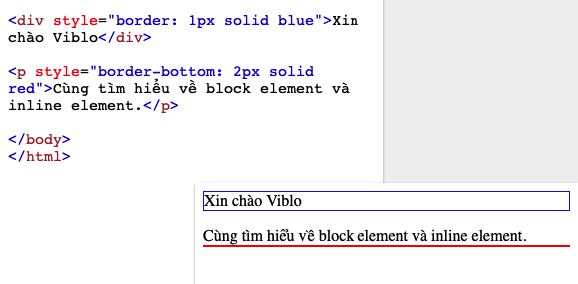
**Block Elements**

Các Block element (phần tử khối) khi hiển thị trên trình duyệt chúng sẽ tự động thêm các ngắt dòng (line break) vào phía trước và phía sau nó. Hiểu một cách đơn giản là khi gọi 2 block elements ra thì mỗi element sẽ chiếm 1 dòng và width của các element này sẽ full luôn dòng đó.

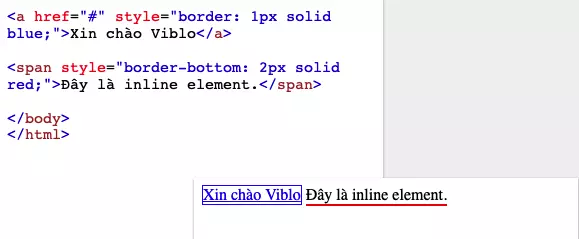


<div style="border: 1px solid blue">Xin chào Viblo</div>

<p style="border-bottom: 2px solid red">Cùng tìm hiểu về block element và inline element.</p>

**Inline Elements**

Trái ngược với **Block Elements** thì ta lại có **Inline Elements**. Các Inline element (phần tử nội tuyến) thường xuất hiện trong một đoạn văn (sentence), khi hiển thị trên trình duyệt nó không tự động thêm các ngắt dòng (line break) vào phía trước và phía sau của nó.



<a href="#" style="border: 1px solid blue;">Xin chào Viblo</a>

<span style="border-bottom: 2px solid red;">Đây là inline element.</span>

Danh sách tất cả các Inline element

| **Col 1** | **Col 2** | **Col 3** | **Col 4** | **Col 5** | **Col 6** | **Col 7** |
| --- | --- | --- | --- | --- | --- | --- |
| <a> | <abbr> | <acronym> | <b> | <bdo> | <big> | <button> |
| <cite> | <code> | <dfn> | <em> | <i> | <img> | <input> |
| <kbd> | <label> | <map> | <object> | <output> | <q> | <samp> |
| <script> | <select> | <small> | <span> | <strong> | <sub> | <sup> |
| <textarea> | <time> | <tt> | <var> |  |  |  |

Cũng tương tự như block element, nếu như muốn các block inline này có bản chất giống như block element thì chỉ cần thêm 1 dòng css như dưới là xong.

display: block

**FLEX**

#### flex-direction

#### This establishes the main-axis, thus defining the direction flex items are placed in the flex container. Flexbox is (aside from optional wrapping) a single-direction layout concept. Think of flex items as primarily laying out either in horizontal rows or vertical columns.

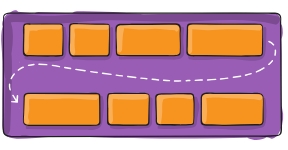
.container {

flex-direction: row | row-reverse | column | column-reverse;

}

* row (default): left to right in ltr; right to left in rtl
* row-reverse: right to left in ltr; left to right in rtl
* column: same as row but top to bottom
* column-reverse: same as row-reverse but bottom to top

#### flex-wrap



By default, flex items will all try to fit onto one line. You can change that and allow the items to wrap as needed with this property.

.container {

flex-wrap: nowrap | wrap | wrap-reverse;

}

* nowrap (default): all flex items will be on one line
* wrap: flex items will wrap onto multiple lines, from top to bottom.
* wrap-reverse: flex items will wrap onto multiple lines from bottom to top.

#### flex-flow

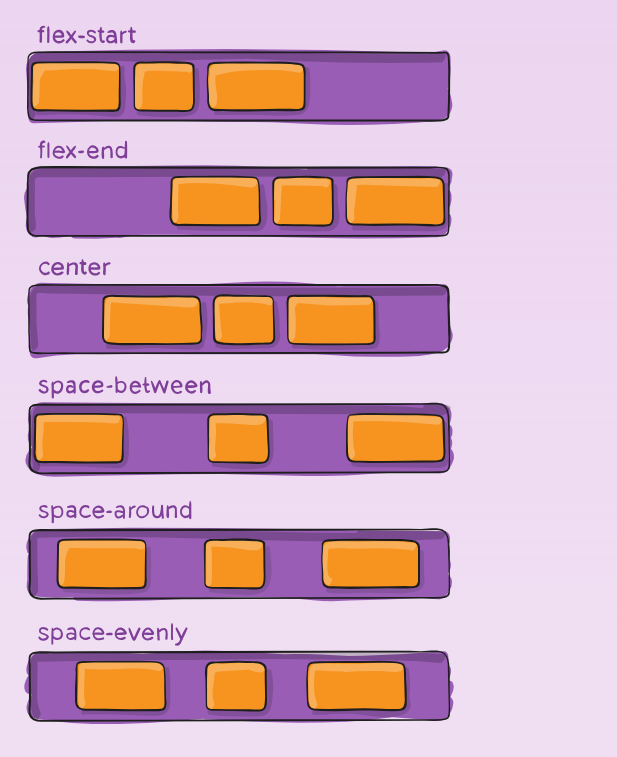
This is a shorthand for the flex-direction and flex-wrap properties, which together define the flex container’s main and cross axes. The default value is row nowrap.

.container {

flex-flow: column wrap;

}

#### justify-content



This defines the alignment along the main axis. It helps distribute extra free space leftover when either all the flex items on a line are inflexible, or are flexible but have reached their maximum size. It also exerts some control over the alignment of items when they overflow the line.

.container {

justify-content: flex-start | flex-end | center | space-between | space-around | space-evenly | start | end | left | right ... + safe | unsafe;

}

* flex-start (default): items are packed toward the start of the flex-direction.
* flex-end: items are packed toward the end of the flex-direction.
* start: items are packed toward the start of the writing-mode direction.
* end: items are packed toward the end of the writing-mode direction.
* left: items are packed toward left edge of the container, unless that doesn’t make sense with the flex-direction, then it behaves like start.
* right: items are packed toward right edge of the container, unless that doesn’t make sense with the flex-direction, then it behaves like end.
* center: items are centered along the line
* space-between: items are evenly distributed in the line; first item is on the start line, last item on the end line
* space-around: items are evenly distributed in the line with equal space around them. Note that visually the spaces aren’t equal, since all the items have equal space on both sides. The first item will have one unit of space against the container edge, but two units of space between the next item because that next item has its own spacing that applies.
* space-evenly: items are distributed so that the spacing between any two items (and the space to the edges) is equal.