Advanced CSS Techniques

The Box Model

Box model adalah dasar dari layout CSS. Setiap elemen di halaman web dianggap sebagai box yang terdiri dari margin, border, padding, dan area konten.

- Margin mengontrol ruang di luar box.
- Border adalah garis yang mengelilingi box.
- Padding adalah ruang antara border dan - konten.
- Konten adalah isi dari box.

```
.box {
  margin: 10px;
  border: 5px solid black;
  padding: 20px;
}
```

Floats in CSS

Float adalah properti CSS yang memungkinkan elemen untuk 'mengapung' ke kiri atau kanan, membiarkan elemen lain mengalir di sekelilingnya. Ini sering digunakan untuk mengatur gambar dalam teks atau untuk membuat layout kolom.

Penggunaan float bisa menjadi problematik, terutama dalam hal mengatur 'clearfix' agar layout tetap rapi.

```
.float-left {
  float: left;
}

.float-right {
  float: right;
}
```

Floats in CSS - Clearfix Hack

Clearfix Hack: adalah teknik yang digunakan untuk mengatasi masalah yang sering terjadi saat menggunakan float. Ketika elemen di dalam container dikenai float, mereka tidak menambah tinggi container, menyebabkan masalah layout.

Clearfix memungkinkan container untuk mengenali tinggi elemen-elemen yang dikenai float, sehingga mencegah masalah seperti overlap atau elemen yang tidak terlihat. Ini penting untuk menjaga stabilitas dan keakuratan layout.

Floats in CSS - Clearfix Hack

```
.clearfix::after {
  content: '';
  display: block;
  clear: both;
}
```

Contoh kode untuk clearfix ini menunjukkan bagaimana menambahkan pseudo-element `::after` pada container. Properti `clear: both;` digunakan untuk memastikan bahwa container mengakomodasi semua elemen yang difloat di dalamnya.

CSS Positioning

Positioning dalam CSS digunakan untuk mengontrol bagaimana elemen diletakkan di halaman. Ada beberapa jenis positioning:

- **Static**: adalah posisi default.
- **Relative**: memposisikan elemen relatif terhadap posisinya yang semula.
- **Absolute**: memposisikan elemen relatif terhadap posisi parent terdekatnya.
- **Fixed**: memposisikan elemen relatif terhadap viewport.

> **Catatan**: Elemen yang dikenai positioning tidak memengaruhi posisi elemen lainnya.

> **Viewports**: Viewport adalah area yang terlihat oleh pengguna di layar. Ini berbeda dari ukuran halaman web, yang dapat lebih besar atau lebih kecil dari viewport.

CSS Positioning

```
.relative {
  position: relative;
  top: 10px;
  left: 20px;
}

.absolute {
  position: absolute;
}

.fixed {
  position: fixed;
}
```

Units in CSS

CSS menawarkan berbagai unit ukuran untuk kebutuhan desain web yang berbeda. Berikut ini adalah ringkasan singkat dari masing-masing unit:

Unit	Deskripsi
Pixels (`px`)	Unit absolut; 1px sama dengan satu piksel pada layar.
Persentase (`%`)	Unit relatif; mengukur ukuran relatif terhadap elemen induk.
EM (`em`)	Ukuran relatif terhadap ukuran font elemen induk.
REM (Root EM; `rem`)	Ukuran relatif terhadap ukuran font elemen root (`<html>`).
Viewport Width (`vw`)	1% dari lebar viewport.
Viewport Height (`vh`)	1% dari tinggi viewport.

Units in CSS

Unit	Deskripsi
Viewport Min (`vmin`)	1% dari dimensi terkecil viewport (tinggi atau lebar).
Viewport Max (`vmax`)	1% dari dimensi terbesar viewport.
Centimeters (`cm`)	Unit berdasarkan ukuran fisik; jarang digunakan untuk layar.
Millimeters (`mm`)	Unit berdasarkan ukuran fisik; jarang digunakan untuk layar.
Inches (`in`)	Unit tradisional dalam tipografi; 1in = 96px.
Points (`pt`)	Unit tradisional dalam tipografi; 1pt = 1/72in.
Picas (`pc`)	Unit tradisional dalam tipografi; 1pc = 12pt.
Ex (`ex`)	Ukuran relatif terhadap tinggi x dari font elemen induk.
Ch (`ch`)	Ukuran relatif terhadap lebar karakter '0' dari font elemen induk.

Units in CSS - Pixels (px)

Pixels (px) adalah unit dasar dalam desain web. Mereka mengukur ukuran layar secara absolut, yang berarti bahwa 1px selalu sama dengan satu piksel pada layar. Unit ini paling sering digunakan untuk ukuran font, border, dan ukuran elemen lain yang memerlukan presisi yang ketat.

```
h1 {
  font-size: 24px;
}
```

Units in CSS - Percentages (%)

Persentase (%) merupakan unit relatif yang mengukur ukuran suatu elemen relatif terhadap ukuran elemen induknya. Unit ini sangat berguna untuk tata letak responsif, karena memungkinkan elemen untuk menyesuaikan ukurannya berdasarkan ukuran kontainer atau jendela browser

```
.container {
  width: 80%;
}
```

Units in CSS - Viewport Units (`vw` , `vh`)

Viewport width (`vw`) dan viewport height (`vh`) adalah unit relatif terhadap ukuran viewport browser.

`1vw` adalah **1% dari lebar viewport**, dan `1vh` adalah **1% dari tinggi viewport**.

Unit ini sangat berguna untuk membuat elemen yang ukurannya disesuaikan dengan ukuran layar browser, seperti hero images atau bagian header

```
header {
  height: 50vh;
  width: 100vw;
}
```

Units in CSS - `em` and `rem`

`em` adalah unit yang relatif terhadap ukuran font elemen induk.

`rem` (Root `em`) relatif terhadap ukuran font elemen root (`<html>`).

`em` biasanya digunakan untuk pengaturan ukuran font yang dinamis, sedangkan `rem` sering digunakan untuk menetapkan ukuran font secara global

```
html {
  font-size: 16px;
}
body {
  font-size: 1.5rem;
}
p {
  font-size: 1.2em;
}
```

Units in CSS - `em` Calculation

```
size in pixels = base size in pixels × value in em
```

Misalnya, jika ukuran font induk adalah `20px` dan Anda menetapkan ukuran font anak menjadi `1.5em`, maka ukuran font anak akan menjadi `30px` (`20px` × `1.5`).

```
.parent {
  font-size: 20px; /* Ukuran induk */
}

.child {
  font-size: 1.5em; /* 20px × 1.5 = 30px */
}
```

Units in CSS - `rem` Calculation

```
size in pixels = root size in pixels × value in rem
```

Jika ukuran font root adalah `16px` dan ukuran font diatur menjadi `1.5rem`, maka ukuran font akan menjadi `24px` (`16px` × `1.5`).

```css
html {
  font-size: 16px; /* Ukuran root */
}

.element {
  font-size: 1.5rem; /* 16px × 1.5 = 24px */
}
```

Styling Forms with CSS

Mengatur style form adalah bagian penting dalam desain web. Dengan CSS, kita dapat meningkatkan tampilan form input, tombol, dan elemen form lainnya, membuat mereka lebih menarik dan konsisten dengan keseluruhan desain situs.

```
input[type='text'] {
  border: 2px solid gray;
  border-radius: 4px;
}

input[type='submit'] {
  background-color: blue;
  color: white;
  padding: 10px;
}
```

Responsive Design

Desain responsif adalah teknik dalam web design yang memastikan tampilan web yang optimal di berbagai perangkat dan ukuran layar.

Media queries dalam CSS memegang peranan kunci dalam desain responsif, memungkinkan kita menerapkan aturan CSS berbeda tergantung pada karakteristik perangkat, seperti lebar layar, tinggi layar, orientasi (portrait atau landscape), dan resolusi.

Responsive Design

```
@media screen and (max-width: 600px) {
  .column {
    width: 100%;
  }
}
```

Contoh kode ini mengubah lebar kolom menjadi 100% ketika lebar layar kurang dari 600px, cocok untuk tampilan pada perangkat mobile.

Responsive Design: Media Queries & Breakpoints

Media Queries: Media queries menggunakan sintaks `@media` untuk menerapkan blok CSS pada kondisi tertentu. Mereka dapat menguji banyak fitur perangkat, tetapi yang paling umum adalah lebar (width) dan tinggi (height) viewport.

Breakpoint dalam desain responsif adalah titik di mana desain situs web berubah untuk mengakomodasi ukuran layar yang berbeda. Memilih breakpoint yang tepat sangat penting untuk memastikan bahwa situs web terlihat bagus dan berfungsi dengan baik di semua perangkat.

Responsive Design: Common Media Queries

Breakpoint untuk Perangkat Mobile:

```
@media screen and (max-width: 600px) {
  .container {
    width: 100%;
    padding: 10px;
  }
}
```

Breakpoint untuk Tablet:

```
@media screen and (min-width: 601px) and (max-width: 1024px) {
  .container {
    width: 80%;
  }
}
```

Responsive Design: Common Media Queries

Breakpoint untuk Desktop:

```
@media screen and (min-width: 1025px) {
  .container {
    width: 60%;
  }
}
```

Media Query untuk Orientasi:

```
@media screen and (orientation: landscape) {
  body {
    background-color: lightblue;
  }
}
```

Responsive Design: Common Media Queries

Device	Media Query
Mobile Phones (Small Devices)	`(max-width: 600px)`
Tablets (Medium Devices)	`(min-width: 601px) and (max-width: 1024px)`
Small Laptops / Large Tablets	`(min-width: 1025px) and (max-width: 1280px)`
Desktops (Extra Large Devices)	`(min-width: 1281px)`
High Resolution / 4K Displays	`(min-width: 1920px)`

Responsive Design: Additional Tips

Saat menggunakan media queries, penting untuk mengambil pendekatan **mobile-first**, di mana Anda mendesain untuk layar kecil terlebih dahulu, kemudian menambahkan media queries untuk layar yang lebih besar.

Ini membantu dalam memastikan bahwa situs web Anda dapat diakses dengan baik pada perangkat mobile.

Media Types in CSS Media Queries

Type	Description	Example
Screen	Used for computer screens, tablets, and phones.	`@media screen { ... }`
Print	Used for content that will be printed.	`@media print { ... }`
All	Applies to all types of media.	`@media all { ... }`
Speech	Intended for screen readers and other assistive devices.	`@media speech { ... }`

Media Types in CSS Media Queries

Dalam prakteknya, `screen` dan `print` adalah tipe media yang paling sering digunakan. Tipe media `screen` digunakan untuk gaya yang diterapkan saat melihat di layar, sementara `print` digunakan untuk gaya yang spesifik ketika konten dicetak, seperti menghilangkan elemen latar belakang atau mengubah warna untuk hemat tinta.

Typography in CSS

Menggunakan web fonts seperti Google Fonts atau Bunny Fonts dapat meningkatkan tampilan teks di situs web kita.

Kita dapat memilih dan menggunakan berbagai font dan memasukkannya ke dalam CSS tanpa perlu menyimpan file font secara lokal.

```
@import url('https://fonts.googleapis.com/css2?family=Roboto:wght@400;700&display=swap');

@import url(https://fonts.bunny.net/css?family=Roboto:400,700);
```

Typography in CSS

Menggunakan font yang telah di-import:

```
body {
  font-family: 'Roboto', sans-serif;
}

h1 {
  font-weight: 700; /* Bold */
}

p {
  font-weight: 400; /* Regular */
}
```

Pseudo Elements

Pseudo elements dalam CSS memungkinkan kita untuk menambahkan style ke bagian tertentu dari elemen. Misalnya, kita dapat mengatur style untuk baris pertama atau huruf pertama dalam paragraf.

Pseudo elements yang paling umum adalah `::before` dan `::after`, yang memungkinkan kita untuk memasukkan konten sebelum atau setelah isi elemen.

Konten ini tidak ada dalam HTML, tapi ditambahkan melalui CSS. Hal ini berguna untuk dekorasi, seperti ikon atau elemen grafis lainnya, tanpa perlu menambahkan markup tambahan ke HTML.

Pseudo Elements

```
.element::before {
  content: '★';
  color: blue;
}

.element::after {
  content: '✓';
  color: green;
}
```

Pseudo Elements

Pseudo Element	Description
`::before`	Menambahkan konten sebelum isi elemen.
`::after`	Menambahkan konten setelah isi elemen.
`::first-letter`	Mengatur style huruf pertama dari sebuah elemen blok, seperti paragraf.
`::first-line`	Mengatur style baris pertama dari sebuah elemen blok.
`::selection`	Mengatur style teks yang dipilih oleh pengguna.
`::placeholder`	Mengatur style teks placeholder dalam elemen input atau textarea.

Pseudo Elements

Pseudo Element	Description
`::marker`	Mengatur style marker dari list (misalnya bullet atau nomor dalam list).
`::backdrop`	Menerapkan style pada latar belakang elemen (full-screen).

Penting untuk dicatat bahwa tidak semua browser mendukung semua pseudo elements ini, jadi disarankan untuk memeriksa kompatibilitas browser sebelum mengimplementasikannya.

Custom Properties

Custom properties, atau sering disebut CSS variables, adalah cara yang sangat fleksibel untuk mengelola style.

Dengan custom properties, kita dapat menentukan nilai yang dapat digunakan kembali.

```
:root {
  --primary-color: #4caf50;
  --secondary-color: #ff9800;
}
```

Custom Properties

Custom properties digunakan dengan mendeklarasikannya sebagai nilai dari properti CSS lain.

```
body {
  background-color: var(--primary-color);
  color: var(--secondary-color);
}
```

Dalam contoh ini, `background-color` dan `color` dari elemen `body` diatur menggunakan custom properties.

Mengubah nilai dari `--primary-color` dan `--secondary-color` akan mempengaruhi semua elemen yang menggunakan variabel-variabel ini.

Advantages of Custom Properties

Custom properties menawarkan beberapa keuntungan signifikan dalam CSS:

- **Pemeliharaan:** Mudah untuk memperbarui nilai-nilai seperti warna, padding, atau margin di seluruh situs.
- **Konsistensi:** Membantu menjaga konsistensi dalam desain.
- **Fleksibilitas:** Memungkinkan lebih banyak kreativitas dan adaptabilitas dalam desain.
- **Interoperabilitas dengan JavaScript:** Memudahkan interaksi dengan JavaScript, memungkinkan manipulasi style secara dinamis.

Flexbox in CSS

Flexbox adalah model tata letak CSS yang memungkinkan kita untuk mengatur item dalam sebuah container dengan mudah dan fleksibel. Flexbox menyediakan properti-properti yang dapat digunakan untuk mengatur alignment, perataan, dan pembagian ruang di antara item.

```
.flex-container {
  display: flex;
  justify-content: space-between;
}
```

Flexbox in CSS: Flex Directions

Properti `flex-direction` dalam Flexbox menentukan arah utama di mana item flex akan diletakkan dalam container. Ada empat nilai utama:

- `row` mengatur item secara horizontal
- `column` secara vertikal.
- `row-reverse` dan `column-reverse` menempatkan item dalam urutan terbalik.

```
.container {
  display: flex;
  flex-direction: row; /* or column, row-reverse, column-reverse */
}
```

Flexbox in CSS: Justify Content Options

Properti `justify-content` dalam Flexbox digunakan untuk mengatur penjajaran item flex di sepanjang sumbu utama container.

Opsi-opsinya termasuk `flex-start`, `flex-end`, `center`, `space-between`, `space-around`, dan `space-evenly`. Masing-masing opsi ini memiliki efek yang berbeda terhadap penyebaran dan penjajaran item dalam container.

Misalnya, `flex-start` menempatkan item di awal sumbu utama, `flex-end` di akhir, `center` di tengah, `space-between` memberikan ruang yang sama antar item, `space-around` memberikan ruang di sekitar setiap item, dan `space-evenly` mendistribusikan ruang secara merata antara semua item.

Flexbox in CSS: Justify Content Options

```
.container {
  display: flex;
  justify-content: flex-start;
}
```

Dalam contoh ini, `justify-content` digunakan untuk mengatur penjajaran item dalam container flex sesuai dengan opsi yang dipilih. Nilai `flex-start` menempatkan item di awal sumbu utama.

Flexbox in CSS: The Flex Property

Properti `flex` adalah properti singkat yang menggabungkan `flex-grow`, `flex-shrink`, dan `flex-basis`.

- `flex-grow` mengontrol kemampuan item untuk tumbuh jika ada ruang tambahan.
- `flex-shrink` mengatur bagaimana item menyusut jika tidak ada cukup ruang.
- `flex-basis` menetapkan ukuran awal item sebelum distribusi ruang tambahan.

Grid

CSS Grid adalah sistem tata letak yang memungkinkan kita untuk membuat tata letak kompleks dengan mudah dan efisien. Grid membagi container menjadi kolom dan baris, dan kita dapat menempatkan elemen di dalam grid tersebut.

Defining a Grid:

```
.container {
  display: grid;
}
```

Grid

Grid Columns and Rows:

```
.container {
  grid-template-columns: 1fr 2fr;
}
```

Grid Gaps: Menetapkan jarak antar kolom dan baris menggunakan.

```
.container {
  grid-gap: 10px;
  grid-row-gap: 10px;
  grid-column-gap: 10px;
}
```

Exercise: Build Your Own Personal Web Page

- Develop a personal webpage using HTML, CSS, and potentially JavaScript.
- Design the page to be responsive across devices using media queries.
- Include the following sections:
  - Header: Name and tagline using Flexbox.
  - About Me: Bio section.
  - Skills: List of abilities using CSS Grid.
  - Projects Portfolio: Showcase of your work.
  - Contact: Professional links or a form.