Let’s roll by 09:05 !!!

CSS 1

Gopi Krishnan R

Topics to be Discussed

- What’s CSS?
- 3 types of CSS inclusions
- CSS Selectors
- CSS Cascade
- Relative lengths
- Absolute lengths

CSS

CSS is used to control the style of a web document in a simple and easy way.

CSS is the acronym for **"Cascading Style Sheet"**. This tutorial covers both the versions CSS1,CSS2 and CSS3, and gives a complete understanding of CSS, starting from its basics to advanced concepts.

3 Types of Inclusion

- Inline
- Internal or Embedded CSS
- External CSS

Inline CSS

Inline CSS contains the CSS property in the body section attached with element is known as inline CSS. This kind of style is specified within an HTML tag using the style attribute.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Inline CSS</title>
    </head>

    <body>
        <p style = "color:#009900; font-size:50px;
                font-style:italic; text-align:center;">
            GeeksForGeeks
        </p>

    </body>
</html>
```

Internal or Embedded CSS

This can be used when a single HTML document must be styled uniquely. The CSS rule set should be within the HTML file in the head section i.e the CSS is embedded within the HTML file.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Internal CSS</title>
        <style>
            .main {
                text-align:center;
            }
            .GFG {
                color:#009900;
                font-size:50px;
                font-weight:bold;
            }
            .geeks {
                font-style:bold;
                font-size:20px;
            }
        </style>
    </head>
    <body>
        <div class = "main">
            <div class ="GFG">GeeksForGeeks</div>

            <div class ="geeks">
                A computer science portal for geeks
            </div>
        </div>
    </body>
</html>
```

External CSS

External CSS contains separate CSS file which contains only style property with the help of tag attributes (For example class, id, heading, ... etc). CSS property written in a separate file with .css extension and should be linked to the HTML document using **link** tag. This means that for each element, style can be set only once and that will be applied across web pages.

- **link** tag is used to link the external style sheet with the html webpage.
- **href** attribute is used to specify the location of the external style sheet file.

geeks.css

```
body {
    background-color:powderblue;
}
.main {
    text-align:center;
}
.GFG {
    color:#009900;
    font-size:50px;
    font-weight:bold;
}
#geeks {
    font-style:bold;
    font-size:20px;
}
```

CSS Selectors

CSS selectors are used to "find" (or select) the HTML elements you want to style.

We can divide CSS selectors into five categories:

- Simple selectors (select elements based on name, id, class, etc.)
- Combinator_selectors (select elements based on a specific relationship between them)
- Pseudo-class selectors (select elements based on a certain state)
- Pseudo-elements selectors (select and style a part of an element)
- Attribute selectors (select elements based on an attribute or attribute value)

Simple Selector

- CSS element Selector

```
p {
  text-align: center;
  color: red;
}
```

- CSS id Selector

```
#para1 {
  text-align: center;
  color: red;
}
```

- CSS Class Selector

```css
.center {
  text-align: center;
  color: red;
}
```

```css
p.center {
  text-align: center;
  color: red;
}
```

- CSS Universal Selector

```css
* {
  text-align: center;
  color: blue;
}
```

- CSS Grouping Selector

```css
h1, h2, p {
  text-align: center;
  color: red;
}
```

CSS Cascade

We'll begin breaking down exactly how styles are rendered by looking at what is known as the cascade and studying a few examples of the cascade in action. Within CSS, all styles cascade from the top of a style sheet to the bottom, allowing different styles to be added or overwritten as the style sheet progresses.

p { background: orange; font-size: 24px; } p { background: green; }

Because the paragraph selector that sets the background color to green comes after the paragraph selector that sets the background color to orange, it will take precedence in the cascade. All of the paragraphs will appear with a green background. The font size will remain 24 pixels because the second paragraph selector didn’t identify a new font size.

Relative Lengths

Unit	Description
em	Relative to the font-size of the element (2em means 2 times the size of the current font)
ex	Relative to the x-height of the current font (rarely used)
ch	Relative to the width of the "0" (zero)
rem	Relative to font-size of the root element
vw	Relative to 1% of the width of the viewport*
vh	Relative to 1% of the height of the viewport*
vmin	Relative to 1% of viewport's* smaller dimension
vmax	Relative to 1% of viewport's* larger dimension
%	Relative to the parent element

Absolute lengths

cm	centimeters
mm	millimeters
in	inches (1in = 96px = 2.54cm)
px *	pixels (1px = 1/96th of 1in)
pt	points (1pt = 1/72 of 1in)
pc	picas (1pc = 12 pt)

Lets code Lengths

- https://www.w3schools.com/cssref/css_units.asp

Let’s roll by 09:05 !!!

CSS 2

Gopi Krishnan R

Topics to be Discussed

- CSS properties
  - Lengths (Remaining)
  - Color and Background
  - Styling text
- CSS Box Model
- CSS Properties
  - CSS margin and padding
  - CSS border
  - Border radius

Lets code Lengths

- https://www.w3schools.com/cssref/css_units.asp

Colors

- Color - text
- RGB

  rgb(red, green, blue)

  Each parameter (red, green, and blue) defines the intensity of the color between 0 and 255.
- HEX

  #rrggbb

  Where rr (red), gg (green) and bb (blue) are hexadecimal values between 00 and ff (same as decimal 0-255).
- HSL

  hsl(hue, saturation, lightness)

  Hue is a degree on the color wheel from 0 to 360. 0 is red, 120 is green, and 240 is blue.

  Saturation is a percentage value, 0% means a shade of gray, and 100% is the full color.

  Lightness is also a percentage, 0% is black, 50% is neither light or dark, 100% is white

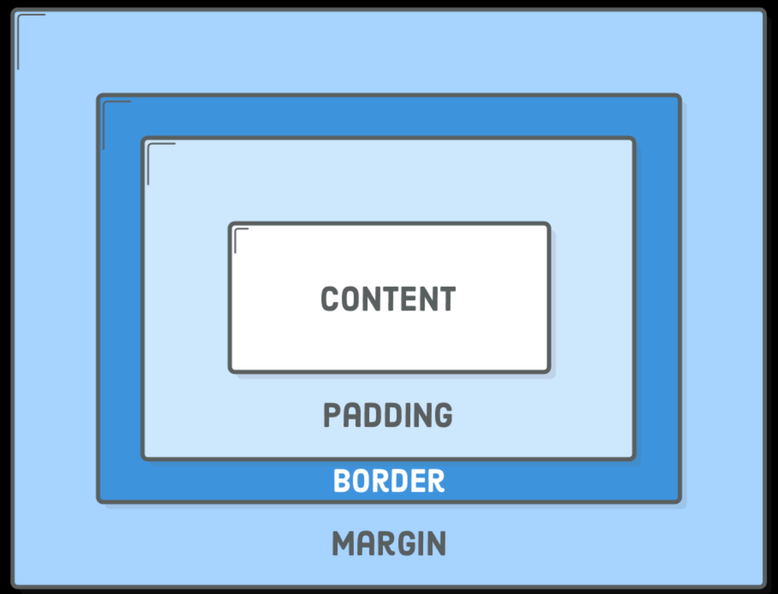
Styling Text

- font-family
- font-size
- font-weight
- font-style
- text-decoration
- text-transform
- text-spacing

CSS Box Model

Every box is composed of four parts (or areas), defined by their respective edges: the content edge, padding edge, border edge, and margin edge.

- The content area, bounded by the content edge, contains the "real" content of the element, such as text, an image, or a video player. Its dimensions are the content width (or content-box width) and the content height (or content-box height). It often has a background color or background image.
- The padding area, bounded by the padding edge, extends the content area to include the element's padding. Its dimensions are the padding-box width and the padding-box height.
- The border area, bounded by the border edge, extends the padding area to include the element's borders. Its dimensions are the border-box width and the border-box height.



CSS Margin & Padding

The CSS margin properties are used to create space around elements, outside of any defined borders.

CSS has properties for specifying the margin for each side of an element:

- margin-top
- margin-right
- margin-bottom
- margin-left

All the margin properties can have the following values:

- auto - the browser calculates the margin
- *length* - specifies a margin in px, pt, cm, etc.
- % - specifies a margin in % of the width of the containing element
- inherit - specifies that the margin should be inherited from the parent element

CSS Padding

The CSS padding properties are used to generate space around an element's content, inside of any defined borders.

CSS has properties for specifying the padding for each side of an element:

- padding-top
- padding-right
- padding-bottom
- padding-left

All the padding properties can have the following values:

- *length* - specifies a padding in px, pt, cm, etc.
- % - specifies a padding in % of the width of the containing element
- inherit - specifies that the padding should be inherited from the parent element

Border

- Border-style
- Border-sides
- Border-width
- Border-color
- Border-radius

Let’s roll by 09:05 !!!

CSS 3

Gopi Krishnan R

Topics to be Discussed

- Selectors (extended)
- CSS Display
- CSS Position
- Pseudo Classes
- Shorthand Notation

Selectors

- Child Selector

```
ol.todos li {
  font-size: 30px;
}
```

- Immediate Child Selector

```
div.container > p {
  text-decoration: underline;
}
```

- Sibling Selector

```
div.container span ~ span {
  color: orange;
}
```

- Immediate Sibling Selector

```
span + p {
  margin-top: 20px;
}
```

CSS Display Properties

CSS specifies three display types - inline, block and none.

inline does just what it says — boxes that are displayed inline follow the flow of a line.

block makes a box standalone, fitting the entire width of its containing box, with an effective line break before and after it.

none, well, doesn’t display a box at all, which may sound pretty useless but can be used to good effect with dynamic effects, such as switching extended information on and off at the click of a link, or in alternative stylesheets.

CSS Position

The position property is used to define whether a box is absolute, relative, static or fixed:

- static is the default value and renders a box in the normal order of things, as they appear in the HTML.
- relative is much like static but the box can be offset from its original position with the properties top, right, bottom and left.
- absolute pulls a box out of the normal flow of the HTML and delivers it to a world all of its own. In this crazy little world, the absolute box can be placed anywhere on the page using top, right, bottom and left.
- fixed behaves like absolute, but it will absolutely position a box in reference to the browser window as opposed to the web page, so fixed boxes should stay exactly where they are on the screen even when the page is scrolled.

Pseudo Classes

- Pseudo classes are specified on selectors to specify a state or relation to the selector.

```
a:link {
    color: blue;
}

a:visited {
    color: purple;
}
```

Shorthand Notation

- Margin
- Padding
- Border

Let’s roll by 09:05 !!!

CSS 4

Gopi Krishnan R

Topics to be Discussed

- CSS Shadows
- CSS Transitions
- CSS Transformations
- Gradients

CSS Shadows

box-shadow is the standard CSS property to add shadows to an element in a web page.

```
box-shadow: none|h-offset v-offset blur spread color |inset|initial|inherit;
```

CSS Transitions

Transitions allow you to easily animate parts of your design without the need for Javascript.

- transition
- transition-delay
- transition-duration
- transition-property
- transition-timing-function

CSS Transformation

The transform property applies a 2D or 3D transformation to an element. This property allows you to rotate, scale, move, skew, etc., elements.

```
transform: none|transform-functions|initial|inherit;
```

CSS Gradient

CSS defines three types of gradients

- Linear Gradients (goes down/up/left/right/diagonally)
- Radial Gradients (defined by their centre)
- Conic Gradients (rotated around a centre point)

Let’s roll by 09:05 !!!

CSS 5

Gopi Krishnan R

Topics to be Discussed

- Gradients
- CSS Flex-box
- CSS Flex-container

CSS Gradient

CSS defines three types of gradients

- Linear Gradients (goes down/up/left/right/diagonally)
- Radial Gradients (defined by their centre)
- Conic Gradients (rotated around a centre point)

CSS Flex-box

- The Flexible Box Module, usually referred to as flexbox, was designed as a one-dimensional layout model, and as a method that could offer space distribution between items in an interface and powerful alignment capabilities.
- When we describe flexbox as being one dimensional we are describing the fact that flexbox deals with layout in one dimension at a time — either as a row or as a column.

Why Flex-box?

- By default, HTML block-level elements stack, so if you want to align them in a row, you need to rely on CSS properties like float, or manipulate the display property with table-ish or inline-block settings.

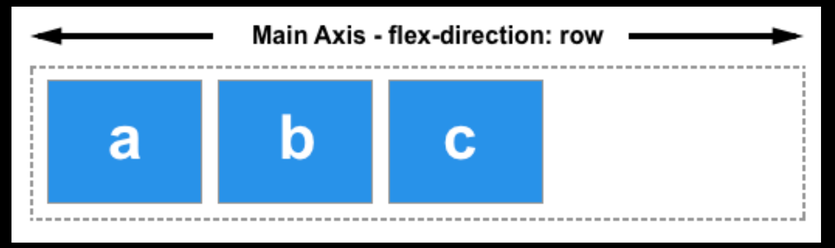
- If you choose to use float (left or right), you must clear the wrapper at some point to push it until the very last floated element, otherwise, they will overflow over anything coming after. However, with float, you are limited to organising elements horizontally.

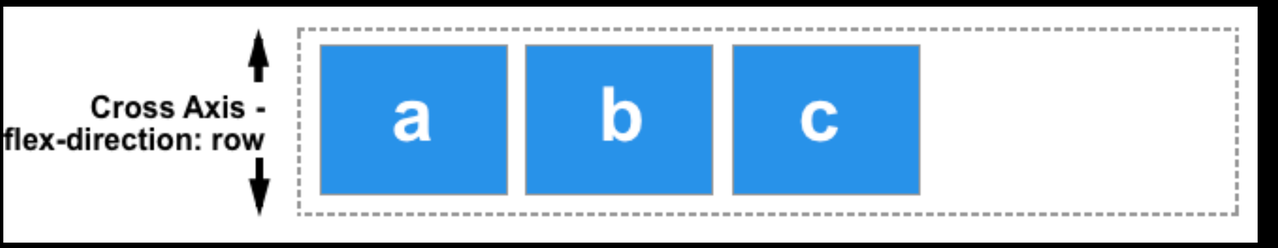
- Alternatively, or in addition, you can manipulate the display property to try to achieve the layout you desire! But this often feels kludgy at best, not to mention tedious, and often results in a fairly fragile layout that won't necessarily render consistently across browsers. This approach is particularly ineffective if you are targeting multiple devices of varying sizes—which is almost always the case nowadays.

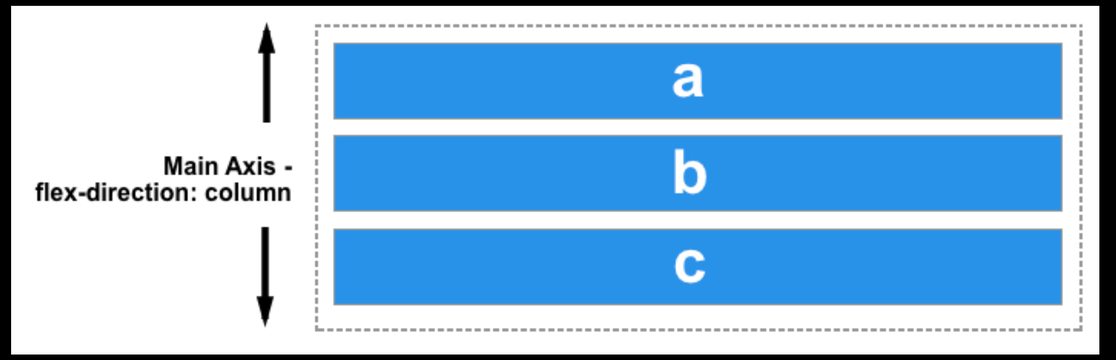
Two axis of Flex

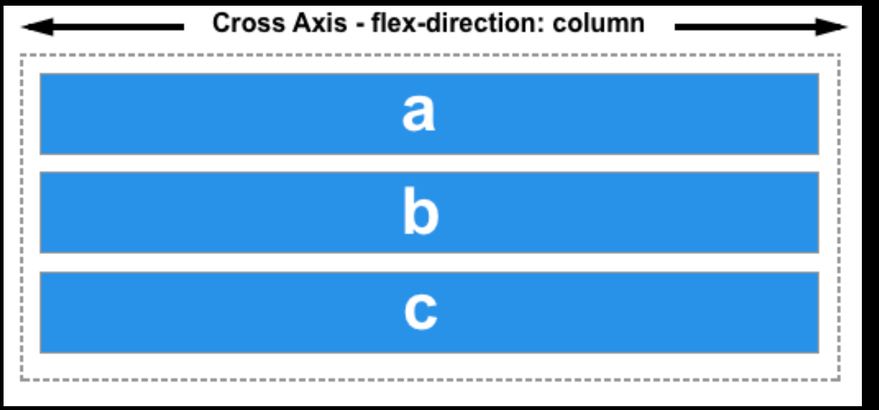
- Main Axis
- Cross Axis

- Values these axis can take:
  - row
  - row-reverse
  - column
  - column-reverse


Main Axis - flex-direction: row
a
b
c


Cross Axis - flex-direction: row
a
b
c


Main Axis - flex-direction: column
a
b
c


Cross Axis - flex-direction: column
a
b
c

Flex Container

The flex container properties are:

- flex-direction
- flex-wrap
- flex-flow
- justify-content
- align-items
- align-content

Let’s roll by 09:05 !!!

CSS 6

Gopi Krishnan R

Topics to be Discussed

- Flex Items
- CSS Grid Layout
- CSS Grid Container
- CSS Grid Item

Flex Items

Flex Items properties

- order
- flex-grow
- flex-shrink
- flex-basis
- flex
- align-self

CSS Grid

The CSS Grid Layout Module offers a grid-based layout system, with rows and columns, making it easier to design web pages without having to use floats and positioning.

```
<div class="grid-container">
  <div class="grid-item">1</div>
  <div class="grid-item">2</div>
  <div class="grid-item">3</div>
  <div class="grid-item">4</div>
  <div class="grid-item">5</div>
  <div class="grid-item">6</div>
  <div class="grid-item">7</div>
  <div class="grid-item">8</div>
  <div class="grid-item">9</div>
</div>
```

Grid Container

- grid-template-columns
- grid-template-rows
- justify-content
- align-content

CSS Grid Item

A grid *container* contains grid *items*.

By default, a container has one grid item for each column, in each row, but you can style the grid items so that they will span multiple columns and/or rows.

- grid-row
- grid-column
- grid-area
- order

Let’s roll by 09:05 !!!

CSS 7

Gopi Krishnan R

Topics to be Discussed

- CSS Grid Item
- Bootstrap
- Inclusion of Bootstrap
- Bootstrap components

CSS Grid Item

A grid *container* contains grid *items*.

By default, a container has one grid item for each column, in each row, but you can style the grid items so that they will span multiple columns and/or rows.

- grid-row - shorthand for grid-row-start and grid-row-end
- grid-column - shorthand for grid-column-start and grid-column-end
- grid-area - shorthand for grid-row-start, grid-column-start, grid-row-end, grid-column-end
- order

Bootstrap

Bootstrap is a powerful front-end framework for faster and easier web development. It includes HTML and CSS based design templates for creating common user interface components like forms, buttons, navigations, dropdowns, alerts, modals, tabs, accordions, carousels, tooltips, and so on.

Bootstrap gives you ability to create flexible and responsive web layouts with much less efforts.

Bootstrap Inclusion

- Use it from https://getbootstrap.com/docs/5.1/getting-started/download/

Bootstrap Components

- Buttons
- List Group
- Tables
- Modal
- Carousel
- Bootstrap Forms

Let’s roll by 09:05 !!!

CSS 8

Gopi Krishnan R

Topics to be Discussed

- Bootstrap components
- Bootstrap Breakpoints
- Bootstrap Layouts
  - Container
  - Grid
  - Columns
  - Gutters

Bootstrap Components

- Buttons
- List Group
- Modal
- Carousel
- Bootstrap Forms

Bootstrap Breakpoints

Breakpoints are customizable widths that determine how your responsive layout behaves across device or viewport sizes in Bootstrap.

- **Breakpoints are the building blocks of responsive design.** Use them to control when your layout can be adapted at a particular viewport or device size.
- Use media queries to architect your CSS by breakpoint. Media queries are a feature of CSS that allow you to conditionally apply styles based on a set of browser and operating system parameters. We most commonly use min-width in our media queries.
- **Mobile first, responsive design is the goal.** Bootstrap’s CSS aims to apply the bare minimum of styles to make a layout work at the smallest breakpoint, and then layers on styles to adjust that design for larger devices. This optimizes your CSS, improves rendering time, and provides a great experience for your visitors.

Breakpoint	Class infix	Dimensions
X-Small	*None*	<576px
Small	`sm`	≥576px
Medium	`md`	≥768px
Large	`lg`	≥992px
Extra large	`xl`	≥1200px
Extra extra large	`xxl`	≥1400px

Bootstrap Container

Containers are the most basic layout element in Bootstrap and are **required when using our default grid system**. Containers are used to contain, pad, and (sometimes) center the content within them. While containers *can* be nested, most layouts do not require a nested container.

Bootstrap comes with three different containers:

- .container, which sets a max-width at each responsive breakpoint
- .container-fluid, which is width: 100% at all breakpoints
- .container-{breakpoint}, which is width: 100% until the specified breakpoint

	Extra small <576px	Small ≥576px	Medium ≥768px	Large ≥992px	X-Large ≥1200px	XX-Large ≥1400px
`.container`	100%	540px	720px	960px	1140px	1320px
`.container-sm`	100%	540px	720px	960px	1140px	1320px
`.container-md`	100%	100%	720px	960px	1140px	1320px
`.container-lg`	100%	100%	100%	960px	1140px	1320px
`.container-xl`	100%	100%	100%	100%	1140px	1320px
`.container-xxl`	100%	100%	100%	100%	100%	1320px
`.container-fluid`	100%	100%	100%	100%	100%	100%

Bootstrap Grid

Bootstrap’s grid system uses a series of containers, rows, and columns to layout and align content. It’s built with flexbox and is fully responsive. Below is an example and an in-depth explanation for how the grid system comes together.

Bootstrap Columns

- Vertical Alignment
- Horizontal Alignment

Let’s roll by 09:05 !!!

CSS 9

Gopi Krishnan R

Topics to be Discussed

- Bootstrap Layouts
  - Container
  - Gutters
  - Z-index
- CSS Preprocessors
- SASS and SCSS

Bootstrap Container

Containers are the most basic layout element in Bootstrap and are **required when using our default grid system**. Containers are used to contain, pad, and (sometimes) center the content within them. While containers *can* be nested, most layouts do not require a nested container.

Bootstrap comes with three different containers:

- .container, which sets a max-width at each responsive breakpoint
- .container-fluid, which is width: 100% at all breakpoints
- .container-{breakpoint}, which is width: 100% until the specified breakpoint

	Extra small <576px	Small ≥576px	Medium ≥768px	Large ≥992px	X-Large ≥1200px	XX-Large ≥1400px
`.container`	100%	540px	720px	960px	1140px	1320px
`.container-sm`	100%	540px	720px	960px	1140px	1320px
`.container-md`	100%	100%	720px	960px	1140px	1320px
`.container-lg`	100%	100%	100%	960px	1140px	1320px
`.container-xl`	100%	100%	100%	100%	1140px	1320px
`.container-xxl`	100%	100%	100%	100%	100%	1320px
`.container-fluid`	100%	100%	100%	100%	100%	100%

Gutters

- Gutters are the gaps between column content, created by horizontal padding. We set padding-right and padding-left on each column, and use negative margin to offset that at the start and end of each row to align content.

- Gutters start at 1.5rem (24px) wide. This allows us to match our grid to the padding and margin spacers scale.

- **Gutters can be responsively adjusted.** Use breakpoint-specific gutter classes to modify horizontal gutters, vertical gutters, and all gutters.

- Horizontal Gutter

- Vertical Gutter

CSS Preprocessor

CSS is primitive and incomplete. Building a function, reusing a definition or inheritance are hard to achieve. For bigger projects, or complex systems, maintenance is a very big problem. On the other hand, web is evolving, new specs are being introduced to HTML as well as CSS. Browsers apply these specs while they are in proposal state with their special vendor prefixes. In some cases (as in background gradient), coding with vendor specific properties become a burden. You have to add all different vendor versions for a single result.

1. Variables
2. Nesting
3. Mixins
4. Extends
5. Conditionals
6. Loops
7. Calculations

SCSS and SASS

SASS (Syntactically Awesome Style Sheets) is a pre-processor scripting language that will be compiled or interpreted into CSS. SassScript is itself a scripting language whereas SCSS is the main syntax for the SASS which builds on top of the existing CSS syntax. It makes use of semicolons and brackets like CSS (Cascaded Style Sheets).

SASS and SCSS can import each other. Sass actually makes CSS more powerful with math and variable support.

SCSS

```
/* .scss file */
$bgcolor: blue;
$textcolor: red;
$fontsize: 25px;

/* Use the variables */
body {
background-color: $bgcolor;
color: $textcolor;
font-size: $fontsize;
}
```

SASS

```
/* SASS */


$primary-color: green

$primary-bg: red


body
  color: $primary-color
  background: $primary-bg
```