## HTML&CSS

## 1. Box - sizing

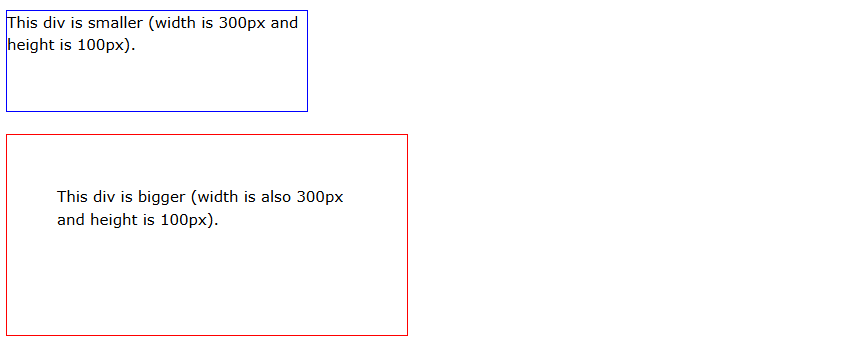
## Without the CSS box-sizing Property

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**

This means: When you set the width/height of an element, the element often appears bigger than you have set (because the element's border and padding are added to the element's specified width/height).

The following illustration shows two <div> elements with the same specified width and height:



The two <div> elements above end up with different sizes in the result (because div2 has a padding specified):

.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:



Here is the same example as above, with box-sizing: border-box; added to both <div> elements:

.div1 {  
  width: 300px;  
  height: 100px;  
  border: 1px solid blue;  
  box-sizing: border-box;  
}  
  
.div2 {  
  width: 300px;  
  height: 100px;  
  padding: 50px;  
  border: 1px solid red;  
  box-sizing: border-box;  
}

Since the result of using the box-sizing: border-box; is so much better, many developers want all elements on their pages to work this way.

The code below ensures that all elements are sized in this more intuitive way. Many browsers already use box-sizing: border-box; for many form elements (but not all - which is why inputs and text areas look different at width: 100%;).

1. **Animation**

****Animation Declaration**:**

animation: bounce 800ms ease-in-out infinite alternate;

· *bounce* specifies the name of the animation (defined in @keyframes).

· *800ms* sets the duration of each bounce cycle to 800 milliseconds.

· *ease-in-out* makes the animation start and end smoothly.

· *infinite* makes the animation loop endlessly.

· *alternate* reverses the animation direction after each cycle, making it move up and down continuously.

****Keyframes Definition**:**

@keyframes bounce {

0% { transform: translateY(-10px); }

100% { transform: translateY(0); }

}

·  *0%* and *100%* define the start and end points of the animation.

· At *0%*, the element is moved up by 10px (translateY(-10px)), and at *100%*, it returns to its original position (translateY(0)).

This results in a smooth up-and-down "bouncing" effect, where the element moves up by 10 pixels and then returns to its original position, repeating continuously.

1. **Add EventListener**

**HTML**

<!-- #PRELOADER -->

    <div class="preloader" data-preloader>

        <div class="preloader-inner">

            <img src="./assets/images/preloader.svg" alt="" class="img">

        </div>

    </div>

**CSS**

.preloader.remove{

  visibility: hidden;

  opacity: 0;

}

**Select the Preloader Element**:

const preloader = document.querySelector("[data-preloader]");

- This line selects an element with the attribute data-preloader. This attribute selector is commonly used to target elements in HTML that have a custom attribute like data-preloader.

**Add Event Listener for** load **Event**:

window.addEventListener("load", () => {

preloader.classList.add("remove");

});

· window.addEventListener("load", ...) waits until the entire page (including images and sub-resources) has loaded.

· When the load event fires, the callback function is executed, which adds the remove class to the *preloader* element.

**Hiding the Preloader**:

The remove class (from your earlier CSS) sets *visibility: hidden*; and *opacity: 0*; for *.preloader.remove,* making the preloader fade out when the page is fully loaded.

This approach is common for displaying a loading screen or animation until the entire content of the page is ready for interaction.