



FPT POLYTECHNIC

THIẾT KẾ WEB ĐÁP ỨNG MỌI THIẾT BỊ

BÀI 3:

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

www.poly.edu.vn

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



Phần I: Dàn bố cục sử dụng Flexbox CSS

- ❖ Các thành phần chính của Flexbox
- ❖ Các thuộc tính của Flex-container

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

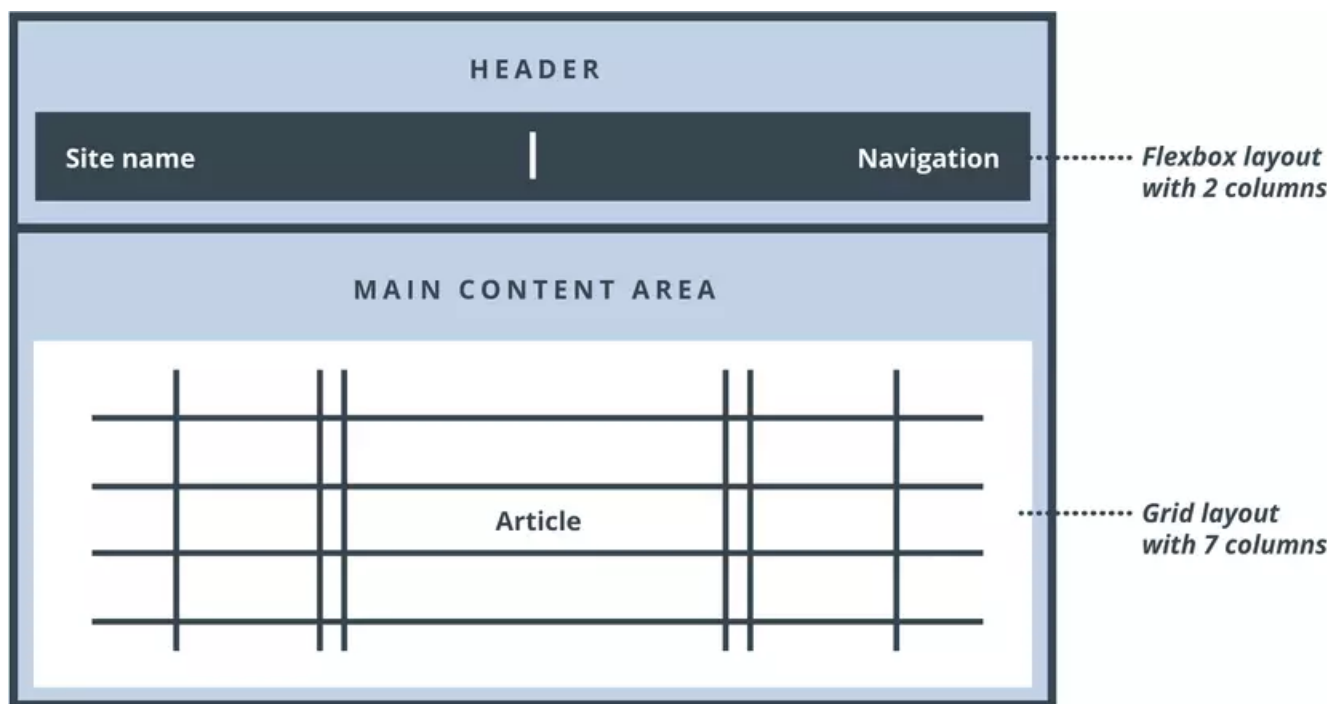
- ❖ Các thuộc tính của Flex-item
- ❖ Kết hợp Flexbox với Media Query



BÀI 3: **THIẾT KẾ WEB ĐÁP ỨNG SỬ DỤNG FLEXBOX CSS**

PHẦN I: DÀN BỐ CỤC SỬ DỤNG FLEXBOX

Giả sử, chúng ta có 1 layout của 1 landing page. Trang web bao gồm 2 sections: header và vùng content chính. Đối với header, sử dụng CSS Flexbox để cho vị trí của site name ở bên cạnh navigation. Còn đối với main content area thì áp dụng Grid Layout để bố trí layout của article thành 7 cột.

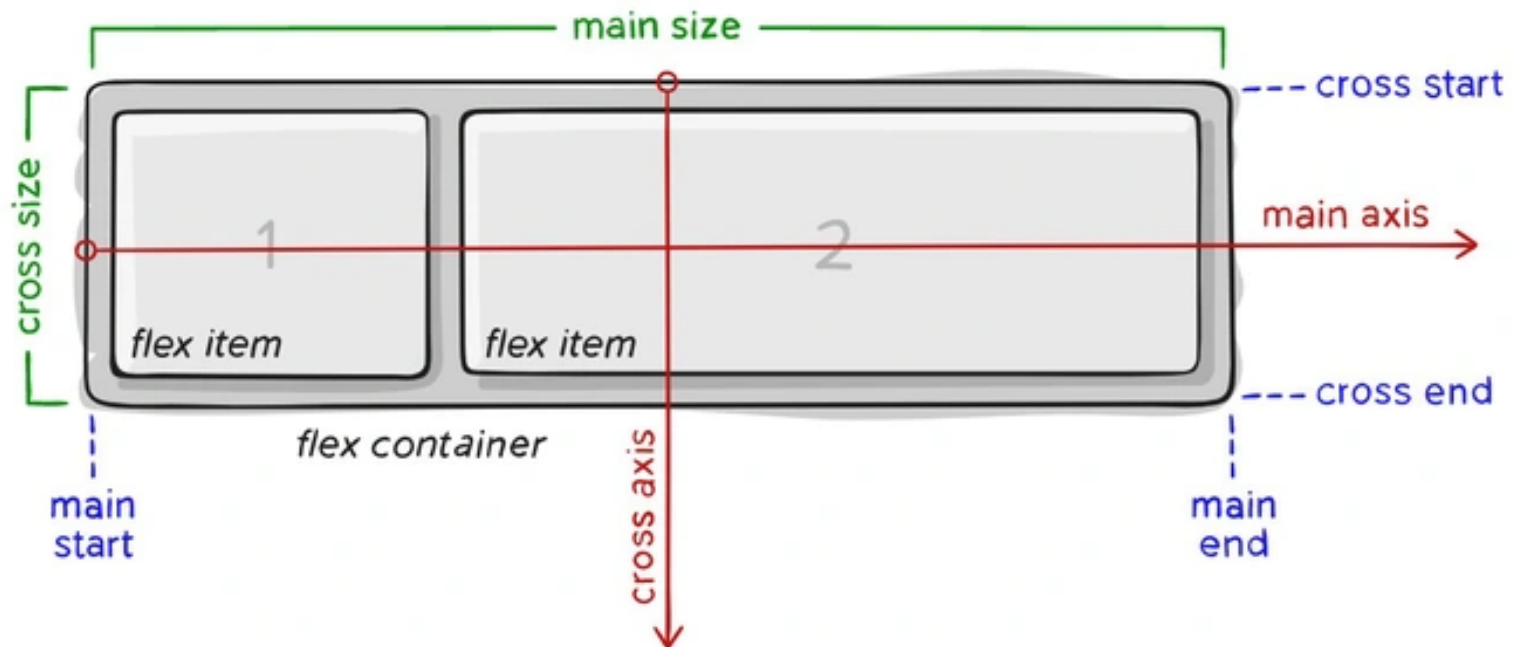


- ❑ Bố cục Flexbox phù hợp nhất với các thành phần của ứng dụng và các bố cục quy mô nhỏ như các khung trong trang web.
- ❑ Ngược lại, đối với các layout phạm vi lớn hơn, ví dụ như chia cột cho website, thì nên sử dụng bố cục lưới (Grid layout).

- ❑ Flexbox là một trong những tính năng của CSS3, thường được sử dụng trong các thiết kế responsive, giúp hiển thị các phần tử linh hoạt trên nhiều thiết bị và kích thước màn hình khác nhau.
- ❑ Flexbox (Flexible Box) là một mô-đun CSS cung cấp một cách hiệu quả hơn để bố trí, căn chỉnh các phần tử và khoảng cách giữa chúng trong một container, ngay cả khi kích thước của các phần tử chưa được quy định hoặc linh hoạt. Đó là lí do nó có tên 'flexible box' (linh hoạt).

CÁC THÀNH PHẦN TRONG FLEXBOX

Các thành phần của flexbox (flex-container và flex-item) được bố trí dần theo trục chính **main axis** (kéo dài từ main-start tới main-end) hoặc trục chéo **cross axis** (từ cross-start đến cross-end).



- ❑ **main axis**: Trục chính của flex container, là trục chính dọc theo đó các phần tử flex được xếp ra. Lưu ý, nó không nhất thiết phải nằm ngang; nó phụ thuộc vào thuộc tính flex-direction.
- ❑ **main-start | main-end**: Các flex items được đặt trong container bắt đầu từ main-start và đến main-end.
- ❑ **main size**: là chiều rộng hoặc chiều cao của một flex item, tùy theo kích thước chính. Thuộc tính main size sẽ có giá trị bằng thuộc tính width hoặc height của item tùy theo chiều của flex-direction là dọc hay ngang.
- ❑ **cross axis**: trục vuông góc với trục chính được gọi là trục chéo. Hướng của nó phụ thuộc vào hướng trục chính.
- ❑ **cross-start | cross-end**: hai trục này vuông góc với main-start và main-end.
- ❑ **cross size**: chiều rộng hoặc chiều cao của phần tử nằm trong flexbox. Thuộc tính này ngược lại so với main size (nếu main size là 'width' thì cross size là 'height' và ngược lại).

Dưới đây là một số thuộc tính có thể sử dụng đối với flex-container:

- ☐ display
- ☐ flex-direction
- ☐ flex-wrap
- ☐ flex-flow
- ☐ justify-content
- ☐ align-items
- ☐ align-content



1. Thuộc tính display:

Dùng để định nghĩa một flex-container, và cũng là thuộc tính bắt buộc nếu muốn làm việc với flexbox.

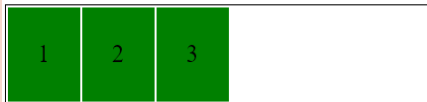
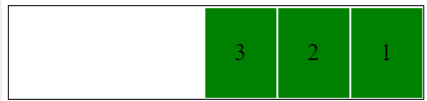


```
<div class="box">  
  <div class="box-item">1</div>  
  <div class="box-item">2</div>  
  <div class="box-item">3</div>  
</div>
```

```
.box {  
  display: flex;  
}  
.box-item {  
  background-color: green;  
  padding: 20px;  
  border: 1px solid #fff;  
}
```



2. Thuộc tính flex-direction:

Dùng để chỉ định hướng hiển thị của các item, việc thay đổi hướng hiển thị flex cũng có thể cho phép thay đổi vị trí của các flex item.

Thuộc tính: giá trị;	Mô tả	Ví dụ
flex-direction: row	Là giá trị mặc định khi sử dụng flexbox, không thực hiện bất kỳ thay đổi nào, chỉ đặt các item từ trái qua phải theo trục chính.	
flex-direction: row-reverse	Ngược lại với row, các item sẽ được đặt từ phải qua trái.	
flex-direction: column	Trục chính sẽ đi từ trên xuống dưới và các items sẽ được xếp thành hàng dọc.	
flex-direction: column-reverse	Các items sẽ được xếp thành hàng dọc nhưng theo chiều ngược lại	

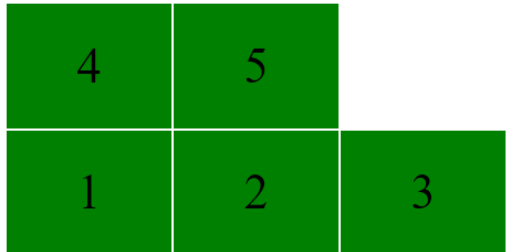
3. Thuộc tính flex-wrap:

Dùng để kiểm soát việc bọc các items nằm gọn trong container. Nếu giảm chiều rộng của trình duyệt, có thể không nhìn thấy một số item trên cùng một dòng.

Thuộc tính flex-wrap có thể giải quyết vấn đề đó:

- ❑ **nowrap**: mặc định, tất cả các item sẽ nằm trên một dòng.
- ❑ **wrap**: khi kích thước container thay đổi và tổng chiều rộng các item cộng lại lớn hơn chiều rộng của container thì item sẽ tự động xuống dòng.
- ❑ **wrap-reverse**: tương tự như **wrap**, nhưng thay vì xuống dòng thì item sẽ tự động nhảy lên trên.

Ví dụ:

Thuộc tính: giá trị;	Ví dụ
<pre>.flex-container { display: flex; flex-wrap: nowrap; }</pre>	
<pre>.flex-container { display: flex; flex-wrap: wrap; }</pre>	
<pre>.flex-container { display: flex; flex-wrap: wrap-reverse; }</pre>	

3. Thuộc tính flex-flow:

Dùng để gộp chung hai thuộc tính flex-direction và flex-wrap.

Cú pháp:

```
flex-flow: <'flex-direction'> || <'flex-wrap'>
```

Ví dụ:

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


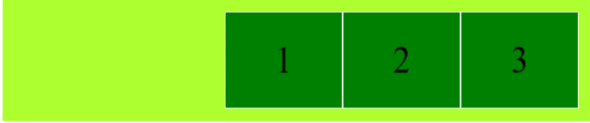





3. Thuộc tính justify-content:

Dùng để điều chỉnh vị trí bắt đầu và căn chỉnh các item bên trong container dọc theo trục **main axis**, chiều ngang hoặc dọc tùy thuộc vào flex-direction.

Cú pháp:

```
.container { justify-content: flex-start | flex-end | center |  
space-between | space-around | space-evenly; }
```

Ví dụ:

Thuộc tính: giá trị;	Ví dụ
justify-content: flex-start;	
justify-content: flex-end;	
justify-content: center;	
justify-content: space-between;	
justify-content: space-evenly;	
justify-content: space-around;	

3. Thuộc tính align-items:

Dùng để điều chỉnh vị trí bắt đầu và căn chỉnh các item bên trong container dọc theo trục **cross axis**, chiều ngang hoặc chiều dọc tùy thuộc vào flex-direction.

Cú pháp:

```
.container { align-items: stretch | flex-start | flex-end |  
center | baseline; }
```

3. Thuộc tính align-content:

Dùng để căn chỉnh khoảng cách các item bên trong container dọc theo trục **cross axis**, chiều ngang hoặc chiều dọc tùy thuộc vào flex-direction.

Cú pháp:

```
.container { align-content: flex-start | flex-end | center |  
space-between | space-around | stretch; }
```



**KẾT HỢP FLEXBOX VỚI MEDIA QUERIES
TẠO BỐ CỤC RESPONSIVE**

Ví dụ dàn 3 thẻ div theo chiều ngang:

```
<div class="container">  
  <div class="list">This is div 1</div>  
  <div class="list">This is div 2</div>  
  <div class="list">This is div 3</div>  
</div>
```

```
.container {  
  display: flex;  
  flex-direction: row;  
}  
.list {  
  width: 100%;  
  padding: 10px 5px;  
  background: green;  
  color: #fff;  
}
```

This is div 1

This is div 2

This is div 3

Kết hợp Flexbox với Media Queries để tạo bố cục responsive:

```
.container {  
    display: flex;  
    flex-direction: row;  
}  
.list {  
    width: 100%;  
    padding: 10px 5px;  
    background: green;  
    color: #fff;  
}  
@media screen and (max-width: 768px){  
    .container{  
        flex-wrap: wrap;  
    }  
}
```



This is div 1
This is div 2
This is div 3

*Giao diện trên màn hình
kích thước < 768px*

BÀI 3: **THIẾT KẾ BỐ CỤC WEBSITE ĐÁP ỨNG**

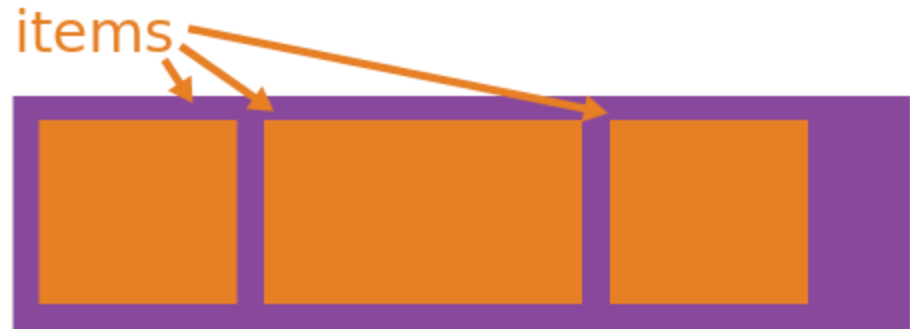
PHẦN II: DÀN BỐ CỤC SỬ DỤNG FLEXBOX (TT)

PHẦN 2 GIỚI THIỆU CÁC THUỘC TÍNH CỦA FLEX-ITEM

CÁC THUỘC TÍNH CỦA FLEX-ITEM

Dưới đây là một số thuộc tính có thể sử dụng đối với **flex-item**:

- ☐ order
- ☐ flex-grow
- ☐ flex-shrink
- ☐ flex-basis
- ☐ flex
- ☐ align-self



3. Thuộc tính order:

Dùng để sắp xếp lại vị trí sắp xếp của các item.

Cú pháp:

```
.item { order: <integer>; /* mặc định là 0 */ }
```

Ví dụ:

```
<div class="flex-container">  
  <div style="order: 3">1</div>  
  <div style="order: 2">2</div>  
  <div style="order: 4">3</div>  
  <div style="order: 1">4</div>  
</div>
```



3. Thuộc tính flex-grow:

Cho phép các phần tử giãn theo độ rộng của container.

Cú pháp:

```
.item { flex-grow: <number>; /* mặc định là 0 */ }
```

```
<div class="flex-container">  
  <div style="flex-grow: 1">1</div>  
  <div style="flex-grow: 1">2</div>  
  <div style="flex-grow: 3">3</div>  
  <div style="flex-grow: 1">4</div>  
</div>
```

Tất cả Item có flex-grow:1 và #item3 có flex-grow:3

Tất cả các item đều được giãn ra lấp đầy phần trống của container, riêng item3 có flex-grow: 3 thì khoảng trống chiếm nhiều hơn khoảng 3 lần.



3. Thuộc tính flex-shink:

Ngược lại với thuộc tính flex-grow, flex-shink cho phép các phần tử co lại theo độ rộng của container.

Cú pháp:

```
.item { flex-shrink: <number>; /* mặc định là 1 */ }
```

Giá trị mặc định trong flex-shrink là 1, cho phép các phần tử co lại bằng nhau khi độ rộng container giảm xuống. Nếu flex-shrink: 0 thì item sẽ không co giãn mà lấy nguyên giá trị của thuộc tính width/height.

3. Thuộc tính flex-basis:

Dùng để xác định độ dài ban đầu của một item.

Cú pháp:

```
.item { flex-basis: <length> | auto; /*mặc định là auto */ }
```

3. Thuộc tính flex:

Dùng để gộp chung ba thuộc tính **flex-grow**, **flex-shrink** và **flex-basis**.

Cú pháp:

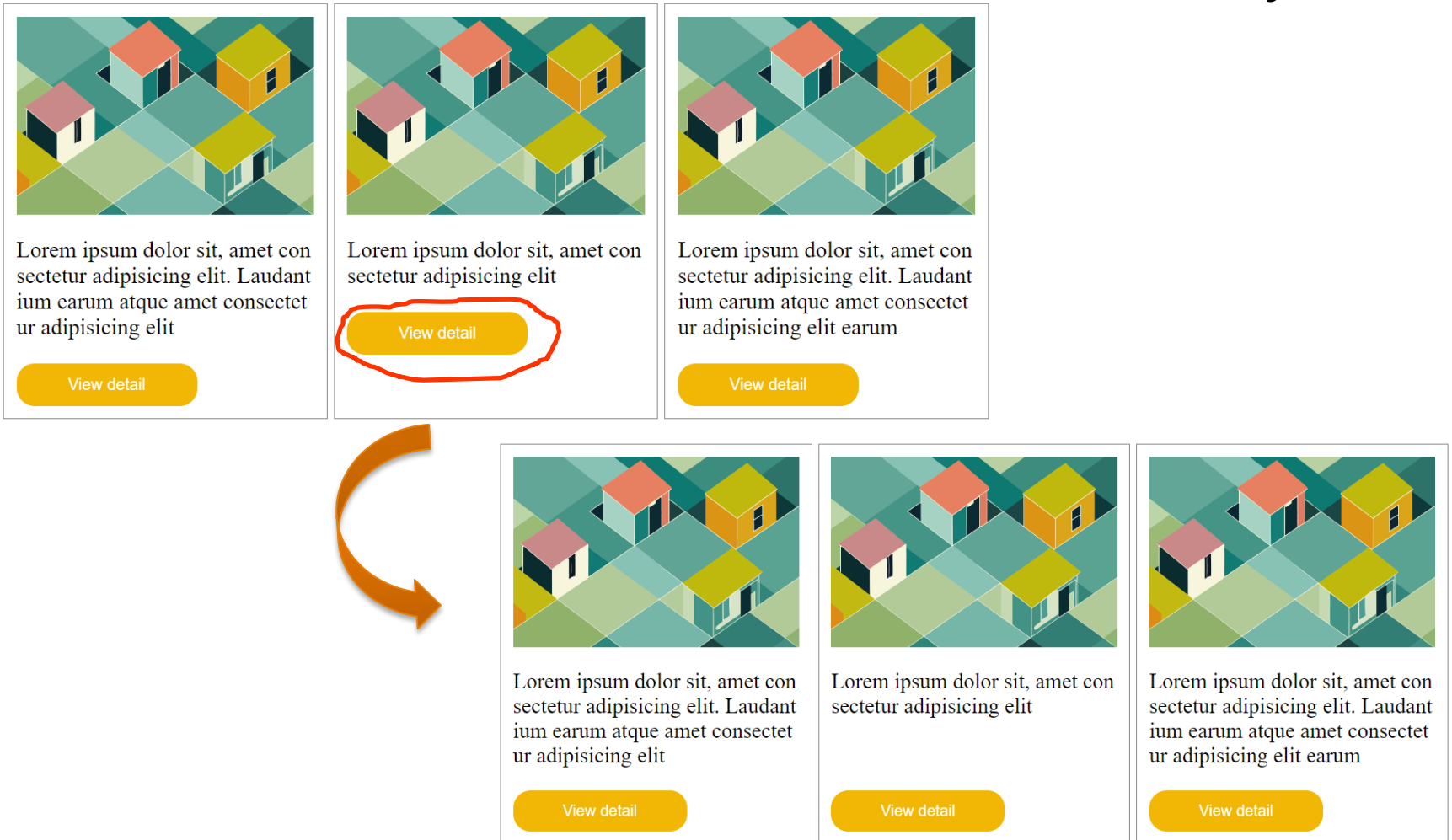
```
.item { flex: none | [ <'flex-grow'> <'flex-shrink'> ||  
<'flex-basis'> ] }
```

Ví dụ: **flex: 1 3 250px;**

Trong đó: **flex-grow: 1;** **flex-shrink: 3;** **flex-basis: 250px;**

demo

Tạo bố cục và xử lý yêu cầu như hình dưới đây:



HTML code: Tạo một hàng ngang **.container** chứa 3 cột **.thumbnail** giống nhau.

```
<div class="container">
  <div class="thumbnail">
    
    <h3>
      Lorem ipsum dolor sit, amet consectetur
      adipisicing elit. Laudantium earum atque
      amet consectetur adipisicing elit
    </h3>
    <button>View detail</button>
  </div>
  <div class="thumbnail"> ...
</div>
  <div class="thumbnail"> ...
</div>
</div>
```

CSS code:

```
.container {
  display: flex;
  flex-wrap: wrap;
  justify-content: space-around;
}
.thumbnail {
  width: 30%;
  border: 1px solid gray;
  padding: 10px;
  margin-bottom: 10px;
  display: flex;
  flex-direction: column;
}
```

```
.img-main {
  width: 100%;
  flex-shrink: 0;
  /*lấy nguyên giá trị của thuộc tính width/height*/
}
h3 {
  font-weight: 500;
  word-break: break-all;
  flex: 1;
  /*flex-grow: 1 tự động giãn ra đều cho vừa với khung*/
}
button {
  background-color: #f2b705;
  color: #fff;
  border: none;
  border-radius: 15px;
  padding: 10px 35px;
  width: 150px;
  flex-shrink: 0;
  /*lấy nguyên giá trị của thuộc tính width/height*/
}
```

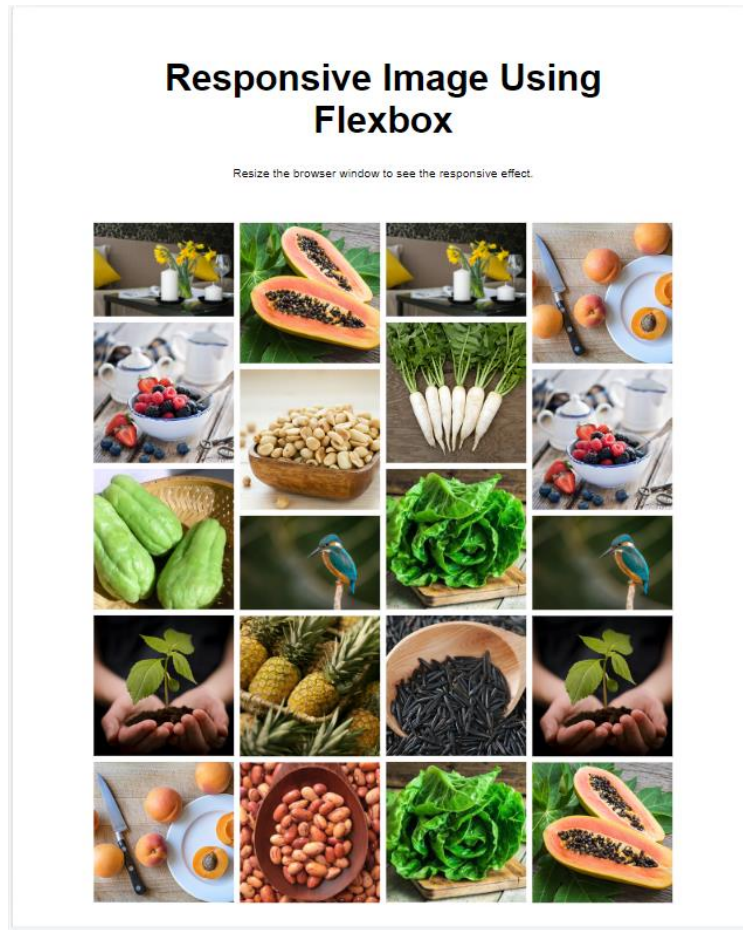
Xử lý giao diện trên các thiết bị:



Xử lý giao diện trên các thiết bị kích thước tối đa 768px và tối đa 480px:

```
@media screen and (max-width: 768px) {  
  .thumbnail {  
    width: calc(48% - 22px);  
  }  
}  
  
@media screen and (max-width: 480px) {  
  .thumbnail {  
    width: 100%;  
  }  
}
```

Tạo thư viện ảnh sử dụng Flexbox:



min-width: 800px



Responsive Image Using Flexbox

Resize the browser window to see the responsive effect.



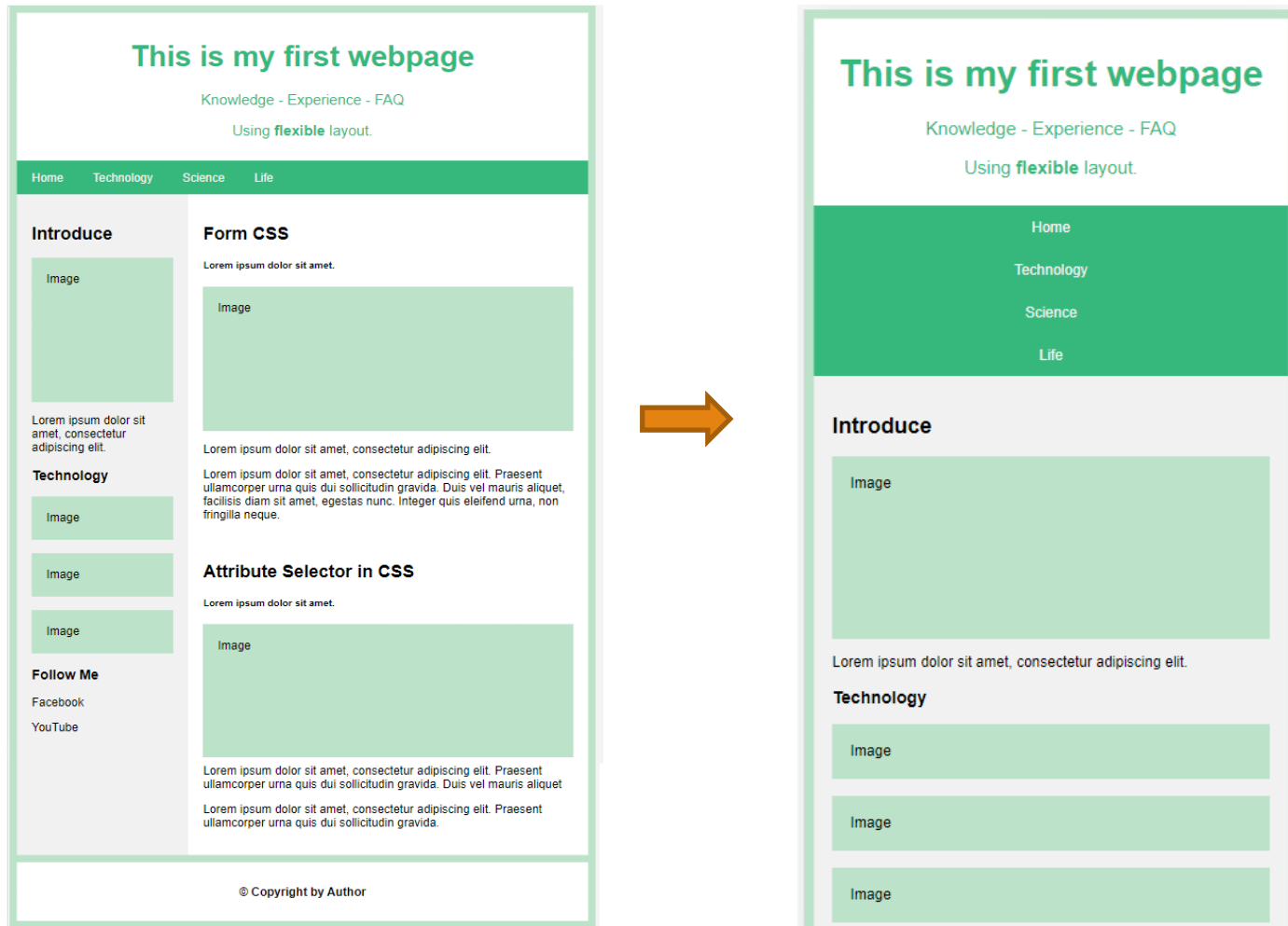
max-width: 800px

Tạo thư viện ảnh sử dụng Flexbox:

```
<div class="header">
  <h1>Responsive Image Using Flexbox</h1>
  <p>Resize the browser window to see the responsive effect.</p>
</div>
<div class="row">
  <div class="column">
    
    
    
    
    
  </div>
  <div class="column">
    
    
    
    
    
  </div>
  <div class="column">
    
    
    
    
    
  </div>
  <div class="column">
    
    
    
    
    
  </div>
</div>
```

```
* { box-sizing: border-box; }
body {
  margin: 0 auto;
  font-family: Arial;
  width: 80%;
}
.header {
  text-align: center;
  padding: 20px;
}
.header h1 { font-size: 5vw; }
.row {
  display: flex;
  flex-wrap: wrap;
  padding: 0 4px;
}
/* Tạo bốn cột bằng nhau nằm cạnh nhau */
.column {
  flex-basis: 25%;
  max-width: 25%;
  padding: 0 4px;
}
.column img {
  margin-top: 8px;
  vertical-align: middle;
}
/* Bố cục linh hoạt: tạo bố cục 2 cột */
@media (max-width: 800px) {
  .column {
    flex-basis: 50%;
    max-width: 50%;
  }
}
/* Bố cục linh hoạt: làm cho hai cột xếp dọc thay vì cạnh nhau */
@media (max-width: 600px) {
  .column {
    flex-basis: 100%;
    max-width: 100%;
  }
}
```

Flexbox Layout website responsive kiểu mẫu:



Flexbox Layout website responsive kiểu mẫu.

HTML code:

```
<!-- Header -->
<div class="header">
  <h1>This is my first webpage</h1>
  <p>Knowledge - Experience - FAQ</p>
  <p>Using <b>flexible</b> layout.</p>
</div>
<!-- Navigation Bar -->
<div class="navbar">
  <a href="#">Home</a>
  <a href="#">Technology</a>
  <a href="#">Science</a>
  <a href="#">Life</a> </div>
```

```
<!-- Nội dung -->
<div class="row">
  <div class="side">
    <h2>Introduce</h2>
    <div class="fakeimg" style="height:200px;">Image</div>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
    <h3>Technology</h3>
    <div class="fakeimg" style="height:60px;">Image</div><br>
    <div class="fakeimg" style="height:60px;">Image</div><br>
    <div class="fakeimg" style="height:60px;">Image</div>
    <h3>Follow Me</h3>
    <p>Facebook</p>
    <p>YouTube</p>
  </div>
  <div class="main">
    <h2>Form CSS</h2>
    <h5>Lorem ipsum dolor sit amet.</h5>
    <div class="newimg" style="height:200px;">Image</div>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent ullamcorper urna quis dui sollicitudin gravida. Duis vel mauris aliquet, facilisis diam sit amet, egestas nunc. Integer quis eleifend urna, non fringilla neque. </p> <br>
    <h2>Attribute Selector in CSS</h2>
    <h5>Lorem ipsum dolor sit amet.</h5>
    <div class="newimg" style="height:200px;">Image</div>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent ullamcorper urna quis dui sollicitudin gravida. Duis vel mauris aliquet</p>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent ullamcorper urna quis dui sollicitudin gravida. </p>
  </div>
</div>
<!-- Footer -->
<div class="footer">
  <h4>&copy; Copyright by Author</h4>
</div>
```

Flexbox Layout website responsive kiểu mẫu.

CSS code:

```

* {
  box-sizing: border-box;
}

body {
  font-family: Arial;
  padding: 10px;
  background: #BCE2C9;
  margin: 0;
}

/* Header/logo/Title */
.header {
  padding: 10px;
  text-align: center;
  background: white;
  color: #36B87D;
}

.header h1 {
  font-size: 40px;
}

.header p {
  font-size: 20px;
}

/* Định dạng navigation bar */
.navbar {
  display: flex;
  background-color: #36B87D;
}

```

```

/* Định dạng link điều hướng */
.navbar a {
  color: #f2f2f2;
  padding: 14px 20px;
  text-decoration: none;
  text-align: center;
}

.navbar a:hover {
  background-color: #BCE2C9;
  color: white;
}

/* Column container */
.row {
  display: flex;
  flex-wrap: wrap;
}

/* Tạo 2 cột trên 1 hàng ngang */
/* Sidebar/Cột trái */
.side {
  flex-basis: 30%;
  background-color: #f1f1f1;
  padding: 20px;
}

/* Cột chính */
.main {
  flex-basis: 70%;
  background-color: white;
  padding: 20px;
}

```

```

/* Hình ảnh tượng trưng */
.newimg {
  background-color: #BCE2C9;
  width: 100%;
  padding: 20px;
}

/* Footer */
.footer {
  padding: 10px;
  text-align: center;
  background: white;
  margin-top: 10px;
}

/* Bố cục linh hoạt: các cột xếp dọc thay vì
hàng khi màn hình có kích thước dưới 600px */
@media screen and (max-width: 600px) {

  .row,
  .navbar {
    flex-direction: column;
  }
}

```

ƯU ĐIỂM VÀ NHƯỢC ĐIỂM CỦA FLEXBOX

1. Flexbox có ưu điểm gì?

- ☐ Dàn trang linh hoạt với nhiều thuộc tính về hướng
- ☐ Tự động căn chỉnh các flex-item với nhau và cân đối so với flex-container.
- ☐ Có thể chọn bố trí dọc theo một hướng trục chính hoặc bao bọc bởi cả hai trục

2. Khuyết điểm

- ☐ Một số trình duyệt không hỗ trợ Flexbox
- ☐ Flexbox có thể làm tăng thời gian tải của trang web trong trường hợp project rất lớn

- ❑ Flexbox là một trong những tính năng của CSS3, thường được sử dụng trong các thiết kế responsive, giúp hiển thị các phần tử linh hoạt trên nhiều thiết bị và kích thước màn hình khác nhau.
- ❑ Các thành phần của flexbox (flex-container và flex-item) được bố trí dọc theo trục chính main axis (kéo dài từ main-start tới main-end) hoặc trục chéo cross axis (từ cross-start đến cross-end).
- ❑ Vận dụng kết hợp Flexbox và Media Query
- ❑ Ưu điểm, nhược điểm của Flexbox





Cảm ơn