MODUL 5

RESPONSIVE WEB DESIGN

A. TUJUAN PRAKTIKUM

1. Mahasiswa mampu menerapkan konsep desain responsif ke dalam website.
2. Mahasiswa mampu menghasilkan desain web responsif.
3. Mahasiswa dapat menerapkan HTML dan CSS tingkat lanjut.

B. ALOKASI WAKTU 1 x 50 menit

C. DASAR TEORI

1. Desain Web Responsif

Desain web responsif (responsive web design atau RWD) adalah sebuah konsep pengembangan aplikasi web yang memungkinkan layout untuk menyesuaikan dengan resolusi layar pengguna (viewport). Desain ini menjadikan aplikasi web mampu menyediakan tampilan yang optimal untuk semua ukuran layar pengguna. Jadi, tidak lagi pengguna yang harus menyesuaikan perangkat yang digunakan (seperti pendekatan lawas).

Konsep RWD di dalam pengembangan aplikasi web direalisasikan melalui fitur CSS3. Melalui fitur media query, pengembang aplikasi dapat menyediakan aturan-aturan CSS yang dapat diberlakukan untuk variasi layar pengguna. Secara sederhana, media query adalah filter yang akan diberlakukan terhadap halaman web sesuai dengan identifikasi browser.

Penerapan RWD sudah menjadi kebutuhan dalam pengembangan aplikasi web modern seperti saat ini. Bagaimanapun, pengguna aplikasi web saat ini tidak hanya memanfaatkan komputer desktop atau laptop ketika mengakses halaman web, namun sangat dinamis dan mencakup peralatan-peralatan bergerak seperti komputer tablet dan telepon pintar (smartphone). Bagian ini akan memberikan gambaran mengenai penerapan konsep RWD di dalam aplikasi web.

2. Langkah Pengembangan Desain Website Responsive

Dalam proses pengembangan desain website responsive, terdapat 3 langkah yang harus diperhatikan :

a. Mendefinisikan Meta Tag Viewport

Mobile browser biasanya akan mengatur skala halaman HTML sesuai lebar viewport, yang akhirnya website dapat tampil pada layar mobile. Meta tag viewport ini digunakan untuk mereset ulang dan untuk memberitahukan kepada browser untuk menggunakan lebar perangkat sebagai acuan lebar viewport serta menonaktifkan skala awal. Anda bisa menyertakan meta tag seperti berikut ini dibagian <HEAD>

```
<html>
    <head>
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>

    </body>
    </body>
</html>
```

Berikut beberapa properti dari meta viewport dan fungsinya:

- **Width**

  Mengontrol ukuran lebar dari viewport. Nilainya dapat diatur secara spesifik berdasarkan pixel seperti `width=600` atau nilai khusus seperti `device-width`, yang berarti 100vw, atau 100% dari lebar viewport. Minimum: `1`. Maximum: `10000`.

- **Height**

  Mengontrol ukuran tinggi dari viewport. Nilainya dapat diatur secara spesifik berdasarkan pixel seperti `height=600` atau nilai khusus seperti `device-height`, yang berarti 100vh, atau 100% dari lebar viewport. Minimum: `1`. Maximum: `10000`.

- **Initial-scale**

  Mengontrol level perbesaran halaman ketika pertama kali di load. Minimum: `0.1`. Maximum: `10`. Default: `1`.

- **Minimum-scale**

  Mengizinkan seberapa besar level zoom out dalam halaman. Minimum: `0.1`. Maximum: `10`. Default: 0.1.

- **Maximum-scale**

  Mengizinkan seberapa besar level zoom in dalam halaman. Minimum: `0.1`. Maximum: `10`. Default: 10.

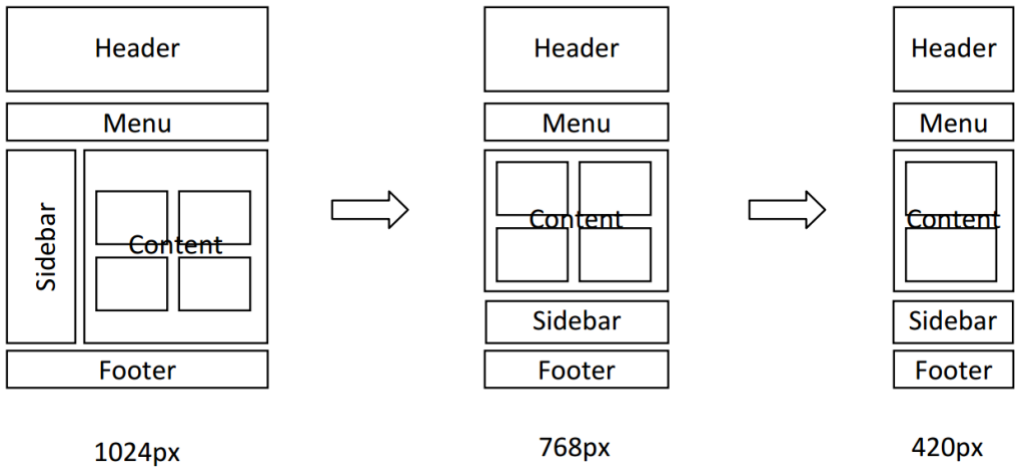
- **User-scalable**

  Mengizinkan proses zoom out maupun zoom in dalam halaman. Nilainya antara: `0, 1, yes` atau `no`. Default: 1, atau yes.

b. Menentukan Layout dan Struktur HTML

Sebuah website biasanya terdiri dari elemen header, menu, sidebar, content, dan footer. Tinggi dan lebar masing-masing elemen sebaiknya direncanakan dulu sesuai kebutuhan sebelum melakukan coding script.

Berikut beberapa layout dan struktur yang biasa digunakan saat proses responsive diukur dari maksimal lebar layar perangkat :



c. Menetapkan Resolusi layar

Meskipun di luar sana terdapat banyak sekali perangkat dengan berbagai ukuran, namun terdapat standar yang disebut sebagai aspect ratio untuk memudahkan pengembang dalam menentukan resolusi layer yang dibangun. Berikut beberapa nama serta ukuran layar yang sudah umum.

Nama	Rasio	Panjang (piksel)	Lebar (piksel)
VGA	4:3	640	480
SVGA	4:3	800	600
XGA	4:2	1024	768
WXGA / HD	16:9	1280	720
HD	~16:9	1360	768
WXGA+	16:10	1440	900
FHD	16:9	1920	1080
WUXGA	16:10	1920	1200
WQHD	16:9	2560	1440
WQXGA	16:10	2560	1600
4K UHD	16:9	3840	2160
8K UHD	16:9	7680	4320

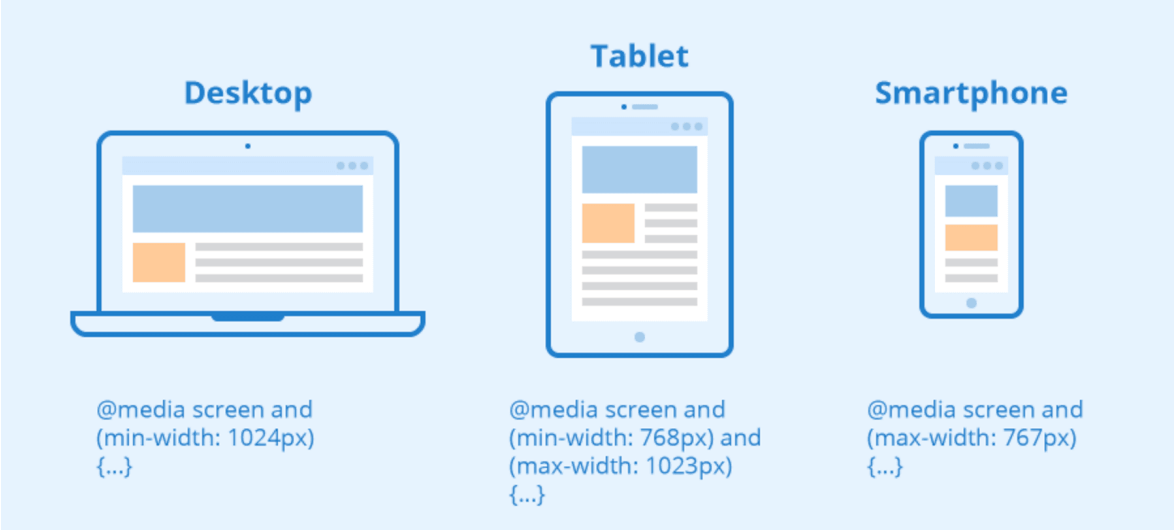
3. Teknik Membuat Responsive Web

a. Media Query CSS

Media Query merupakan perintah di CSS3 untuk memerintahkan browser untuk proses rendering agar mengikuti ukuran sesuai script yang dituliskan. Contoh script media query adalah sebagai berikut :

```
/* Jika ukuran layar 680px atau kurang dari itu */
@media screen and (max-width:768px) {
    #content {
        width: auto;
        float: none;
    }

    #sidebar{
        width: auto;
        float: none;
    }
}
```

Dengan memperhatikan besaran pada masing-masing perangkat, maka kita dapat menentukan CSS yang berbeda pada perangkat yang memiliki ukuran layar yang berbeda.



Saat ini media query sudah disupport oleh beragam browser. Menurut web https://caniuse.com/css-mediaqueries, berikut adalah browser yang support media query

Menulis media query dapat digabungkan dengan beberapa ukuran lain secara bersamaan. Penggunaan logika seperti and dan or sangat dimungkinkan di dalam penulisan media query.

Logika AND

```
@media (min-width: 600px) and (max-width: 800px)
```

Logika OR

```
@media (max-width: 600px), (min-width: 800px)
```

Logika Complement

```
@media screen and not (max-width: 600px), (min-width:
800px)
```

Dalam penulisan media query ada dua buah pendekatan: graceful degradation dan mobile first. Graceful Degradation apabila penulisan media query diawali dari tampilan desktop, sementara mobile first melakukan penulisan CSS dengan mengutamakan tampilan mobile terlebih dahulu. Berikut adalah penulisan CSS untuk graceful degradation

```
/* Default styles first for the largest screen; then media
queries for smaller screens */ ...
@media screen and (max-width: 1400px) {...}
@media screen and (max-width: 1000px) {...}
@media screen and (max-width: 600px) {...}
@media screen and (max-width: 400px) {...}
```

Selanjutnya berikut adalah contoh penulisan untuk mobile-first.

```
/* Default styles first for the smallest screen; then media
queries for larger screens*/ ...
@media screen and (min-width: 400px) {...}
@media screen and (min-width: 600px) {...}
@media screen and (min-width: 1000px) {...}
@media screen and (min-width: 1400px) {...}
```

b. Flex Box CSS

Flex Box adalah model layout 1 dimensi yang dapat mengatur jarak dan alignment antar item dalam sebuah container. 1 dimensi artinya teknik pengaturan baris saja atau kolom saja. Flexbox menawarkan cara yang efektif untuk menyusun, mensejajarkan dan mendistribusikan jarak antar item di dalam sebuah container, meskipun ukurannya tidak kita ketahui dan bersifat dinamis.

```
.container {
    display: flex; /* atau inline-flex */
}
```

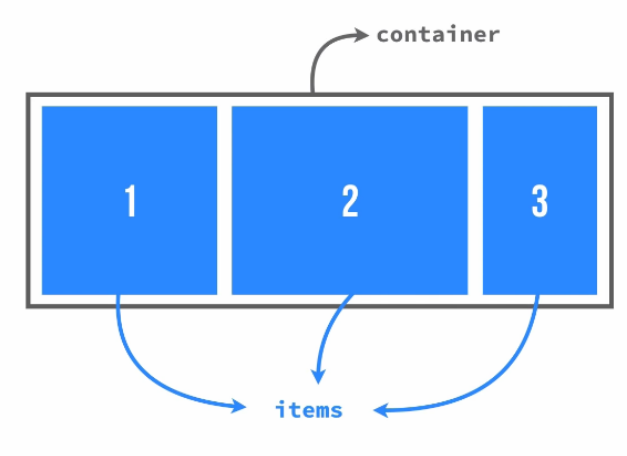
Berikut beberapa elemen yang perlu diketahui :

a. Container

Tempat meletakkan elemen yang bersifat sebagai pembungkus / parent

b. Items

Elemen yang akan dikenai konsep fleksibel box / child

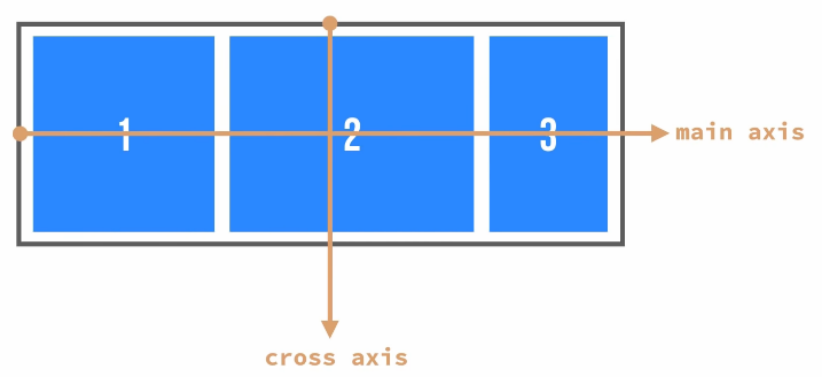


c. Main axis

Sumbu utama dari sebuah container yang menentukan urutan dari penempatan items secara horizontal.

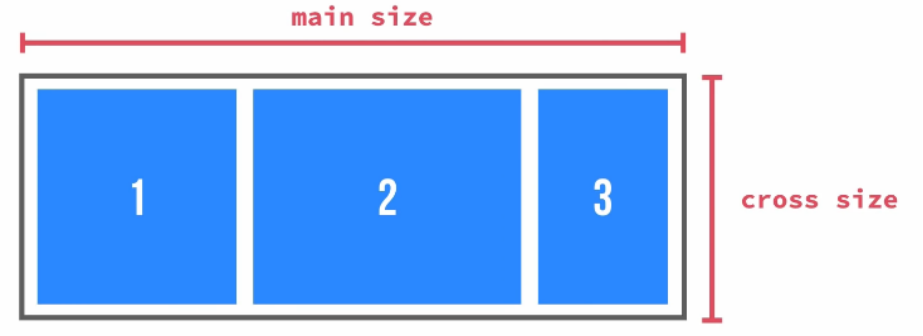
d. Cross axis

Sumbu pendukung dari sebuah container yang menentukan urutan dari penempatan items secara vertical.



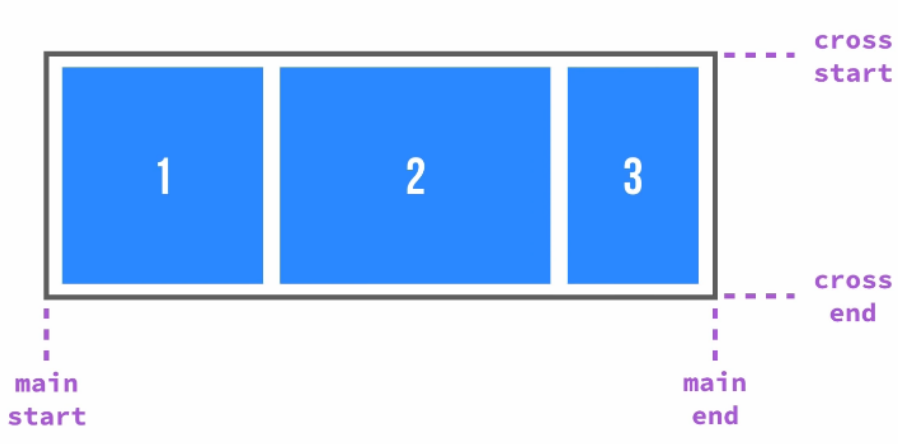
e. **Main size**
Ukuran panjang dari container yang menjadikan items bersifat relative secara horizontal.

f. **Cross size**
Ukuran lebar dari container yang menjadikan items bersifat relative secara vertical.



g. **Main start / end**
Mulai dan berakhirnya items yang disimpan dalam container secara horizontal

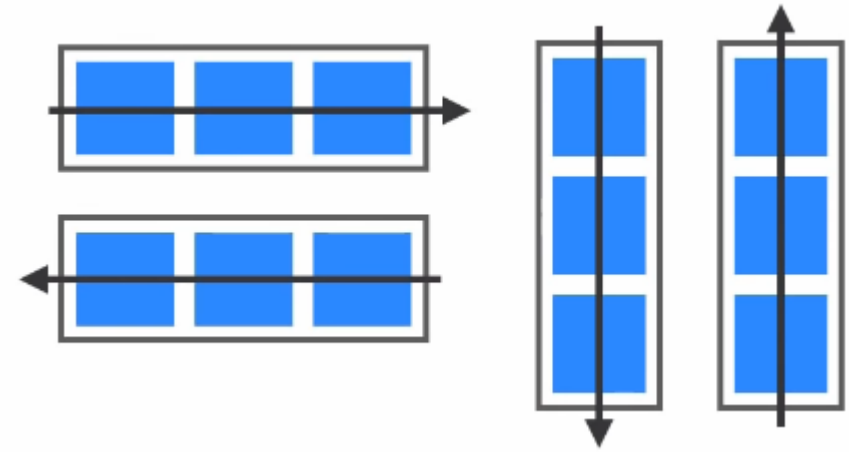
h. **Cross start / end**
Mulai dan berakhirnya items yang disimpan dalam container secara vertical



Flex-direction

Flex Box sebagai sarana alignment 1 dimensi memiliki flex-direction untuk menentukan ke arah mana items tersebut akan diurutkan di dalam container. Adapun value dari property flex-direction adalah sebagai berikut :

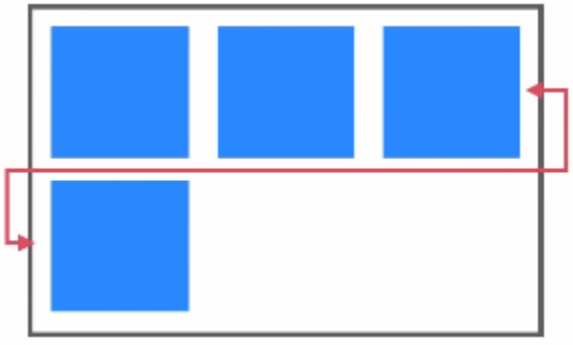
```
.container {
  flex-direction: row | row-reverse | column | column-reverse;
}
```



Flex-wrap

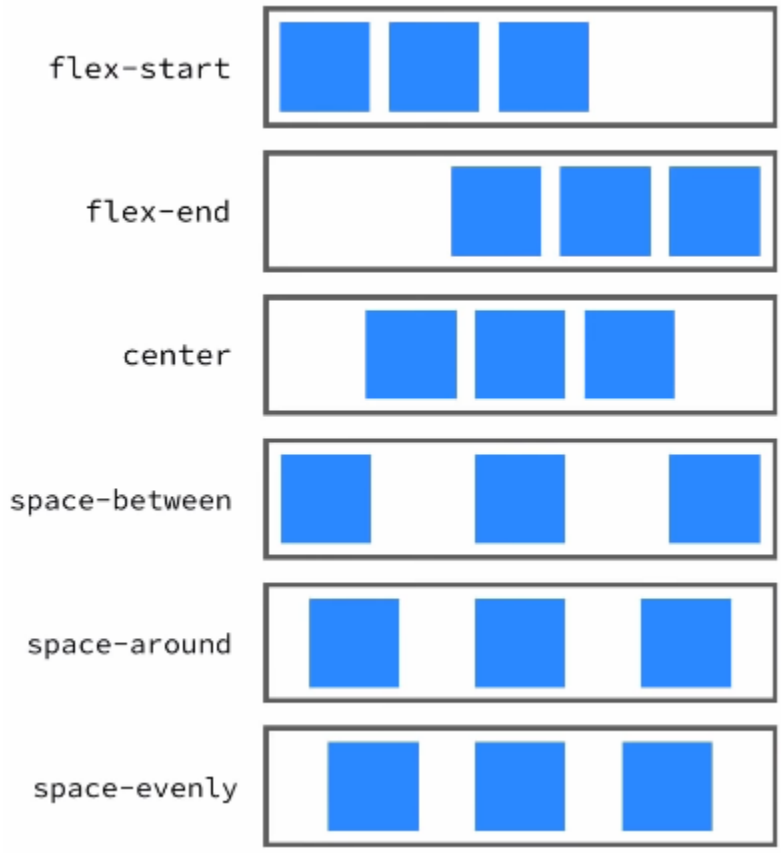
Flex-wrap akan membuat items yang berada di dalam satu baris pada container menjadi pindah ke baris selanjutnya. Pada umumnya semua items yang ada di dalam container flex box akan berada pada satu baris meskipun ukurannya sudah tidak cukup.

```
.container {
  flex-wrap: nowrap | wrap | wrap-reverse;
}
```



Justify-content

Justify-content merupakan property yang membantu penataan jarak antar items yang ada di dalam container terhadap main axis.

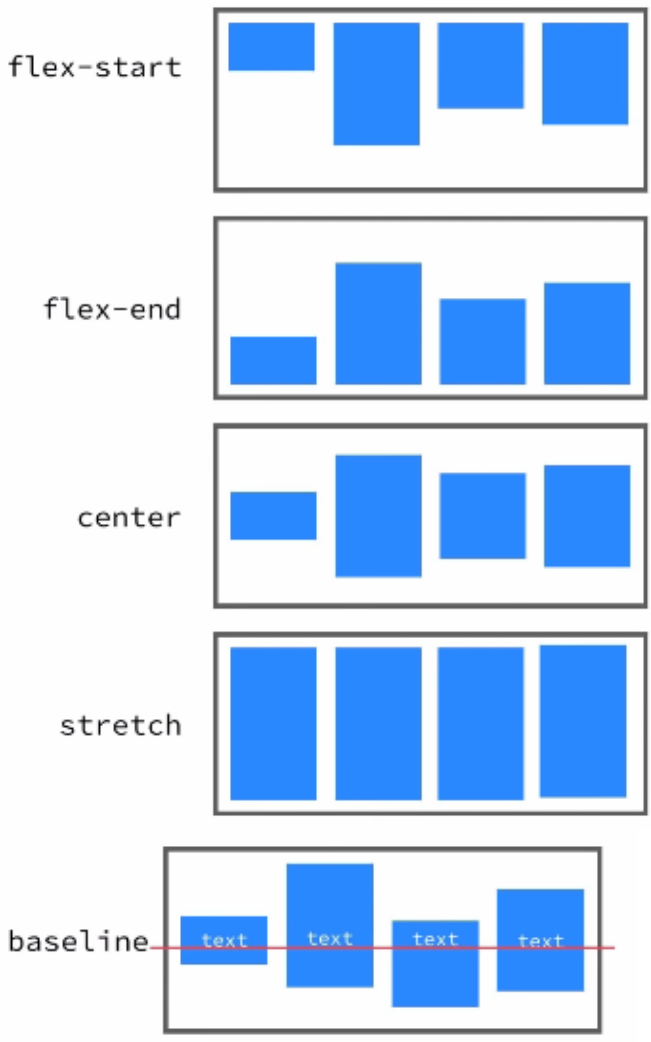



```
.container {
  justify-content: flex-start | flex-end | center
      space-between | space-around | space-evenly;
}
```

Align-items

Align-items merupakan property yang membantu pengaturan kesejajaran antar items yang ada di dalam container terhadap cross axis.

```
.container {
  align-items: flex-start | flex-end | center
               stretch | baseline;
}
```



D. LATIHAN

1. Menggunakan Media Query

Konsep media query ini mirip dengan media type yang diperkenalkan oleh CSS 3, yakni ketika kita menyediakan tampilan khusus untuk dokumen tertentu. Kita bisa merasakan bagaimana mudahnya memisahkan style untuk halaman utama dan halaman khusus lainnya.

Sebagai contoh awal untuk mengetahui cara kerja media query, kita akan menyediakan aturan khusus pada perangkat smartphone. Dalam hal ini kita tetapkan misalkan ukuran lebar layar smartphone adalah 428 piksel (contoh iPhone 13). Aturan yang kita terapkan sangat sederhana, yakni memberi background warna kuning jika layar teridentifikasi berukuran maksimal 428 piksel.

```
<!DOCTYPE html>
<html>
<head>
    <title>Menggunakan media Query CSS3</title>
    <meta name="viewport" content="width-device-width, initial-scale=1.0">

    <style>
        .smartphone{
            display: none;
        }

        @media only screen and (max-width: 428px){
            body{
                background-color: yellow;
            }

            .smartphone{
                display: block;
            }
        }
    </style>

</head>
<body>
    <h3>Menggunakan Media Query CSS 3</h3>
    <div class="smartphone">
        <h2>Terdeteksi ukuran smartphone</h2>
    </div>

</body>
</html>
```

Simpan kode tersebut dengan nama **latihan1a.html.** Buka kode HTML di browser desktop dan hasilnya akan terlihat seperti pada gambar di bawah ini.

Selanjutnya, resize lebar browser hingga merepresentasikan ukuran layar smartphone. Seharusnya hasilnya akan terlihat seperti pada Gambar di bawah ini.

Terlihat bahwa aturan CSS media query yang kita tetapkan sudah bekerja dan teridentifikasi dengan baik.

Dengan cara serupa, munculkan logo UNS (file cari sendiri) yang hanya nampak ketika ukuran layar lebih dari 1024 piksel dan simpan dengan nama latihan1b.html!

2. Desain Halaman Web Responsif

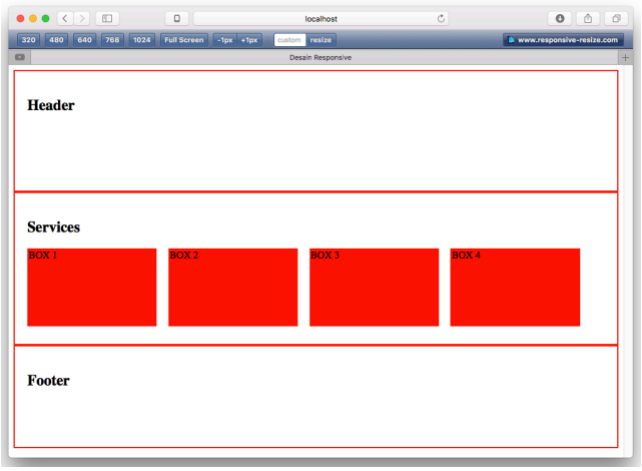
Setelah memahami konsep desain web responsif, selanjutnya kita bisa menerapkan di dalam pembuatan aplikasi web. Ada tiga sasaran perangkat yang akan kita identifikasi, yaitu komputer desktop, tablet, dan smartphone.

```
<!DOCTYPE html>
<html>
<head>
    <title>Desain Responsif</title>
    <meta name="viewport" content="width-device-width, initial-scale=1.0">

    <style type="text/css">
        #wrapper{
            margin: auto;
            width: 100%;
        }

        #header{
            height: 150px;
            padding: 20px;
            border: 2px solid red;
        }

        #nav{
            margin: auto;
            padding: 20px;
            height: 200px;
            border: 2px solid red;
        }

        .box{
            float: left;
            width: 22%;
            margin: 0 2% 0 0;
            height: 120px;
            background: red;
            border: 2px solid red;
        }

        #footer{
            padding: 20px;
            height: 120px;
            border: 2px solid red;
        }

```

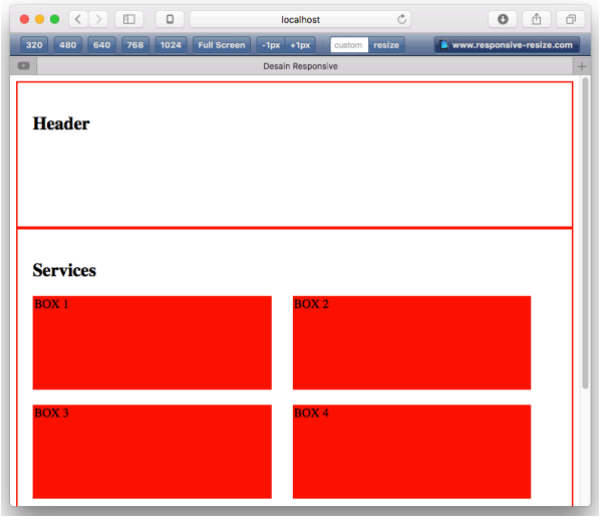
```
41          /*Tablet*/
42          @media only screen and (max-width: 1080px){
43              #nav{
44                  height: 360px;
45              }
46
47              .box{
48                  width: 45%;
49                  margin: 0 4% 20px 0;
50              }
51          }
52
53          /*Smartphone*/
54          @media only screen and (max-width: 640px){
55              #nav{
56                  height: 640px;
57              }
58
59              .box{
60                  width: 100%;
61                  margin: 0 0 20px 0;
62              }
63          }
64
65      </style>
66
67  </head>
```

```
68  <body>
69
70      <div id="wrapper">
71          <div id="header">
72              <h2>Header</h2>
73          </div>
74
75          <div id="nav">
76              <h2>Services</h2>
77              <div class="box"> BOX 1 </div>
78              <div class="box"> BOX 2 </div>
79              <div class="box"> BOX 3 </div>
80              <div class="box"> BOX 4 </div>
81          </div>
82
83          <div id="footer">
84              <h2>Footer</h2>
85          </div>
86      </div>
87
88  </body>
89  </html>
```

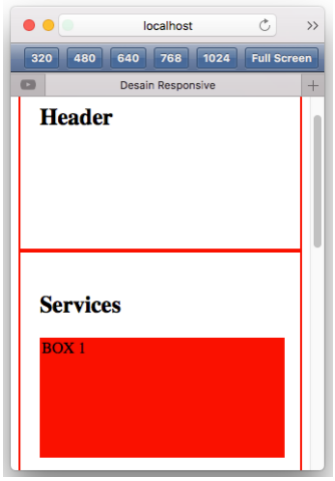
Simpan kode tersebut dengan nama **latihan2a.html.** Buka kode HTML di browser desktop dan hasilnya akan terlihat seperti pada gambar di bawah ini.



Aturan yang kita tetapkan untuk ukuran layar komputer tablet adalah posisi box menjadi dua kolom.



Untuk ukuran layar smartphone, sebagaimana umumnya, kita buat posisi box menjadi satu kolom.



Ubah media query di atas menjadi bentuk responsive menggunakan flex lalu simpan dengan nama latihan2b.html!

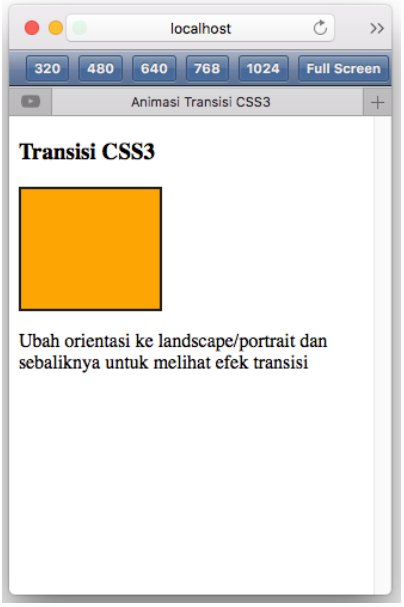
3. Animasi Transisi

Kemampuan lain dari CSS3 adalah menghasilkan objek-objek bergerak atau animasi. Fitur animasi transisi merepresentasikan perpindahan dari satu status visual ke status visual lainnya atau dari satu aturan ke aturan lainnya. Sederhananya, ini merupakan bentuk transisi titik ke titik. Transisi CSS merupakan jenis animasi yang paling mendasar. Realisasi konsep animasi transisi ini diimplementasikan melalui sebuah properti bernama –webkit-transition.

```
<!DOCTYPE html>
<html>
<head>
    <title>Animasi Transisi CSS3</title>
    <meta name="viewport" content="width-device-width, initial-scale=1.0">

    <style type="text/css">
        .box{
            width: 40%;
            height: 100px;
            background: orange;
            border: 2px solid #212121;

            -webkit-transition: width 1s ease-in-out;
            -o-transition: width 1s ease-in-out;
            -moz-transition: width 1s ease-in-out;
            transition: width 1s ease-in-out;
        }

        /*landscape*/
        @media (orientation:landscape){
            .box{
                width: 100%;
            }
        }
    </style>

</head>
<body>
    <h3>Transisi CSS3</h3>

    <div class="box"></div>
    <p>
        ubah orientasi ke landscape/potrait dan sebaliknya untuk melihat efek transisi
    </p>
</body>
</html>
```

Simpan kode tersebut dengan nama **latihan3a.html.** Buka kode HTML di browser desktop dan hasilnya akan terlihat seperti pada gambar di bawah ini.

Ketika layar diperkecil seukuran layar smartphone, objek akan mengikuti dengan diiringi efek animasi transisi.



Dengan transition, ubah bentuk objek menjadi oval dan warna dari objek menjadi biru pada saat landscape!

Masih pada file yang sama, berikan animasi (animation dan keyframes) objek berputar 360 derajat infinite linier dengan tempo 2 detik pada saat landscape sehingga nampak seperti baling-baling pesawat terbang!

Simpan dengan nama latihan3b.html

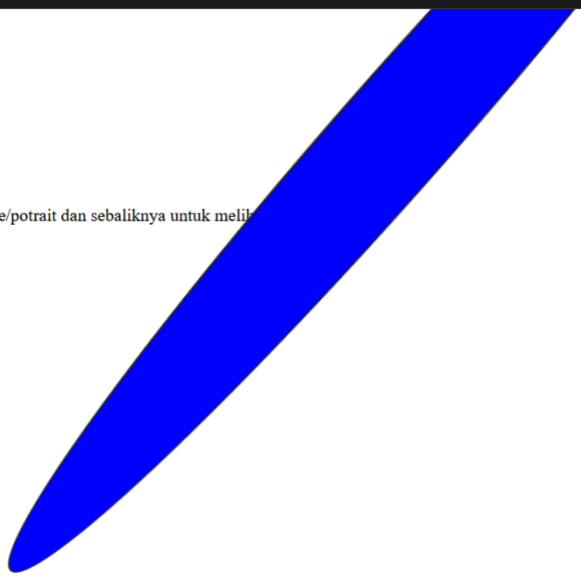
Transisi CSS3

ubah orientasi ke landscape/potrait dan sebaliknya untuk melihat efek transisi

Transisi CSS3

ubah orientasi ke landscape/potrait dan sebaliknya untuk melihat efek transisi

Transisi CSS3



ubah orientasi ke landscape/potrait dan sebaliknya untuk melih