



# CSS3

## Cascading order

All the styles in a page will "cascade" into a new "virtual" style sheet by the following rules, where number one has the highest priority:

- 1. Inline style (inside an HTML element)
- 2. External and internal style sheets (in the head section)
- 3. Browser default

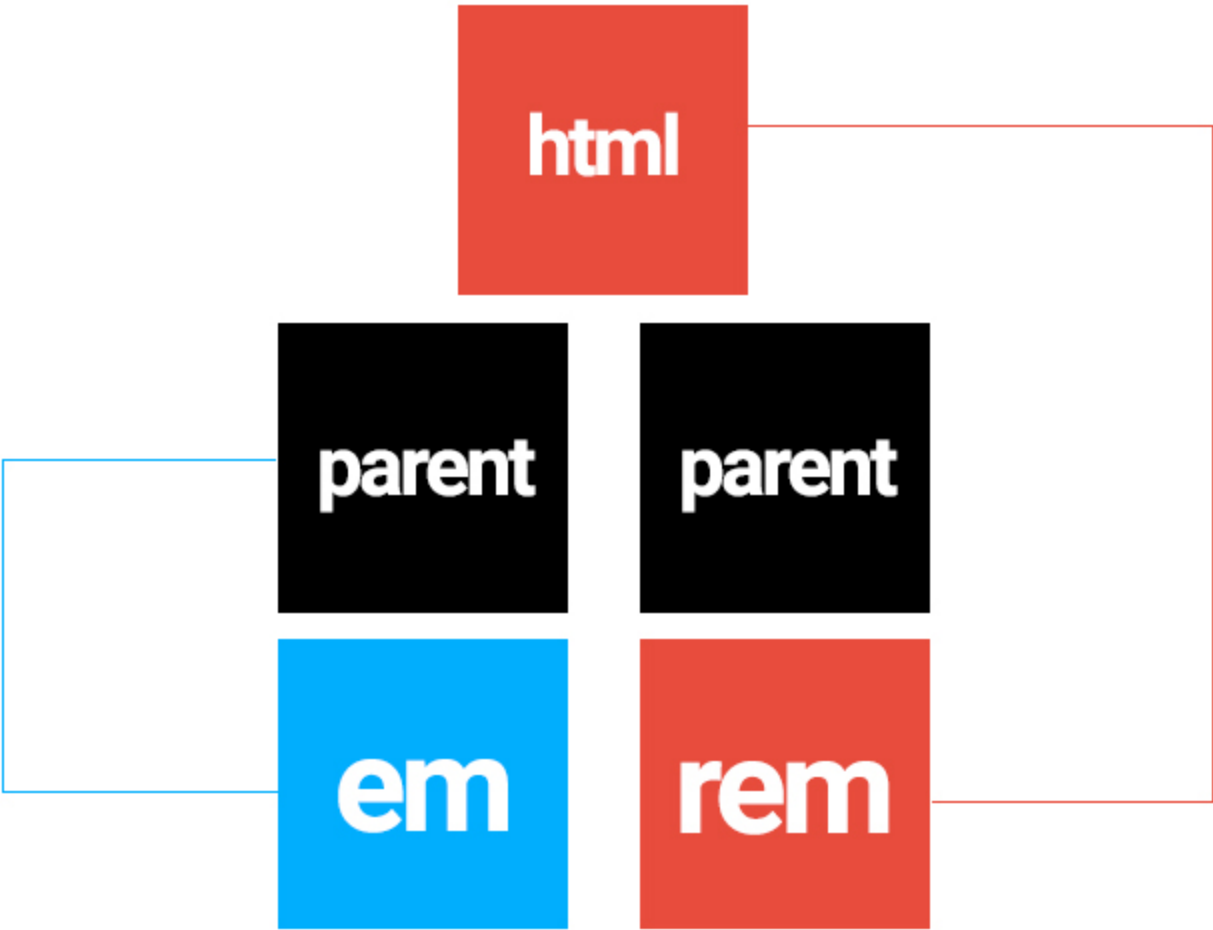
## Position trong CSS

<code>relative</code>	Căn chỉnh phần tử mà không gây ảnh hưởng đến các phần tử khác. Bình thường chúng ta dùng <code>margin</code> hay <code>padding</code> chắc chắn sẽ đẩy các phần tử khác ra một đoạn gây ảnh hưởng tới layout.	Không ảnh hưởng elements khác
<code>absolute</code>	Thường thường giá trị <code>absolute</code> này khi được sử dụng cho phần tử mà phần tử cha của nó đang có <code>relative</code> hoặc <code>absolute</code> . Để lúc này nó sẽ chạy theo phần tử cha đó	Ăn theo cha
<code>fixed</code>	Giá trị này không phụ thuộc vào phần tử cha hay gì cả. Khi nào scroll trình duyệt là nó hoạt động,	Đứng yên

## "em" và "rem"

`rem` : là đơn vị tham chiếu tỷ lệ so với phần tử gốc của website ở đây là thẻ `<html>` dựa vào giá trị của thuộc tính **font-size**

`em` : là đơn vị tham chiếu tỷ lệ so với phần tử cha trực tiếp chứa nó hoặc chính nó dựa vào giá trị của thuộc tính là **font-size**



không set giá trị của thuộc tính `font-size` cho thẻ `html` thì `font-size` mặc định của thẻ `html` là 16px.

```
<div class="box">
  <div class="em">EM</div>
  <div class="rem">REM</div>
</div>
```

```
html {
  font-size: 15px;
}
.box {
  font-size: 20px;
  color: white;
  display: flex;
}
.em{
  width: 10em;
  height: 10em;
  background-color: red;
}

.rem{
  width: 10rem;
  height: 10rem;
  background-color: blue;
}
```

10em = 10 \* 20px = 200px

10rem = 10 \* 15px = 150px;


## Single-colon vs. Double-colon

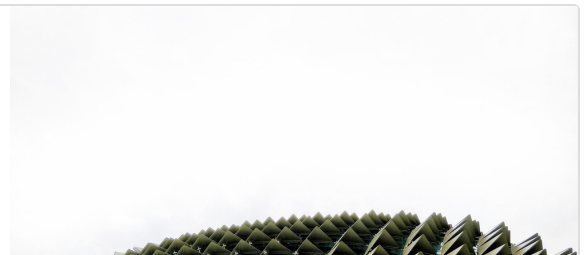
There's a bit of discussion about the right way of using pseudo-elements – the old style single-colon ( `:before` ), used in CSS specifications 1 and 2, versus the CSS3 recommendation, double-colon ( `::before` ), mainly to “*establish a discrimination between pseudo-classes and pseudo-elements*”.

But for compatibility reasons, the single-colon method is still accepted. Keep in mind that IE8 supports the single-colon notation only.

### The Best CSS Examples and CSS3 Examples

CSS provides the style of a website. The background property lets you use images and colors to create backgrounds for your web pages. Background Color ExampleThe background color property allows you to choose the color of your element. This can be the background for the entire page or the

 <https://www.freecodecamp.org/news/css-example-css3/#background-color-example>



## CSS Flexbox

The flex container properties are:

- `flex-direction`
- `flex-wrap`
- `flex-flow`
- `justify-content`
- `align-items`
- `align-content`

`flex-flow` is a shorthand for `flex-direction` and `flex-wrap`

### The justify-content Property

used to align the **flex items**

center, flex-start, flex-end

The `space-around` value displays the flex items with space before, between, and after the lines

The `space-between` value displays the flex items with space between the lines

### The align-items Property

used to align the flex items.

center, flex-start, flex-end

The `stretch` value stretches the flex items to fill the container (this is default)

The `baseline` value aligns the flex items such as their baselines aligns

The align-content Property

used to align the flex lines.

The flex item properties are:


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

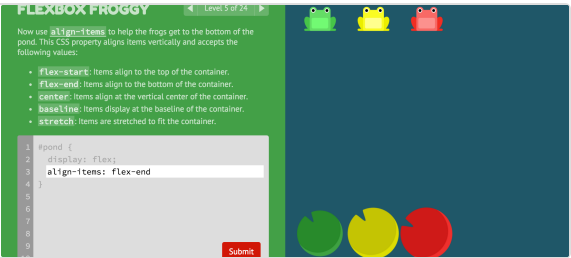
`flex` is a shorthand for `flex-grow`, `flex-shrink`, and `flex-basis`

```
/*Make the third flex item not growable (0), not shrinkable (0),
and with an initial length of 200 pixels:*/
<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div style="flex: 0 0 200px">3</div>
  <div>4</div>
</div>
```

Flexbox Froggy

A game for learning CSS flexbox


 <https://flexboxfroggy.com/>




<https://viblo.asia/p/huong-dan-day-du-ve-css-flexbox-maGK7J9a5j2>

CSS Flexbox


Try it Yourself " Before the Flexbox Layout module, there were four layout modes: Block, for sections in a webpage Inline, for text Table, for two-dimensional table data Positioned, for explicit position of an element The Flexible Box Layout Module, makes it easier to design flexible responsive layout structure


 [https://www.w3schools.com/css/css3\\_flexbox.asp](https://www.w3schools.com/css/css3_flexbox.asp)



A Complete Guide to Flexbox

Our comprehensive guide to CSS flexbox layout. This complete guide explains everything about flexbox, focusing on all the different possible properties for the parent element (the flex container) and the child elements (the flex items). It also includes history, demos, patterns, and a browser support

 <https://css-tricks.com/snippets/css/a-guide-to-flexbox/>



CSS Selectors

Five categories:

`Simple selectors` (select elements based on name, id, class)

`Combinator selectors` (select elements based on a specific relationship between them)

**Pseudo-class selectors** (select elements based on a certain state)

**Pseudo-elements selectors** (select and style a part of an element)

**Attribute selectors** (select elements based on an attribute or attribute value)

HTMLT elements with a specified class:

```
p.center {
  text-align: center;
  color: red;
}
```

All CSS Simple Selectors

Selector	Example	Example description
<u>#id</u>	#firstname	Selects the element with id="firstname"
<u>.class</u>	.intro	Selects all elements with class="intro"
<u>element.class</u>	p.intro	Selects only <p> elements with class="intro"
<u>*</u>	*	Selects all elements
<u>element</u>	p	Selects all <p> elements
<u>element,element,..</u>	div, p	Selects all <div> elements and all <p> elements

CSS combinators

There are four different combinators in CSS:

- descendant selector (space)
- child selector (>)
- adjacent sibling selector (+)
- general sibling selector (~)

**adjacent sibling selector (+)**

Sibling elements must have the same parent element, and "adjacent" means "immediately following".

**general sibling selector (~)**

The general sibling selector selects all elements that are next siblings of a specified element.

CSS pseudo-class

it can be used to:

- Style an element when a user mouses over it
- Style visited and unvisited links differently
- Style an element when it gets focus

:link, :visited, :hover, :active, :focus, :first-child, :lang

Anchor Pseudo-classes

```
a:link {
  color: red;
}

/* visited link */
a:visited {
  color: green;
}
```

```
/* mouse over link */
a:hover {
  color: hotpink;
}

/* selected link */
a:active {
  color: blue;
}
```

a:hover MUST come after a:link and a:visited  
a:active MUST come after a:hover

## Simple Tooltip Hover

```
p {
  display: none;
  background-color: yellow;
  padding: 20px;
}

div:hover p {
  display: block;
}
```

## Pseudo-elements

It can be used to:

- Style the first letter, or line, of an element
- Insert content before, or after, the content of an element

::first-line, ::first-letter, ::before, ::after, ::marker, ::selection

## Specificity Hierarchy

Every selector has its place in the specificity hierarchy. There are four categories which define the specificity level of a selector:

**Inline styles** - An inline style is attached directly to the element to be styled. Example: <h1 style="color: #ffffff;">.

**IDs** - An ID is a unique identifier for the page elements, such as #navbar.

**Classes, attributes and pseudo-classes** - This category includes .classes, [attributes] and pseudo-classes such as :hover, :focus etc.

**Elements and pseudo-elements** - This category includes element names and pseudo-elements, such as h1, div, :before and :after.



**!important** will override ALL previous styling rules for that specific property on that element!

## Hide an Element - display:none or visibility:hidden?

Hiding an element can be done by setting the **display** property to **none**. The element will be hidden, and the page will be displayed as if the element is not there:

```
h1.hidden {
  display: none;
}
```

**visibility: hidden**; also hides an element.

However, the element will still take up the same space as before. The element will be hidden, but still affect the layout:

```
h1.hidden {  
  visibility: hidden;  
}
```

---

## @media only

It notifies older browsers that don't support the queries to ignore the code

The only query is for the screen

## @media only screen

*means this setting is only for the device that is being used to visit the site with*

## @media only screen and

Says this is media only for a screen **and** the settings are

**@media only screen and (min-width: 400px){}**

**only**: The only keyword prevents older browsers that do not support media queries with media features from applying the specified styles. It has no effect on modern browsers.

The **screen** keyword references that it's a computer, mobile phone or tablet etc. There are two other media types, **print** and **speech**, as well as the default **all**.

Simply: The **only** keyword is optional and used for backwards compatibility. The **screen** keyword will default to **all**.