

CSS Layout - The position Property

[< Previous](#)[Next >](#)

The **position** property specifies the type of positioning method used for an element (static, relative, fixed, absolute or sticky).

The position Property

The **position** property specifies the type of positioning method used for an element.

There are five different position values:

- **static**
- **relative**
- **fixed**
- **absolute**
- **sticky**

Elements are then positioned using the top, bottom, left, and right properties. However, these properties will not work unless the **position** property is set first. They also work differently depending on the position value.

position: static;

HTML elements are positioned static by default.

Static positioned elements are not affected by the top, bottom, left, and right properties.

An element with **position: static;** is not positioned in any special way; it is always positioned according to the normal flow of the page:

This <div> element has **position: fixed;**

This <div> element has position: static;

Here is the CSS that is used:

Example

```
div.static {  
  position: static;  
  border: 3px solid #73AD21;  
}
```

Try it Yourself »

position: relative;

An element with `position: relative;` is positioned relative to its normal position.

Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.

This <div> element has position: relative;

Here is the CSS that is used:

Example

```
div.relative {  
  position: relative;  
  left: 30px;  
  border: 3px solid #73AD21;  
}
```

Try it Yourself »

This <div> element has `position: fixed;`

position: fixed;

An element with `position: fixed;` is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled. The top, right, bottom, and left properties are used to position the element.

A fixed element does not leave a gap in the page where it would normally have been located.

Notice the fixed element in the lower-right corner of the page. Here is the CSS that is used:

Example

```
div.fixed {  
  position: fixed;  
  bottom: 0;  
  right: 0;  
  width: 300px;  
  border: 3px solid #73AD21;  
}
```

Try it Yourself »

position: absolute;

An element with `position: absolute;` is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed).

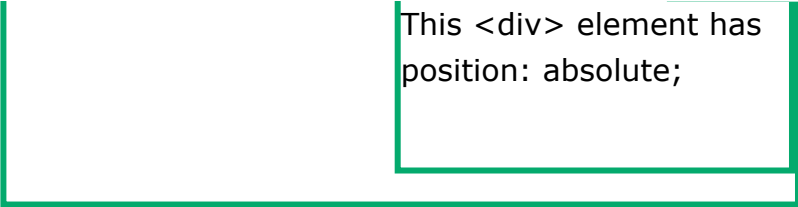
However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.

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

Here is a simple example:

This <div> element has position: relative;

This <div> element has `position: fixed;`



This <div> element has
position: absolute;

Here is the CSS that is used:

Example

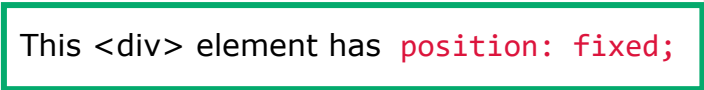
```
div.relative {  
  position: relative;  
  width: 400px;  
  height: 200px;  
  border: 3px solid #73AD21;  
}  
  
div.absolute {  
  position: absolute;  
  top: 80px;  
  right: 0;  
  width: 200px;  
  height: 100px;  
  border: 3px solid #73AD21;  
}
```

Try it Yourself »

position: sticky;

An element with `position: sticky;` is positioned based on the user's scroll position.

A sticky element toggles between `relative` and `fixed`, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like `position: fixed`).



This <div> element has `position: fixed;`

Try to scroll inside this frame to understand how sticky positioning works.

I am sticky!

Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id

Note: Internet Explorer does not support sticky positioning. Safari requires a -webkit- prefix (see example below). You must also specify at least one of **top**, **right**, **bottom** or **left** for sticky positioning to work.

In this example, the sticky element sticks to the top of the page (**top: 0**), when you reach its scroll position.

Example

```
div.sticky {  
  position: -webkit-sticky; /* Safari */  
  position: sticky;  
  top: 0;  
  background-color: green;  
  border: 2px solid #4CAF50;  
}
```

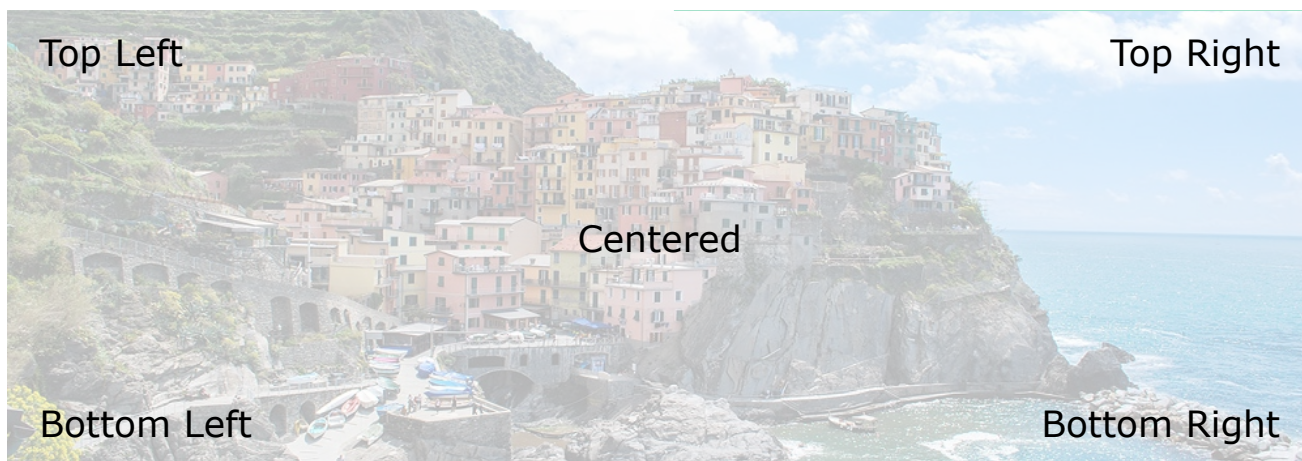
Try it Yourself »

Positioning Text In an Image

How to position text over an image:

Example

This <div> element has **position: fixed;**



Try it Yourself:

[Top Left »](#)[Top Right »](#)[Bottom Left »](#)[Bottom Right »](#)[Centered »](#)

More Examples

Set the shape of an element

This example demonstrates how to set the shape of an element. The element is clipped into this shape, and displayed.

Test Yourself With Exercises

Exercise:

Position the `<h1>` element to always be 50px from the top, and 10px from the right, relative to the window/frame edges.

```
<style>
h1 {
    position: fixed;
    top: 50px;
    right: 10px;
}
```

This `<div>` element has `position: fixed;`

```
</style>
```

```
<body>  
  <h1>This is a heading</h1>  
  <p>This is a paragraph</p>  
  <p>This is a paragraph</p>  
</body>
```

[Submit Answer »](#)[Start the Exercise](#)

All CSS Positioning Properties

Property	Description
<u>bottom</u>	Sets the bottom margin edge for a positioned box
<u>clip</u>	Clips an absolutely positioned element
<u>left</u>	Sets the left margin edge for a positioned box
<u>position</u>	Specifies the type of positioning for an element
<u>right</u>	Sets the right margin edge for a positioned box
<u>top</u>	Sets the top margin edge for a positioned box

[< Previous](#)[Next >](#)

NEW This <div> element has **position: fixed;**