

# Chapter 6: Centering

## Section 6.1: Using Flexbox

HTML:

```
<div class="container">
  
</div>
```

CSS:

```
html, body, .container {
  height: 100%;
}
.container {
  display: flex;
  justify-content: center; /* horizontal center */
}
img {
  align-self: center; /* vertical center */
}
```

[View Result](#)

HTML:

```

```

CSS:

```
html, body {
  height: 100%;
}
body {
  display: flex;
  justify-content: center; /* horizontal center */
  align-items: center; /* vertical center */
}
```

[View Result](#)

See Dynamic Vertical and Horizontal Centering under the Flexbox documentation for more details on flexbox and what the styles mean.

### Browser Support

Flexbox is supported by all major browsers, [except IE versions before 10](#).

Some recent browser versions, such as Safari 8 and IE10, require [vendor prefixes](#).

For a quick way to generate prefixes there is [Autoprefixer](#), a third-party tool.

For older browsers (like IE 8 & 9) a [Polyfill is available](#).

For a more detailed look at flexbox browser support, see [this answer](#).

## Section 6.2: Using CSS transform

CSS transforms are based on the size of the elements so if you don't know how tall or wide your element is, you can position it absolutely 50% from the top and left of a relative container and translate it by 50% left and upwards to center it vertically and horizontally.

Keep in mind that with this technique, the element could end being rendered at a non-integer pixel boundary, making it look blurry. See [this answer in SO](#) for a workaround.

### HTML

```
<div class="container">
  <div class="element"></div>
</div>
```

### CSS

```
.container {
  position: relative;
}

.element {
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
}
```

[View example in JSFiddle](#)

### CROSS BROWSER COMPATIBILITY

The transform property needs prefixes to be supported by older browsers. Prefixes are needed for Chrome<=35, Safari<=8, Opera<=22, Android Browser<=4.4.4, and IE9. CSS transforms are not supported by IE8 and older versions.

Here is a common transform declaration for the previous example:

```
-webkit-transform: translate(-50%, -50%); /* Chrome, Safari, Opera, Android */
-ms-transform: translate(-50%, -50%); /* IE 9 */
transform: translate(-50%, -50%);
```

For more information see [canluse](#).

### MORE INFORMATION

- The element is being positioned according to the first non-static parent (**position**: **relative**, **absolute**, or **fixed**). Explore more in this [fiddle](#) and this documentation topic.
- For horizontal-only centering, use **left**: **50%** and **transform**: **translateX(-50%)**. The same goes for vertical-only centering: center with **top**: **50%** and **transform**: **translateY(-50%)**.
- Using a non-static width/height elements with this method of centering can cause the centered element to appear squished. This mostly happens with elements containing text, and can be fixed by adding: **margin-right**: **-50%**; and **margin-bottom**: **-50%**; View this [fiddle](#) for more information.

## Section 6.3: Using margin: 0 auto;

Objects can be centered by using `margin: 0 auto;` if they are block elements and have a defined width.

### HTML

```
<div class="containerDiv">
  <div id="centeredDiv"></div>
</div>

<div class="containerDiv">
  <p id="centeredParagraph">This is a centered paragraph.</p>
</div>

<div class="containerDiv">
  
</div>
```

### CSS

```
.containerDiv {
  width: 100%;
  height: 100px;
  padding-bottom: 40px;
}

#centeredDiv {
  margin: 0 auto;
  width: 200px;
  height: 100px;
  border: 1px solid #000;
}

#centeredParagraph {
  width: 200px;
  margin: 0 auto;
}

#centeredImage {
  display: block;
  width: 200px;
  margin: 0 auto;
}
```

Result:



This is a centered paragraph.



JSFiddle example: [Centering objects with margin: 0 auto;](#)

## Section 6.4: Using text-align

The most common and easiest type of centering is that of lines of text in an element. CSS has the rule **text-align: center** for this purpose:

### HTML

```
<p>Lorem ipsum</p>
```

### CSS

```
p {  
  text-align: center;  
}
```

*This does not work for centering entire block elements.* text-align controls only alignment of inline content like text in its parent block element.

See more about text-align in Typography section.

## Section 6.5: Using position: absolute

*Working in old browsers (IE >= 8)*

Automatic margins, paired with values of zero for the **left** and **right** or **top** and **bottom** offsets, will center an absolutely positioned elements within its parent.

[View Result](#)

### HTML

```
<div class="parent">
  
</div>
```

## CSS

```
.parent {
  position: relative;
  height: 500px;
}

.center {
  position: absolute;
  margin: auto;
  top: 0;
  right: 0;
  bottom: 0;
  left: 0;
}
```

Elements that don't have their own implicit width and height like images do, will need those values defined.

Other resources: [Absolute Centering in CSS](#)

## Section 6.6: Using calc()

The calc() function is the part of a new syntax in CSS3 in which you can calculate (mathematically) what size/position your element occupies by using a variety of values like pixels, percentages, etc. Note: Whenever you use this function, always take care of the space between two values `calc(100% - 80px)`.

## CSS

```
.center {
  position: absolute;
  height: 50px;
  width: 50px;
  background: red;
  top: calc(50% - 50px / 2); /* height divided by 2*/
  left: calc(50% - 50px / 2); /* width divided by 2*/
}
```

## HTML

```
<div class="center"></div>
```

## Section 6.7: Using line-height

You can also use line-height to center vertically a single line of text inside a container :

## CSS

```
div {
  height: 200px;
  line-height: 200px;
}
```

That's quite ugly, but can be useful inside an `<input />` element. The `line-height` property works only when the text to be centered spans a single line. If the text wraps into multiple lines, the resulting output won't be centered.

## Section 6.8: Vertical align anything with 3 lines of code

[Supported by IE11+](#)

[View Result](#)

Use these 3 lines to vertical align practically everything. Just make sure the div/image you apply the code to has a parent with a height.

### CSS

```
div.vertical {  
  position: relative;  
  top: 50%;  
  transform: translateY(-50%);  
}
```

### HTML

```
<div class="vertical">Vertical aligned text!</div>
```

## Section 6.9: Centering in relation to another item

We will see how to center content based on the height of a near element.

Compatibility: IE8+, all other modern browsers.

### HTML

```
<div class="content">  
  <div class="position-container">  
    <div class="thumb">  
        
    </div>  
    <div class="details">  
      <p class="banner-title">text 1</p>  
      <p class="banner-text">content content content content content content content  
content content content content content content content</p>  
      <button class="btn">button</button>  
    </div>  
  </div>  
</div>
```

### CSS

```
.content * {  
  box-sizing: border-box;  
}  
.content .position-container {  
  display: table;  
}  
.content .details {  
  display: table-cell;  
  vertical-align: middle;
```

```

width: 33.33333%;
padding: 30px;
font-size: 17px;
text-align: center;
}
.content .thumb {
width: 100%;
}
.content .thumb img {
width: 100%;
}

```

Link to [JSFiddle](#)

The main points are the 3 `.thumb`, `.details` and `.position-container` containers:

- The `.position-container` must have **display: table**.
- The `.details` must have the real width set **width: ...** and **display: table-cell**, **vertical-align: middle**.
- The `.thumb` must have **width: 100%** if you want that it will take all the remaining space and it will be influenced by the `.details` width.
- The image (if you have an image) inside `.thumb` should have **width: 100%**, but it is not necessary if you have correct proportions.

## Section 6.10: Ghost element technique (Michał Czernew's hack)

This technique works even when the container's dimensions are unknown.

Set up a "ghost" element inside the container to be centered that is 100% height, then use **vertical-align: middle** on both that and the element to be centered.

### CSS

```

/* This parent can be any width and height */
.block {
text-align: center;

/* May want to do this if there is risk the container may be narrower than the element inside */
white-space: nowrap;
}

/* The ghost element */
.block:before {
content: '';
display: inline-block;
height: 100%;
vertical-align: middle;

/* There is a gap between ghost element and .centered,
caused by space character rendered. Could be eliminated by
nudging .centered (nudge distance depends on font family),
or by zeroing font-size in .parent and resetting it back
(probably to 1rem) in .centered. */
margin-right: -0.25em;
}

```

```

/* The element to be centered, can also be of any width and height */
.centered {
    display: inline-block;
    vertical-align: middle;
    width: 300px;
    white-space: normal; /* Resetting inherited nowrap behavior */
}

```

## HTML

```

<div class="block">
    <div class="centered"></div>
</div>

```

## Section 6.11: Centering vertically and horizontally without worrying about height or width

The following technique allows you to add your content to an HTML element and center it both horizontally and vertically **without worrying about its height or width**.

### The outer container

- should have `display: table;`

### The inner container

- should have `display: table-cell;`
- should have `vertical-align: middle;`
- should have `text-align: center;`

### The content box

- should have `display: inline-block;`
- should re-adjust the horizontal text-alignment to eg. `text-align: left;` or `text-align: right;`, unless you want text to be centered

## Demo

### HTML

```

<div class="outer-container">
    <div class="inner-container">
        <div class="centered-content">
            You can put anything here!
        </div>
    </div>
</div>

```

### CSS

```

body {
    margin : 0;
}

.outer-container {
    position : absolute;
    display: table;
    width: 100%; /* This could be ANY width */
}

```

```

height: 100%; /* This could be ANY height */
background: #ccc;
}

.inner-container {
display: table-cell;
vertical-align: middle;
text-align: center;
}

.centered-content {
display: inline-block;
text-align: left;
background: #fff;
padding: 20px;
border: 1px solid #000;
}

```

See also [this Fiddle!](#)

## Section 6.12: Vertically align an image inside div

### HTML

```

<div class="wrap">
  
</div>

```

### CSS

```

.wrap {
height: 50px; /* max image height */
width: 100px;
border: 1px solid blue;
text-align: center;
}

.wrap:before {
content: "";
display: inline-block;
height: 100%;
vertical-align: middle;
width: 1px;
}

img {
vertical-align: middle;
}

```

## Section 6.13: Centering with fixed size

If the size of your content is fixed, you can use absolute positioning to 50% with margin that reduces half of your content's width and height:

### HTML

```

<div class="center">
  Center vertically and horizontally
</div>

```

## CSS

```
.center {  
    position: absolute;  
    background: #ccc;  
  
    left: 50%;  
    width: 150px;  
    margin-left: -75px; /* width * -0.5 */  
  
    top: 50%;  
    height: 200px;  
    margin-top: -100px; /* height * -0.5 */  
}
```

### Horizontal centering with only fixed width

You can center the element horizontally even if you don't know the height of the content:

## HTML

```
<div class="center">  
    Center only horizontally  
</div>
```

## CSS

```
.center {  
    position: absolute;  
    background: #ccc;  
  
    left: 50%;  
    width: 150px;  
    margin-left: -75px; /* width * -0.5 */  
}
```

### Vertical centering with fixed height

You can center the element vertically if you know the element's height:

## HTML

```
<div class="center">  
    Center only vertically  
</div>
```

## CSS

```
.center {  
    position: absolute;  
    background: #ccc;  
  
    top: 50%;  
    height: 200px;  
    margin-top: -100px; /* width * -0.5 */  
}
```

## Section 6.14: Vertically align dynamic height elements

Applying css intuitively doesn't produce the desired results because

- **vertical-align:middle** *isn't* applicable to block-level elements
- **margin-top:auto** and **margin-bottom:auto** used values would compute as **zero**
- **margin-top:-50%** percentage-based margin values are calculated relative to the **width** of containing block

For widest browser support, a workaround with helper elements:

### HTML

```
<div class="vcenter--container">
  <div class="vcenter--helper">
    <div class="vcenter--content">
      <!--stuff-->
    </div>
  </div>
</div>
```

### CSS

```
.vcenter--container {
  display: table;
  height: 100%;
  position: absolute;
  overflow: hidden;
  width: 100%;
}
.vcenter--helper {
  display: table-cell;
  vertical-align: middle;
}
.vcenter--content {
  margin: 0 auto;
  width: 200px;
}
```

[jsfiddle](#) from [original question](#). This approach

- works with dynamic height elements
- respects content flow
- is supported by legacy browsers

## Section 6.15: Horizontal and Vertical centering using table layout

One could easily center a child element using **table** display property.

### HTML

```
<div class="wrapper">
  <div class="parent">
    <div class="child"></div>
  </div>
</div>
```

```
.wrapper {  
  display: table;  
  vertical-align: center;  
  width: 200px;  
  height: 200px;  
  background-color: #9e9e9e;  
}  
.parent {  
  display: table-cell;  
  vertical-align: middle;  
  text-align: center;  
}  
.child {  
  display: inline-block;  
  vertical-align: middle;  
  text-align: center;  
  width: 100px;  
  height: 100px;  
  background-color: teal;  
}
```