CSS Layout - Horizontal & Vertical Align

Center Align Elements

To horizontally center a block element (like), use `margin: auto;`

Setting the width of the element will prevent it from stretching out to the edges of its container.

The element will then take up the specified width, and the remaining space will be split equally between the two margins:

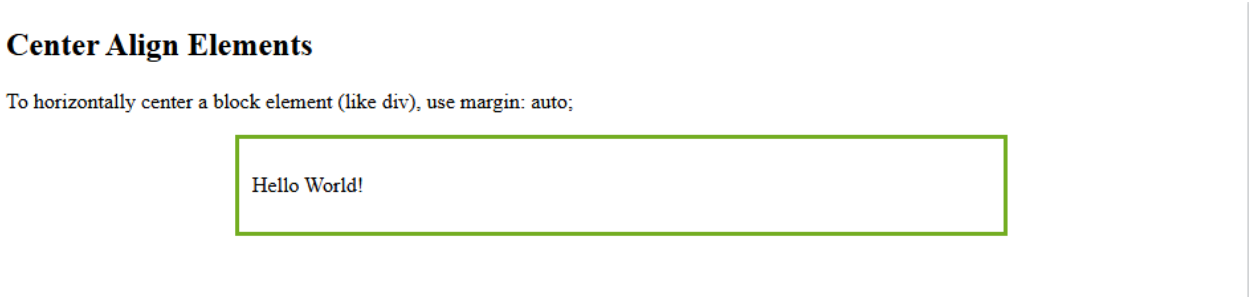
This div element is centered.

Example

```
.center {
  margin: auto;
  width: 50%;
  border: 3px solid green;
  padding: 10px;
}
Example:-
<!DOCTYPE html>
<html>
<head>
<style>
.center {
  margin: auto;
  width: 60%;
  border: 3px solid #73AD21;
  padding: 10px;
}
</style>
</head>
<body>
```

```
<h2>Center Align Elements</h2>
<p>To horizontally center a block element (like div), use margin:
auto;</p>

<div class="center">
  <p>Hello World!</p>
</div>

</body>
</html>
```

Output:-



Note: Center aligning has no effect if the `width` property is not set (or set to 100%).

Center Align Text

To just center the text inside an element, use `text-align: center;`

This text is centered.

Example

```
.center {
  text-align: center;
  border: 3px solid green;
}
```

Center an Image

To center an image, set left and right margin to `auto` and make it into a `block` element:

Example

```
img {
  display: block;
  margin-left: auto;
  margin-right: auto;
  width: 40%;
}
```

Left and Right Align - Using position

One method for aligning elements is to use `position: absolute;`:

In my younger and more vulnerable years my father gave me some advice that I've been turning over in my mind ever since.

Example

```
.right {
  position: absolute;
```

```
  right: 0px;
  width: 300px;
  border: 3px solid #73AD21;
  padding: 10px;
}
```

Example :

```
<!DOCTYPE html>

<html>

<head>

<style>

.right {

 position: absolute;

 right: 0px;

 width: 300px;

 border: 3px solid #73AD21;

 padding: 10px;

}

</style>

</head>

<body>


<h2>Right align with the position property</h2>


<p>An example of how to right align elements with the position property:</p>


<div class="right">

 <p>In my younger and more vulnerable years my father gave me some advice that I've been turning over in my mind ever since.</p>
```

</body>

</html>

Output:-

Note: Absolute positioned elements are removed from the normal flow, and can overlap elements.

Left and Right Align - Using float

Another method for aligning elements is to use the `float` property:

Example

```
.right {
  float: right;
  width: 300px;
  border: 3px solid #73AD21;
  padding: 10px;
}
```

Example :-

<!DOCTYPE html>

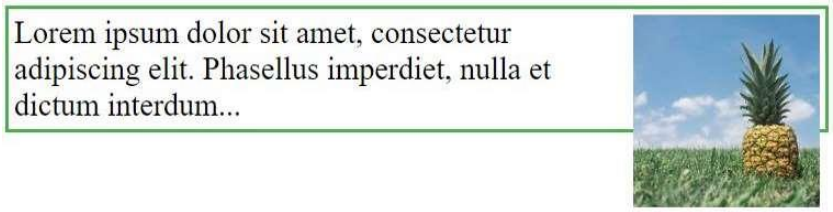
```
<html>

<head>

<style>

.right {

  float: right;

  width: 300px;

  border: 3px solid #73AD21;

  padding: 10px;

}

</style>

</head>

<body>


<h2>Right align with the float property</h2>


<p>An example of how to right align elements with the float property:</p>


<div class="right">

  <p>In my younger and more vulnerable years my father gave me some advice that I've been turning over in my mind ever since.</p>

</div>


</body>
```

</html>

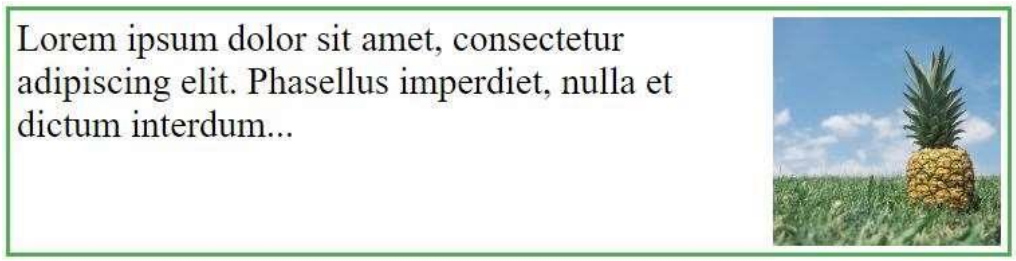
The clearfix Hack

Note: If an element is taller than the element containing it, and it is floated, it will overflow outside of its container. You can use the "clearfix hack" to fix this (see example below).

Without Clearfix



With Clearfix



Then we can add the clearfix hack to the containing element to fix this problem:

Example

```
.clearfix::after {
  content: "";
  clear: both;
```

```
  display: table;
}
```

Example :-

```
<!DOCTYPE html>

<html>

<head>

<style>

div {

  border: 3px solid #4CAF50;

  padding: 5px;

}


.img1 {

  float: right;

}


.img2 {

  float: right;

}


.clearfix::after {

  content: "";

  clear: both;
```

```
  display: table;
}
</style>
</head>
<body>

<h2>Without Clearfix</h2>

<p>This image is floated to the right. It is also taller than the element containing it, so it overflows outside of its container:</p>

<div>
  <img class="img1" src="pineapple.jpg" alt="Pineapple" width="170" height="170">
  Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet...
</div>

<h2 style="clear:right">With New Modern Clearfix</h2>
<p>Add the clearfix hack to the containing element, to fix this problem:</p>

<div class="clearfix">
  <img class="img2" src="pineapple.jpg" alt="Pineapple" width="170" height="170">
  Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet...
</div>
```

</body>

</html>

Center Vertically - Using padding

There are many ways to center an element vertically in CSS. A simple solution is to use top and bottom `padding`:

I am vertically centered.

Example

```
.center {
  padding: 70px 0;
  border: 3px solid green;
}
```

To center both vertically and horizontally, use `padding` and `text-align: center`:

I am vertically and horizontally centered.

Example

```
.center {
  padding: 70px 0;
  border: 3px solid green;
  text-align: center;
}
```

Center Vertically - Using line-height

Another trick is to use the `line-height` property with a value that is equal to the `height` property:

I am vertically and horizontally centered.

Example

```
.center {
  line-height: 200px;
  height: 200px;
  border: 3px solid green;
  text-align: center;
}

/* If the text has multiple lines, add the following: */
.center p {
  line-height: 1.5;
  display: inline-block;
  vertical-align: middle;
}
```

Example :-

<!DOCTYPE html>

<html>

<head>

<style>

.center {

line-height: 200px;

height: 200px;

border: 3px solid green;

text-align: center;

}

```
.center p {
  line-height: 1.5;
  display: inline-block;
  vertical-align: middle;
}
</style>
</head>
<body>

<h2>Center with line-height</h2>

<p>In this example, we use the line-height property with a value that is equal to the height property to center the div element:</p>

<div class="center">
  <p>I am vertically and horizontally centered.</p>
</div>

</body>
</html>
```

Center Vertically - Using position & transform

If `padding` and `line-height` are not options, another solution is to use positioning and the `transform` property:

I am vertically and horizontally centered.

Example

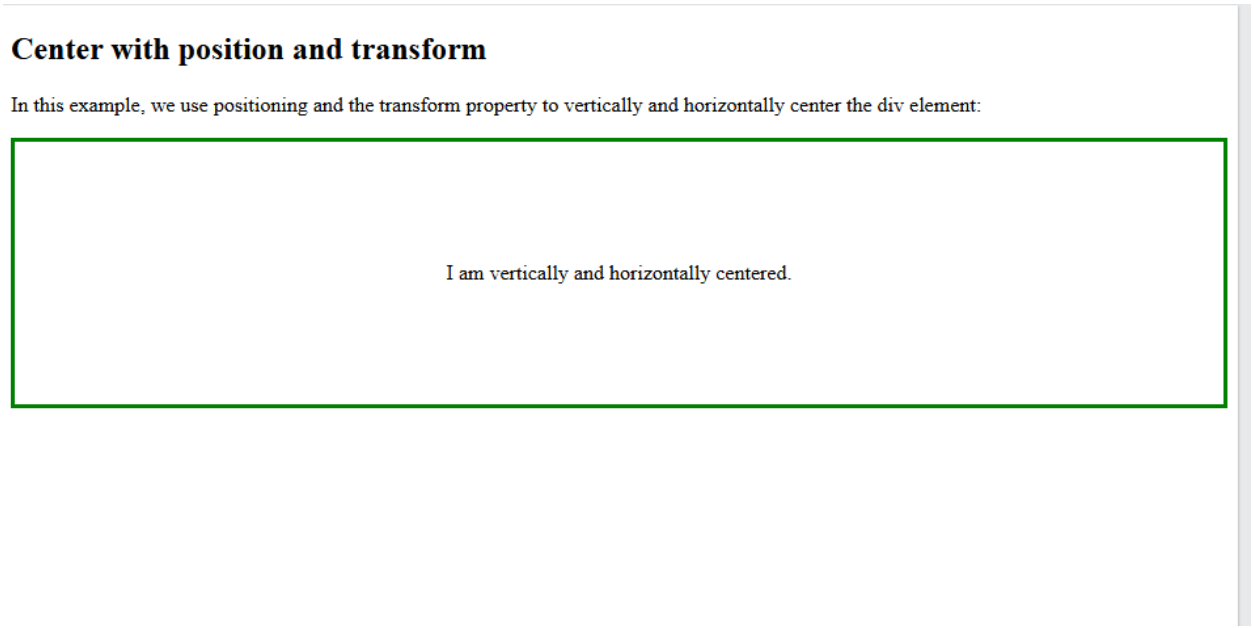
```
.center {
  height: 200px;
  position: relative;
  border: 3px solid green;
}

.center p {
  margin: 0;
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
}
```

Example

```
<!DOCTYPE html>

<html>

<head>

<style>

.center {

 height: 200px;

 position: relative;

 border: 3px solid green;

}
```

```
.center p {
  margin: 0;
  position: absolute;
  top: 50%;
  left: 50%;
  -ms-transform: translate(-50%, -50%);
  transform: translate(-50%, -50%);
}
</style>
</head>
<body>

<h2>Center with position and transform</h2>

<p>In this example, we use positioning and the transform property to vertically and horizontally center the div element:</p>

<div class="center">
  <p>I am vertically and horizontally centered.</p>
</div>

</body>
</html>.
```

Output:-



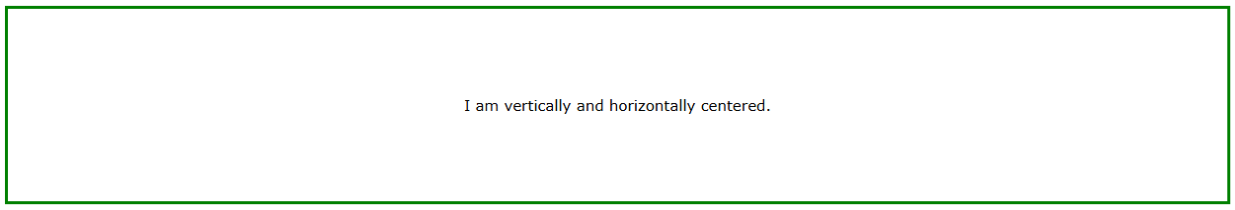
Center Vertically - Using Flexbox

You can also use flexbox to center things. Just note that flexbox is not supported in IE10 and earlier versions:

I am vertically and horizontally centered.

Example

```
.center {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 200px;
  border: 3px solid green;
}
```




```
<!DOCTYPE html>
```

```
<html>
<head>
<style>
.center {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 200px;
  border: 3px solid green;
}
</style>
</head>
<body>

<h2>Flexbox Centering</h2>

<p>A container with both the justify-content and the align-items properties set to <em>center</em> will align the item(s) in the center (in both axis).</p>

<div class="center">
  <p>I am vertically and horizontally centered.</p>
</div>

</body>
</html>
```

CSS `align-items` and `align-self` are properties that help with vertical alignment in Flexbox, especially when the flex container's height is larger than the content's height. Here's a breakdown of how they work with examples:

1. `align-items`: Aligns all flex items within a flex container along the cross axis (usually the vertical axis, unless `flex-direction` is `column`).

2. `align-self`: Allows individual flex items to override the alignment set by `align-items`.

Flexbox Alignment Properties:

- **`align-items`** (on container)
- **`align-self`** (on individual items)

Here are the values for both:

- **`flex-start`**: Aligns the items to the start of the cross axis.
- **`flex-end`**: Aligns the items to the end of the cross axis.
- **`center`**: Aligns the items to the center of the cross axis.
- **`baseline`**: Aligns items along their baselines (useful for text).
- **`stretch`** (default): Stretches the items to fill the container along the cross axis.

Example 1: Using `align-items`

In this example, all items are aligned vertically to the **center** of the container.

HTML:

```
Html code:-
<div class="container">
  <div class="item">Item 1</div>
  <div class="item">Item 2</div>
  <div class="item">Item 3</div>
</div>
```

CSS:

```
Css code
.container {
  display: flex;
  height: 200px; /* Container height to show vertical alignment */
  align-items: center; /* Vertically center the items */
  background-color: lightblue;
  gap: 10px; /* Space between items */
}

.item {
```

```
  width: 100px;
  height: 50px;
  background-color: lightcoral;
  display: flex;
  justify-content: center;
  align-items: center; /* Center text within each item */
}
```

Explanation:

- `align-items: center;` aligns all flex items vertically in the center of the container.
- The height of the container is `200px`, which gives space to see the vertical alignment in action.

Example 2: Using `align-self` for Individual Items

Now, we will override the `align-items` value for a specific item using `align-self`.

HTML:

```
Html code
<div class="container">
  <div class="item">Item 1</div>
  <div class="item item-special">Item 2 (Aligned differently)</div>
  <div class="item">Item 3</div>
</div>
```

CSS:

```
Css code
.container {
  display: flex;
  height: 200px; /* Container height */
  align-items: flex-start; /* Align all items to the start of the container
*/
  background-color: lightblue;
  gap: 10px;
}

.item {
  width: 100px;
  height: 50px;
  background-color: lightcoral;
  display: flex;
  justify-content: center;
  align-items: center; /* Center text inside each item */
}

.item-special {
  align-self: flex-end; /* Align only Item 2 to the end of the container */
}
```

Explanation:

- `align-items: flex-start;` aligns all items at the top of the container.
- `align-self: flex-end;` on `.item-special` overrides the container's alignment and aligns "Item 2" to the bottom of the container.

Example 3: Using `align-items` with `flex-direction: column`

If you change the flex direction to `column`, the cross axis becomes horizontal, and the alignment affects the horizontal axis.

HTML:

```
Html code
<div class="container">
  <div class="item">Item 1</div>
  <div class="item">Item 2</div>
  <div class="item">Item 3</div>
</div>
```

CSS:

```
Css code
.container {
  display: flex;
  flex-direction: column; /* Stack items vertically */
  height: 200px; /* Container height */
  align-items: center; /* Horizontally center the items */
  background-color: lightblue;
  gap: 10px;
}

.item {
  width: 100px;
  height: 50px;
  background-color: lightcoral;
  display: flex;
  justify-content: center;
  align-items: center; /* Center text inside each item */
}
```

Explanation:

- `flex-direction: column;` stacks the items vertically.
- `align-items: center;` aligns the items horizontally in the center of the container.

Example 4: Using `align-items: baseline` for Text Alignment

This example demonstrates how `align-items: baseline` aligns items based on their text baseline.

HTML:

Html code:-

```
<div class="container">
  <div class="item">Short Text</div>
  <div class="item">Much longer text here</div>
</div>
```

CSS:

Css code:

```
.container {
  display: flex;
  align-items: baseline; /* Align items based on their text baseline */
  height: 100px; /* Container height */
  gap: 10px;
  background-color: lightblue;
}

.item {
  background-color: lightcoral;
  padding: 10px;
  font-size: 18px;
}
```

Explanation:

- `align-items: baseline;` aligns the items based on the baseline of their text, which ensures the text in both items aligns properly.

Summary:

- Use **`align-items`** on the container to align all items along the cross axis (vertically if `flex-direction` is `row`, horizontally if `flex-direction` is `column`).
- Use **`align-self`** on individual items to override the container's `align-items` alignment for that specific item.

These Flexbox properties are highly flexible and can be combined to create complex and responsive layouts!