CSS-Grid

Obyektif:
- Memahami konsep CSS-Grid

Daftar Isi

CSS Grid

Secanggih apapun pemrograman JavaScript, namun pemakai (user) menilai aplikasi tetap dari sisi User Interface - User Experience (UIUX). Kunci dari keberhasilan UIUX tersebut ada pada CSS, terutama terhadap desain layout yang dibuat.

CSS3 menawarkan banyak fitur baru dalam membuat Layout, antara lain dengan menggunakan FlexBox dan terakhir menggunakan CSS-Grid. FlexBox merupakan desain satu dimensi sedangkan CSS-Grid menggunakan dua dimensi, artinya Grid berkonsentrasi atas baris dan kolom (row - column).

Sebagai Developer MWS, kedua konsep ini harus dikuasai dengan baik. Pada Udacity elearning baru dijelaskan tentangflexbox, sedangkan CSS-Grid belum beredar. Hal ini karena CSS-Grid masih relatif baru dan belum semua browser mendukung CSS-Grid.

Tutorial berikut menjelaskan awal penggunaan CSS-Grid untuk dapat memahami cara pengguaannya secara praktis. Contoh tersebut tentunya tidak menggali seluruh fitur CSS-Grid yang ada, namun menjelaskan esensi CSS-Grid sehingga desain layout sudah dapat terbentuk dengan baik.

Front-End Developer wajib memahami CSS-Grid!

Layout Grid

Berikut adalah sebuah container yang menggunakan Grid, membungkus elemen , yang disebut sebagai children. Dalam hal ini container adalah **Parent**, dan elemen div dibawahnya adalah **children**.

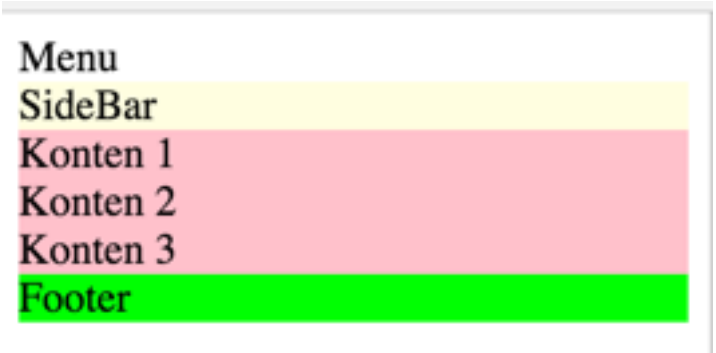
Parent menentukan template, dan **Children** memposisikan diri sesuai dengan penempatan pada template tersebut.

```
<div class="container">
  <div>Menu</div>
  <div>SideBar</div>
  <div>Konten 1</div>
  <div>Konten 2</div>
  <div>Konten 3</div>
  <div>Footer</div>
</div>
```

Menciptakan container menjadi Grid (parent) cukup mudah, dengan menggunakan CSS display: grid.

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

Untuk membedakan setiap komponen div, diberikan nama *menu, sidebar, konten* dan *footer*, kemudian diberi warna untuk memudahkan membedakan komponen tersebut.



File: grid1.html

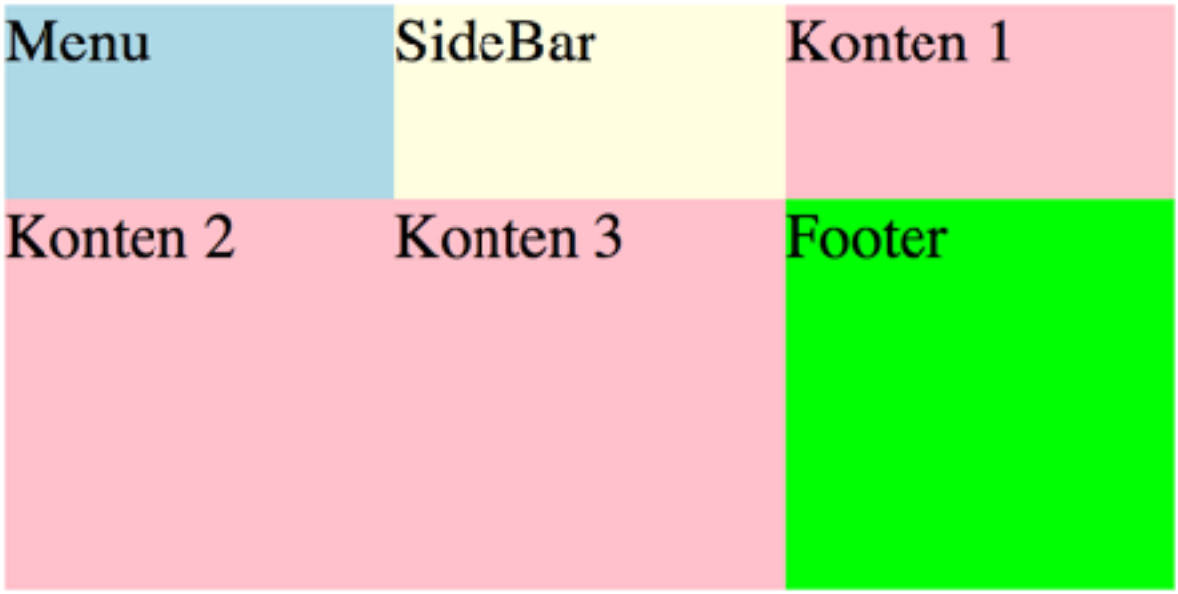
```
<html>
<head>
<style>
   .container {
      display: grid;
   }

   .menu {
      background: lightblue;
   }
   .sidebar {
      background: lightyellow;
   }
   .konten {
      background: pink;
   }
   .footer {
      background: lime;
   }
</style>
</head>
<body>
<div class="container">
  <div class='menu'>Menu</div>
  <div class='sidebar'>SideBar</div>
  <div class='konten'>Konten 1</div>
  <div class='konten'>Konten 2</div>
  <div class='konten'>Konten 3</div>
  <div class='footer'>Footer</div>
</div>
</body>
</html>
```

Kolom dan Baris

Untuk membuat grid menjadi 2 dimensi, maka diperlukan definisi kolom dan baris . Berikut pada ***parent*** dibuat template untuk kolom dan baris.

```
.container {
    display: grid;
    grid-template-columns: 100px 100px 100px;
    grid-template-rows: 50px 100px;
}
```



Pada contoh ditulis 3 kolom @ 100 px, dan 2 baris - masing-masing 50 px.

Template tersebut dapat diubah misalnya dengan mengunakan besaran piksel yang berbeda, 3 buah kolom dengan 2 buah baris.

Salin program grid1.html menjadi grid2.html, dan lakukan perubahan sesuai dengan nilai yang baru.

```
.container {
    display: grid;
    grid-template-columns: 200px 100px 50px;
    grid-template-rows: 75px 25px;
}
```

Periksa sendiri hasilnya, apakah sesuai dengan pemahaman.

Penempatan Elemen

Berikut adalah penempatan elemen di grid, seperti contoh adalah 3x3 grid menggunakan markup seperti sebelumnya. Untuk memudahkan visualisasi, gunakan grid-gap.

```
.container {
    display: grid;
    grid-template-columns: 100px 100px 100px;
    grid-template-rows: 100px 100px 100px;
    grid-gap: 4px;
}
```

Periksa hasilnya.

Ternyata harusnya 3x3 Grid, tapi karena hanya ada 6 elemen, maka yang terlihat hanya elemen tersebut.

Untuk memposisikan elemen menguasai kolom atau baris, digunakan **grid-column-start** dan **grid-column-end.**

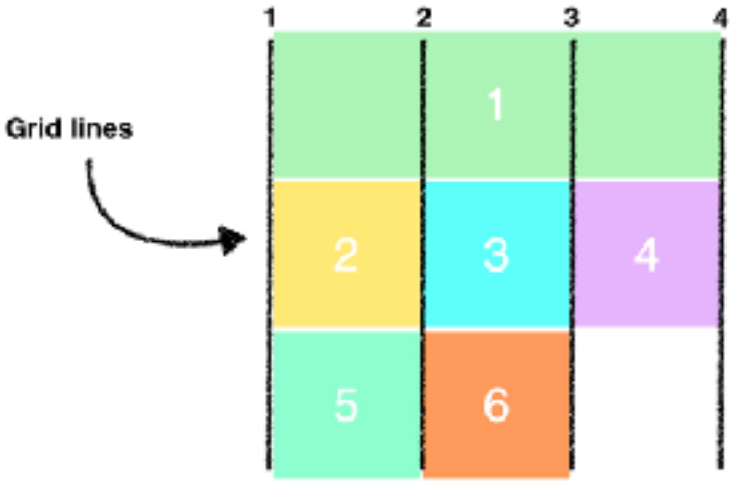
```
.menu {
    background: lightblue;
    grid-column-start: 1;
    grid-column-end: 4;
}
```

Agar menu tidak terlalu tinggi, ganti t*emplate row* dengan **30px.**

```
.container {
    display: grid;
    grid-template-columns: 100px 100px 100px;
    grid-template-rows: 30px 100px 100px;
    grid-gap: 4px;
}
```

Baris Grid

Penghitungan nomor kolom dan baris dimulai dengan angka 1:



Penulisan grid-column-start:1 , dan grid-column-end: 4;

Demikian juga dengan baris, grid-row-start :1 dan grid-row-end: 4.

Penulisan ini:

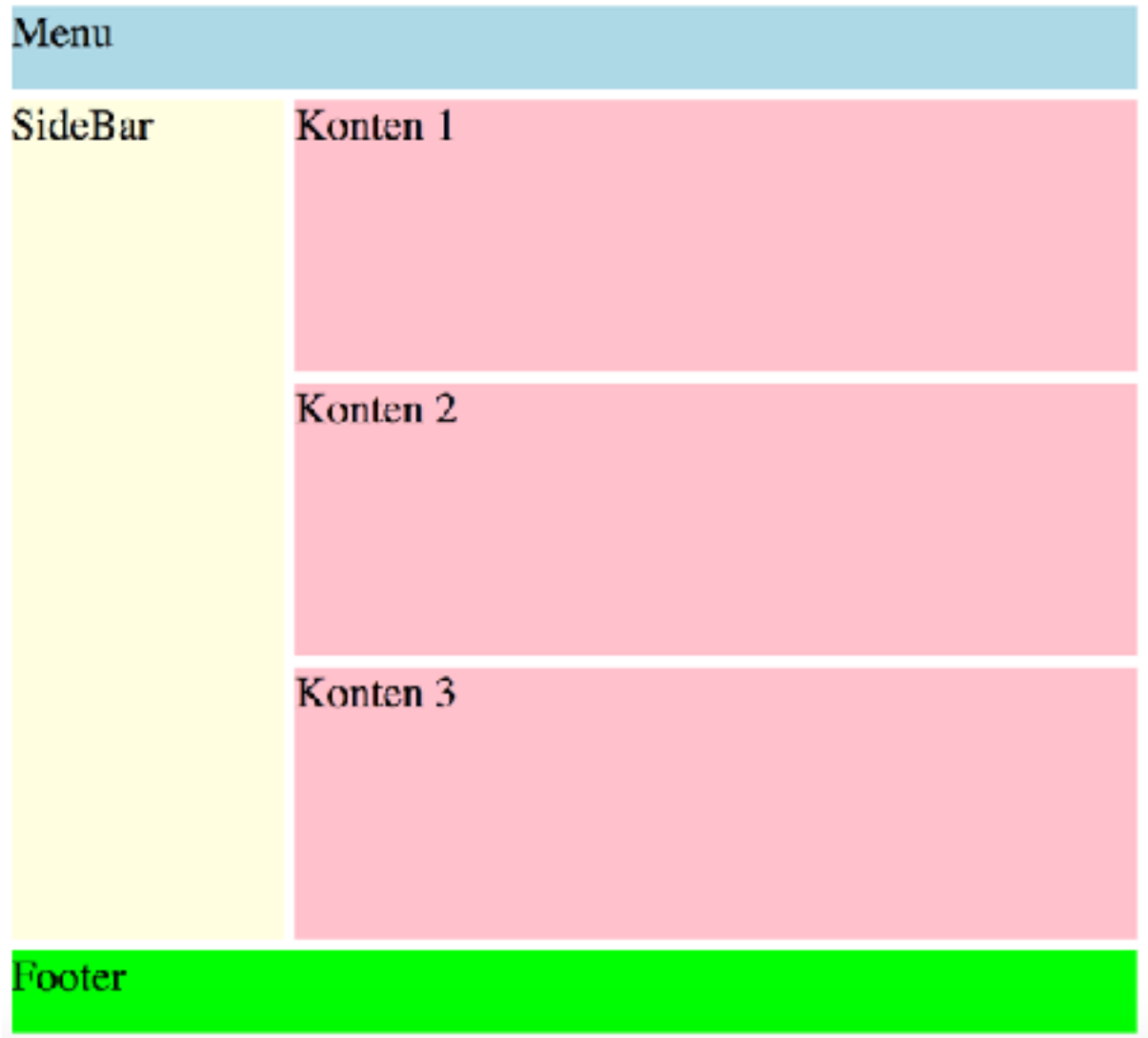
```
.menu {
    background: lightblue;
    grid-column-start: 1;
    grid-column-end: 4;
}
```

Dapat disederhanakan menjadi:

```
.menu {
    background: lightblue;
    grid-column: 1/4;
}
```

1/4 dibaca sebagai grid-column dimulai dari kolom 1 dan berakhir sebelum kolom nomor 4.

Upayakan agar sidebar menguasai 1 baris kolom, kemudian diisi dengan konten dan diakhiri dengan footer, dibuat konfigurasi seperti pada tugas berikut:



Eksperimen:

Buat desain user-interface dengan menggunakan 4 kolom dan 5 baris (dimensi 4x5)

```
.container {
      display: grid;
      grid-template-columns:  100px 100px 100px 100px;
      grid-template-rows: 30px 100px 100px 100px 30px;
      grid-gap: 4px;
   }
```

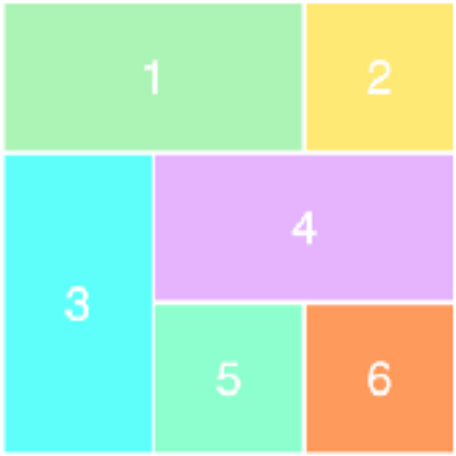
Solusi:

```
<html>
<head>
<style>
   .container {
      display: grid;
      grid-template-columns:  100px 100px 100px 100px;
      grid-template-rows: 30px 100px 100px 100px 30px;
      grid-gap: 4px;
   }
   .menu {
      background: lightblue;
      grid-column-start: 1;
      grid-column-end: 5;
   }
   .sidebar {
      background: lightyellow;
      grid-row-start: 2;
      grid-row-end: 5;
      grid-column-end: 1/2;
   }
   .konten {
      background: pink;
      grid-column: 2/5;
   }
   .footer {
      background: lime;
      grid-column: 1/5;
      grid-row : 5/5;
   }
</style>
</head>
<body>
<div class="container">
  <div class='menu'>Menu</div>
  <div class='sidebar'>SideBar</div>
  <div class='konten'>Konten 1</div>
  <div class='konten'>Konten 2</div>
  <div class='konten'>Konten 3</div>
  <div class='footer'>Footer</div>
</div>
</body>
</html>
```

Catatan: program diatas belum responsif

Tugas: (opsional)

Buat Figur berikut ini dengan menggunakan css-grid.



Reponsive CSS-Grid

Agar responsive, lebar dari grid-row ditentukan dengan width: 100% .

```
<meta name="viewport" content="width=device-width, initial-
scale=1">
<style>
   .container {
      display: grid;
      width: 100%;
      grid-template-columns:  1fr 1fr 1fr 1fr;
      grid-template-rows: 30px 100px 100px 100px 30px;
      grid-gap: 4px;
   }
...
</style>
```

Kolom kali ini tidak menggunakan ukuran pixel, tapi menggunakan fr (fraction). 1fr adalah satu fraction, yang merupakan 1 grid. Dengan demikian :

```
grid-template-columns:  1fr 1fr 1fr 1fr;
```

Membagi kolom dalam 4 fraction yang sama.

Pada contoh diatas, maka jumlah kolom adalah 4, dan baris dihitung dengan 1/5, sedangkan jumlah row (baris) adalah 5.

```
.menu {
    background: lightblue;
    grid-column-start: 1;
    grid-column-end: 5;
 }
 .sidebar {
    background: lightyellow;
    grid-row-start: 2;
    grid-row-end: 5;
    grid-column-end: 1/2;
 }
.konten {
    background: pink;
    grid-column: 2/5;
 }
 .footer {
    background: lime;
    grid-column: 1/5;
    grid-row : 5/6;
 }
```

Penyederhanaan Tulisan dengan repeat

CSS instruksi seperti ini:

```
grid-template-columns:  1fr 1fr 1fr 1fr;
```

dapat disederhanakan dengan menggunakan ***repeat(brp-kali, 1fr)*** .

```
grid-template-columns: repeat(4, 1fr);
```

Hal ini sangat membantu, bila jumlah kolom sangat banyak (misalnya 20).