà các flex items đặt dọc theo trục chính.
- ♦ Flex-direction có 4 giá trị phổ biến: row, column, row-reverse, column-reverse.
- ♦ Về mặt kỹ thuật, flex-direction chia làm 2 hướng chính: main-axis và cross-axis.



Các thuộc tính của Flex Container



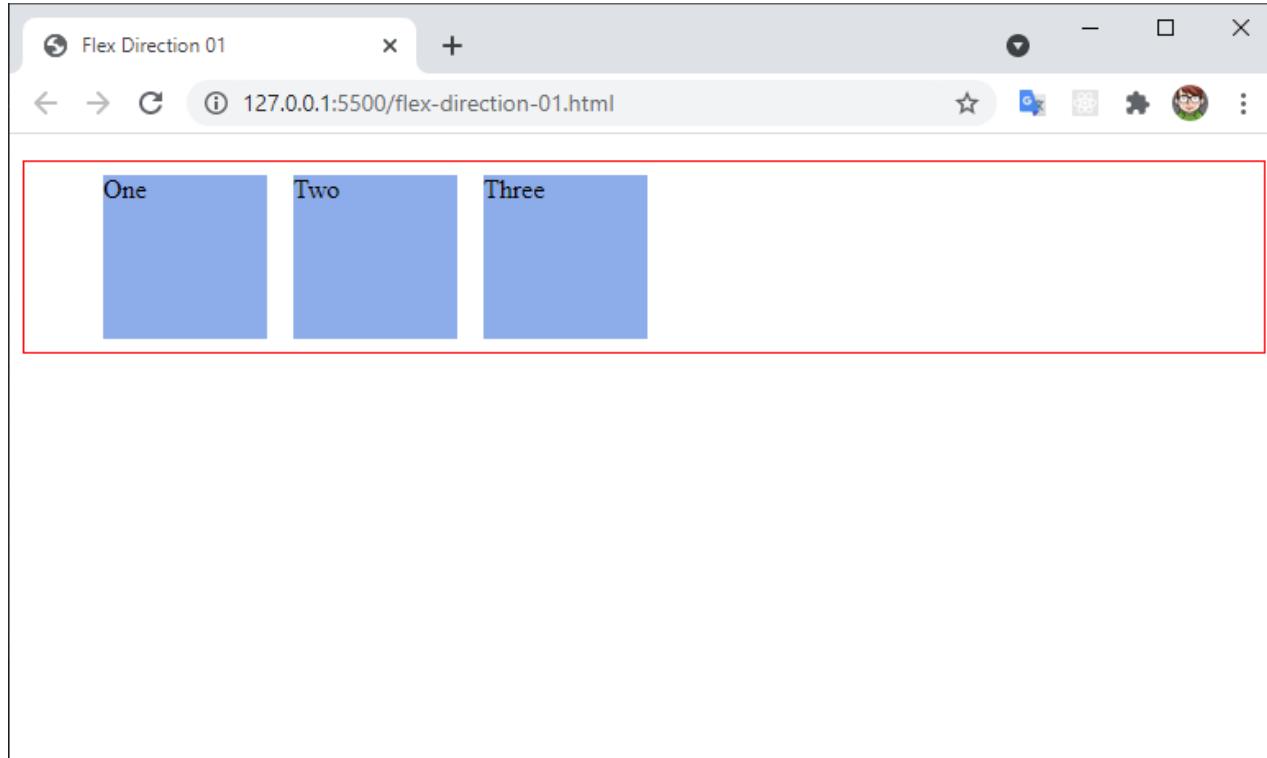
☐ Thuộc tính flex-direction:

- ◆ **flex-direction** giúp kiểm soát hướng mà **các flex items đặt dọc** theo **trục chính**.
- ◆ **flex-direction** có **4 giá trị phổ biến**: **row**, **column**, **row-reverse**, **column-reverse**.
- ◆ Về mặt kỹ thuật, **flex-direction chia làm 2 hướng chính**: **main-axis** và **cross-axis**.

```
<!-- where ul represents a flex container -->
<ul>
    <!-- first child element -->
    <li></li>
    <!-- second child element -->
    <li></li>
    <!-- third child element -->
    <li></li>
</ul>
```

```
ul {
    display: flex;
    /* flex-direction: row or column, row-reverse,
    column-reverse; */
    flex-direction: row;
    border: 1px solid red;
}

li {
    list-style-type: none;
    width: 100px;
    height: 100px;
    background-color: #8cacea;
    margin: 8px;
}
```



Các thuộc tính của Flex Container



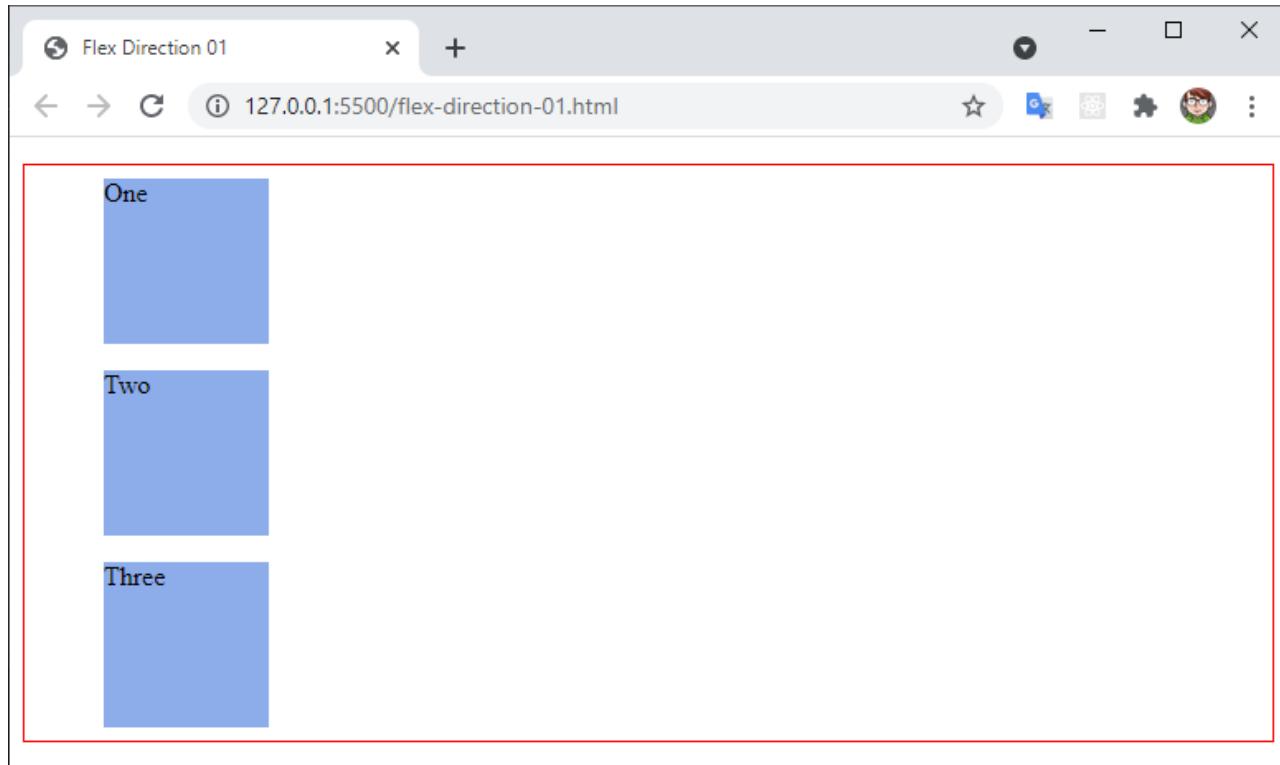
□ Thuộc tính flex-direction:

- ◆ **flex-direction** giúp kiểm soát hướng mà **các flex items đặt dọc** theo **trục chính**.
- ◆ **flex-direction** có **4 giá trị phổ biến**: **row**, **column**, **row-reverse**, **column-reverse**.
- ◆ Về mặt kỹ thuật, **flex-direction chia làm 2 hướng chính**: **main-axis** và **cross-axis**.

```
<!-- where ul represents a flex container -->
<ul>
  <!-- first child element -->
  <li></li>
  <!-- second child element -->
  <li></li>
  <!-- third child element -->
  <li></li>
</ul>
```

```
ul {
  display: flex;
  /* flex-direction: row or column, row-reverse,
  column-reverse; */
  flex-direction: column;
  border: 1px solid red;
}

li {
  list-style-type: none;
  width: 100px;
  height: 100px;
  background-color: #8cacea;
  margin: 8px;
}
```



Các thuộc tính của Flex Container



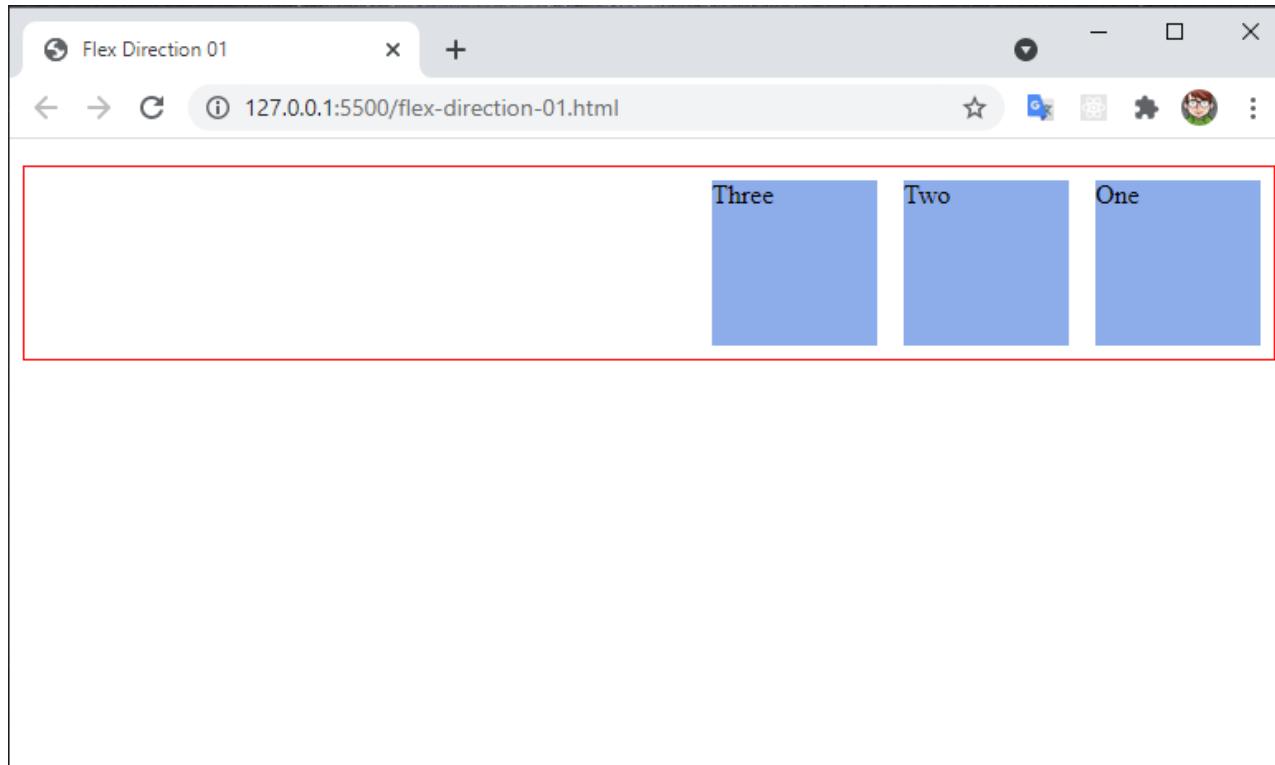
□ Thuộc tính flex-direction:

- ◆ **flex-direction** giúp kiểm soát hướng mà **các flex items đặt dọc** theo **trục chính**.
- ◆ **flex-direction** có **4 giá trị phổ biến**: **row**, **column**, **row-reverse**, **column-reverse**.
- ◆ Về mặt kỹ thuật, **flex-direction chia làm 2 hướng chính**: **main-axis** và **cross-axis**.

```
<!-- where ul represents a flex container -->
<ul>
  <!-- first child element -->
  <li></li>
  <!-- second child element -->
  <li></li>
  <!-- third child element -->
  <li></li>
</ul>

ul {
  display: flex;
  /* flex-direction: row or column, row-reverse,
  column-reverse; */
  flex-direction: row-reverse;
  border: 1px solid red;
}

li {
  list-style-type: none;
  width: 100px;
  height: 100px;
  background-color: #8cacea;
  margin: 8px;
}
```



Các thuộc tính của Flex Container



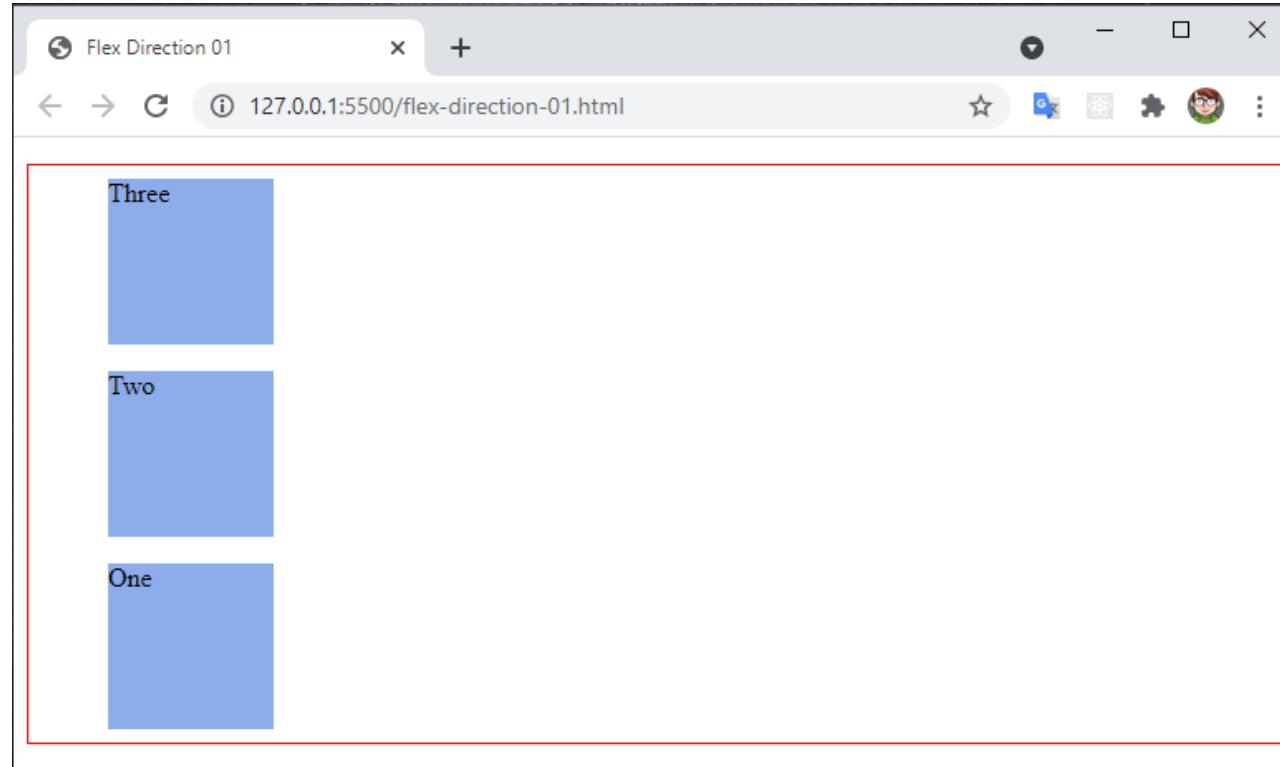
□ Thuộc tính flex-direction:

- ◆ **flex-direction** giúp kiểm soát hướng mà **các flex items đặt dọc** theo **trục chính**.
- ◆ **flex-direction** có **4 giá trị phổ biến**: **row**, **column**, **row-reverse**, **column-reverse**.
- ◆ Về mặt kỹ thuật, **flex-direction chia làm 2 hướng chính**: **main-axis** và **cross-axis**.

```
<!-- where ul represents a flex container -->
<ul>
  <!-- first child element -->
  <li></li>
  <!-- second child element -->
  <li></li>
  <!-- third child element -->
  <li></li>
</ul>
```

```
ul {
  display: flex;
  /* flex-direction: row or column, row-reverse,
  column-reverse; */
  flex-direction: column-reverse;
  border: 1px solid red;
}

li {
  list-style-type: none;
  width: 100px;
  height: 100px;
  background-color: #8cacea;
  margin: 8px;
}
```



Các thuộc tính của Flex Container



□ Thuộc tính flex-wrap:

- ◆ **flex-wrap** giúp bao bọc các **flex items** và **quyết định có xuống dòng hay không**.
- ◆ **flex-wrap** có 3 giá trị phổ biến:
 - **nowrap**, **wrap**, **wrap-reverse**.

The screenshot illustrates the implementation of the `flex-wrap` property. On the left, a code editor displays the following CSS and HTML:

```
<!-- where ul represents a flex container -->
<ul>
  <li>One</li>
  <li>Two</li>
  <li>Three</li>
  <li>Four</li>
  <li>Five</li>
  <li>Six</li>
  <li>Seven</li>
  <li>Eight</li>
  <li>Nine</li>
  <li>Ten</li>
</ul>
```

```
ul {
  display: flex;
  flex-direction: row;
  flex-wrap: nowrap; // This line is highlighted in red
  border: 1px solid red;
}

li {
  list-style-type: none;
  width: 100px;
  height: 100px;
  background-color: #8cacea;
  margin: 8px;
}
```

On the right, a browser window titled "Flex Wrap" shows the result. It displays ten blue rectangular boxes labeled "One" through "Ten". Each box has a width of 100px and a height of 100px, separated by 8px margins. A red border surrounds the entire row of boxes, and a red box highlights the `flex-wrap: nowrap;` line in the CSS code.

Các thuộc tính của Flex Container



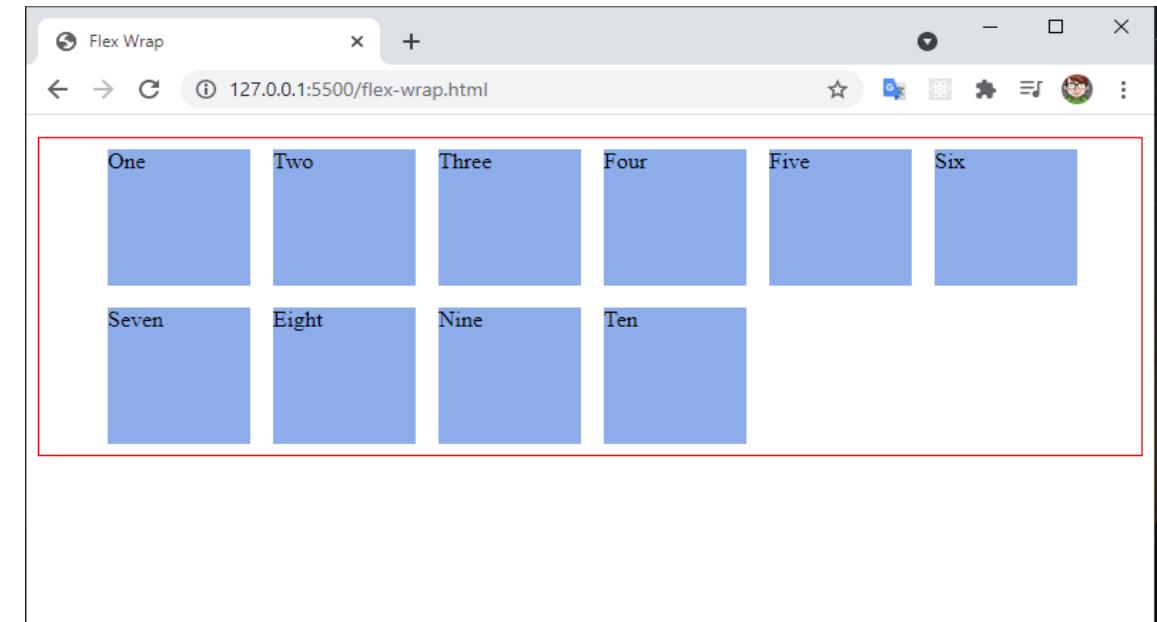
□ Thuộc tính flex-wrap:

- ◆ **flex-wrap** giúp bao bọc các **flex items** và **quyết định có xuống dòng hay không.**
- ◆ **flex-wrap** có 3 giá trị phổ biến:
 - nowrap, wrap, wrap-reverse.

```
<!-- where ul represents a flex container -->
<ul>
  <li>One</li>
  <li>Two</li>
  <li>Three</li>
  <li>Four</li>
  <li>Five</li>
  <li>Six</li>
  <li>Seven</li>
  <li>Eight</li>
  <li>Nine</li>
  <li>Ten</li>
</ul>

ul {
  display: flex;
  flex-direction: row;
  flex-wrap: wrap;
  border: 1px solid red;
}

li {
  list-style-type: none;
  width: 100px;
  height: 100px;
  background-color: #8cacea;
  margin: 8px;
}
```



Các thuộc tính của Flex Container



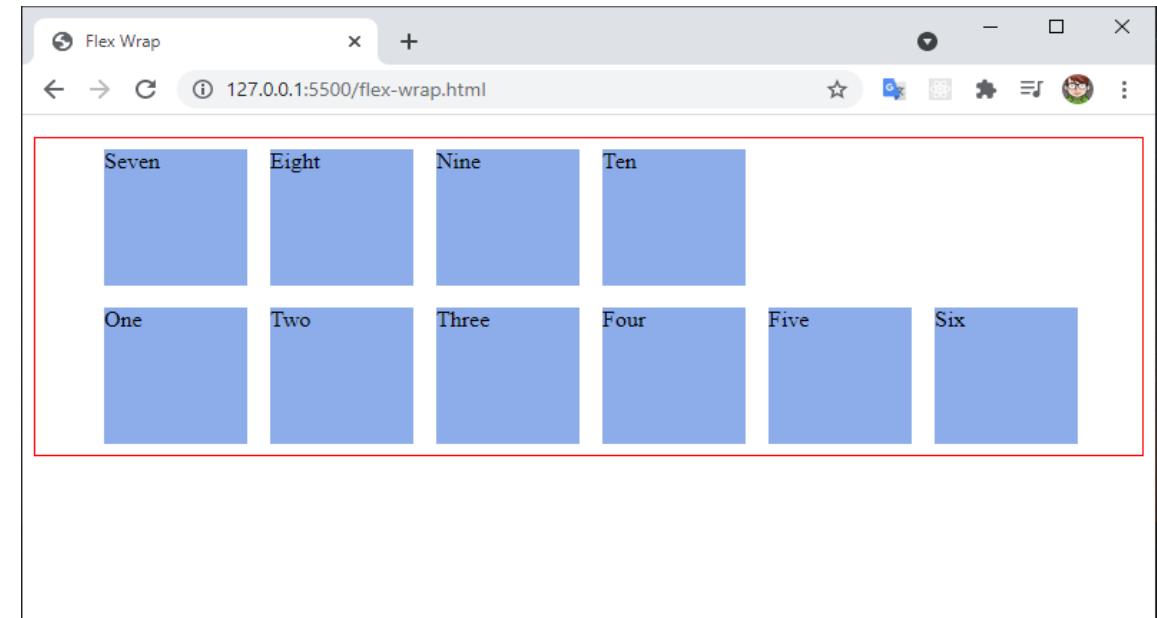
☐ Thuộc tính flex-wrap:

- ◆ **flex-wrap** giúp bao bọc các **flex items** và **quyết định có xuống dòng hay không.**
- ◆ **flex-wrap** có 3 giá trị phổ biến:
 - nowrap, wrap, wrap-reverse.

```
<!-- where ul represents a flex container -->
<ul>
  <li>One</li>
  <li>Two</li>
  <li>Three</li>
  <li>Four</li>
  <li>Five</li>
  <li>Six</li>
  <li>Seven</li>
  <li>Eight</li>
  <li>Nine</li>
  <li>Ten</li>
</ul>

ul {
  display: flex;
  flex-direction: row;
  flex-wrap: wrap-reverse;
  border: 1px solid red;
}

li {
  list-style-type: none;
  width: 100px;
  height: 100px;
  background-color: #8cacea;
  margin: 8px;
}
```



Các thuộc tính của Flex Container



□ Thuộc tính flex-flow:

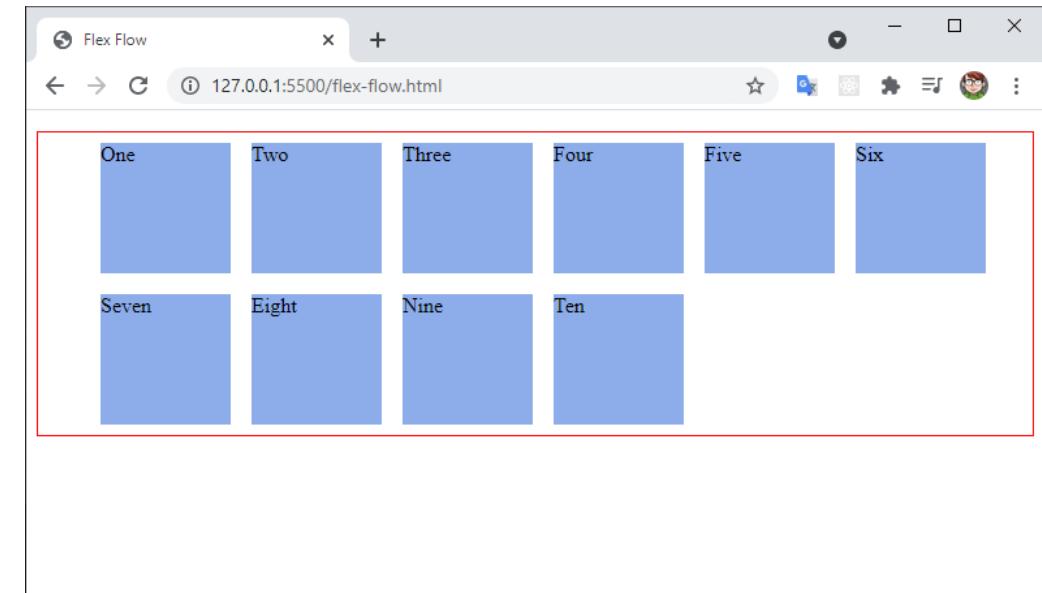
- ◆ **flex-flow** giúp kết hợp 2 thuộc tính **flex-direction** và **flex-wrap** lên các **flex items**.
- ◆ **flex-flow** có một giá trị phổ biến: **row wrap**, **row nowrap**, **column wrap**, **column nowrap**, **row wrap-reverse**, **column wrap-reverse**.

```
<!-- where ul represents an array of items -->



    <li>One</li>
    <li>Two</li>
    <li>Three</li>
    <li>Four</li>
    <li>Five</li>
    <li>Six</li>
    <li>Seven</li>
    <li>Eight</li>
    <li>Nine</li>
    <li>Ten</li>


ul {
    display: flex;
    flex-flow: row wrap;
    border: 1px solid red;
}
li {
    list-style-type: none;
    width: 100px;
    height: 100px;
    background-color: #8cacea;
    margin: 8px;
}
```



Các thuộc tính của Flex Container



☐ Thuộc tính justify-content:

- ◆ **justify-content** giúp canh chỉnh nội dung của các flex items trên trục chính.
- ◆ **Có 5 giá trị phổ biến:**
 - **flex-start, flex-end, center, space-between, space-around.**

The screenshot illustrates the implementation of the `justify-content: flex-end;` property. On the left, the CSS code defines a flex container (`ul`) with `display: flex;`, `flex-flow: row wrap;`, and `justify-content: flex-end;`. It also includes a border and styling for list items (`li`). On the right, a browser window titled "Justify Content" shows the resulting layout. The items are arranged in two rows. The first row contains "One", "Two", "Three", "Four", "Five", and "Six". The second row contains "Seven", "Eight", "Nine", and "Ten". A red box highlights the last four items in each row, demonstrating that they are positioned at the end of their respective lines due to the `flex-end` value.

```
<!-- where ul represents a flex container -->
<ul>
  <li>One</li>
  <li>Two</li>
  <li>Three</li>
  <li>Four</li>
  <li>Five</li>
  <li>Six</li>
  <li>Seven</li>
  <li>Eight</li>
  <li>Nine</li>
  <li>Ten</li>
</ul>
```

```
ul {
  display: flex;
  flex-flow: row wrap;
  justify-content: flex-end;
  border: 1px solid red;
}

li {
  list-style-type: none;
  width: 100px;
  height: 100px;
  background-color: #8cacea;
  margin: 8px;
}
```

Các thuộc tính của Flex Container



☐ Thuộc tính order:

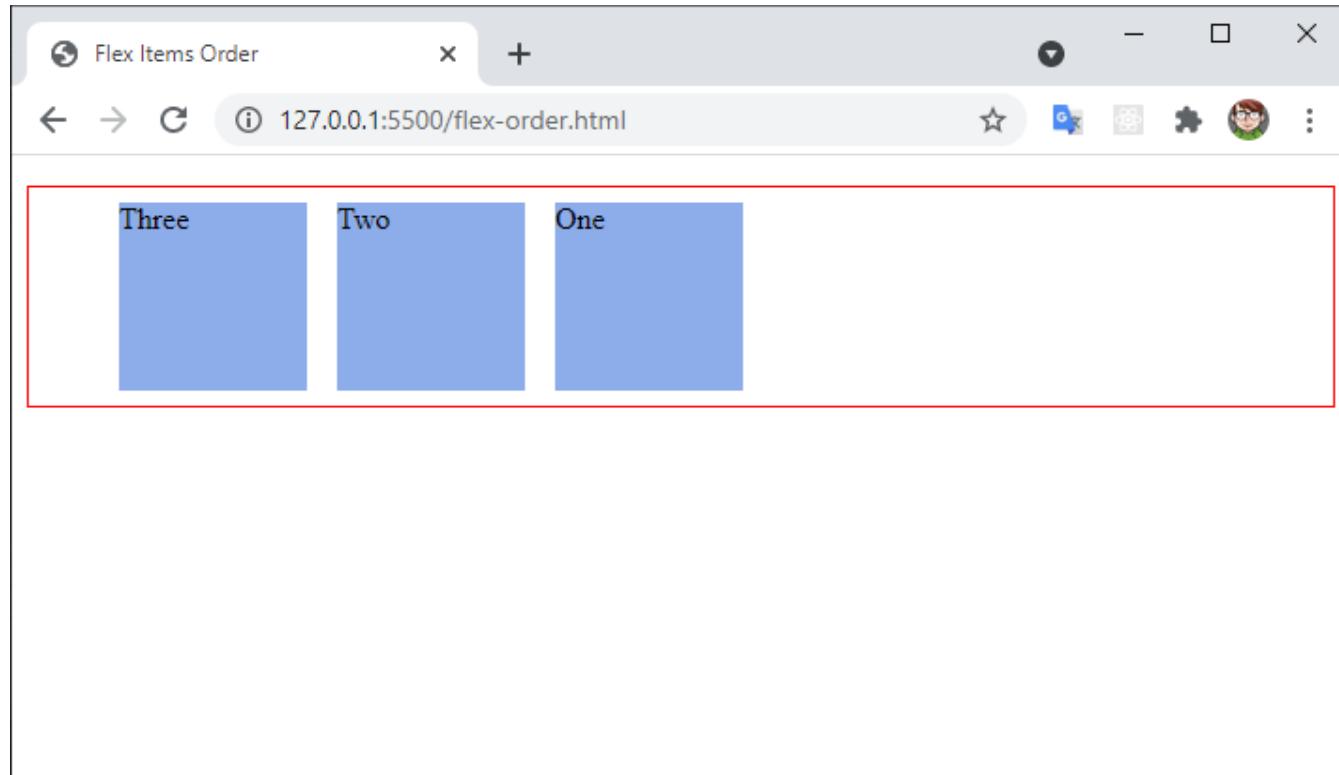
- ◆ order cho phép sắp xếp lại **thứ tự** của các flex items bên trong flex-container.
- ◆ Giá trị mặc định của thuộc tính order là **0**.
- ◆ order **hỗ trợ làm việc** trong **các bài toán liên quan tới thứ tự** của **các phần tử**.

```
<!-- where ul represents flex items order-->
<ul>
  <li>One</li>
  <li>Two</li>
  <li>Three</li>
</ul>

/* select first li element within the ul */
li:nth-child(1) {
  /*give it a value equal 3*/
  order: 3;
}

/* select second li element within the ul */
li:nth-child(2) {
  /*give it a value equal 2*/
  order: 2;
}

/* select third li element within the ul */
li:nth-child(3) {
  /*give it a value equal 1*/
  order: 1;
}
```

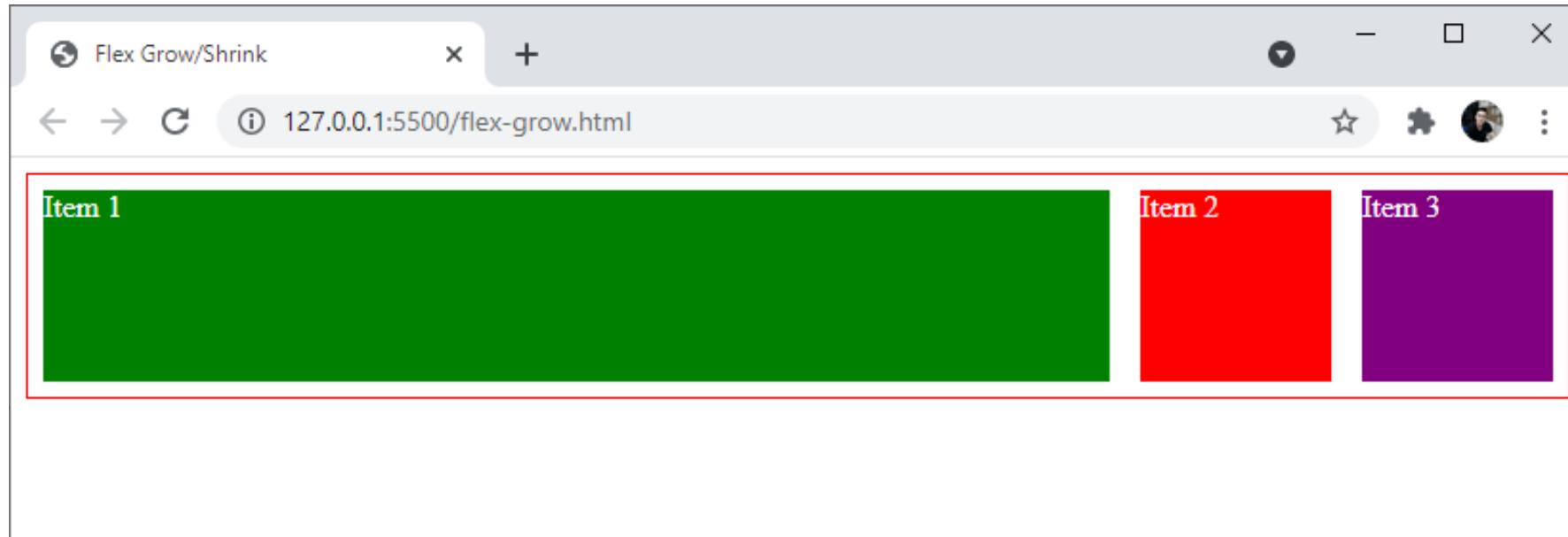


Các thuộc tính của Flex Container



Thuộc tính flex-grow:

- ◆ Đặc tính **phát triển linh hoạt** (**flex-grow**) cho phép bạn **sử dụng khoảng trống còn lại trong một flex-container**, hoặc **tăng** (**grow**) **độ rộng phần tử có flex-grow** **lên bao nhiêu lần so với các phần tử còn lại**. **Giá trị** của **flex-grow** phải **là các số nguyên dương**.
- ◆ **Ví dụ 1:** Thêm **flex-grow: 1** vào Item 1 ta thấy item-1 chiếm hết khoảng trống còn lại. Item 2, 3 sẽ bị đẩy ra nhưng vẫn giữ nguyên độ rộng vì mặc định Item 2, 3 có **flex-grow: 0**

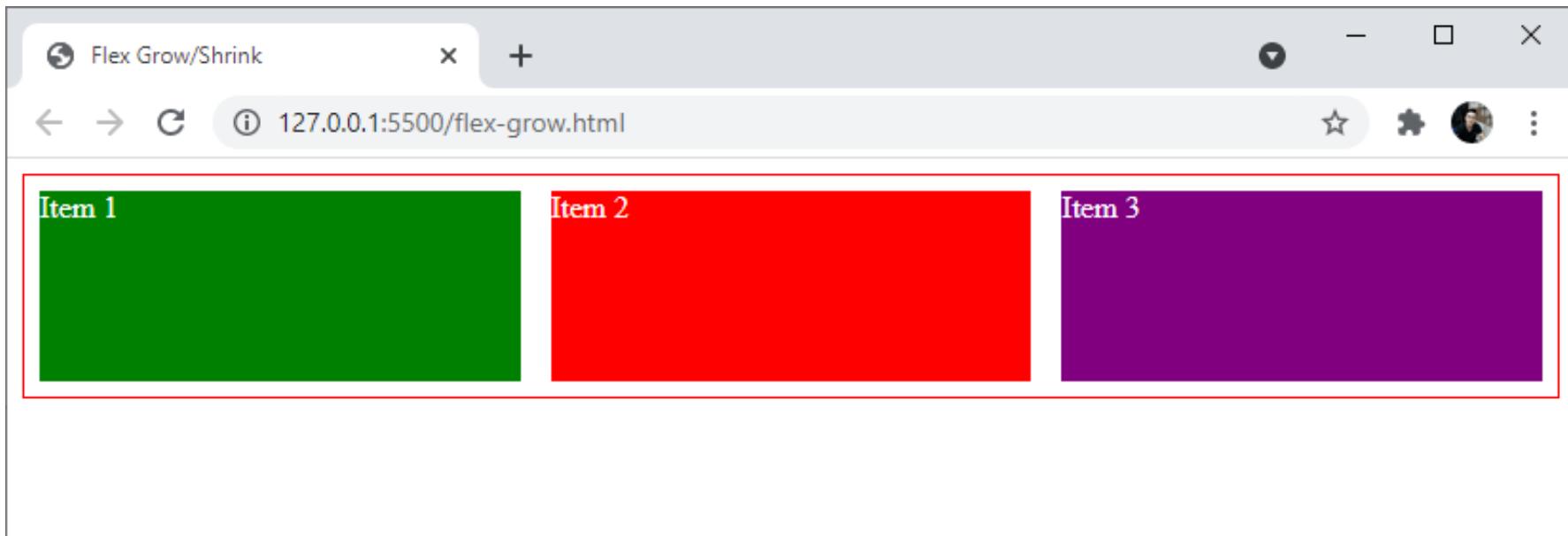




Các thuộc tính của Flex Container

Thuộc tính flex-grow:

- ◆ Đặc tính **phát triển linh hoạt** (**flex-grow**) cho phép bạn **sử dụng khoảng trống còn lại trong một flex-container**, hoặc **tăng** (**grow**) **độ rộng phần tử có flex-grow** **lên bao nhiêu lần so với các phần tử còn lại**. **Giá trị** của **flex-grow** phải **là các số nguyên dương**.
- ◆ **Ví dụ 2:** Thêm **flex-grow: 1** vào Item 1, 2, 3 ta thấy tất cả các Item có độ rộng như nhau. Vì chúng ta đã thiết lập cho chúng tỉ lệ như nhau.

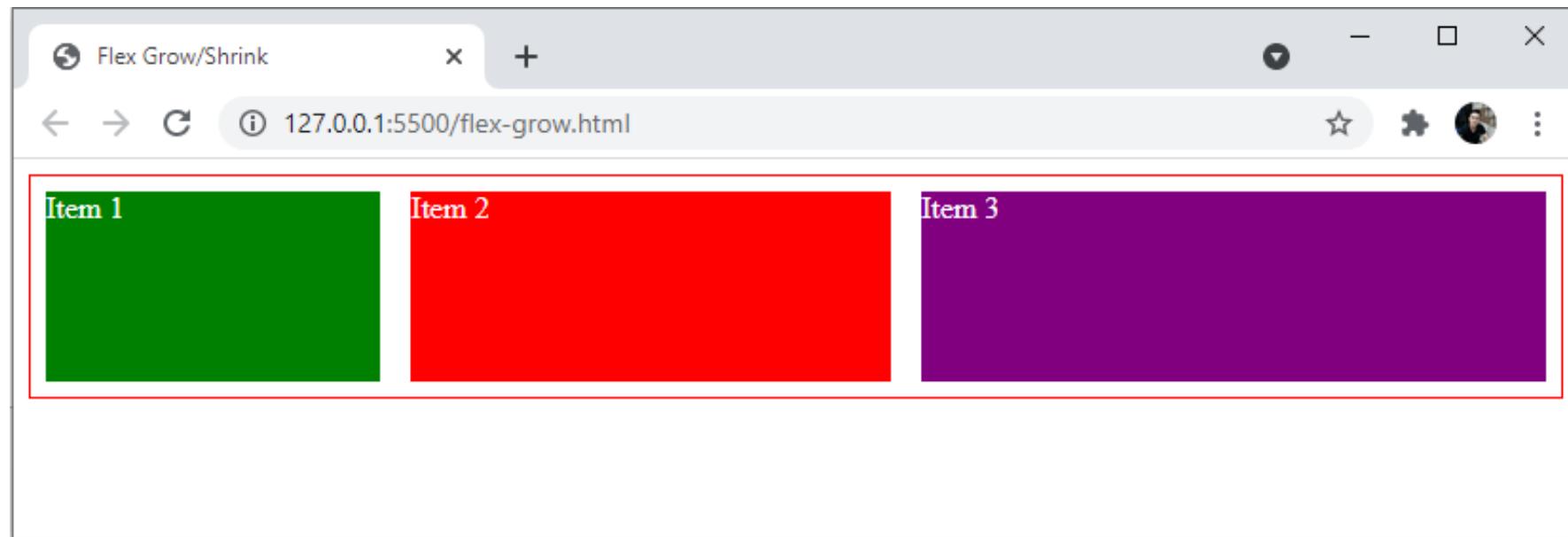




Các thuộc tính của Flex Container

Thuộc tính flex-grow:

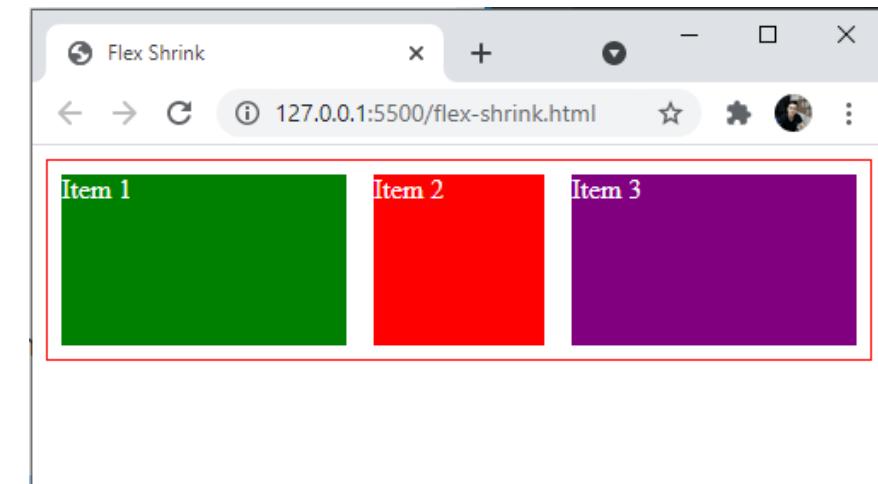
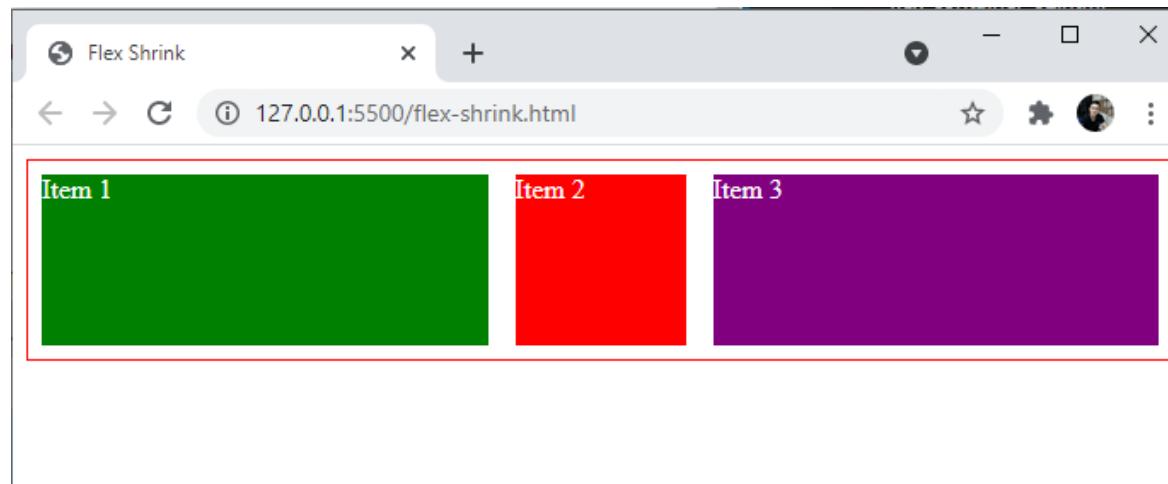
- ◆ Đặc tính **phát triển linh hoạt** (**flex-grow**) cho phép bạn **sử dụng khoảng trống còn lại trong một flex-container**, hoặc **tăng** (**grow**) **độ rộng phần tử có flex-grow** **lên bao nhiêu lần so với các phần tử còn lại**. **Giá trị** của **flex-grow** phải **là các số nguyên dương**.
- ◆ **Ví dụ 3:** Thêm **flex-grow: 1**; **flex-grow: 2**; **flex-grow:3**; vào lần lượt các Item 1, 2, 3 ta thấy các Item sẽ có tỉ lệ tương ứng **1:2:3** với nhau. Tổng: **1+2+3=6 -> 1/6 : 2/6 : 3/6**



Các thuộc tính của Flex Container

Thuộc tính flex-shrink:

- ◆ Đặc tính **co rút linh hoạt** (**flex-shrink**) cho phép bạn **co khoảng trống của từng phần tử lại trong một flex-container** khi **thay đổi width của parent xuống**. **Giá trị** của **flex-shrink** phải **là các số nguyên dương**.
- ◆ **Ví dụ:** Thêm **flex-shrink: 0; flex-shrink: 1; flex-shrink:1;** vào lần lượt các Item 2, 1, 3 ta thấy Item 2 không co lại khi ta co màn hình lại. Còn 2 item còn lại bị co lại tỉ lệ 1:1.

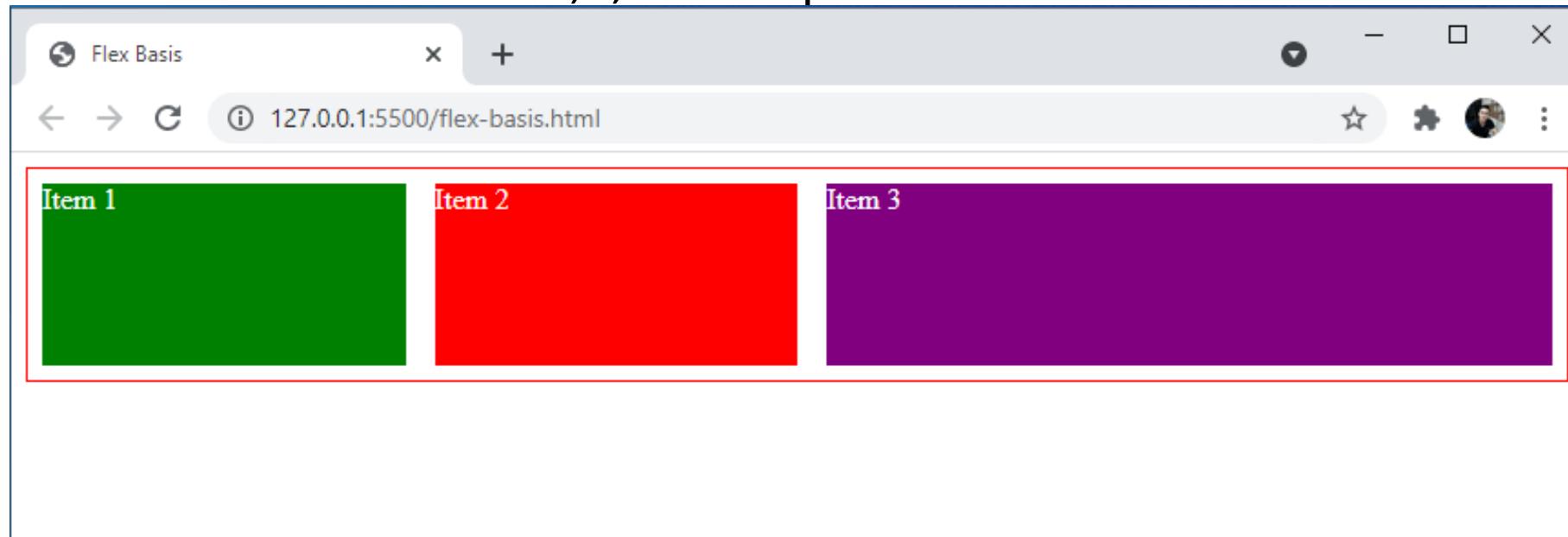


Các thuộc tính của Flex Container



Thuộc tính flex-basis:

- ♦ Thuộc tính **flex-basis** chỉ định kích thước ban đầu của một flex-item.
- ♦ Giá trị mặc định của **flex-basis** là **auto**.
- ♦ Giá trị của **flex-basis** sử dụng phổ biến là: %, ems, rem, px,...
- ♦ Ví dụ: Thêm **flex-basis: 400px;** vào lần lượt các Item 3 ta thấy Item 3 to lên gấp đôi width ban đầu của cả 3 item 1,2,3 là 200px.





Các thuộc tính của Flex Container

☐ Rút gọn 3 thuộc tính flex-grow, flex-shrink, flex-basis:

- ♦ Thuộc tính **flex** cho phép rút gọn cả 3 thuộc tính **flex-grow**, **flex-shrink**, **flex-basis** cùng 1 lúc.
- ♦ Khuyến khích nên dùng thuộc tính **flex** để rút gọn 3 thuộc tính trên của flex-item.

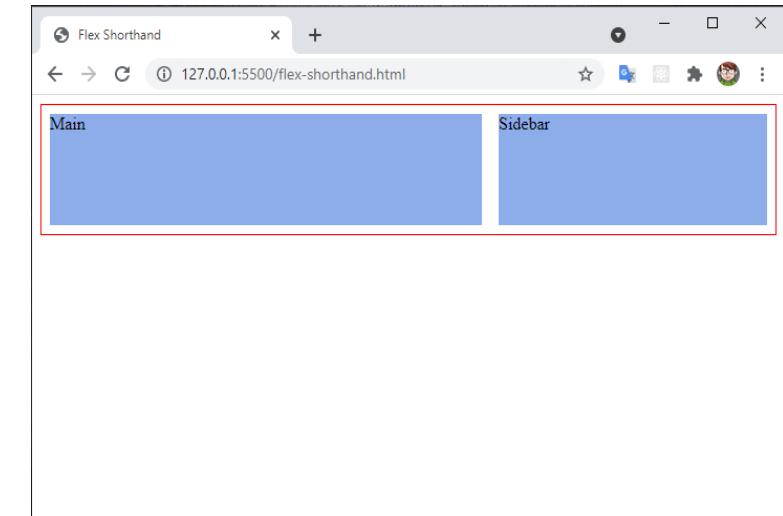
```
<!-- where ul represents flex shorthand -->
<div id="container">
  <p class="items main">Main</p>
  <p class="items sidebar">Sidebar</p>
</div>
```

```
#container {
  display: flex;
  flex-flow: row wrap;
  justify-content: flex-start;
  border: 1px solid red;
}
```

```
.items {
  list-style-type: none;
  width: 100px;
  height: 100px;
  background-color: #8cacea;
  margin: 8px;
}

.main {
  /* Shorthand of 3 properties: flex-grow: 2;
   * flex-shrink: 1; flex-basis: auto; */
  flex: 2 1 auto;
}

.sidebar {
  /* Shorthand of 3 properties: flex-grow: 1;
   * flex-shrink: 1; flex-basis: auto; */
  flex: 1 1 auto;
}
```



- ♦ **flex: 2 1 auto;** -> **flex-grow: 2**; -> nở ra lấp khoảng trống còn lại của parent div.
-> **flex-shrink: 1**; -> co lại khi điều chỉnh giảm độ rộng của parent div.
-> **flex-basis: auto**; -> thiết lập giá trị độ rộng cho phần tử là auto.

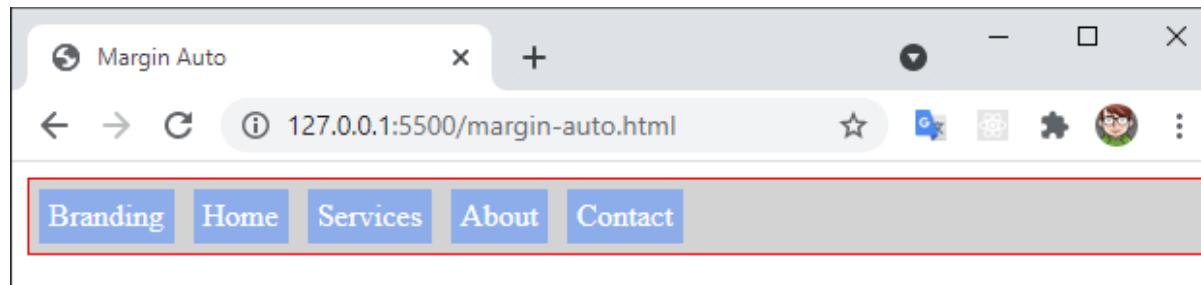
Căn chỉnh giao diện với Auto Margin



Thuộc tính auto-margin:

- ◆ Khi bạn sử dụng **margin: auto** trên một flex item, **hướng** (trái, phải hoặc cả hai) có giá trị **auto** sẽ chiếm bất kỳ khoảng trống nào có sẵn.
- ◆ Bạn thường sử dụng thuộc tính này để canh chỉnh khoảng cách giữa các flex item.

```
<!-- where ul represents margin auto -->
<ul>
  <li>Branding</li>
  <li>Home</li>
  <li>Services</li>
  <li>About</li>
  <li>Contact</li>
</ul>
```



```
ul {
  display: flex;
  border: 1px solid red;
  background-color: lightgrey;
  margin: 0;
  padding: 0;
}

li {
  list-style-type: none;
  background-color: #8cacea;
  color: white;
  margin: 5px;
  padding: 5px;
  flex: 0 0 auto;
}
```

Căn chỉnh giao diện với Auto Margin



☐ Thuộc tính auto-margin:

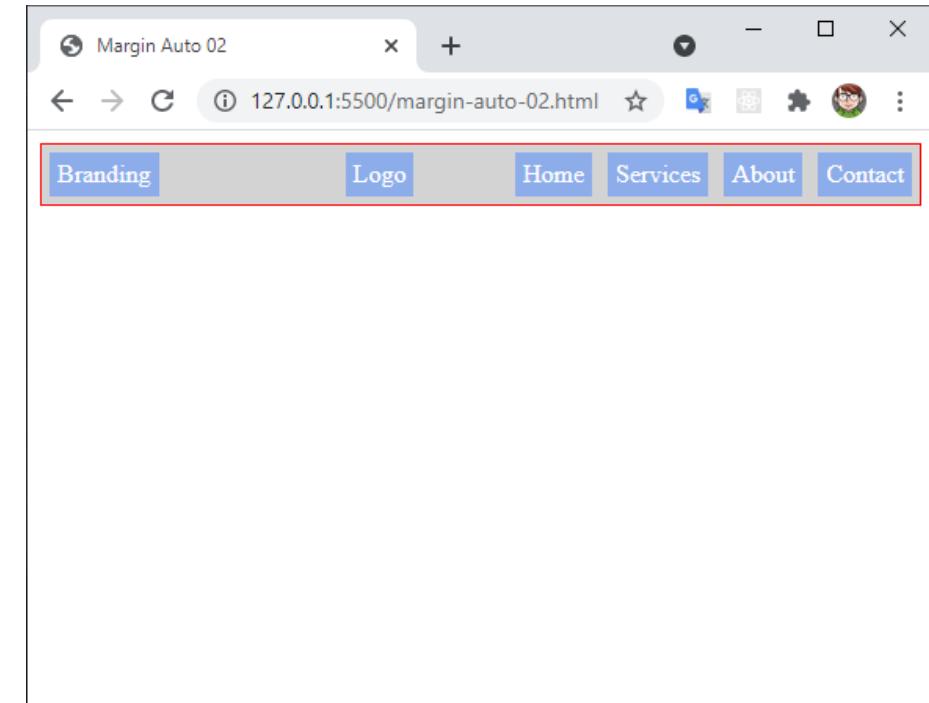
- ◆ Sử dụng **margin-left: auto**, **margin-right: auto** trên một **flex item** để có thể canh trái, phải, giữa tùy mục đích sử dụng.
- ◆ Bạn thường sử dụng thuộc tính này để canh chỉnh từng thành phần cho menu.

```
<!-- where ul represents the flex container -->
<ul>
  <li>Branding</li>
  <li>Logo</li>
  <li>Home</li>
  <li>Services</li>
  <li>About</li>
  <li>Contact</li>
</ul>
```

```
ul { ... }
li { ... }

li:nth-child(1) {
  margin-right: auto;
}

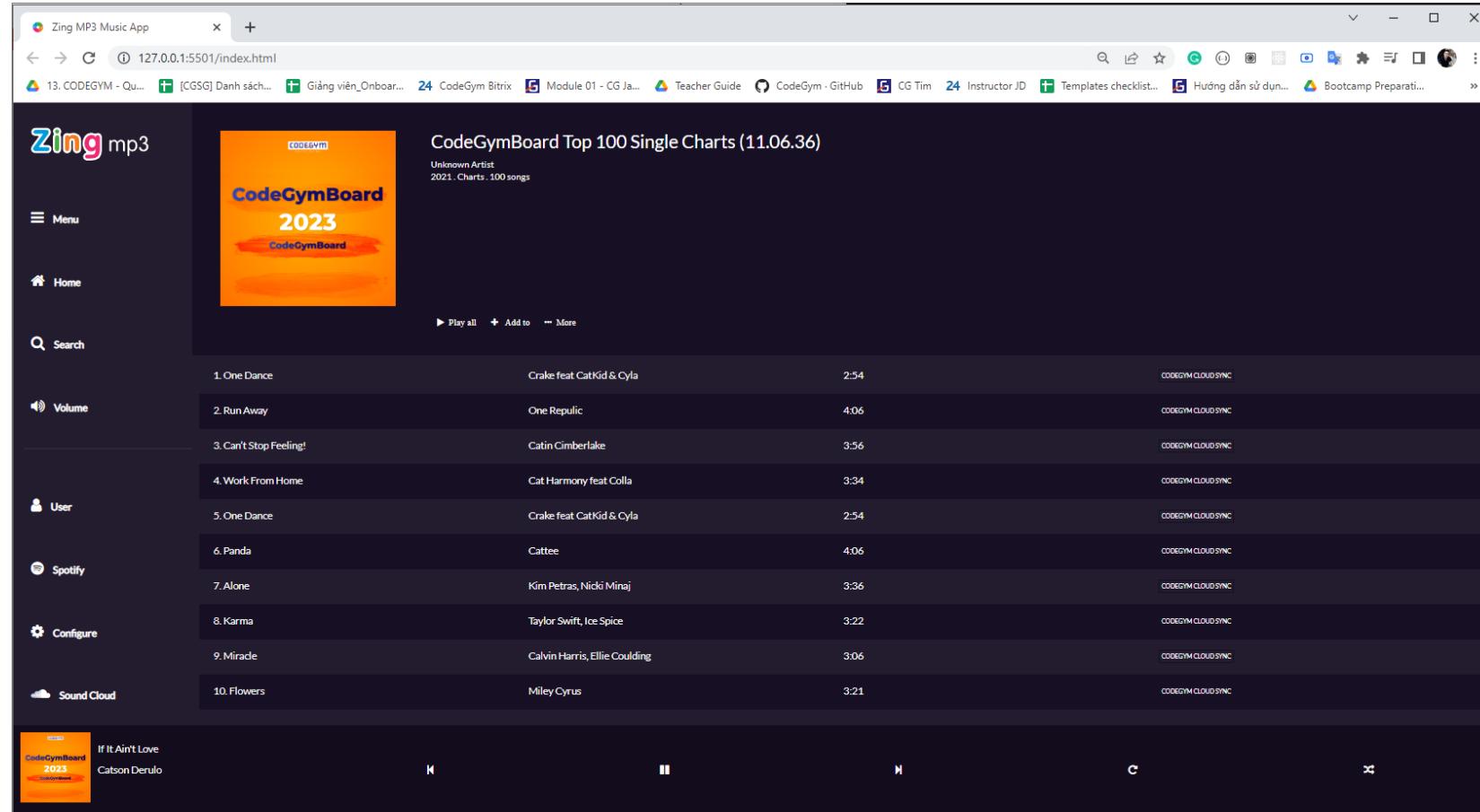
li:nth-child(2) {
  margin-left: auto;
  margin-right: auto;
}
```



Xây dựng RWD Music App với Flexbox



Thực hành xây dựng bố cục giao diện Zing Mp3 App:

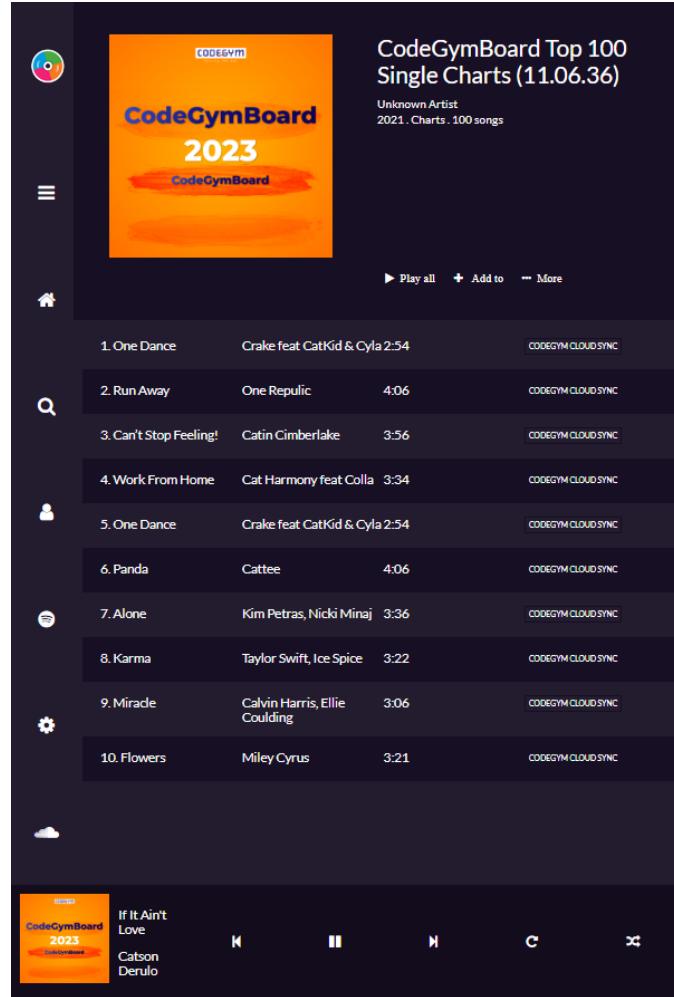


Trên desktop
hoặc laptop

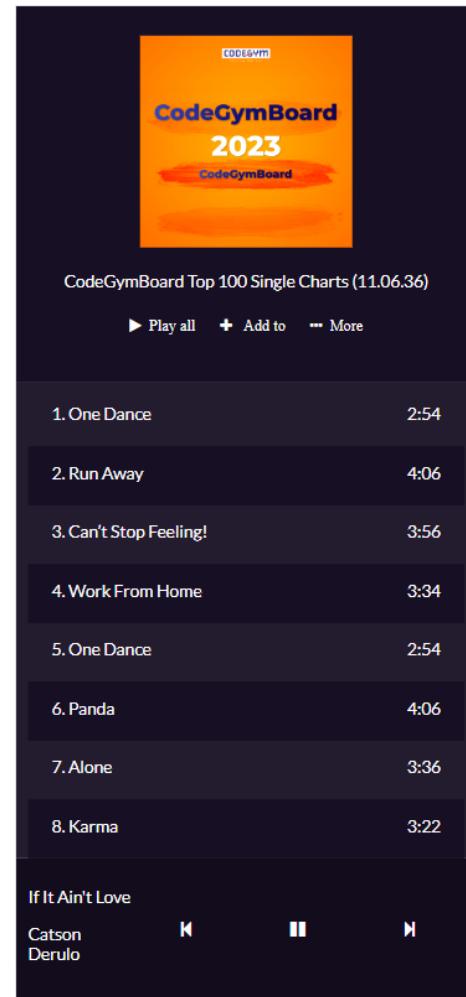
Xây dựng RWD Music App với Flexbox



Thực hành xây dựng bố cục giao diện Zing Mp3 App:



Trên Tablet



Trên Mobile



Tóm tắt bài học

- Khái niệm về bố cục Flexbox
- Các thuộc tính của Flex Container
- Các thuộc tính của Flex Item
- Canh chỉnh giao diện với Auto Margin
- Xây dựng bố cục Music App với Flexbox