**CSS Grid**

CSS Grid is a powerful layout system used to create complex, responsive layouts. It works by defining a grid with rows and columns, allowing items to be placed precisely.

* **Key Properties**:
  + display: grid; – Initializes grid container.
  + grid-template-columns and grid-template-rows – Define the number and size of columns and rows.
  + gap – Controls the spacing between grid items.
  + grid-area – Used to specify the placement of grid items.

.grid-container {

display: grid;

grid-template-columns: repeat(3, 1fr);

grid-template-rows: 100px 200px;

gap: 10px;

}

**CSS Preprocessors (SCSS)**

SCSS (Sassy CSS) is a CSS preprocessor that adds features like variables, nested rules, and functions, making CSS easier to write and maintain.

* **Key Features**:
  + **Variables**: $primary-color: #3498db;
  + **Nesting**: Nesting of selectors makes CSS more readable.
  + **Mixins**: Reusable blocks of styles.

$primary-color: #3498db;

$padding: 20px;

.box {

background-color: $primary-color;

padding: $padding;

}

**CSS @property**

* **Definition**: This is a way to define custom properties (variables) in CSS. It allows you to specify a type, syntax, and initial value for a custom property.
* **Usage**:
  + Use @property to create a variable that can be animated.

@property --my-color {

syntax: '<color>';

inherits: false;

initial-value: red;

}

**CSS Box Sizing**

* **Definition**: Controls how the width and height of elements are calculated in CSS.
* **Properties**:
  + box-sizing: content-box: The default. Width and height only apply to the content, not the padding or border.
  + box-sizing: border-box: Width and height include padding and border. This is often easier to work with.

\* {

box-sizing: border-box;

}

**CSS Media Queries**

* **Definition**: A technique that allows you to apply styles based on the size of the viewport (the visible area of the web page).
* **Usage**: Helps make designs responsive, meaning they adapt to different screen sizes.

@media (max-width: 600px) {

body {

background-color: lightblue;

}

}

**CSS Flexbox**

* **Definition**: A layout model that allows you to design a complex layout structure more easily and efficiently.
* **Properties**:
  + display: flex: Enables flexbox on a container.
  + flex-direction: Defines the direction of the flex items (row or column).
  + justify-content: Aligns items along the main axis (e.g., center, space-between).
  + align-items: Aligns items along the cross axis (e.g., center, stretch).

.flex-container {

display: flex;

justify-content: center;

align-items: center;

flex-direction: row; /\* Items will be placed in a row \*/

}

**CSS Rounded Corners**

* **Definition**: You can create rounded corners for elements using the border-radius property.

.box {

border: 2px solid black;

border-radius: 15px; /\* 15px rounded corners \*/

}

**CSS Border Images**

* **Definition**: Allows you to use an image as the border of an element.
* **Properties**:
  + border-image-source: The image you want to use.
  + border-image-slice: How to slice the image.
  + border-image-width: Width of the border image.

.bordered {

border: 10px solid transparent;

border-image-source: url('border.png');

border-image-slice: 30;

}

**CSS Backgrounds**

* **Definition**: Controls the background of elements.
* **Properties**:
  + background-color: Sets a solid color.
  + background-image: Adds an image.
  + background-size: Controls the size of the image.

.background {

background-color: lightgrey;

background-image: url('image.jpg');

background-size: cover; /\* Image covers the whole area \*/

}

**CSS Colors**

* **Definition**: Specifies colors for text and backgrounds.
* **Formats**:
  + Named colors (e.g., red).
  + HEX values (e.g., #ff0000).
  + RGB (e.g., rgb(255, 0, 0)).
  + RGBA (e.g., rgba(255, 0, 0, 0.5) for transparency).

p {

color: rgba(0, 0, 0, 0.8);

}

**CSS Color Keywords**

* **Definition**: Predefined names for colors.
* **Examples**: black, white, blue, red, yellow, etc.
* **Usage**: Quick and easy way to set colors.

h1 {

color: blue;

}

**CSS Gradients**

* **Definition**: Creates a smooth transition between two or more colors.
* **Types**:
  + Linear gradients: Transitions along a line.
  + Radial gradients: Transitions outward from a center point.

.gradient {

background: linear-gradient(to right, red, yellow);

}

**CSS Shadows**

* **Definition**: Adds shadow effects to elements.
* **Types**:
  + Box shadow for elements.
  + Text shadow for text.

.box {

box-shadow: 5px 5px 10px rgba(0, 0, 0, 0.5);

}

**CSS Text Effects**

* **Definition**: Styles applied to text to enhance appearance.
* **Examples**:
  + Text shadow: Gives depth to text.
  + Letter spacing: Adjusts the space between letters.

h2 {

text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.3);

letter-spacing: 1px;

}

**CSS Web Fonts**

* **Definition**: Custom fonts that can be used on web pages.
* **Usage**: Can be added using @font-face or via services like Google Fonts.

@font-face {

font-family: 'MyCustomFont';

src: url('myfont.woff2') format('woff2');

}

body {

font-family: 'MyCustomFont', sans-serif;

}

**CSS 2D Transforms**

* **Definition**: Allows elements to be transformed in 2D space (like rotating, scaling, or moving).

.rotate {

transform: rotate(45deg);

}

**CSS 3D Transforms**

* **Definition**: Enables 3D transformations to create depth effects.
* **Properties**:
  + transform-style: Controls how child elements are rendered in 3D.
  + perspective: Defines the perspective for 3D space.

.container {

perspective: 1000px; /\* Set perspective for 3D \*/

}

.cube {

transform: rotateY(45deg); /\* Rotate in 3D space \*/

}

**CSS Transitions**

* **Definition**: Smoothly animates changes to CSS properties over time.

.button {

transition: background-color 0.3s ease;

}

.button:hover {

background-color: green;

}

**CSS Animations**

* **Definition**: Allows more complex animations using keyframes.
* **Usage**: Use @keyframes to define an animation, then apply it with the animation property.

@keyframes slide {

from { transform: translateX(0); }

to { transform: translateX(100px); }

}

.animate {

animation: slide 2s infinite;

}

**CSS Tooltips**

* **Definition**: Small pop-up boxes that appear when you hover over an element, providing additional information.
* **Implementation**: Usually created with the :hover pseudo-class.

.tooltip {

position: relative; /\* Position relative for absolute positioning \*/

}

.tooltip:hover::after {

content: "Tooltip text"; /\* Text shown in the tooltip \*/

position: absolute;

background: black;

color: white;

padding: 5px;

border-radius: 5px;

bottom: 100%; /\* Position above the element \*/

left: 50%; /\* Center horizontally \*/

transform: translateX(-50%); /\* Adjust positioning \*/

}

**CSS Style Images**

* **Definition**: Adjusts how images are displayed on the web.
* **Properties**:
  + object-fit: Defines how an image should be resized to fit its container.
  + object-position: Sets the alignment of the image within the container.

img {

object-fit: cover; 🡪 Cover the container

object-position: center;

}

**CSS Image Reflection**

* **Definition**: Creates a reflection effect for images (though not widely supported in modern browsers).

img {

-webkit-box-reflect: below 0 linear-gradient(transparent, transparent 20%, rgba(255, 255, 255, 0.5));

}

**CSS Masking**

* **Definition**: Hides parts of an element based on a mask image or shape.

.masked {

mask-image: url('mask.png');

mask-size: cover;

}

**CSS Buttons**

* **Definition**: Custom styles for buttons to improve aesthetics and usability.

.button {

background-color: blue;

color: white;

padding: 10px 20px;

border: none;

border-radius: 5px;

cursor: pointer;

}

.button:hover {

background-color: darkblue;

}

**CSS Pagination**

* **Definition**: Styles for pagination links or components that allow navigation through multiple pages of content.

.pagination {

display: flex;

list-style: none;

}

.pagination li {

margin: 0 5px;

}

.pagination a {

text-decoration: none;

padding: 5px 10px;

border: 1px solid lightgray;

border-radius: 5px;

}

.pagination a:hover {

background-color: lightgray;

}

**CSS Multiple Columns**

* **Definition**: Splits content into multiple columns for better readability.
* **Properties**:
  + column-count: Number of columns.
  + column-gap: Space between columns.

.multi-column {

column-count: 3;

column-gap: 20px;

}

**CSS User Interface**

* **Definition**: Styles to enhance user interactions and experiences (e.g., form elements, buttons).

input[type="text"] {

border: 1px solid lightgray;

padding: 10px;

border-radius: 5px; /\* Rounded input fields \*/

}

**CSS Variables**

* **Definition**: Allows you to define reusable values using custom properties.
* **Usage**: Use --variable-name syntax to define and var(--variable-name) to use.

:root {

--main-color: #3498db;

}

body {

background-color: var(--main-color);

}