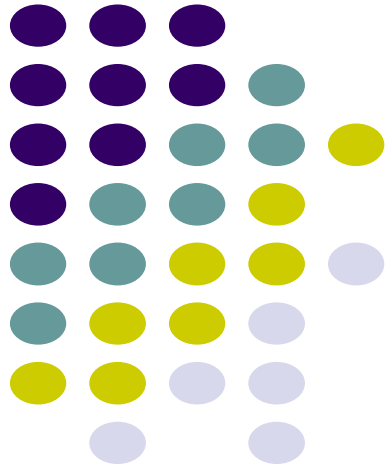
CSS Positioning

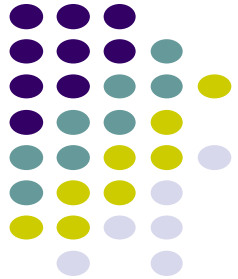
Dr. Arul Xavier V M
Assistant Professor



Normal Flow

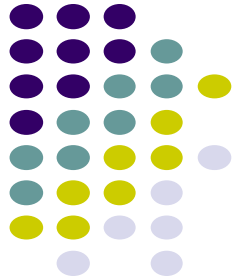
- Normal flow is the way that elements are displayed in a web page in most circumstances.
- All elements in HTML are inside boxes which are either inline elements or block level elements.
- The normal flow can be altered via.
  - CSS Position
  - CSS Floats

CSS Positioning



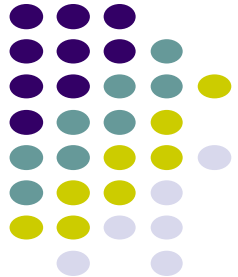
- CSS Layout - The position Property
  - The **position** property specifies the type of positioning method used for an element.
  - There are four different position values:
    - static
    - relative
    - absolute
    - fixed
    - sticky
  - Positioned elements can also utilized top, bottom, left, and right properties. However, these properties will not work unless the position property is set first.

CSS 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.

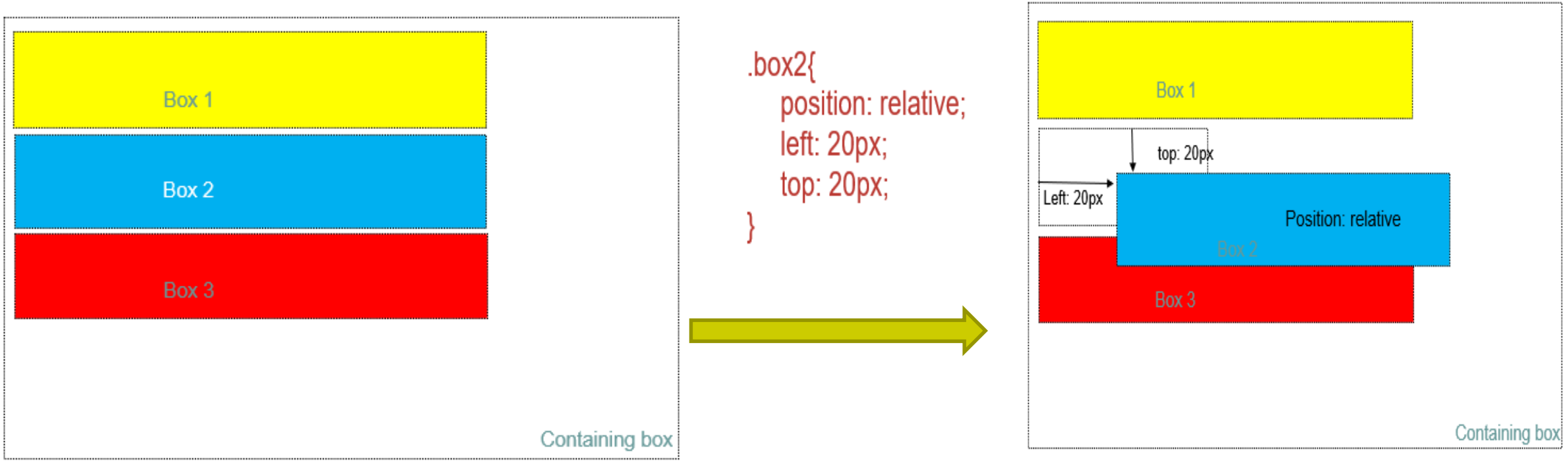
Relative Position



- A relatively positioned element will stay exactly where it is, in relation to the normal flow.
- You can then offset its position “relative” to its original position in the normal flow:

Relative Position

- In this example, **box 2** is offset 20px, top and left. The result is the box is offset 20px from its original position in the normal flow. Box 2 may overlap other boxes in the flow, but other boxes still recognize its original position in the flow.



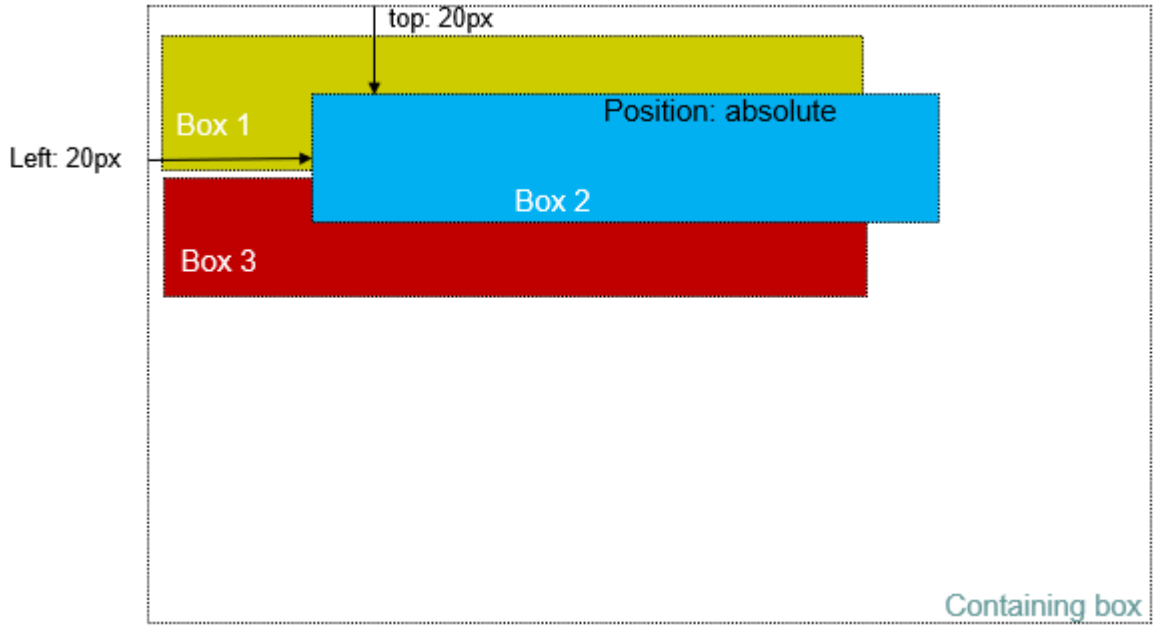
Example: relative Position

```css
.container {
    height: 400px;
    border: 1px solid black;
    padding: 5px;
}
.box1{
    height: 100px;
    background-color: blue;
}
.box2{
    height: 100px;
    background-color: red;
    position: relative;
    top:30px;
    left:100px;
}
.box3{
    height: 100px;
    background-color: green;
}
```

```html
<div class="container">
    <div class="box1"></div>
    <div class="box2"></div>
    <div class="box3"></div>
</div>
```

Absolute Position

- An absolutely positioned box is taken out of the normal flow, and positioned in relation to its nearest positioned ancestor (i.e. its containing box) but other boxes within the block (and still within the normal flow) act as if the box wasn't there.
- If there is no positioned(relative) ancestor box, it will be positioned in relation to the initial containing block, usually the browser window.



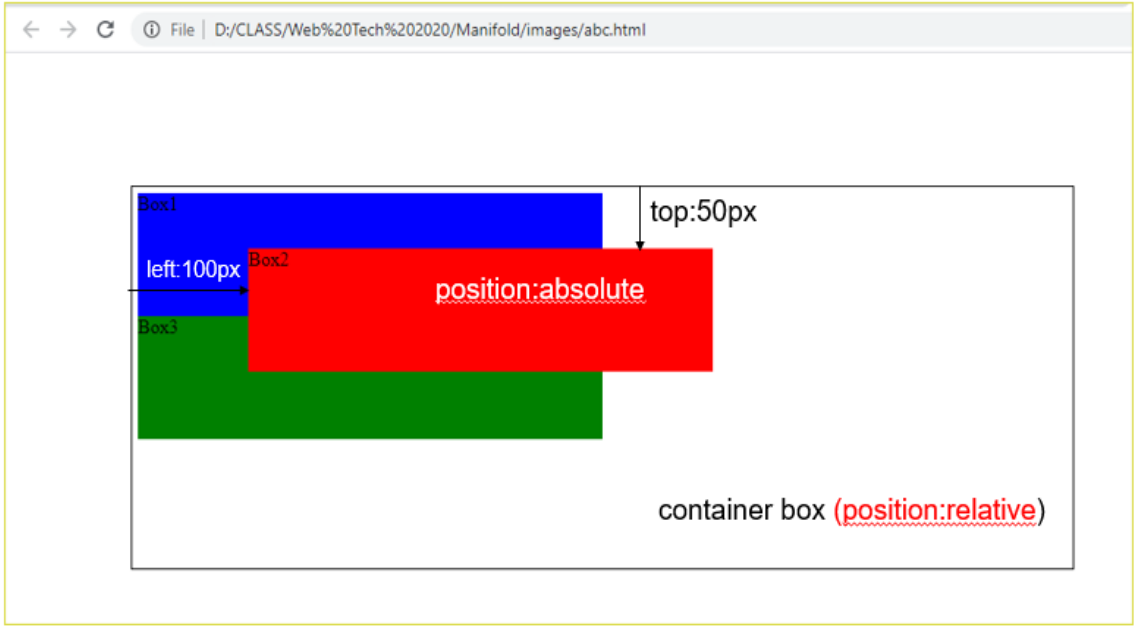
CSS Code:

```
.container {
    width:800px;
    height: 300px;
    border: 1px solid black;
    padding: 5px;
    position: relative;
    top:100px;
    left:100px;
}
.box1{
    height: 100px;
    width: 400px;
    background-color: blue;
}
.box2{
    height: 100px;
    width: 400px;
    background-color: red;
    position: absolute;
    top:50px;
    left:100px;
}
.box3{
    height: 100px;
    width: 400px;
    background-color: green;
}
```

```
<div class="container">
    <div class="box1"></div>
    <div class="box2"></div>
    <div class="box3"></div>
</div>
```

Absolute Position: Example

If the container box is positioned as relative, the absolute positioned elements move from its container box border.



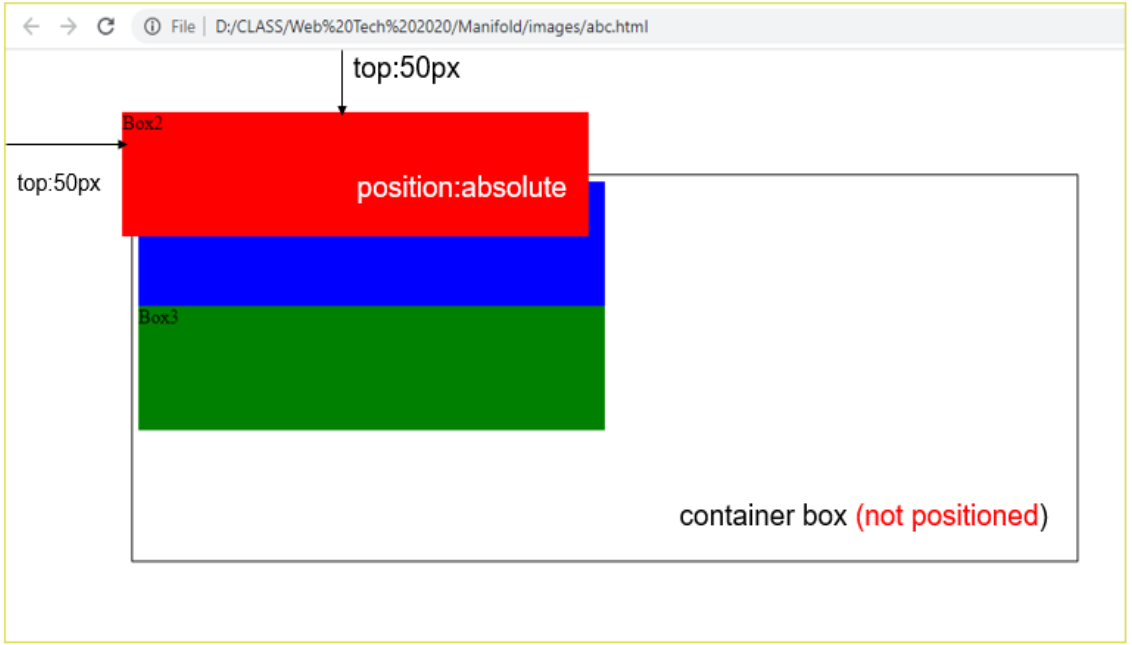
CSS Code:

```
.container {
    width:800px;
    height: 300px;
    border: 1px solid black;
    padding: 5px;
    margin-left: 100px;
    margin-top: 100px;
}
.box1{
    height: 100px;
    width: 400px;
    background-color: blue;
}
.box2{
    height: 100px;
    width: 400px;
    background-color: red;
    position: absolute;
    top:50px;
    left:100px;
}
.box3{
    height: 100px;
    width: 400px;
    background-color: green;
}
```

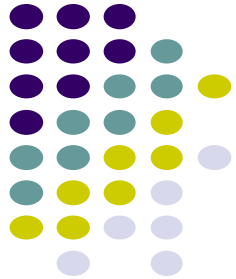
```
<div class="container">
    <div class="box1"></div>
    <div class="box2"></div>
    <div class="box3"></div>
</div>
```

Absolute Position: Example

If container box is not positioned as **relative**? it will be positioned in relation to the initial containing block, usually the browser window



CSS Positioning: Fixed Positioning



- Fixed Positioning is a sub-category of Absolute Positioning
- Allows the creation of floating elements that are always fixed in the same position in the browser window, while the rest of the content scrolls as normal.
  - Applications
    - Header
    - Navigation Bar
    - Footer

Example: Fixed Position

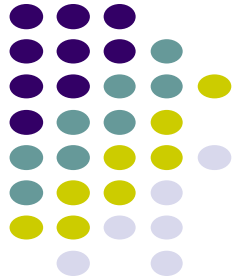
CSS Code:

```
* {
    margin: 0px;
  }
.header {
  position: fixed;
  top:0px;
  width: 100%;
  height: 50px;
  background-color: blue;
  color: white;
  font-size: 2em;
  text-align: center;
 }
body {
  background-color: pink;
 }
div {
  width:80%;
  height: 200px;
  background-color: yellow;
  color: blue;
  padding:20px;
  text-align: justify;
  margin: 10px auto;
}
```

```
<body>
    <h1 class="header">Welcome</h1>
    <div>The idea for the MITE took birth with the
    <div>The idea for the MITE took birth with the
    <div>The idea for the MITE took birth with the
    <div>The idea for the MITE took birth with the
    <div>The idea for the MITE took birth with the
</body>
```

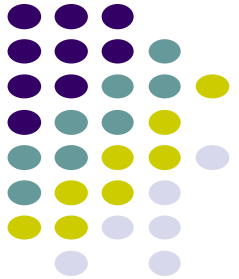
Here, the header div box showing “Welcome” as title is positioned as fixed at top:0px. Due to this, that box remains fixed in that position. This fixed box does not move even when you scroll the web page also.

Sticky Positioning



- 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).

Sticky position – Example



```css
* {
    margin: 0px;
  }
.header {
  position: sticky;
  top:0px;
  width: 100%;
  height: 50px;
  background-color: blue;
  color: white;
  font-size: 2em;
  text-align: center;
 }
body {
  background-color: pink;
 }
div {
  width:80%;
  height: 200px;
  background-color: yellow;
  color: blue;
  padding:20px;
  text-align: justify;
  margin: 10px auto;
}
```

HTML Code:

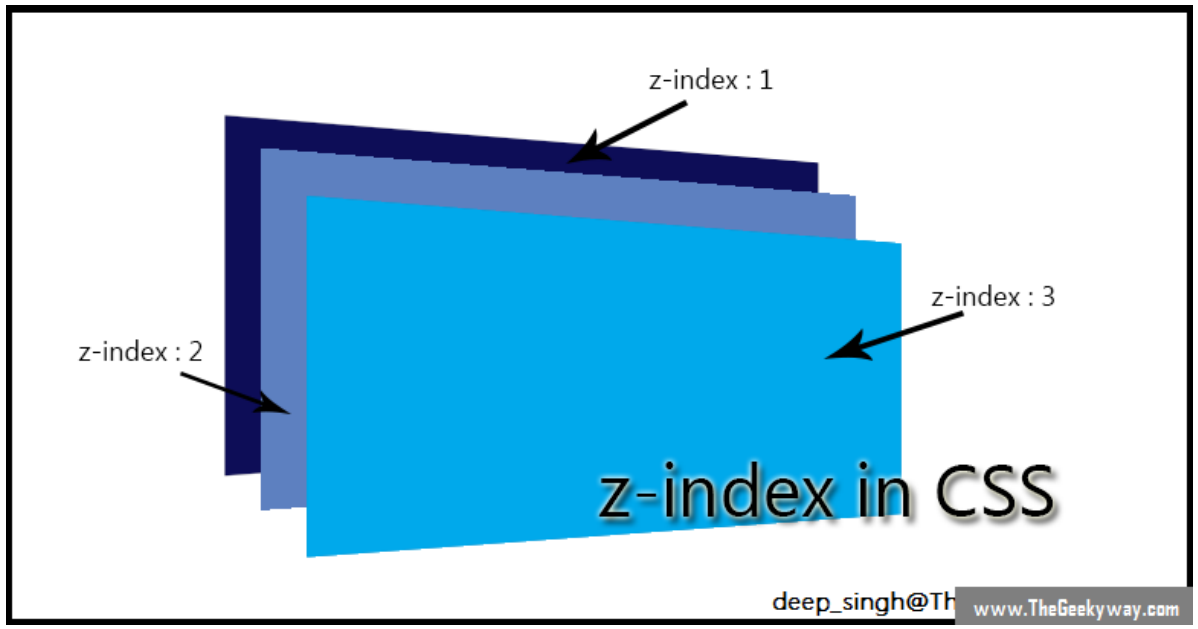
```html
<body>
    <p style="text-align: center;">
        <img src="Manifold-Logo.png">
    </p>
    <h1 class="header">Welcome</h1>
    <div>The idea for the MITE took birth with the aim of aiding budding en
    <div>The idea for the MITE took birth with the aim of aiding budding en
    <div>The idea for the MITE took birth with the aim of aiding budding en
    <div>The idea for the MITE took birth with the aim of aiding budding en
    <div>The idea for the MITE took birth with the aim of aiding budding en
</body>
```

Sticky position – Example

Here, the first output screenshot shows the page before scrolling the page where the Manifold logo visible before the header box. When you scroll the page, it is not visible and but the header box is still sticky there in top:0px, since it is positioned as “sticky” at the offset top:0px. When it reaches top:0px it sticky in that position

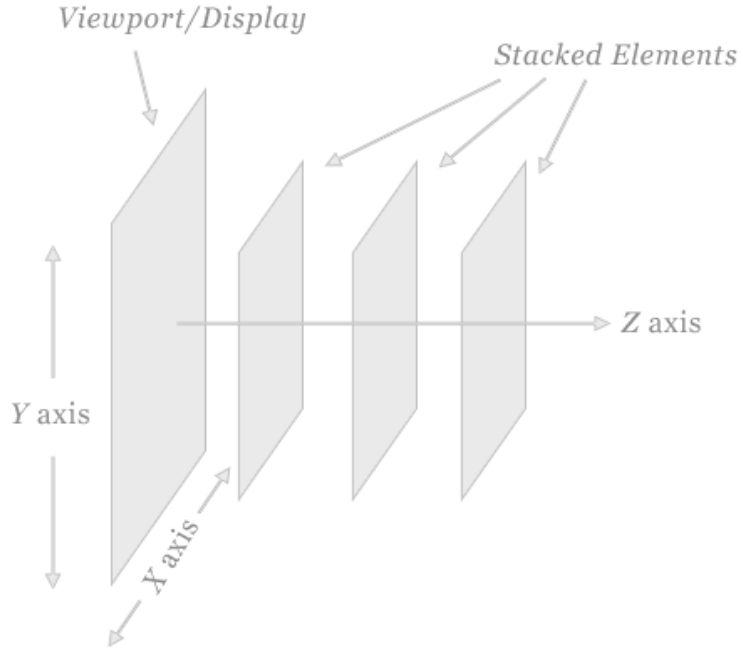
Overlapping Elements: Z-index

- When elements are positioned, they can overlap other elements.
- The z-index property specifies the stack order of an element (which element should be placed in front of, or behind, the others).
  - An element can have a positive or negative stack order:
  - An element with greater stack order is always in front of an element with a lower stack order.



Z-Index

- Modifies the stacking order of elements on a Web page.
- The default z-index value is “0".
- Elements with higher z-index values will appear stacked on top of elements with lower z-index values rendered on the same area of the page.



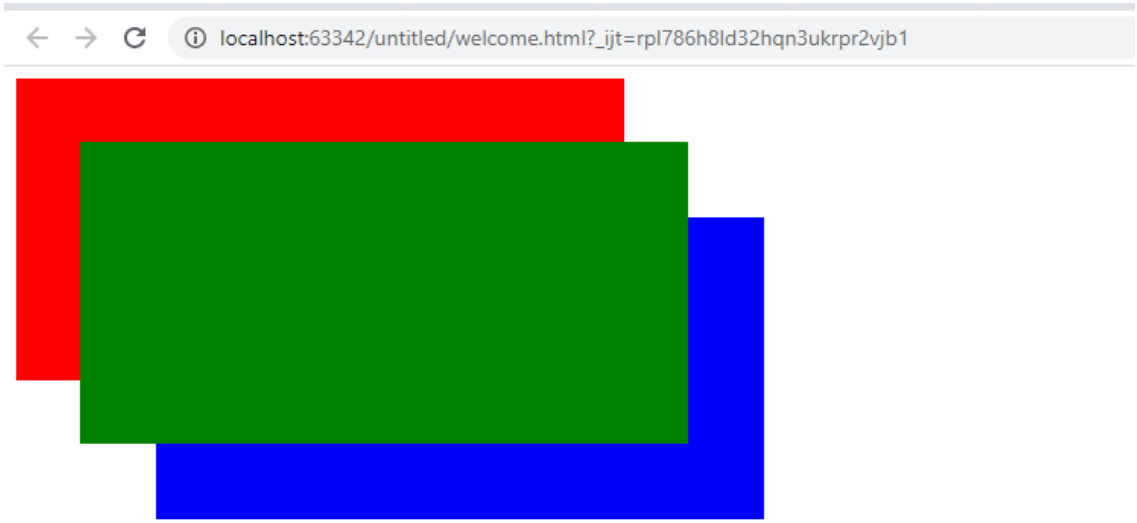
Z-Index Example

CSS Code

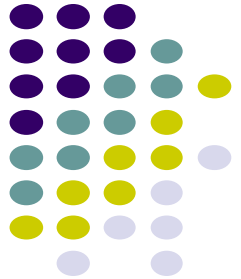
```
<style>
    div {
        height: 200px;
        width: 400px;
        font-size: 30px;
        color: white;
    }
    .b1{
        position: absolute;
        background-color: red;
        z-index: 1;
    }
    .b2{
        position: absolute;
        left:50px;
        top:50px;
        background-color: green;
        z-index: 10;
    }
    .b3{
        position: absolute;
        left:100px;
        top:100px;
        background-color: blue;
        z-index:3;
    }
</style>
```

HTML Code

```
<body>
  <div class="b1"></div>
  <div class="b2"></div>
  <div class="b3"></div>
</body>
```

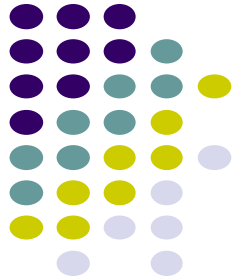


CSS Float



- The normal flow can be altered via - > CSS Floats
  - The CSS float property specifies how an element should float.
  - The CSS clear property specifies what elements can float beside the cleared element and on which side.

float property



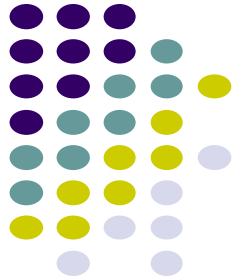
- The float property is used for positioning and formatting content or blocks.
- The float property can have one of the following values:
  - left - The element floats to the left of its container
  - right- The element floats to the right of its container
  - none - The element does not float (will be displayed just where it occurs in the text). This is default
- In its simplest use, the float property can be used to wrap text around images.

Example for float property

```
<style>
    div {
        width:200px;
        background-color: blue;
        height: 200px;
        color:white;
        margin: 5px;
    }
    .d1 {
        float: left;
    }
    .d2 {
        float: left;
    }
    .d3 {
        float:right;
    }
</style>
```

```
<body>
    <div class="d1">Box1</div>
    <div class="d2">Box2</div>
    <div class="d3">Box3</div>
</body>
```

clear property

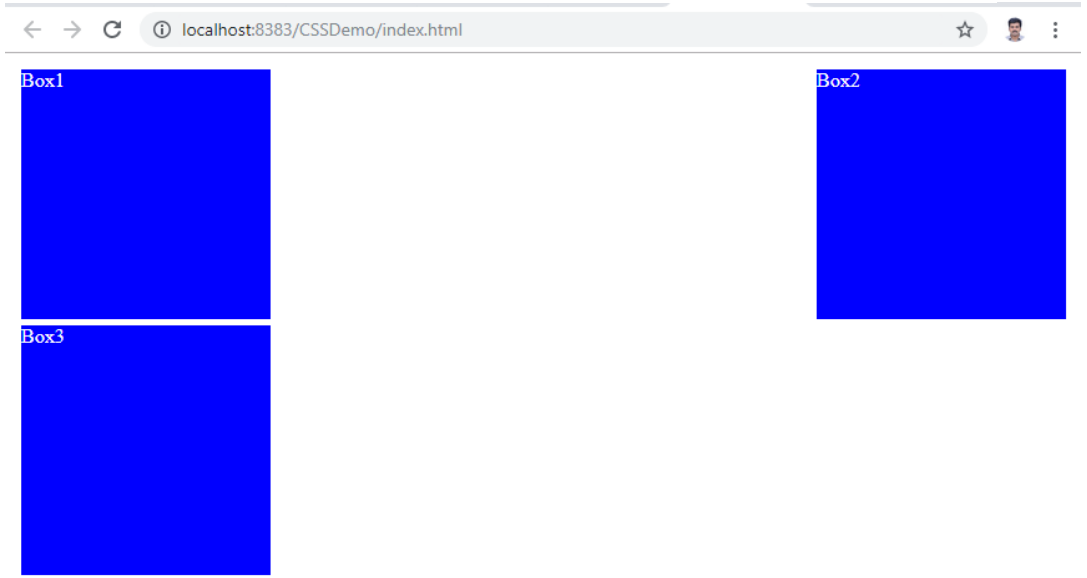


- The clear property specifies what elements can float beside the cleared element and on which side.
- The clear property can have one of the following values:
  - none - Allows floating elements on both sides. This is default
  - left - No floating elements allowed on the left side
  - right- No floating elements allowed on the right side
  - both - No floating elements allowed on either the left or the right side
- The most common way to use the clear property is after you have used a float property on an element.

Example for clear property

```
<style>
    div {
        width:200px;
        background-color: blue;
        height: 200px;
        color:white;
        margin: 5px;
    }
    .d1 {
        float: left;
    }
    .d2 {
        float: right;
    }
    .d3 {
        clear:right;
    }
</style>
```

```
<body>
    <div class="d1">Box1</div>
    <div class="d2">Box2</div>
    <div class="d3">Box3</div>
</body>
```



opacity property

- The opacity property specifies the opacity/transparency of an element.

Transparent Image

The `opacity` property can take a value from 0.0 - 1.0. The lower value, the more transparent:

opacity 0.2

opacity 0.5

opacity 1
(default)

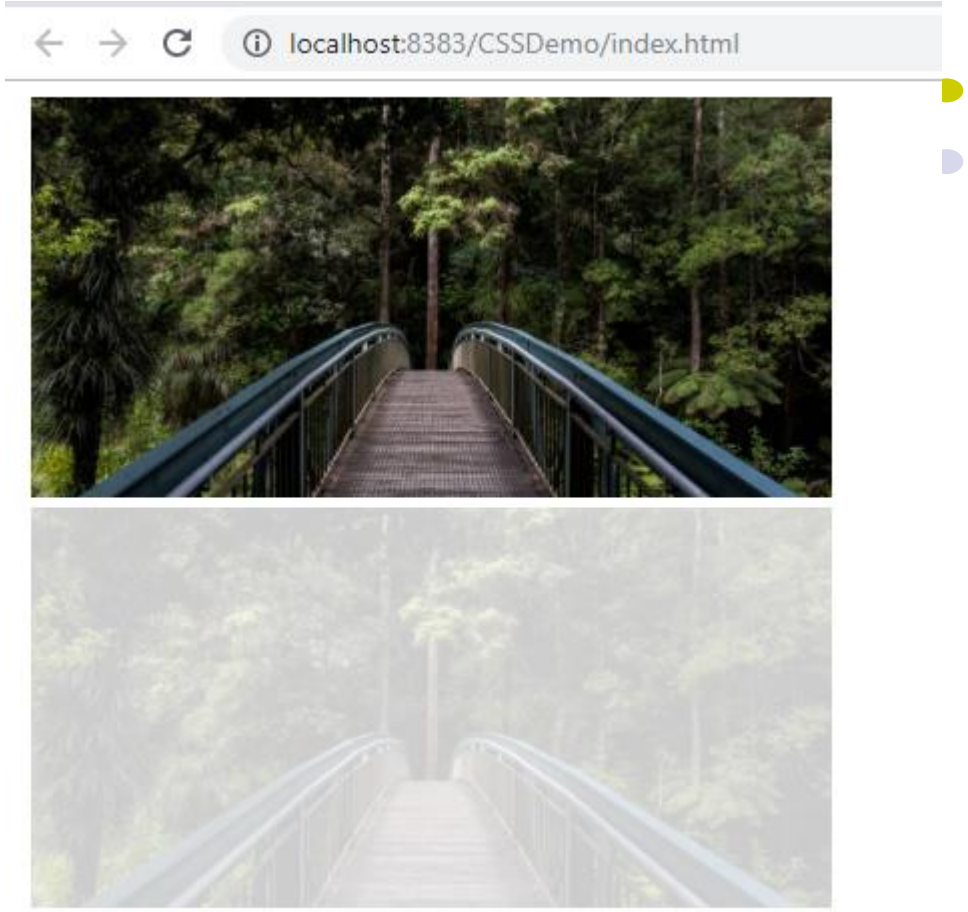
opacity property example

```css
.d1 {
    width:400px;
    height: 200px;
    background-image: url(forest.jpg);
    background-repeat: no-repeat;
    background-position: center;
    background-size: cover;
    color:white;
    margin: 5px;
    opacity: 1;
}
```

```css
.d2 {
    width:400px;
    height: 200px;
    background-image: url(forest.jpg);
    background-repeat: no-repeat;
    background-position: center;
    background-size: cover;
    color:white;
    margin: 5px;
    opacity: 0.2;
}
```

opacity:1

opacity:0.2



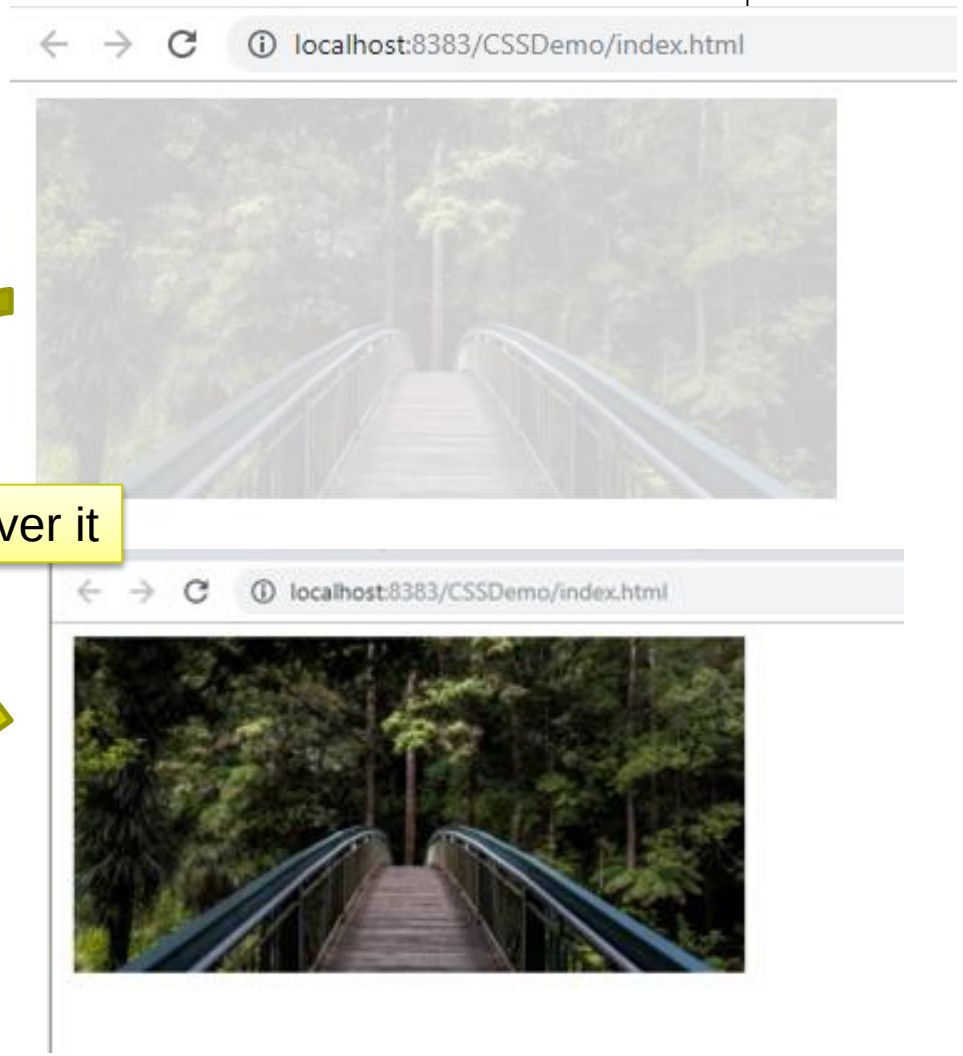

```html
<body>
    <div class="d1"> </div>
    <div class="d2"> </div>
</body>
```

hover effect

- Style an element when a user mouse over it.

```
<style>
    .d1 {
        width:400px;
        height: 200px;
        background-image: url(forest.jpg);
        background-repeat: no-repeat;
        background-position: center;
        background-size: cover;
        color:white;
        margin: 5px;
        opacity: 0.2;
    }
    .d1:hover {
        opacity: 1;
    }
</style>
<body>
    <div class="d1"> </div>
</body>
```

On mouse over it



Box-shadow

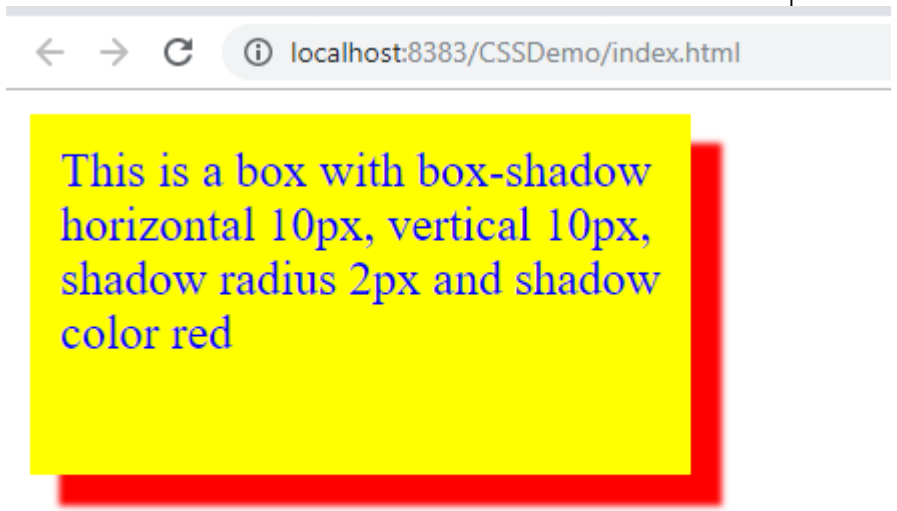
- CSS box-shadow Property
  - The CSS box-shadow property applies shadow to elements.

This is a yellow element with a black box-shadow

This is a yellow element with a grey box-shadow

Box-shadow

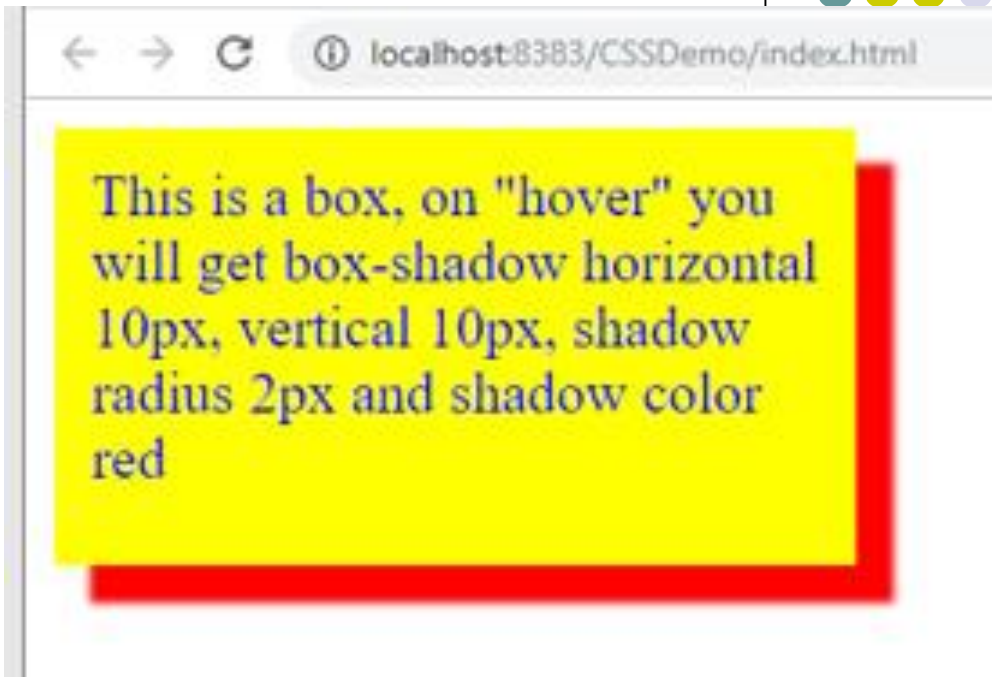
```
<style>
    div {
        background-color: yellow;
        color:blue;
        padding: 10px;
        width: 200px;
        height: 100px;
        box-shadow: 10px 10px 2px red;
    }
</style>
```




```
<body>
    <div>
        This is a box with box-shadow horizontal 5px,
        vertical 5px, shadow radius 2px and shadow color red
    </div>
</body>
```

Box-shadow on hover effect

```
<style>
    div {
        background-color: yellow;
        color:blue;
        padding: 10px;
        width: 200px;
        height: 100px;
    }
    div:hover {
         box-shadow: 10px 10px 2px red;
    }
</style>
```




```
<body>
    <div>
        This is a box, on "hover" you will get box-shadow horizontal 10px,
        vertical 10px, shadow radius 2px and shadow color red
    </div>
</body>
```