CSS

Topics:

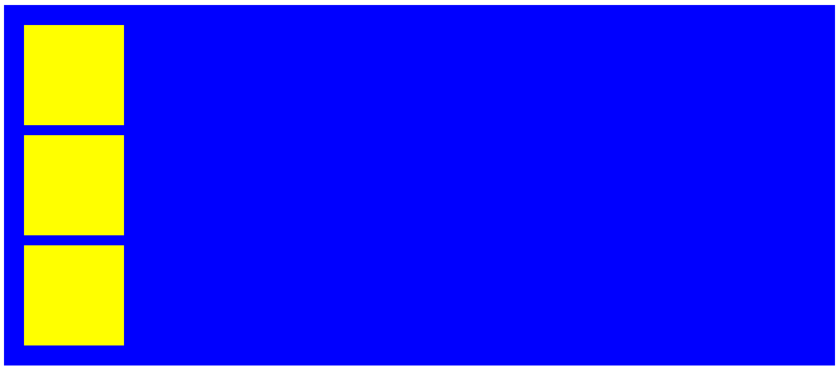
- Positions
- List Styling
- CSS Reset
- Pseudo Classes / Selectors
- Box Shadow
- Transitions
- Transform

Positions:

- Static
- Fixed
- Relative
- Absolute

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.

This fixed position is with respect to the viewport or the screen size that a user would be able to see and related to it, the position of an element is fixed.

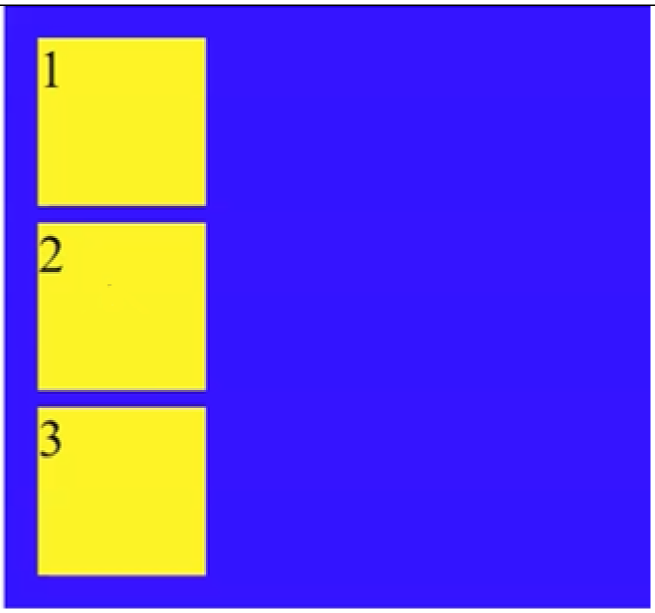
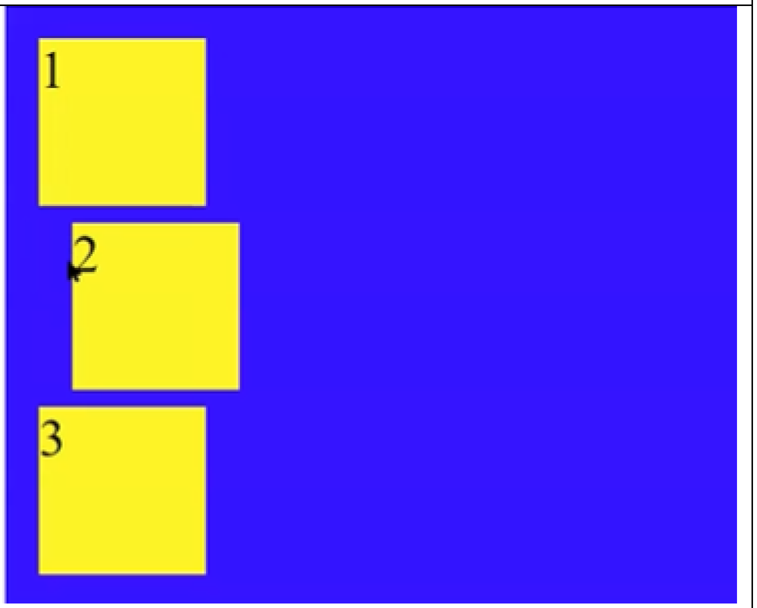
Position-Relative:

In html:

```
<div class="box-container">
        <div class="box">1</div>
        <div class="box box-2">2</div>
        <div class="box">3</div>
</div>
```

In css:

```
.box-2
{
        position: relative;
        bottom: 0px;
        left: 10px;
}
```

Before making any changes,	**After making changes**
	

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.

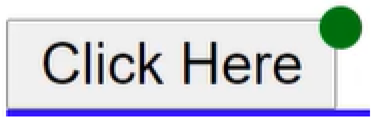
Position-Absolute:

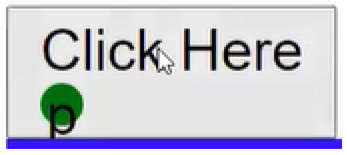
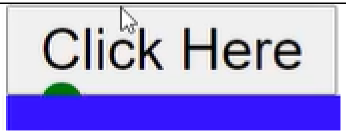
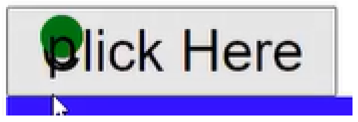
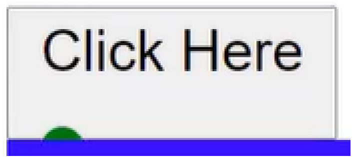
An element with position: absolute is positioned relative to the nearest position ancestor (instead of positioned relative to the viewport, like fixed). However, if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along the page scrolling.

In css:	In html

.box-container {
    background-color: blue;
    padding: 5px;
    position: relative;
}
.box {
    height: 50px;
    width: 50px;
    margin: 5px 5px;
    background-color: yellow;
}
.box-2
{
    position: absolute;
    bottom: 10px;
    left: 100px;
}
	<div class="box-container">
	    <div class="box">1</div>
	    <div class="box box-2">2</div>
	    <div class="box">3</div>
	</div>

In css:	**In html:**

```
.badge {
    width: 10px;
    height: 100px;
    background-color: green;
    position: absolute;
    border-radius: 50%;
    top: -5px;
    right: -8px;
    display: inline;
}
button {
    position: relative;
}
```
	<button>click here<div class="badge"></div></button>
	
	OUTPUT:
	

If the positions of the badge element and the button element are removed, the badge would still be inside the button element.	 this is the normal flow of the document.
To this, if we put in position: absolute the badge element, the div element is not shown in the button, as the button is continued directly by the next element	
Now, if we put top:10px in the badge element, the position of the green dot would be clearly visible on the button.	
If we put position: relative, the space would be present but the position of the dot has moved down 10px;	

The normal flow of the document is still intact, because my button is still considering the amount of spacing it needs to accommodate the green dot (div element).

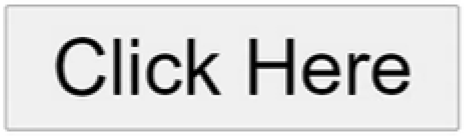
Position-Static:

If you have not used any of the three position-attributes (fixed, absolute or relative), then by default the position of the element would be static.

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:

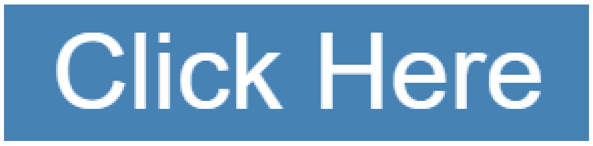
Pseudo Classes / Selectors:

Pseudo Classes are specified on selectors to specify a state or relation to the selector.

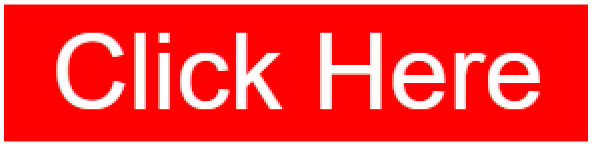


<button class="btn">Click Here</button>

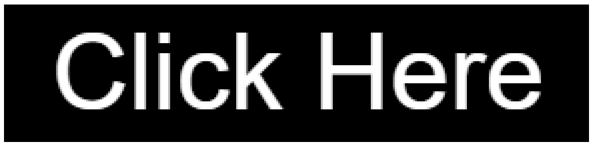
If we add some properties to the button such as,

.btn {
background-color: steelblue;
color: white;
border: none;
outline: none;
}
	

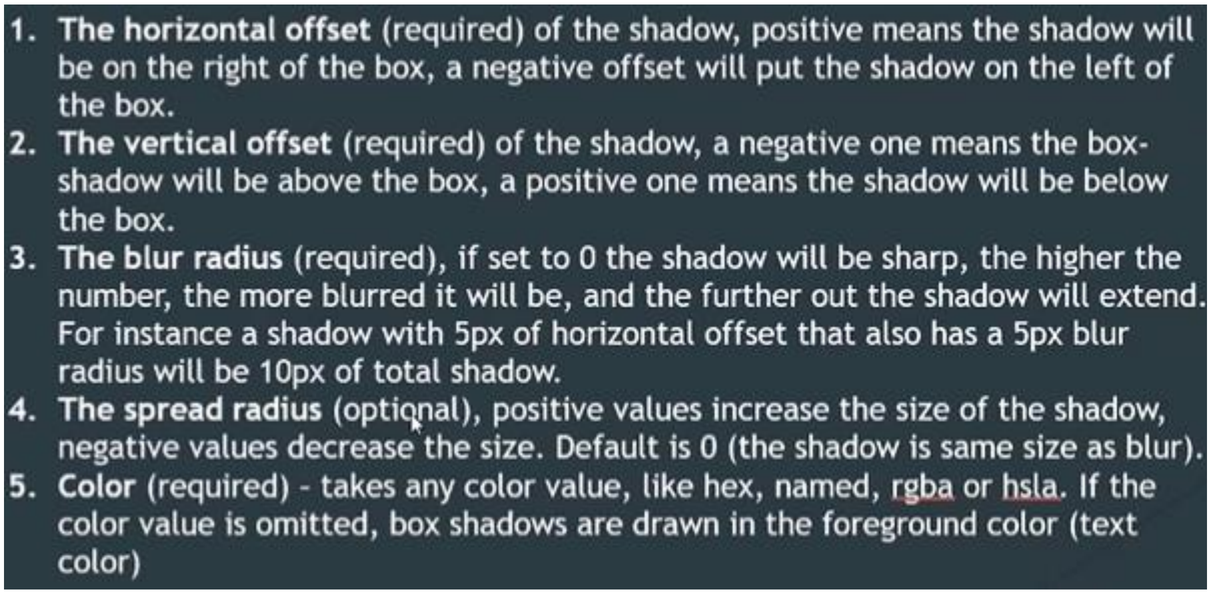
When add a hover class, then every time, the cursor hovers over the button, the properties of the button changes. It is based on the properties that we have mentioned in the .btn:hover {}.

.btn:hover {
background-color: red;
}
	

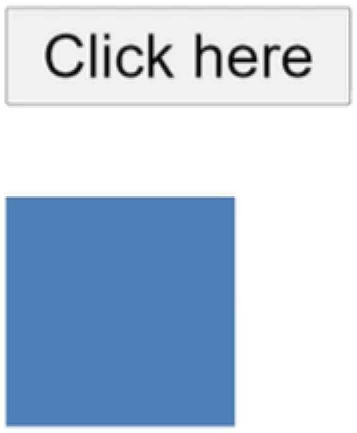
If we click and hold the button, it would change to black background with white color text. All this because we have used another pseudo class/selector → active

.btn:active {
background-color: black;
color: white;
}
	

Box Shadow:



1. **The horizontal offset** (required) of the shadow, positive means the shadow will be on the right of the box, a negative offset will put the shadow on the left of the box.
2. **The vertical offset** (required) of the shadow, a negative one means the box-shadow will be above the box, a positive one means the shadow will be below the box.
3. **The blur radius** (required), if set to 0 the shadow will be sharp, the higher the number, the more blurred it will be, and the further out the shadow will extend. For instance a shadow with 5px of horizontal offset that also has a 5px blur radius will be 10px of total shadow.
4. **The spread radius** (optional), positive values increase the size of the shadow, negative values decrease the size. Default is 0 (the shadow is same size as blur).
5. **Color** (required) - takes any color value, like hex, named, rgba or hsla. If the color value is omitted, box shadows are drawn in the foreground color (text color)

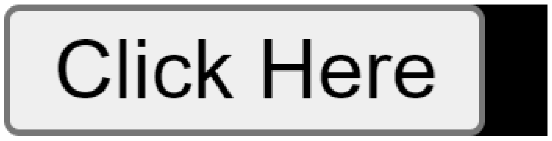
In html
```<button class="">Click Here</button>```
```<div class="box"></div>```
 	
In css:
```.box {```
```background-color: steelblue;```
```width: 50px;```
```height: 50px;```
```margin: 20px;```
```}```
	

Syntax:

box-shadow: none|*h-offset v-offset blur spread color* |inset|initial|inherit;

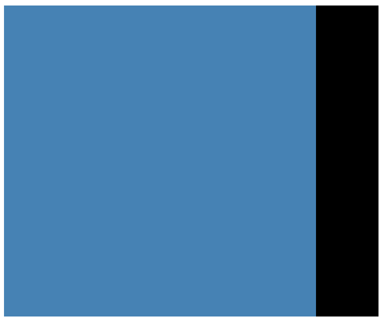
In html

```
<button class="">Click Here</button>
<div class="box"></div>
```

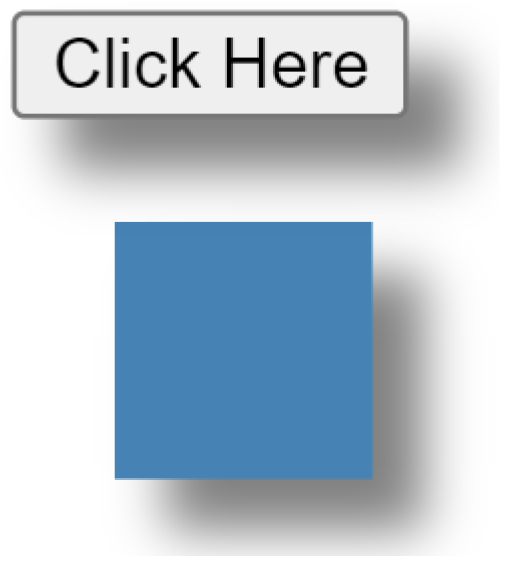


In css:

```
.box {
        background-color: steelblue;
        width: 50px;
        height: 50px;
        margin: 20px;
}
.shadow {
        box-shadow: 10px 0px 0px rgba(0,0,0,1);
}
```



When we change the box-shadow value to box-shadow: 10px 10px 5px rgba(0,0,0,1) and change the shadow color opacity in rgba(), then the output would be shown as follows



The fourth parameter controls the *blur-spread-radius* and depending on the value given to it, the radius of the blur depends.

https://css-tricks.com/pseudo-class-selectors/