**BÁO CÁO TUẦN 5**

Mục lục

[Lời nói đầu 3](#_Toc21516638)

[Bảng phân công 3](#_Toc21516639)

1. [**Lịch sử Web Layout 3**](#_Toc21516640)

[1. Presentational HTML Tags 3](#_Toc21516641)

[2. The Single Pixel GIF Trick 3](#_Toc21516642)

[3. Table Layout 3](#_Toc21516643)

[4. Frames 3](#_Toc21516644)

[5. Client-Side Imagemaps 3](#_Toc21516645)

[6. Cascading Style Sheets 3](#_Toc21516646)

[*6.1. DIVS and the "Box Model" 3*](#_Toc21516647)

[*6.2. CSS Position Property 3*](#_Toc21516648)

[*6.3. Floated Layouts 3*](#_Toc21516649)

[*6.4. Mobile and Responsive Sites 3*](#_Toc21516650)

[*6.5. CSS Frameworks 3*](#_Toc21516651)

[*6.6. Flexbox 3*](#_Toc21516652)

1. [**CSS Grid 3**](#_Toc21516653)
2. [**Ứng dụng CSS Grid 3**](#_Toc21516654)
3. [**Tổng kết 3**](#_Toc21516655)

Lời nói đầu

Bảng phân công

|  |  |  |  |
| --- | --- | --- | --- |
| MSSV | Họ tên | Công việc | Đánh giá |
| 3115410042 | Trần Huy Hoàng | Chương I, IV |  |
| 3115410046 | Nguyễn Nhật Hùng | Chương II, III |  |
| 3115410049 | Hà Quốc Huy | Chương II, III |  |
| 3115410054 | Thái Minh Hưng | Chương I, IV |  |

1. **Lịch sử Web Layout**
2. **Presentational HTML Tags**

Giữa những năm 1990 là thời đại mà các nhà sản xuất trình duyệt lớn (Netscape và Microsoft) đã thêm các thẻ để cho phép các nhà thiết kế kiểm soát giao diện trang của họ bằng các thẻ "mang tính trình bày" như in đậm và <i> cho chữ nghiêng. Một loạt các thẻ cũng cho phép thay đổi kích thước phông chữ <font size = "+ 2">, mặt phông chữ <font face = "verdana"> và màu của các thành phần trang với các thẻ như <font color = "blue"> và <body bgcolor = "# e6e6fa">. Thậm chí một số mức độ căn chỉnh hình ảnh có thể được thực hiện bằng một thuộc tính cho thẻ <img>, chẳng hạn như <img src = "smiley.gif" align = "left">.

Sử dụng HTML để kiểm soát bố cục sớm trở thành một vấn đề lớn đối với các nhà phát triển web để duy trì các trang. Thay đổi sẽ phải được thực hiện trong nhiều trường hợp trên nhiều trang của một trang web. Thêm vào đó, những người sáng lập web, như Tim Berners-Lee, đã hình dung HTML chỉ được sử dụng để xác định cấu trúc của một tài liệu, không phải để kiểm soát việc trình bày.

## The Single Pixel GIF Trick

Một yêu thích thời bấy giờ là việc sử dụng các ảnh trong suốt pixel (GIF) để thêm khoảng cách giữa các thành phần trên một trang.

Một hình ảnh pixel đơn được tải xuống nhanh chóng và vì nó không nhìn thấy (trong suốt), nhà thiết kế có thể chỉ cần thay đổi khoảng cách ngang và / hoặc dọc bằng cách thêm các thuộc tính khoảng cách như <img src = "smiley.gif" hspace = "75"> hoặc chiều rộng và / hoặc các thuộc tính chiều cao, chẳng hạn như <img src = "single.gif" width = "150"> để tạo bất kỳ lượng không gian trống nào mong muốn.

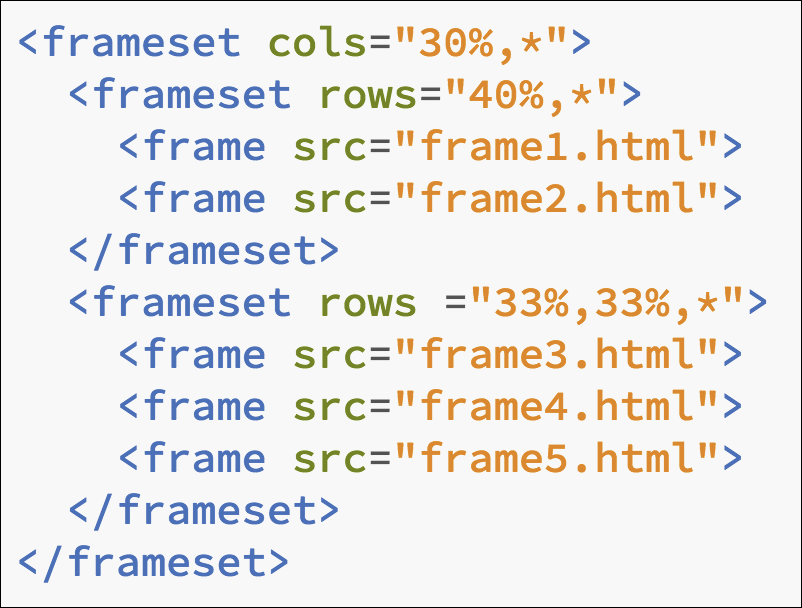
## Table Layout

Sử dụng hình ảnh để trính bày hạn chế, Người ta đã sớm phát hiện ra rằng sử dụng các bảng dữ liệu có thể là một công cụ mạnh mẽ để xử lý bố cục. Đến năm 1996, việc sử dụng các bảng để bố trí đã cất cánh. Các bảng được thiết kế theo thông số kỹ thuật ban đầu cho HTML để giữ dữ liệu dạng bảng trong các hàng và cột, nhưng rõ ràng là các ô riêng lẻ của một bảng có thể được sử dụng để giữ các thành phần trang với nhau theo bố cục đa chiều.

Mặc dù không được thiết kế để bố trí, các bảng đã trở thành phương pháp "đi đến" để bố trí nội dung trang tốt vào những năm 2000. Giống như việc sử dụng các thẻ trình bày được nêu ở trên, việc sử dụng các bảng HTML để kiểm soát bản trình bày đã phá vỡ sự hợp lý của HTML chỉ được sử dụng để mô tả cấu trúc của một trang. Chỉnh sửa các trang là cồng kềnh và khó khăn.

## Frames

Cũng trong khoảng thời gian các bảng được sử dụng để bố trí, việc sử dụng các khung HTML là phổ biến. Sử dụng thẻ <frameset>, một tài liệu HTML có thể chỉ định các khu vực của trang nơi các tệp HTML khác có thể được bao gồm bằng cách sử dụng thẻ <frame src>. Các khung thậm chí có thể được lồng vào nhau để kiểm soát tốt hơn. Ví dụ dưới đây cho thấy năm trang riêng biệt được bao gồm trong thẻ <frameset> chính.



## Client-Side Imagemaps

Client-side imagemaps là hình ảnh lớn với các khu vực có thể nhấp (điểm nóng) liên kết đến các trang khác. Các nhà thiết kế đồ họa yêu thích bản đồ hình ảnh vì họ có thể tạo ra các hình ảnh tinh vi trong Photoshop và biết rằng thiết kế sẽ trông giống hệt như khi xem trên trình duyệt. Điểm nóng của bản đồ hình ảnh được tạo bằng một số mã HTML đơn giản chỉ định tọa độ cho hình chữ nhật, hình tròn hoặc đa giác.

Với hình ảnh tưởng tượng, không có mối quan tâm về cách kiểm soát bố cục trang. Toàn bộ bố cục của một trang có thể được thực hiện với một hình ảnh lớn. Tất nhiên, điều đó có nghĩa là thời gian tải xuống dài và đau đầu cho bất cứ ai muốn thực hiện thay đổi. Cũng như các bảng và khung, một lần nữa nó đã vi phạm mục tiêu của web để tách nội dung khỏi bản trình bày.

**CASCADING STYLE SHEETS**

Vào cuối năm 1990, một cuộc cải cách lớn đã được tiến hành để mang lại một số trật tự cho HTML. Những ông lớn trong các cuộc chiến các trình duyệt (Netscape và Microsoft) thực sự đã hợp tác với nhau và đồng ý loại bỏ các presentational tags (thẻ trình bày) .

World Wide Web Consortium đã đưa ra khuyến nghị đầu tiên cho CSS (CSS1) vào năm 1996. Một bản cập nhật, CSS Cấp 2, được xuất bản năm 1998 và nó cung cấp các khả năng bố cục mới như fixed (position: fixed), absolute (position: absolute), and relative position: relative) và chỉ mục z-index. Việc áp dụng rất chậm vì phải đến tận thế kỷ trước, bất kỳ trình duyệt nào cũng hỗ trợ đầy đủ cho CSS.

Bản nháp CSS 3 được xuất bản lần đầu tiên vào năm 1999. CSS 3 là thông số kỹ thuật hiện tại cho CSS và tiếp tục cho thấy các mô-đun mới, nhiều trong số đó đã cung cấp cho các công cụ mới mạnh mẽ để bố trí (layout) như Flexbox và CSS Grid.

### **DIVS AND THE "BOX MODEL**"

### Phần tử <div> được tạo như một phần của CSS để chia trang thành các phần logic. Thẻ <div> có thể được sử dụng để giữ văn bản và hình ảnh trong bố cục. Những <div> này cũng có thể thiết lập cho chiều rộng và chiều cao. Chúng có thể có đường viền và thậm chí có thể cho phép đệm (không gian xung quanh nội dung trong hộp) và lề (không gian xung quanh bên ngoài hộp).



### **CSS POSITION PROPERTY**

Theo mặc định, <div> s là các thành phần cấp khối và chúng xếp chồng lên nhau, trừ khi được định vị (positioned) theo một cách nào đó. Công cụ CSS đầu tiên để xử lý căn chỉnh các yếu tố này là **position** -thuộc tính vị trí (position:fixed).   
Thuộc tính **position** vẫn là một phần của thông số kỹ thuật và vẫn còn rất hữu ích cho đến ngày nay. Có năm giá trị **position** khác nhau:

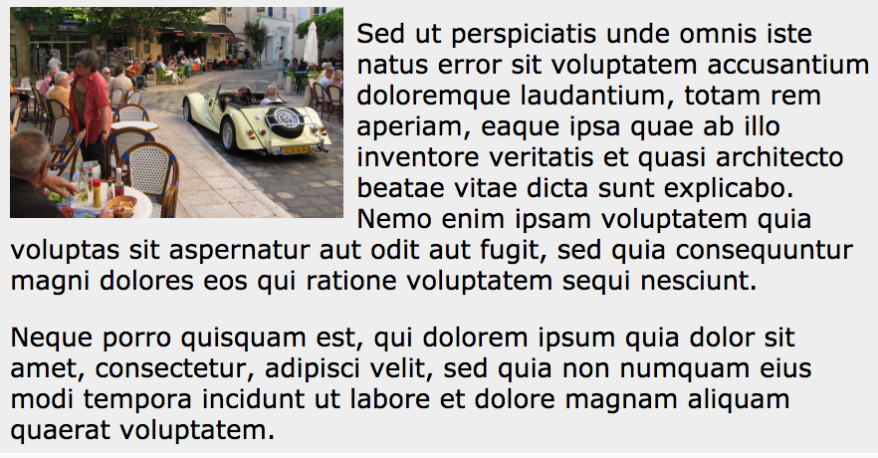
*position:***static** (tĩnh) là mặc định được định vị.   
*position:* vị trí **relative** (tương đối) so với vị trí bình thường của họ.  
*position:***fixed** được định vị tương đối với chế độ xem (cửa sổ trình duyệt), có nghĩa là nó vẫn ở vị trí trên màn hình ngay cả khi cuộn trang.   
*position:* vị trí **absolute** (tuyệt đối) liên quan đến ancestor (gốc) vị trí gần nhất.   
*position:* vị trí **sticky** (dính) dựa trên vị trí cuộn của người dùng.

Đôi khi các yếu tố cần chồng chéo các yếu tố khác. Để xác định thứ tự sắp xếp trục Z-axis (chiều sâu), z-index đã được tạo. Ví dụ: một phần tử có z-index: 2 sẽ hiển thị trên đầu của một phần tử có z-index: 1.

Định vị **Absolute** và **relative** hữu ích cho việc căn chỉnh một số yếu tố, nhưng bị hạn chế trong khả năng kiểm soát bố cục hiệu quả cho toàn bộ trang

### **FLOATED LAYOUTS**

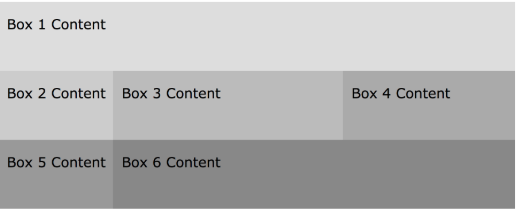
Hình ảnh dưới đây cho thấy Float được sử dụng như thế nào. Trong trường hợp này, hình ảnh được thả sang bên trái với **float: left** được áp dụng cho hình ảnh.



Khi chiều rộng của các hộp liền kề với nhau vượt quá chiều rộng của container, <div> tiếp theo chỉ cần chuyển xuống hàng tiếp theo.

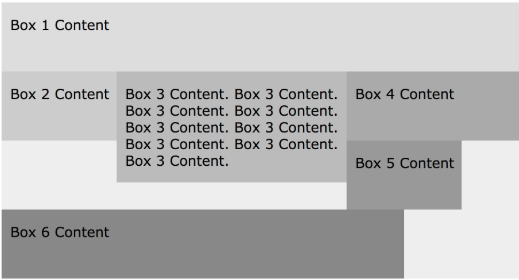
**FLOAT IN ACTIONS**

Trong ví dụ này, container có chiều rộng 900 pixel. Các hộp lồng nhau đều được đặt thành **float: left**. Hộp 1 rộng 900 pixel, vì vậy nó chính xác vừa với hàng đầu tiên và hộp 2 phải chuyển xuống hàng tiếp theo. Hộp 2 rộng 200 pixel. Hộp 3 rộng 400 pixel và Hộp 4 rộng 300 pixel. Chung với nhau, các hộp 2, 3 và 4 có chiều rộng 900 pixel, vì vậy tất cả chúng khớp với nhau trên hàng thứ hai. Hộp 5 không vừa, vì vậy nó di chuyển xuống hàng dưới cùng. Nó rộng 200 pixel. Cuối cùng, hộp 6 là 700 pixel, do đó, nó chiếm không gian còn lại ở hàng dưới cùng.



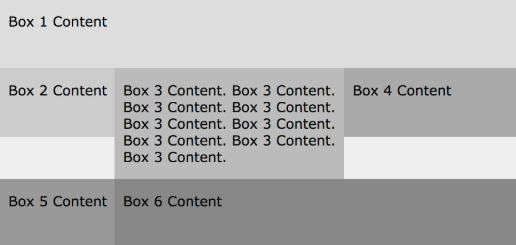
**THE HANGING FLOAT**

Một vấn đề phổ biến với float layout được gọi là **hanging float**. Nếu các phần tử float bên trái, một **hanging float** được tạo khi một phần tử trên cùng một hàng cao hơn nội dung của một hộp ở bên phải của nó. Trong ví dụ này, hộp 5 sẽ xóa tất cả các hàng tiếp theo, nhưng thay vào đó, bị cuốn vào hộp có chiều cao lớn hơn. Trong trường hợp này, hộp 5 đang bị treo lên trên hộp 3.



**CLEARING FLOATS**

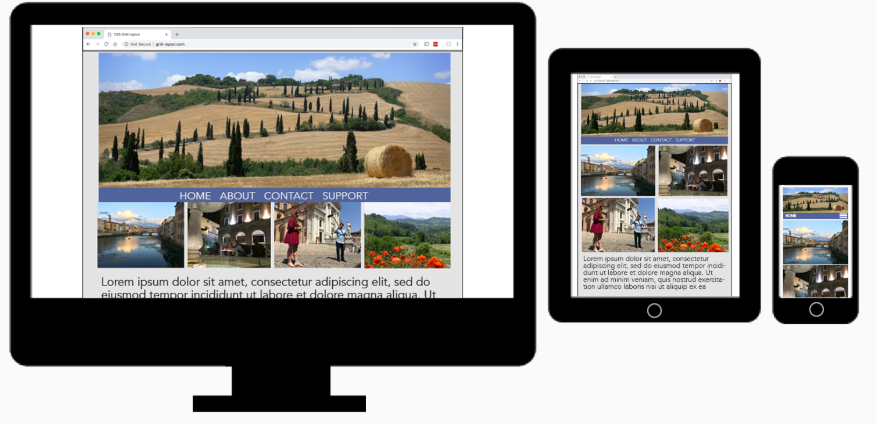
Sửa hanging floats yêu cầu bạn xóa float trên phần tử. Vì vậy, chúng ta phải thêm clear: left or clear: both vào <div> cho hộp 5. Bây giờ, nó sẽ đi xuống hàng tiếp theo và bắt đầu ở phía bên trái của bố cục.



### **MOBILE AND RESPONSIVE SITES**

Ngày nay, xu hướng là tạo một trang web phù hợp với kích thước hiển thị của thiết bị. Thiết kế web đáp ứng (RWD) cố gắng làm cho các trang web hiển thị tốt trên nhiều thiết bị và kích cỡ màn hình. Mục tiêu là làm cho các trang vừa dễ sử dụng vừa thỏa mãn.

Hình minh họa dưới đây cho thấy một trang web cần thay đổi bố cục như thế nào để thích ứng với các kích thước màn hình khác nhau.



Thiết kế web **Responsive** dựa trên các truy vấn phương tiện để biết khi nào cần thay đổi bố cục. Truy vấn phương tiện là một đoạn mã CSS xác định độ phân giải của thiết bị hiển thị trang. Sau đó, nó phục vụ các CSS hoặc bố cục khác nhau dựa trên những gì nó xác định. Trong ví dụ bên dưới, văn bản bên trong tiêu đề sẽ hiển thị ở mức 1,2em nếu độ phân giải màn hình là 768 pixel hoặc rộng hơn.

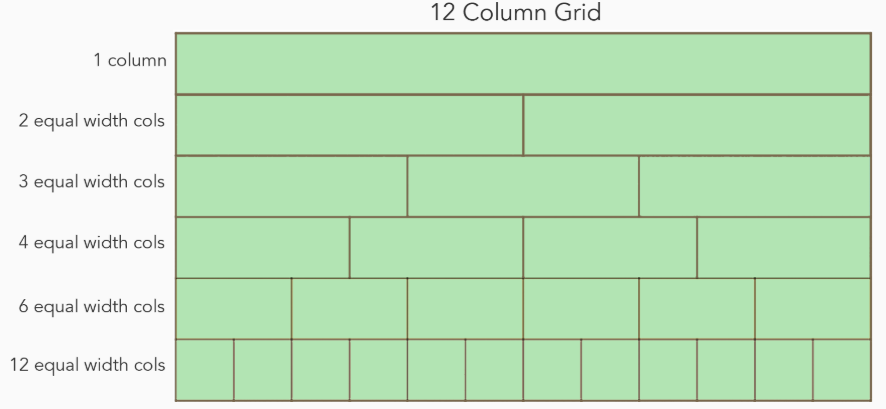
@media only screen and (min-width: 768px) {  
header {  
font-size: 1.2em;  
}  
}

CSS Grid đặc biệt phù hợp để tạo các trang web **Responsive**.

### **CSS FRAMEWORKS**

Khi nhu cầu về các trang web đáp ứng đã tăng lên, sự phức tạp của bố cục. Sự phức tạp này đã dẫn đến sự phát triển của một số **Frameworks** để giúp việc tạo trang web đáp ứng dễ dàng hơn. Hai loại được sử dụng rộng rãi nhất là **Foundation** và **Bootstrap**.

Hầu hết các khung sử dụng hệ thống lưới 12 cột tích hợp để tạo  responsive layouts. Với 12 cột, người ta có thể nhận được số lượng cột có chiều rộng bằng nhau lớn nhất trên một hàng. Grid (lưới) bên dưới cho thấy có thể có các cột có chiều rộng bằng 1, 2, 3, 4, 6 và 12 với lưới 12 cột.



Một ví dụ thực tế hơn về coding được hiển thị dưới đây.



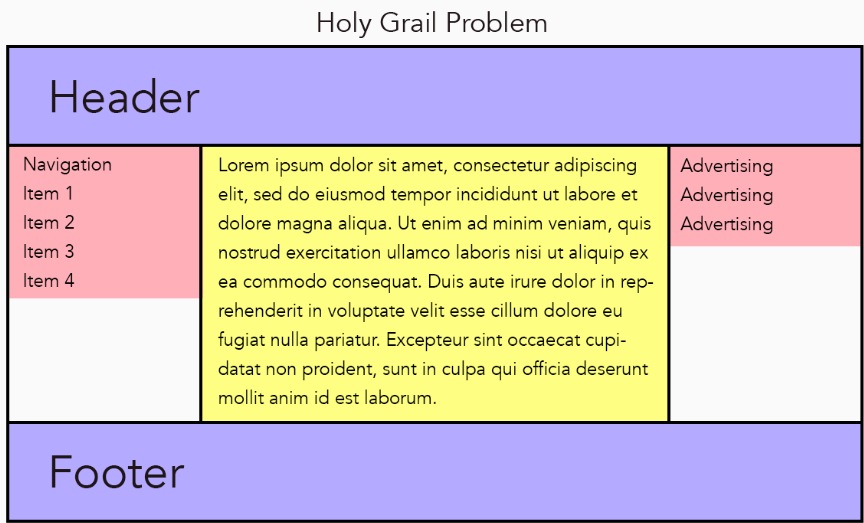
### **FLEXBOX**

**Flexbox** là một công cụ layout một chiều để bố trí các phần tử trong các hàng hoặc cột. Nó được thiết kế để giải quyết nhiều vấn đề với **position** và **float**. Bản thảo làm việc đầu tiên cho Flexbox đã được xuất bản vào năm 2009, nhưng nó đã trải qua một số thay đổi và bản dự thảo làm việc đã không được xuất bản cho đến năm 2013.

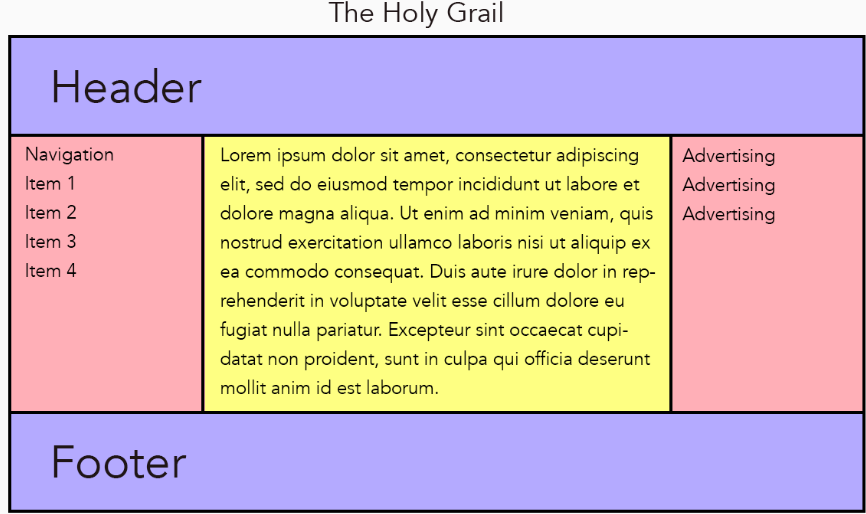
#### **Finding the Holy Grail**

Một trong những vấn đề mà Flexbox (và CSS Grid) đã giải quyết là một thứ gọi là Holy Grail Layout .  
Việc tạo các cột có chiều cao bằng nhau trên cùng một hàng với số lượng nội dung khác nhau đã thách thức các nhà thiết kế web trong nhiều năm.   
Trước Flexbox, tất cả các giải pháp đã biết đều có nhược điểm. Các giải pháp bao gồm divs with table-property displays, absolutely positioned divs, and JavaScript hacks.

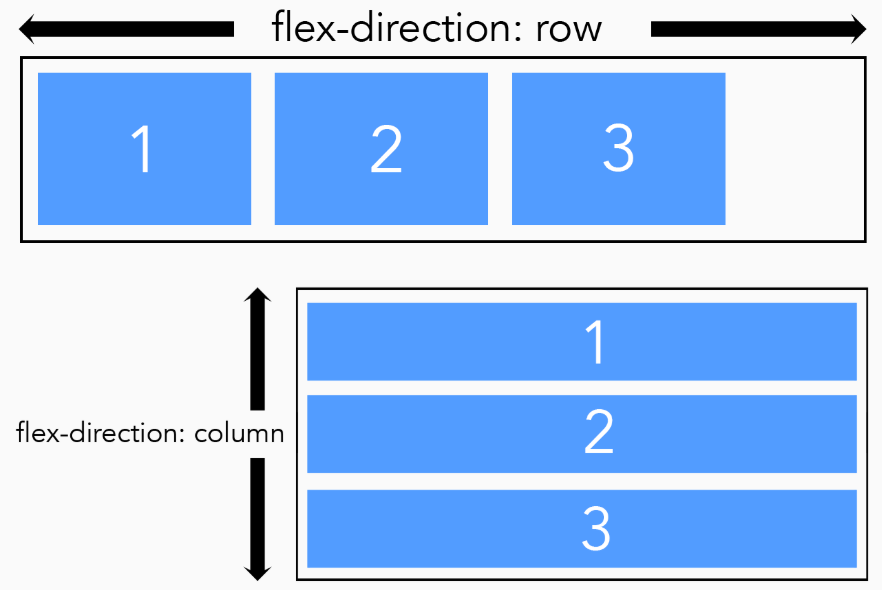
Trong hình minh họa bên dưới, bạn có thể thấy rằng hàng thứ hai có ba cột. Cột giữa là cao nhất, do đó nó lấp đầy tất cả không gian có sẵn. Tuy nhiên, các yếu tố thứ nhất và thứ ba có ít nội dung hơn. Để mỗi yếu tố chiếm toàn bộ chiều cao mà không cần thêm nhiều nội dung là thách thức.



Ví dụ dưới đây hiển thị the elusive Holy Grail - làm cho cả ba cột có chiều cao bằng nhau.



Flexbox là một cải tiến lớn ở chỗ nó cho phép căn chỉnh và căn giữa nội dung dễ dàng hơn và nó cho phép các cột có chiều cao bằng nhau, bất kể lượng nội dung là bao nhiêu.  
Flexbox Layout được tạo thành từ thùng chứa flex chứa các vật phẩm flex. Bạn có thể bố trí một thùng chứa theo chiều ngang (hàng) hoặc chiều dọc (cột). Bất cứ điều gì bạn chọn được gọi là trục chính (thường là hàng) và sau đó cái khác được gọi là trục chéo.   
Các mục **flex** được đặt dọc theo trục chính và chúng có thể phát triển để lấp đầy không gian không sử dụng. Bằng cách lồng các thùng chứa flex khác bên trong thùng chứa chính, bạn có thể đạt được sự kiểm soát đối với cả căn chỉnh ngang và dọc, mặc dù Flexbox về bản chất là một chiều.



Flexbox cung cấp các công cụ mạnh mẽ và hữu ích để tạo bố cục web.

## Cascading Style Sheets

### *DIVS and the "Box Model"*

### *CSS Position Property*

### *Floated Layouts*

### *Mobile and Responsive Sites*

### *CSS Frameworks*

### *Flexbox*

1. **CSS Grid**
2. **Ứng dụng CSS Grid**
3. **Tổng kết**