Lesson:

CSS Custom Properties

CSS
HTML
<HTML>
<HEAD>
<TITLE> Lorem ipsum </TITLE>
</HEAD>
<BODY>
<H1> Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam</H1>
<P> <IMG SRC= "Lorem ipsum"></P>

SKILLS

Topics Covered

- Understanding CSS Custom properties
- Example of CSS Custom properties
- cal() func

Understanding CSS Custom properties

CSS custom properties which are sometimes referred to as CSS variables or cascading variables are entities defined by CSS authors that contain specific values to be reused throughout a document.
They are set using custom property notation example

- - main-color:black and are accessed using the **var()** function i.e **color: var(--main-color)**;

Complex websites have very large amounts of CSS, often with a lot of repeated values, For example, the same color might be used in hundreds of different places, requiring global search and replace if that color needs to change. Custom properties allow a value to be stored in one place and then referenced in multiple other places. An additional benefit is semantic identifiers. For example, **--main-text-color** is easier to understand than **#00ff00**

Note - When naming CSS variables, it contains only letters and dashes just like other CSS properties. Examples like **line-height, -moz-box-sizing**. But it should start with double dashes (--)

These are invalid variable names

```
--123color: blue;
--#color: red;
--bg_color: yellow;
--$width: 100px
```

These are valid variable names

```
--color: red;
--bg-color: yello;
--width: 100px
```

Example of CSS Custom properties

CSS custom properties example can be of Variable Color, Variable Dimensions and much more.

Variable Color

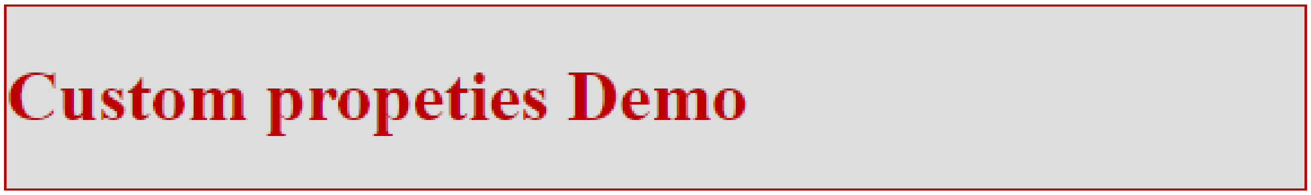
index.html

```
<div class="container">
      <h1>Custom propeties Demo</h1>
    </div>
```

style.css

```
:root {
  --red: #b00;
  --blue: #4679bd;
  --grey: #ddd;
}
.container {
  color: var(--red);
  background: var(--grey);
  border: 1px solid var(--red);
}
```

Browser output -



```
Custom propeties Demo
```

Variable Dimensions

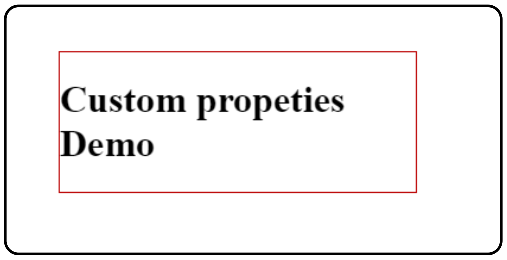
index.html

```
  <div class="container">
      <h1>Custom propeties Demo</h1>
    </div>
```

style.css

```
:root {
  --w200: 300px;
  --m10: 40px;
  --red: #b00;
}
.container {
  border: 1px solid var(--red);
  width: var(--w200);
  margin: var(--m10); }
```

Browser output -


Custom propeties
Demo

Example of Variable Cascading

CSS variables cascade in much the same way as other properties, and can be restated safely. Variables can be defined multiple times and only the definition with the highest specificity will apply to the element selected.

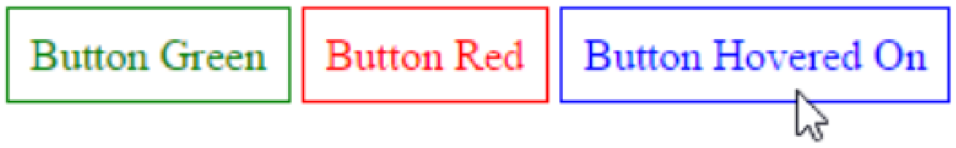
index.html

```
  <body>
     <a class="button" href="#"> Button Green</a>
     <a class="button btn_red" href="#"> Button Red</a>
     <a class="button" href="#"> Button Hover On</a>
   </body>
```

style.css

```
.button {
  --color: green;
  padding: 0.5rem;
  border: 1px solid var(--color);
  color: var(--color);
}
.button:hover {
  --color: blue;
}
.btn_red {
  --color: red;
}
```

Browser output -



cal() func

You can utilize the calc() CSS function to execute calculations while defining values for CSS properties.

Syntax:

```
property: calc(expression)
```

The expression can be any simple expression combining the following operators, using standard operator precedence rules:

- + Addition.
- - Subtraction.
- * Multiplication
- / Division

Example:-

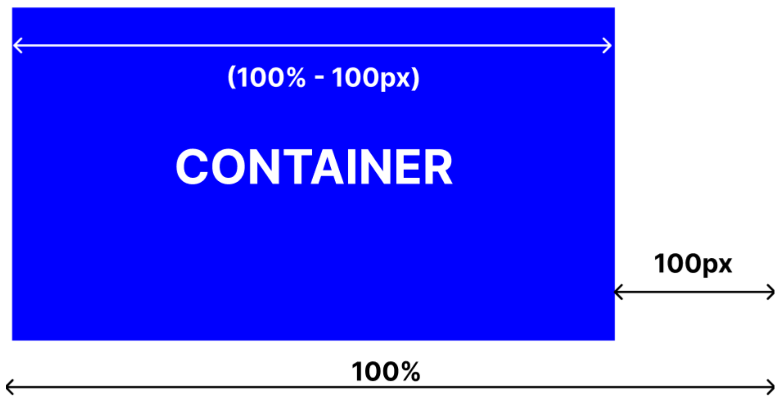
```
<style>
    #container {
      width: calc(100% - 100px);
      height: 50vh;
      background-color: blue;
   }
</style>
<body>
   <div id="container"></div>
</body>
```

In the above example, we have set the width of the **container** by calculating using cal() function as follows,

width = calc(100% - 100px)

It means, the final width will be the total width available - 100px.

Browser output -


(100% - 100px)
CONTAINER
100px
100%