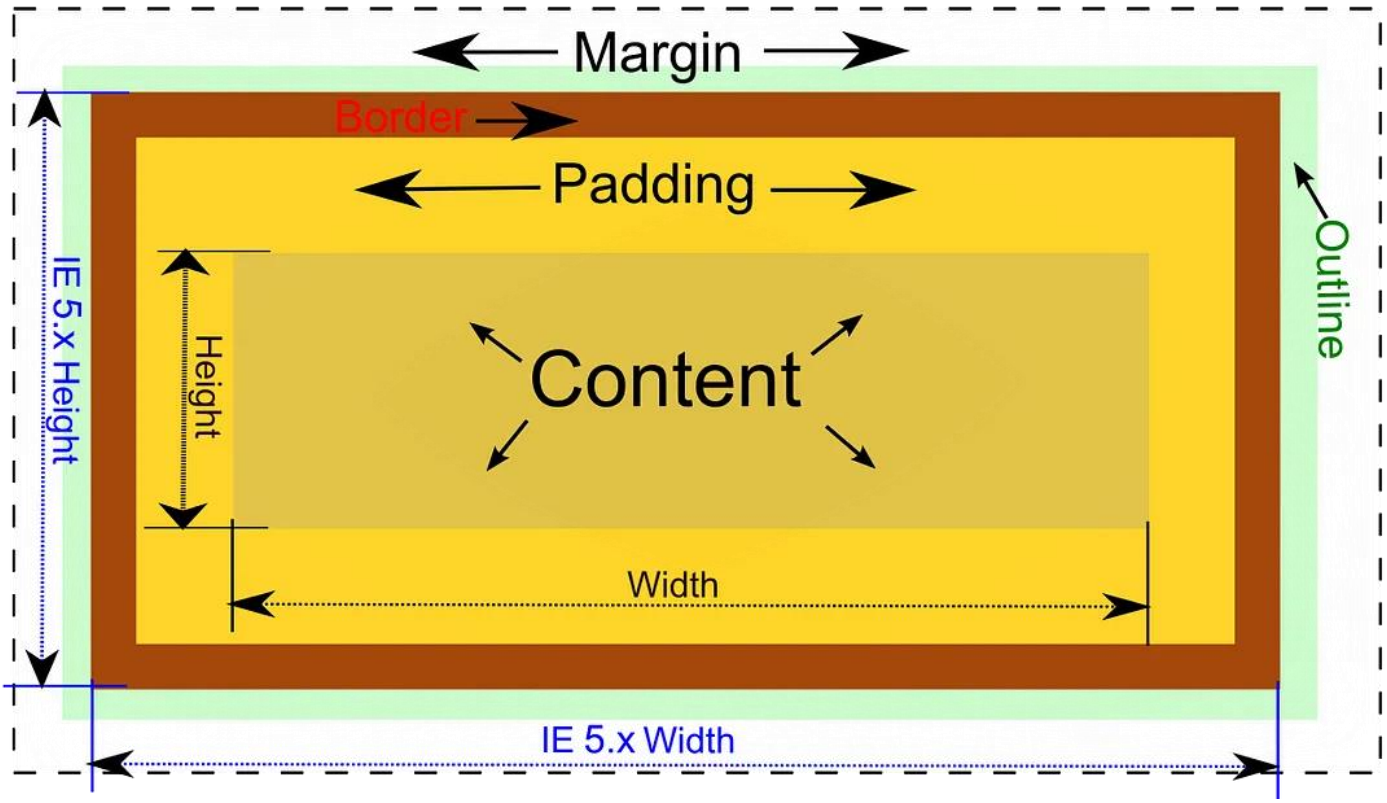
## CSS MARGINS, PADDINGS, WIDTH AND HEIGHT, BORDER

## REFERENCE IMAGE FOR BELOW TOPICS




## MARGINS

- Margin Property is used to create space around element's border.
- It can be set for all sides or individually for each side (top, right, bottom, left).

## Different Properties of Margins:

**1.margin-top: 10px;**

**2.margin-right: 20px;**

**3.margin-bottom: 15px;**

**4.margin-left: 5px;**

**5.margin: 10px; /* All margins are 10px */**

**6.margin: 10px 20px; /* Top/bottom: 10px, Right/left: 20px */**

**7.margin: 10px 20px 15px; /* Top: 10px, Right/left: 20px, Bottom: 15px */**

**8.margin: 10px 20px 15px 5px; /* Top, Right, Bottom, Left */**

**Note: Pixels Can Be Changed and Negative Pixels Are Also Allowed**

- **We Can Set Margins in different ways:**

1. **Setting Margin for All Sides:**

```
.element {
   margin: 10px; /* Applies 10 pixels margin to all sides */
}
```

2. **Setting Margin for Individual Sides:**

```
.element {
margin-top: 10px;
margin-right: 20px;   /*Applies different pixels for each side */
margin-bottom: 15px;
margin-left: 5px;
}
```

3. **Shorthand for Margin:**

```
.element {
   margin: 10px 20px 15px 5px; /* top right bottom left */
}
```

**NOTE:**

- **If you provide two values, they apply to top/bottom and right/left respectively.**
- **If you provide three values, they apply to top, right/left, and bottom respectively**

4. **Negative Margin**

```
.element {
   margin-left: -10px; /* Moves the element 10 pixels to the left */
}
```

5. **Auto Margin**

```
.element {
    margin: auto; /* Centre's the element horizontally */
}
```

## PADDINGS

- Padding Property is used to create space inside an element, between the elements content and its border.
- Padding can be set for all sides or individually for each side (top, right, bottom, left)

## Different Properties of Padding

1. **padding-top: 10px;**
2. **padding-right: 20px;**
3. **padding-bottom: 15px;**
4. **padding-left: 5px;**
5. **padding: 10px; /* All paddings are 10px */**
6. **padding: 10px 20px; /* Top/bottom: 10px, Right/left: 20px */**
7. **padding: 10px 20px 15px; /* Top: 10px, Right/left: 20px, Bottom: 15px */**
8. **padding: 10px 20px 15px 5px; /* Top, Right, Bottom, Left */**

**Note:** Pixels Can Be Changed **and** Negative Pixels Are Not Allowed

- **We Can Set Paddings in different ways:**

1. **Setting Padding for All Sides:**

```
.element {
    Padding : 10px; /* Applies 10 pixels padding to all sides */
}
```

2. **Setting Padding for Individual Sides:**

```
.element {
    padding-top: 10px;
    padding-right: 20px;
```

```
    padding-bottom: 15px;
    padding-left: 5px;
}
```

3. **Shorthand for Padding:**

```
.element {
    padding: 10px 20px 15px 5px; /* top right bottom left */
}
```

- **If you provide two values, they apply to top/bottom and right/left respectively.**
- **If you provide three values, they apply to top, right/left, and bottom respectively.**

## PADDING AND BACKGROUND COLOR:

- Padding affects the background color of an element. If you set a background color on an element, the padding area will also be filled with that background color.

## WIDTH AND HEIGHT

### Width Properties:

• **width:** Sets the width of an element. You can use values like pixels (px), percentages (%), viewport width (vw), and more.

• **min-width:** Specifies the minimum width that an element can have. Useful for responsive design to prevent elements from becoming too narrow.

• **max-width:** Specifies the maximum width that an element can have. Useful for responsive design to limit how wide an element can become.

### Height Properties

- **height:** Sets the height of an element. Similar to width, you can use values like pixels (px), percentages (%), viewport height (vh), and more.
- **min-height:** Specifies the minimum height that an element can have. Useful for responsive design to prevent elements from becoming too short.

- **max-height:** Specifies the maximum height that an element can have. Useful for responsive design to limit how tall an element can become.

## 1.Setting Width and Height:

**To set the width of an element:**

```
.element {
   width: 200px; /* Sets the width to 200 pixels */
}
```

**To set the height of an element:**

```
.element {
   height: 100px; /* Sets the height to 100 pixels */ }
```

## 2. Percentage Width and Height:

```
.element {
    width: 50%; /* Sets the width to 50% of its containing element */
    height: 75%; /* Sets the height to 75% of its containing element */
    }
```

## 3. Minimum and Maximum Width and Height:

You can specify minimum and maximum values for width and height using min-width, max-width, min-height, and max-height properties. This is useful for creating responsive layouts or preventing elements from becoming too small or too large:

```
.element {
   min-width: 100px; /* Sets the minimum width to 100 pixels */
   max-width: 500px; /* Sets the maximum width to 500 pixels */
   min-height: 50px; /* Sets the minimum height to 50 pixels */
   max-height: 300px; /* Sets the maximum height to 300 pixels */
}
```

## 4. Auto Width and Height:

You can use auto to let the browser automatically determine the width or height based on the content:

```
.element {
  width: auto; /* Automatically adjusts the width based on content */
  height: auto; /* Automatically adjusts the height based on content */
}
```

**5. Viewport Units:**

CSS also supports viewport units (vw and vh) for setting width and height relative to the viewport size:

```
.element {
  width: 50vw; /* Sets the width to 50% of the viewport width */
  height: 75vh; /* Sets the height to 75% of the viewport height */
}
```

## BORDERS

- Border Property is the primary way to set border properties aroung an element.
- We Have Many Controls Over Border

**1. Border Style:** Specifies the style of the border, such as solid, dashed, dotted, double, groove, ridge, inset, outset, or none.

```
.element {
  border-style: solid; /* Solid border style */
}
```

**2. Border Width:** Sets the width of the border. You can use values like pixels (px), ems (em), or keywords like thin, medium, thick.

```
.element {
  border-width: 2px; /* Border width of 2 pixels */
}
```

```
.element {
  border-bottom-width: 2px; /* Sets the bottom border width to 2 pixels */
}
```

**3. Border Color:** Sets the color of the border. You can use color names, hex codes, RGB values, or the currentColor keyword

```
.element {
  border-color: #333; /* Border color as hex code */
}
```

**4. Border Shorthand:** The border shorthand property allows you to set all border properties (style, width, color) in one declaration.

```
.element {
  border: 2px solid #333; /* Width, Style, Color */
}
```

**5. Individual Borders:** You can set borders for specific sides of an element using individual properties like border-top, border-right, border-bottom, and border-left. This allows you to customize borders differently for each side if needed.

```
.element {
  border-top: 2px solid #333; /* Top border */
  border-bottom: 1px dashed #999; /* Bottom border */
}
```

**6. Border Radius:** Creates rounded corners for borders. You can specify the radius of each corner separately (border-radius-top-left, border-radius-top-right, border-radius-bottom-right, border-radius-bottom-left) or use a single value for all corners.

```
.element {
  border-radius: 5px; /* Applies rounded corners to all corners */
```

```
}
```

**Example with individual corner radius:**

```
.element {
  border-top-left-radius: 10px;
  border-top-right-radius: 5px;
  border-bottom-right-radius: 20px;
  border-bottom-left-radius: 15px;
}
```

**7. Border Image:** Allows you to use an image as a border instead of a solid color. This property is used in conjunction with border-width, border-style, and border-color to define the border image.

```
.element {
  border-image: url(border.png) 30 round; /* Border image */
}
```

## DISPLAY PROPERTIES AND BEHAVIOUR

### 1. display: none;

When you set an element's display to none, it completely removes the element from the document layout. The element will not take up any space on the page, and it won't be visible or interactable.

```
.hidden-element {
  display: none;
}
```

**The visibility: hidden;** CSS property is used to hide an element while still taking up space in the layout. This means that the element becomes invisible, but it still occupies the same space as if it were visible. The visibility property does not affect the layout of other elements around the hidden element.

```
.hidden-element {
   visibility: hidden;
}
```

## 2. display: inline;

Elements with display: inline; behave like inline elements. They flow along with the surrounding text or other inline elements. Inline elements do not start on a new line and only take up as much width as necessary for their content.

```
.inline-element {
   display: inline;
}
```

## 3. display: inline-block;

inline-block; behave like inline elements in terms of flowing with the text, but they also have block-level properties like setting width, height, margins, and paddings.

Inline-block elements start on a new line like block-level elements but flow inline with the surrounding content.

```
.inline-block-element {
   display: inline-block;
}
```

## 4. display: block;

behave as block-level elements. They start on a new line and occupy the full width available to them unless specified otherwise.

Block-level elements can have margins, paddings, and widths set, and they stack vertically one after another.

```
.block-element {
   display: block;
}
```

## DIFFERENCE BETWEEN INLINE AND INLINE-BLOCK

The main difference between inline and inline-block display values in CSS lies in how they interact with other elements and how they handle properties like width, height, margins, and padding.

**1. Inline Elements (display: inline;):** Inline elements flow along with the surrounding text or other inline elements. They do not start on a new line, and they only take up as much width as necessary for their content.

Inline elements do not have properties like width, height, margins, or paddings applied to them directly. Any attempt to set these properties will not have any effect.

**Examples of inline elements include , <a>, , , <img>, <input>, etc.**

**2. Inline-Block Elements (display: inline-block;):**

Inline-block elements behave like inline elements in terms of flowing with the text, but they also have block-level properties like setting width, height, margins, and paddings.

Inline-block elements start on a new line like block-level elements but flow inline with the surrounding content.

Unlike inline elements, inline-block elements can have properties like width, height, margins, paddings, and they will be respected by the browser.

**Examples of inline-block elements include ,  with display: inline-block;, <img> with display: inline-block;, etc.**

**Inline Element**

```
.inline-element {
    display: inline;
    width: 100px; /* This width will not have any effect */
    margin: 10px; /* This margin will not have any effect */
    padding: 5px; /* This padding will not have any effect */
}
```

**Inline-Block Element**

```
.inline-block-element {
```

```
    display: inline-block;
    width: 100px; /* Width will be applied */
    margin: 10px; /* Margin will be applied */
    padding: 5px; /* Padding will be applied */
}
```