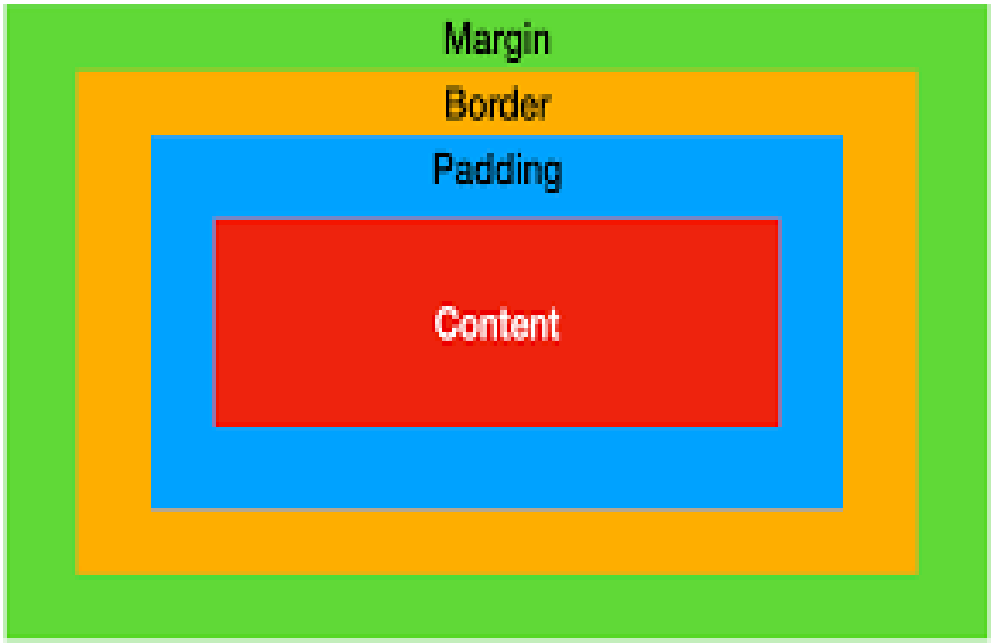
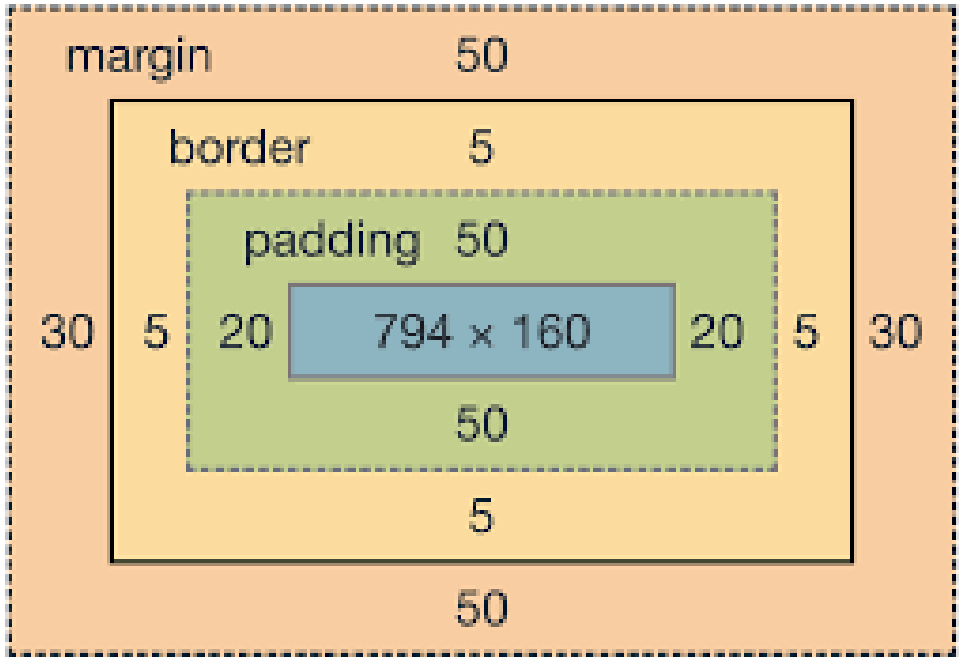
CSS Box Model

- In CSS, the term "**box model**" is used when talking about **design and layout**.

- The CSS box model is **essentially a box** that wraps around every HTML element.
- It consists of: **margins, borders, padding, and the actual content**. The image below illustrates the box model:



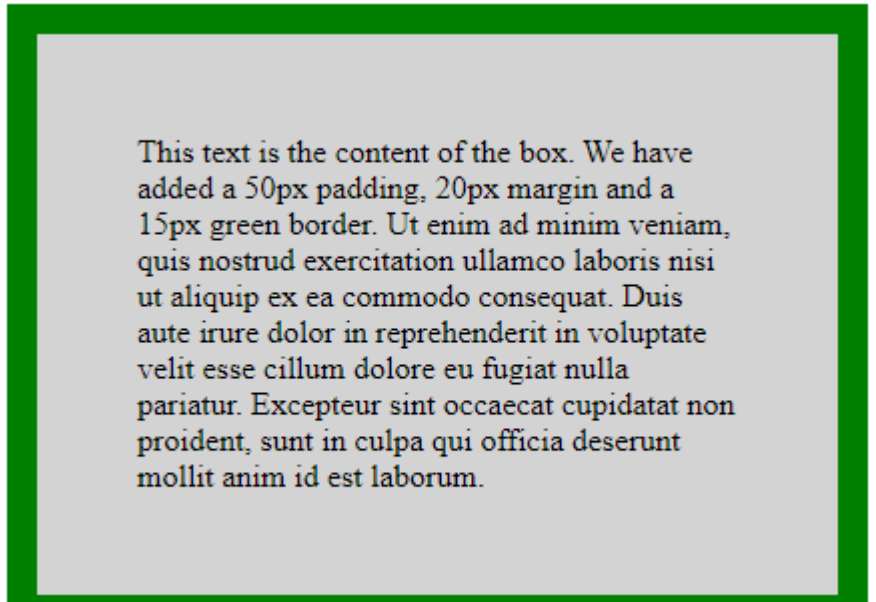


- **Content** - The **content of the box**, where text and images appear
- **Padding** - Clears an area around the content. The padding is transparent
- **Border** - A border that goes around the padding and content
- **Margin** - **Clears an area outside the border**. The margin is transparent

Demonstration of the box model

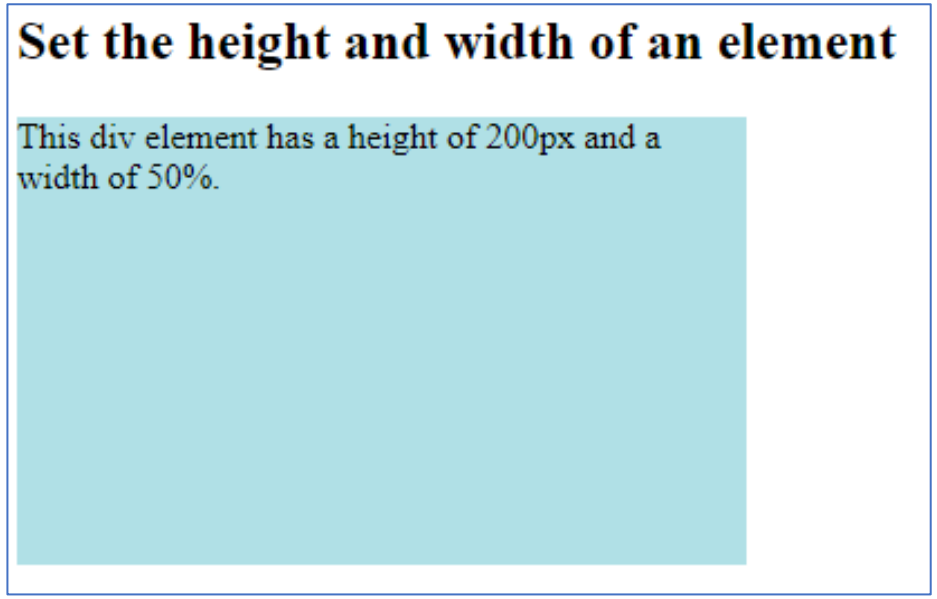
```
<!- box model -->
<html>
<head>
<style>
div
{
  background-color: lightgrey;
  width: 300px;
  border: 15px solid green;
  padding: 50px;
  margin: 20px;
}
</style>
</head>
<body>
```

```
<h2>Demonstrating the Box Model</h2>

<div>
This text is the content of the box. We have added a 50px padding, 20px margin and a 15px green border.
This text is the content of the box. We have added a 50px padding, 20px margin and a 15px green border.

This text is the content of the box. We have added a 50px padding, 20px margin and a 15px green border.

</div>

</body>
</html>
```



CSS Height, Width and Max-width

- The CSS **height and width** properties are used to set the **height and width of an element.**
- The CSS max-width property is used to **set the maximum width** of an element.
- This element has a height of 50 pixels and a width of 100%.




```
div {
  height: 200px;
  width: 50%;
  background-color: aqua;
  max-width: fit-content;
}
<body>
<h2>Set the height and width of an element</h2>
<div>This div element has a height of 200px and a width of 50%.</div>
</body>
```

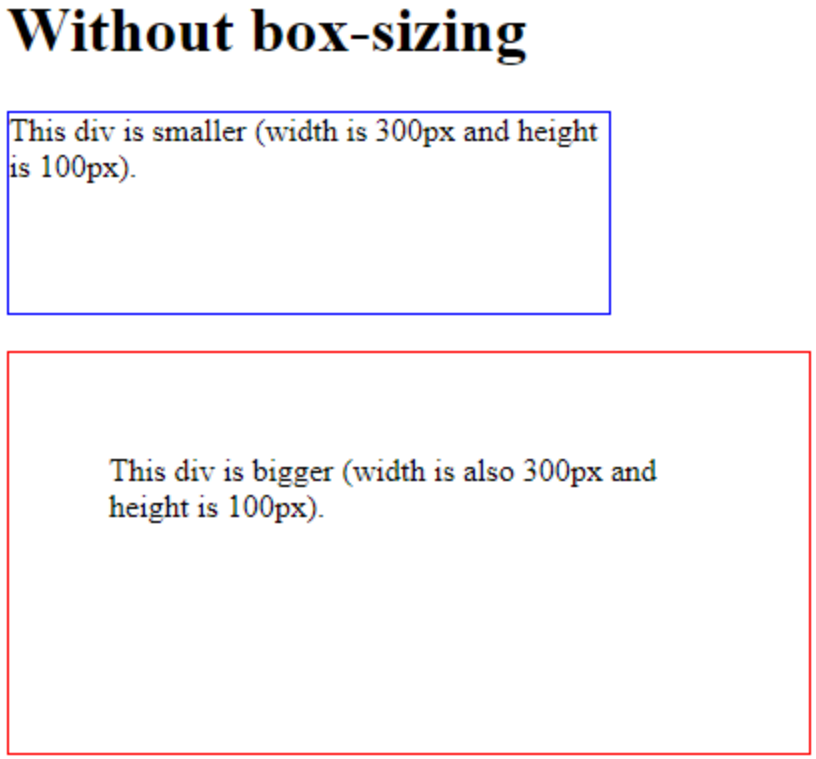
CSS Box Sizing

- The CSS **box-sizing property** allows us to include the **padding and border** in an element's **total width and height.**

Without the CSS box-sizing Property:

- By default, the width and height of an element is calculated like this:
  - **width + padding + border** = **actual width** of an element
  - **height + padding + border** = **actual height** of an element

```
.div1 {
  width: 300px;
  height: 100px;
  border: 1px solid blue;
}

.div2 {
  width: 300px;
  height: 100px;
  padding: 50px;
  border: 1px solid red;
}
```

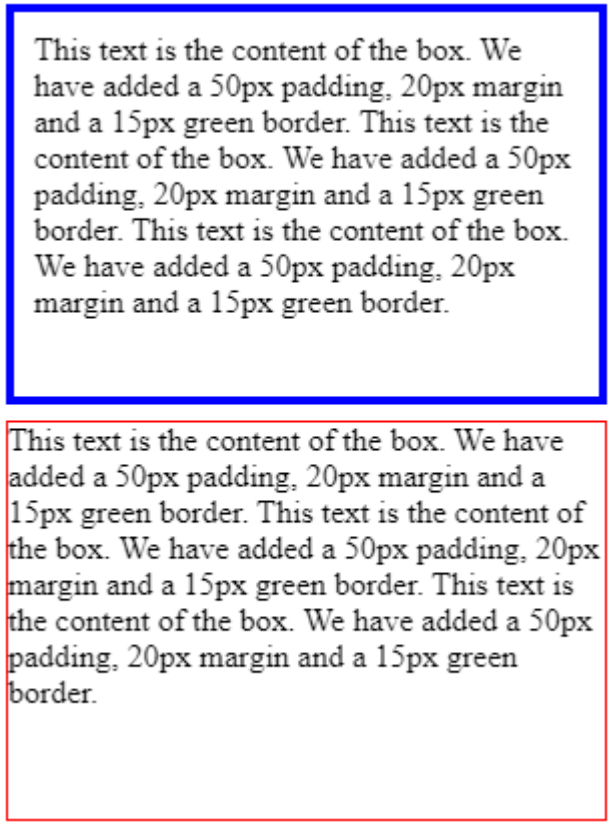


With the CSS box-sizing Property

- The box-sizing property allows us to include the **padding and border** in an element's total width and height.

- If you set **box-sizing: border-box**; on an element, **padding and border are included in the width and height:**

```
.div1 {
  width: 300px;
  height: 200px;
  border: 1px solid blue;
  padding: 10px;
  border-width: 3pt;
  box-sizing: border-box;
}

.div2 {
  width: 300px;
  height: 200px;
  margin-top: inherit;
  border: 1px solid red;
  box-sizing: border-box;
}
```



Border Radius

- The CSS box model includes the **border radius** property, which allows you to **specify the curvature of the corners of an element's border.**

- Specify the value for this property in several ways, including:
  - A **single value → sets the same radius for all four corners of the border.**
  - **Two values →** set different radii for the **top and bottom** corners and the **left and right** corners.
  - **Four values →** set **different radii for each individual corner** of the border.

SINGLE VALUE

For example, to set a border radius of **10 pixels for all corners** of an element's border, you would use the following CSS code:

```
border-radius: 10px;
```

TWO VALUES

To set different radii for the **top and bottom corners**, and the **left and right corners**, you would use **two values** like this:

```
border-radius: 10px 20px;
```

FOUR VALUES

This would set a radius of 10 pixels for the **top and bottom** corners, and a radius of 20 pixels for the **left and right corners.**

```
border-radius: 10px 20px 30px 40px;
```

```
<!- box model with border radius-->
<html>
<head>
    <title>Border Radius Example</title>
    <style>
        .box {
            background-color: lightblue;
            padding: 20px;
            border-radius: 10px;
        }
    </style>
</head>
<body>
    <div class="box">
        <p>This is an example of an element
with rounded corners.</p>
    </div>
</body>
</html>
```

This is an example of an element with rounded corners.

```
.box1 {
background-color: lightblue;
padding: 20px;
border-radius: 10px 30px;
/*top=10 bottom=30 left=10 right=30*/
}
```

top=10 bottom=30 left=10 right=30

```
.box3 {
background-color: lightblue;
padding: 20px;
border-radius: 10px 20px 40px 30px;
}
```

top-left=10 top-right=20 bottom-right=40 bottom-left=30

Padding

- In CSS, the **padding** property is used to **add space between an element's content and its border**.
- The padding property can be **applied to any HTML element, including text, images, and containers.**

The padding property can be set in several ways:

- **padding: value;** sets the **same padding value for all four sides** of the element.

Ex.
padding: 10px;
→ add **10** pixels of padding to the **top, bottom, left, and right** sides of the element.

- **padding: vertical horizontal;** sets the **padding value for the top and bottom, and left and right sides** of the element separately.

Ex.
padding: 10px 20px
→ add 10 pixels of padding to the **top and bottom**,
20 pixels of padding to the **left and right**.

- **padding: top right bottom left;** sets the padding value for each side of the element separately.

Ex.
padding: 10px 20px 30px 40px
→add 10 pixels of padding to the **top**,
20 pixels of padding to the **right**,
30 pixels of padding to the **bottom**
40 pixels of padding to the **left**

```
<!-- box model with padding-->
<html>
<head>
<style>
.box3
{
    background-color: lightblue;
    padding: 20px;
    border-radius: 10px;
}
</style>
</head>
<body>
<div class="box3">
 padding 20px for all sides
  i.e. top, bottom, left and right
</div>
</body>
</html>
```

padding 20px for all sides i.e. top, bottom, left and right

```
.box3
{
    background-color: lightblue;
    padding: 10px 20px;
    border-radius: 10px;
}
```

padding 10 pixels for top and botton
padding 20 pixels for left and right

```
.box3
{   background-color: lightblue;
    padding: 10px 20px 30px 40px;
    border-radius: 10px;
}
```

10 pixels of padding to the top
20 pixels of padding to the right
30 pixels of padding to the bottom
40 pixels of padding to the left

Margins

- In CSS, the **margin** property is used to **add space between an element's border and the adjacent elements**
- The margin property can be applied to any HTML element, including text, images, and containers.

The **margin** property can be set in several ways:

- **margin: value; sets the same margin value for all four sides of the element**

For example,
margin: 10px;
→ add 10 pixels of margin to the top, bottom, left, and right sides of the element.

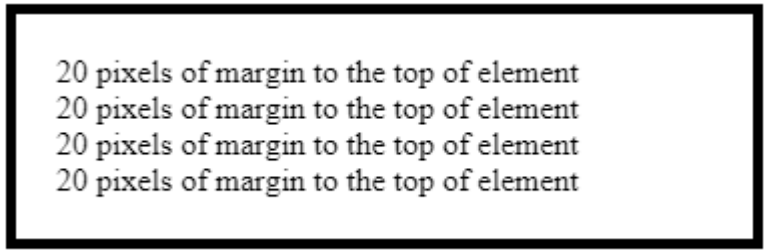
- **margin: vertical horizontal; sets the margin value for the top and bottom, and left and right sides of the element separately..**

For example,
margin: 10px 20px;
→ add 10 pixels to the top and bottom,
20 pixels to the left and right.

- **margin: top right bottom left; sets the margin value for each side of the element separately.**

For example,
margin: 10px 20px 30px 40px;
→ add 10 pixels of margin to the top,
→ 20 pixels of margin to the right,
→ 30 pixels of margin to the bottom, and
→ 40 pixels of margin to the left.

```
<!-- box model with margin->
<html>
<head>
<style>
.box {
    border: 5px solid black;
    padding: 20px;
    margin: 20px;
}
</style>
</head>
<body>
<div class="box">
     20 pixels of margin to the top, bottom,
left  and righ of element<br>

</div>
</body>
</html>
```



auto margin

- The **auto value** for the margin property can be used to horizontally center an element within its parent container.

- Here's an example of how to use **margin: auto** to center an element horizontally:

- In this example, we have defined a element with a class of "box".

margin: auto;

```
.box {
  background-color: lightblue;
  padding: 20px;
  margin: auto;
  width: 50%;
  }
```

```
<!– auto margin demo->
<html>
<head>
        <title>Auto Margin Example</title>
        <style>
        </style>
</head>
<body>
        <div class="box">
                <p>This is an example of an
element centered using auto margins.</p>
        </div>
</body>
</html>
```

Margin collapse

- In CSS, margin collapse is a behavior where **adjacent margins between two elements collapse into a single margin.**
- This can affect the layout of elements in unexpected ways
- Ex. Here two ** **elements with different background colors, padding, and margins.

- The first has a margin-bottom of 20 pixels,
- second element has a margin-top of 30 pixels.


```
<html>
<head> <style>
        .box1 {
                background-color: lightblue;
                margin-bottom: 20px;
                padding: 20px;
                }
        .box2 {
                background-color: pink;
                margin-top: 30px;
                padding: 20px;
                }
        </style>
</head>
<body>
<div class="box1">
        <p>This is the first box.</p>
</div>
<div class="box2">
        <p>This is the second box.</p>
        </div> </body>
</html>
```

Margin collapse

This is the first box.

This is the second box.

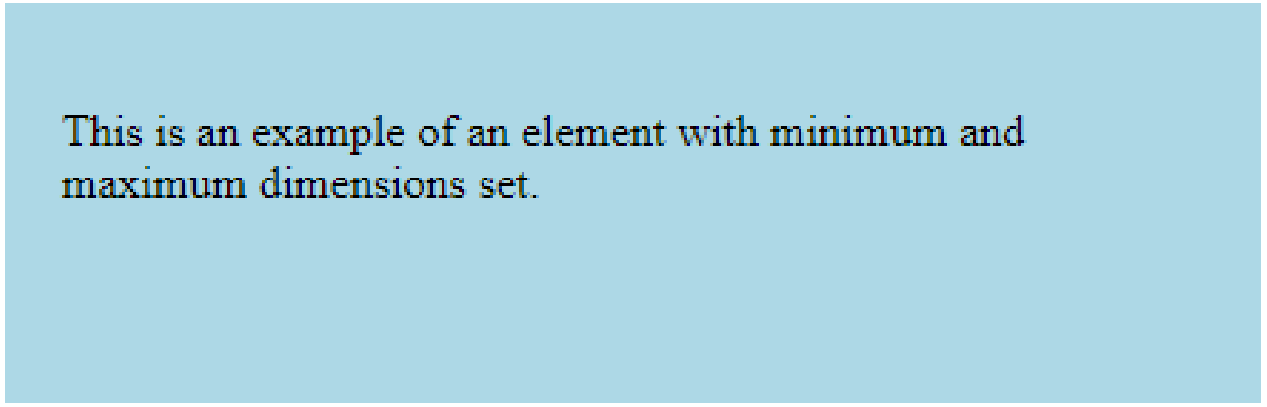
- Here the margin between the two boxes is **not the sum (20px + 30px = 50px),** but instead is **the max value** of the two margins (30px).
- This is because the **margins have collapsed** into a single margin between the two boxes.

- Margin collapse can occur in other situations as well, such as between siblings, or between a parent and its first or last child element.
- To prevent margin collapse, you can use techniques such as **setting a padding value on the parent element**, or **using a border or outline property.**

```
<html>
<head> <style>
        .box1 {
                background-color: lightblue;
                margin-bottom: 20px;
                padding: 20px;
                }
        .box2 {
                background-color: pink;
                margin-top: 30px;
                padding: 20px;
                }
        </style>
</head>
<body>
<div class="box1">
        <p>This is the first box.</p>
</div>
<div class="box2">
        <p>This is the second box.</p>
        </div> </body>
</html>
```

Minimum and Maximum Height and Width

- In CSS, the min-height, max-height, min-width, and max-width properties can be used to set minimum and maximum dimensions for an element in the box model.
- We have set the **min-height** property to 100px and the **max-height** property to 300px.
- We have also set the **min-width** property to 200px and the **max-width** property to 400px.



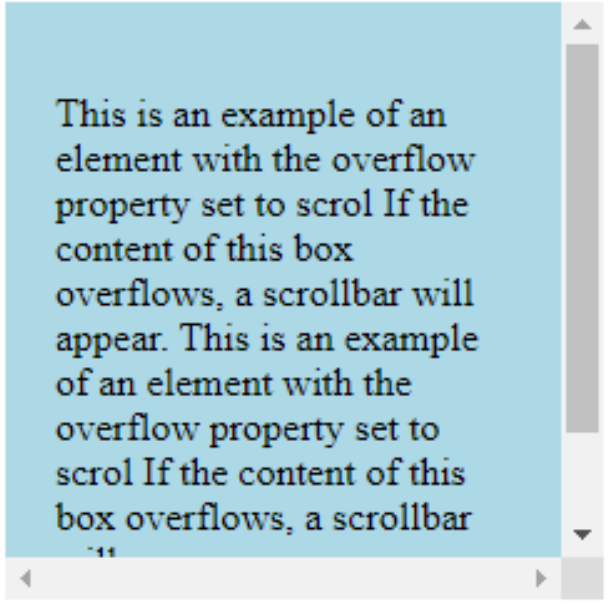

```
<html>
<head>
    <title>Minimum and Maximum Dimensions Example</title>
    <style>
    .box {
        background-color: lightblue;
        padding: 20px;
        min-height: 100px;
        max-height: 300px;
        min-width: 200px;
        max-width: 400px;
        }
    </style>
</head>
<body>
<div class="box">
    <p>This is an example of an element with minimum and maximum dimensions set.</p>
</div> </body>
</html>
```

Overflow

- In CSS, the **overflow** property is used to **control what happens when the content of an element overflows** its allocated space within the box model.

Here are some possible values for the overflow property:

- **scroll:** Content that overflows is clipped, and a **scrollbar is added** to allow scrolling to see the rest of the content.




```
<!DOCTYPE html>
<html>
<head>
        <title>Overflow Example</title>
        <style>
                .box {
                background-color: lightblue;
                padding: 20px;
                width: 200px;
                height: 200px;
                overflow: scroll;
                }
        </style>
</head>
<body>  <div class="box">
                <p>This is an example of an
element with the overflow property set to scroll. If the
content of this box overflows, a scrollbar will
appear.</p>
        </div>
</body>
</html>
```

Overflow

- **visible:** This is the default value. Content overflows outside the box model and can overlap other elements.

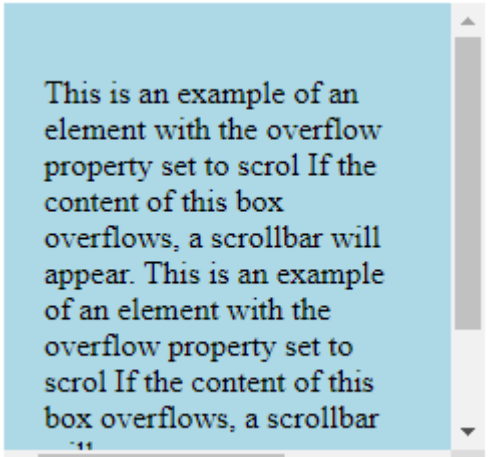
overflow:visible:

This is an example of an element with the overflow property set to scrol If the content of this box overflows, a scrollbar will appear. This is an example of an element with the overflow property set to scrol If the content of this box overflows, a scrollbar will appear dffdfdffdfdfdfdfdfdfdfdfdfdfdfdfdfdfdfdffdffdffdff

- hidden: Content that overflows is clipped and not visible outside the box model.
- **overflow: hidden;**

This is an example of an element with the overflow property set to scrol If the content of this box overflows, a scrollbar will appear. This is an example of an element with the overflow property set to scrol If the content of this box overflows, a scrollbar will appear dffdfdffdfdfdfdfdfdfdfdfdfdfdfd:

- auto: Content that overflows is clipped and a scrollbar is added only if necessary.
- **overflow: auto;**



Resetting Defaults

```
/* CSS Reset */
html, body, div, span, applet, object, iframe,
h1, h2, h3, h4, h5, h6, p, blockquote, pre,
a, abbr, acronym, address, big, cite, code,
del, dfn, em, img, ins, kbd, q, s, samp,
small, strike, strong, sub, sup, tt, var,
b, u, i, center,
dl, dt, dd, ol, ul, li,
fieldset, form, label, legend,
table, caption, tbody, tfoot, thead, tr, th, td {
    margin: 0;
    padding: 0;
    border: 0;
    font-size: 100%;
    font: inherit;
    vertical-align: baseline;
}
```

- Resetting defaults in CSS can be achieved by using a **CSS reset stylesheet**.
- A CSS reset stylesheet is a set of CSS rules that are designed to **reset or override the default styles** applied by browsers to HTML elements.
- Here's an example of a simple CSS reset stylesheet:

```
/* HTML5 display-role reset for older browsers */
article, aside, details, figcaption, figure,
footer, header, hgroup, menu, nav, section {
    display: block;
}
body {   /* Body styles */
    line-height: 1;
}
a {/* Links */
    color: #000;
    text-decoration: none;
}
a:hover {
    text-decoration: underline;
}
```

Visibility

- The **visibility** property in CSS is used to **control the visibility of an element**.

The visibility property can have the following values:

- **visible:** The element is visible.
- **hidden**: The element is hidden, but still takes up space.
- **collapse:** Only for table elements. The **row or column is removed from the table layout** and the **space is made available for other content.**

```
<!DOCTYPE html>
<html>
<head>
        <title>Visibility Example</title>
        <style>
            .box {
                background-color: lightblue;
                padding: 20px;
                width: 200px;
                height: 200px;
                visibility: hidden;
                }
        </style>
</head>
<body>
        <div class="box">
                <p>This is an example of an
element with the visibility property set to hidden. The
element is hidden, but still takes up space.</p>
        </div>
</body>
</html>
```

Visibility

- In this example we set the visibility property to hidden.

- When you view this code in a web browser, you will **see a box with light blue background color and 20 pixels of padding,** but there will be **no content visible inside the box**.
- This is because the visibility property has been set to hidden, which hides the content of the element but still takes up space.

```
<!DOCTYPE html>
<html>
<head>
        <title>Visibility Example</title>
        <style>
            .box {
                    background-color: lightblue;
                    padding: 20px;
                    width: 200px;
                    height: 200px;
                    visibility: hidden;
                    }
        </style>
</head>
<body>
        <div class="box">
                    <p>This is an example of an
element with the visibility property set to hidden. The
element is hidden, but still takes up space.</p>
        </div>
</body>
</html>
```