1-) SAAS EĞİTİMİ

TEML BAZI CSS ÖZELLİKLERİ
1-) SATIRI TAM DOLDURMAK
display: block;

2-) formun belli max büyüklük ve ortalanması
```
.form{
  max-width: 900px;
  margin: 0 auto;
}
```

3-) animasyonlar yapmak:
```
@keyframes textComeFromRight {
    0% {
       transform: translateX(50%);
     }

     70% {
        transform: translateX(-20%);
     }

     100% {
         transform: translateX(0%);
     }
}
```

4-)Bir item üzerine geldiğinde animasyonlu döndürmek için:
```
.site-logo{
    height: 3rem;
    transition: .5s;
    &:hover{
        transform: rotate(360deg);
    }
}
```

5-)box içini düzenleme??
Box-size:box-border

6-)float kavramı ve clear
Float ekrandaki kuralları yok sayıyor ve **sağ sola sabitlemek için** kullanılır.

Gelen diğer elemanlar kalan boşluktan devam eder.Eğer alt alta devam etmesini istiyorsak **clear**:left veya right veya **both** ikisinide sıfırlayabiliriz.

7-)nowrap ve wrap kavramı:

Default olarak flex **nowrap** kullanır tek satıra sığdır deriz.
Eğer fazlalığı alt satıra indirmek istiyorsak o zaman flex değeri **wrap** yapmalıyız.

8-)flex flow kavramı:
***Boş alanları pay etmeyi sağlıyor.**
İki özelliği tek seferde yapmak istersek bu özelliği kullanırız
Örnek:
flex-flow: cloumn wrap

Flex-grow:
Varsayılan flex-frow:0 dır eğer bunu 1 veya 2 yaparsak ekran genişledikçe kalan px değeri bu iki alan arasında var olan genişlik ve yüksekliğe eklenir

O olan item sabit kalırken diğer itemlar verilen değerlere göre artar.
9-) flex-basis kullanımı:

Flex-basis bulunduğu alanda var olan boyutları yok sayarak flex-grow değerlerine göre değer alarak eşit katlarda değer almasını yapar
Örnek:

```
.item {
  width: 200px;
  min-height: 100px;
  border: 1px solid black;
  background-color: orange;
  margin: 10px;
}
.item1 {
  flex-grow: 0;
}
.item2 {
  flex-grow: 1;
  flex-basis: 0;
}
.item3 {
  flex-grow: 2;
  flex-basis: 0;
}
```

10-)flex-shrink:
Belli boyutlarda küçültmek için kullanılır.
Varsayılan flex-shrink:0 dır

Örnek:
Flex-shrink:1
Flex-shrink:3

Dersek belli bir küçültmeden sonra biri diğerine göre 3 kat küçülür.

11-)align-self:
Ekranda başta ortada veya sonda sıralanmasını sağlar.

12-) flex-grow,flex-shrink,flex-basis kısaca yazımı:
Flex:1 0 0

13-) varsayılan sitelerde tümünü böyle yap:

```
*{
   box-sizing: border-box;
}
body{
   margin: 0;
   padding: 0;
   font-size: 20px;
   font-family: Georgia, 'Times New Roman', Times, serif;
}
```

14-)grid kullanımı:

```
.grid-container{
   display: grid;
   grid-template-columns: 300px auto 300px;  // yatay genişlik x ekseninde yer kaplar
   grid-template-rows: 50px 700px 50px; /*the height of the rows*/ // dikey Yükseklik y ekseninde yer kaplar
}
```

15-) fraction kullanımı:
Flex gibi alan bölmeye yarar

Örnek:

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

16-) grid-gap kullanımı:
*satır ve sütunlar arasında boşluk verme için kullanılır
Örnek:

```
grid-column-gap: 10px;
grid-row-gap:    10px;
/** Or in shorthand = **/
grid-gap:        10px;
```

17-) grid ile 3 coloumn oluşturmak :

Sol taraf 100px olsun **ortası** otomatik en **sağ tarafta 100px** olsun deriz
```
.container{
    background-color: brown;
    text-align: center;
    font-size: 2rem;
    color: #ffeead;

    display: grid;
    grid-template-columns: 100px auto 100px;
}
```
18-) grid ile satırlara yükselik vermek

```
.container{
    background-color: brown;
    text-align: center;
    font-size: 2rem;
    color: #ffeead;
    display: grid;

   grid-template-rows: 50px  400px;

}
```

19-) grid ile satır ve sütunlara arası buluş vermek:
row-gap: 10px; // sadece **yatay** boşluk vermek için
column-gap: 10px; // sadece **dikey** boşluk vermek için
gap:15px; // ikisine aynı değer vereceksek bunu kullanırız

20-)grid kullanımında aynı işlemin tekrarı ve responsive yapan:
*responsive yapı için fr kullanmalıyız

Örnek:
```
 grid-template-columns: repeat(3, fr);
```
//Kısa yolu
```
 grid-template: 200px 200px repeat(3, fr);
```
21-) grid area responsive site alt yapsı:
```
.container{
    background-color: #96ceb4;
   height: 100vh;
    display: grid;
    grid-template-columns: repeat(8, fr);
    grid-template-rows: 100px auto  100px;
    gap:5px;
   grid-template-areas:
   "header header header header header header header header"
   "m c c c c c c c"
```

```
    "f f f f f f f f";
}

.header{
    background-color: #ccc;
    grid-area: header;
}

.menu{
    background-color: #ff6f69;
    grid-area: m;
}
.content{
    background-color: #ffcc5c;
    grid-area: c;
}
.footer{
    background-color: #88d8b0;
    grid-area: f;
}
```

22-) ileri seviye responsive yapı kurmak

```
.container4{
    display: grid;
    grid-template-columns: repeat(auto-fit, 1fr);
    grid-template-rows: repeat(3, 150px);
    gap:5px;
}
```

23-) position kullanımı:

1-) relative yazınca kullanabilir duruma gelen özellikler şunlar

```
    position: relative; // Default değeri statik
    top: 20px;
    left: 20px;
    right: 20px;
    bottom: 20px;
```

Not: **position** verilen item **hangi div içinde** ise onun position **değerini relative** yapmalıyız ki istediğimiz noktaya konumlandıralım.

Örnek:

```
body{
    font-size: 32px;
    height: 200vh;
    margin: 1rem;
    border: 2px solid red;
}
.parent{
```

```
    border: 2px solid blue;
    padding: 2rem;
    background-color: chocolate;
    position: relative;
}
.item1{
    padding: 2rem;
    background-color: darkmagenta;
}
.item2{
    padding: 2rem;
    background-color: teal;
    position: absolute;
    bottom: 0px;
    left: 0px;
}
.item3{
    padding: 2rem;
    background-color: gold;
}
```

Not2: absulate ve fixed arasındaki fark absulate sayfa kayarken beraber kayma yaparken fixed ise olduğu yerde kalır sayfada scroll edilse de edilmesede konumunu daima korur.

Örnek:

```
    background-color: gold;
    position: fixed;
    bottom: 10px;
    right: 10px;
```

Not3: **sticky** kullanımı :
*relative ve fixed arasında birşeydir.
Ekran belli bir yer kaydırılırsa yazılan kurala göre örneğin top:0px üst kısma gelirse kendini üst kısma fixed eder ve sayfa kaymaya devam eder.

```
    background-color: teal;
    position: sticky;
    top: 0px;
    right: 0px;
```

24-) z-index kullanımı:

- Hangi item ın üste geleceğini belirlemek için kullanırız default değerler **z-index:0** değerdir

```
.item1{
    padding: 2rem;
    background-color: darkmagenta;
    position: relative;
```

```
    top: 60px;
    z-index: 2;
}
.item2{
    padding: 2rem;
    background-color: teal;
    position: sticky;
    top: 0px;
    z-index: 1;
}
```

25-) overflow kullanımı:
*div içine **sığmayan** yazı veya başka elemanları belirtilen boyutta scroll edip görüntülenmesini sağlar
*genelde şöyle tanımlarız

```
  overflow: auto;

.container{
    height: 100px;
    width: 100px;
    border: 2px solid red;
    overflow: auto;
}
```

26-) ana renkleri global bir yerden yönetmek:
Örnek: Bu şekilde tek bi yerden değiştirmiş oluruz

```
:root{
    --main-color:green;
 --my-border: 3px solid purple;

}

h2{
    background-color: red;
    color: var(--main-color);
    border: var(--my-border);

    padding: .5rem;
}
p{
    color: var(--main-color);
   border: var(--my-border);

}
```

Not :ana yazıyı değiştirmeden bazı yerlerde ana rengi değiştirmek

```
p{
```

```
    --main-color:orange;
    color: var(--main-color);
    border: var(--my-border);
}
```

27-) specifity kavramı:

***** Daha belirlilik yapmak için
öncelik id ile işaretlemektir.Bundan daha öncelikli olan ise inline style ile bir özellik
vermek en baskın ve geçerli olan olur.Bu birbirini eoverride yani ezme ile ilgilidir

*öncelik olarak hangi property lerin baskın olacağını belirlemek için .override yani ezme ile ilgili bir durum
***!important bunu hangi renk için kullansak inline bakmaz id ye veya sınıfa bakmaz direk bildiğini okur.**
not: ***!important bu kullanım önerilen bir bakış değildir çok kullanılmaz.**
Örnek:
!important;

```
  div span{
      color: red;
   }
   span{
      color: blue !important;; // normalde div span öncelikli iken !important yazarak önceliği buna verdik
   }
```

28-) advance selectors kullanımı:

```
   <style>
         body{
      font-size: 2rem;
   }
   div span{ /* Div elementi içindeki tüm spanlar seçldi */
      color: red;
   }
   div >span{ /* Div in direk çoçuğu olan span ları seçer */
      color: purple;
   }
   div ~ p{ /* Div in kardeşi olan ve div den SONRA GELEN tüm p ler seçilir  */
      background-color: aquamarine;
   }

   </style>
</head>
<body>
```

```
    <h2>Başlık2</h2>
    <p>paragraf 1</p>
    <p>paragraf 2</p>
    <div>
        <p>Bu etklenmeyecek</p>
        <span>
            Dişardaki span elementi
            <span>içerdeki span ekelemnti</span>
        </span>
    </div>
    <p>div in kardeşi</p>

</body>

ÖRNEK: 2
  <style>
            body{
        font-size: 2rem;
    }
    div span{ /* Div elementi içindeki tüm spanlar seçldi */
        color: red;
    }
    div >span{ /* Div in direk çoçuğu olan span ları seçer */
        color: purple;
    }
    div ~ p{ /* Div in kardeşi olan ve div den SONRA GELEN tüm p ler seçilir  */
        background-color: aquamarine;
    }
    div + p{ /* div elementin den hemen sonra gelen kardeş element olan p yi
seç BİR TANE SEÇİYOR DİKKAT  */
       background-color: orange;
    }
    p + div{ /* burda da tersi geçerli yani p elementin den hemen sonra gelen
kardeş element olan div i seç BİR TANE SEÇİYOR DİKKAT  */
       background-color: teal;
    }
    h2.h2class{ /* sınıfı h2class olanların h2 olan kısmına etki eder */
        background-color: brown;
    }

    h2:not(.h2class) { /* sınıfı h2class olanları HARİÇ DİĞER h2 olan kısmına etki
eder */
        background-color: chartreuse;
    }
    input:not(:checked){ /* seçili değilse margin 30px olsun */
        margin: 30px;
```

```
    }
    input[type="text"]{  /* Sadece type kısmı TEXT olanları sadece geçerli olsun
*/
       background-color: red;
    }
    input[type]{  /* Sadece type property si TANIMLANANDA olanlarda sadece
geçerli olsun */
       background-color: orange;
    }
    input[data-margin="20"]{ /* data-margin  property sine SAHİP olanlara
uygula demek  */
       background-color: aquamarine;
    }
    input[deger*="mehmet"]{ /* deger property içinde mehmet GEÇENLERE
uygula demek  */
       background-color: pink;
    }
    input[deger^="abc"]{ /* deger property içinde abc ile BAŞLAYANLARA
uygula demek  */
       background-color: burlywood;
    }
    input[deger$="eee"]{ /* deger property içinde eee ile BİTENLERE uygula
demek  */
       background-color: grey;
    }
    /*https://www.w3schools.com/cssref/css_selectors.php*/
   </style>
</head>
<body>

   <input type="checkbox" />
   <input type="text" data-margin="20" />
   <input type="text" deger="mehmet" />
   <h2>Başlık2</h2>
   <h2 class="h2class">Başlık2</h2>
   <p>paragraf 1</p>
   <p>paragraf 2</p>
   <div>
      <p>Bu etklenmeyecek</p>
      <span>
         Dişardaki span elementi
         <span>içerdeki span ekelemnti</span>
      </span>
   </div>
   <p>div in kardeşi</p>
</body>
```

29-)calc kullanımı:
Var olan genişiliği alıp üzerine eklemek istediğim bir değeri yüzde olarak veya rem olarak ekleyebiliriz

Örnek:
transform: translate(-50%, **calc(-100% - 30%))**;

30-)Tooltip örnek:

```
<style>
    *,
    *::before,
    *::after {
        box-sizing: border-box;
    }

    body {
        margin-top: 300px;
        font-size: 2rem;
    }

    [data-tooltip]::before,
    [data-tooltip]::after {
        position: absolute;
        left: 50%;
        --tooltip-color: #333;
        /* değişken olrak tanımlamak ve aşağıda her yerde kullanmak için */
        --tooltip-ok-size: 1rem;
    }

    [data-tooltip] {
        background-color: red;
        position: relative;
        display: inline-block;
    }

    [data-tooltip]:hover::before {
        content: attr(data-tooltip);
        /* atribute değerini direk almak için*/
        transform: translate(-50%, calc(-100% - var(--tooltip-ok-size)));
        /* x ekseninde 50%  kaydır y ekseninde ise 110% yukarı kaydır demek */
        left: 50%;
        /* yüzde 50 kadar soldan kaydır demek*/
        background-color: var(--tooltip-color);
        border-radius: .3rem;
        color: white;
```

```
            padding: .3rem;
            width: max-content;
            max-width: 100%;
            /* inline elementlerde genişlik yükseklik kullanılmaz boyutu kadar
yer kaplar*/
            /* inle elementleri ya div ile değişrmeliyiz yada onları block şeklinde
belirtmeliyiz örnek :
            display: inline-block*/
            text-align: center;
            /*tooltip içindeki text büyükse ortaya hizalansın*/
        }

        [data-tooltip]:hover::after {
            content: '';
            border: calc(var(--tooltip-ok-size)/ 2) solid transparent;
            color: white;
            border-top-color: var(--tooltip-color);
            top: 0px;
            width: var(--tooltip-ok-size);
            height: var(--tooltip-ok-size);
            transform: translate(-50%, -100%);

        }
    </style>
</head>

<body>

    <span class="tooltip" data-tooltip="1-2-3-4-5 1-2-3-4-5 1-2-3-4-5
1-2-3-4-5 1-2-3-4-5 1-2-3-4-5">1-2-3-4-5 1-2-3-4-5
        1-2-3-4-5 1-2-3-4-5 1-2-3-4-5 1-2-3-4-5 1-2-3-4-5 1-2-3-4-5
1-2-3-4-5</span>
</body>

</html>
```

31-) Normalize css kullanımı:
*Hazır bazı ayarlardır
*Bu siteden indiriliyor
https://necolas.github.io/normalize.css/8.0.1/normalize.css

normalize.css dosyasını oluşturup içine ekleriz.Bu başka **browserlarda** web uygulamamız aynı şekilde görünmesini sağlar. her site de vardır.

Bir de reset css var
*Bu sitedeki kodu css dosyası olarak eklersek browserdaki tüm font ayarlarını sıfırlar biz hepsini tek tek tanımlayıp istediğimiz değerleri veriri

https://meyerweb.com/eric/tools/css/reset/
<link rel="stylesheet" href="css/reset.css">

Yeni çıkan özelliklerden hangi browserlar destekliyor hangisi desteklemiyor
Örnek: revert renge göre tersini yap ve ya gap satır sütün arasında boşluk bırakmak için yaparız
*Bu sitede bakıp destekleyenleri görebiliriz
https://caniuse.com/

32-) Browser da destek eklemek:

```
    <style>
        div {
            background-color: blue;
            font-size: 2rem;
            color: red;
        }
        /* Browserda destek ekelmek*/
        @supports (background-color: revert) or (background-color: red) {
            .div {
                background-color: revert;
            }
        }
        @supports not (background-color:revert){
            .div{
                background-color: green;
            }
        }
    </style>
</head>

<body>
    <div class="div">My div element</div>
</body>
```

33-) Media query ile responsive yapılar:

```
    <title>Media Query ile responsive tasarımlar</title>
    <style>
        .div{
            font-size: 200%;
            color: purple;
            font-weight: bold;
        }

 /*      @media print{
            .div{
                color: orange;
```

```
    }
}
@media all and (max-width:768px){
    .div{
        color: teal;
    }
}
@media (min-width:1024px) and (max-width:1300px){
    .div{
        color: red;
    }
}
*/
/* First mobile önce mobil tasarımı yap sonra diğer büyük ekranlara göre yap*/


.container div{
    display: none;
    font-size: 200%;
    font-weight: bold;
}
@media (orientation:landscape){
    .container div { /* Genişlik yükseklikten büyük olunca etki eder*/
        background-color: orange;
    }
}
@media (max-width:480px){
    div.mobile{
        display: block;
    }
}
@media (min-width:481px) and (max-width:768px){
    div.tablet{
        display: block;
    }
}
@media (min-width:769px) and (max-width:1024px){
    div.laptop{
        display: block;
    }
}
@media (min-width:1025px) and (max-width:1200px){
    div.desktop{
        display: block;
    }
}
```

```
        @media (min-width:1201px){
            div.large{
                display: block;
            }
        }
    </style>
</head>
<body>
    <div class="container">
        <div class="mobile">Mobile</div>
        <div class="tablet">Tablet</div>
        <div class="laptop">Laptop</div>
        <div class="desktop">Desktop</div>
        <div class="large">Large</div>
    </div>
</body>
</html>
```

34-) Font ayarlamak:
***Fonts klasöründe tutulur.**
*önerilen **font-display: swap;**
*Google fonts sitesinden istenen fontlar indirilir.
*localde bunlar tutulur.

Örnek:

```
        @font-face {
            font-family: salih;
            src: url('fonts/Mali-Regular.ttf') format('truetype'),
                url('fonts/Mali-Regular.ttf') format('truetype');
            /* bulamazsa bu fonto kullan demek */
            font-weight: normal;
            font-display: swap;
        }

        @font-face {
            font-family: salih;
            src: url('fonts/Mali-Bold.ttf') format('truetype'),
                url('fonts/Mali-Bold.ttf') format('truetype');
            /* bulamazsa bu fonto kullan demek */
            font-weight: bold;
            font-display: swap;// önerilen swap tır yada optional yap browser a bırak
        }
```

not: https://glitch.com/~font-display

35-)ikon kullanımı:

```
    <style>
      .icon-container{
         height: 200px;
         width: 200px;
         color: orange;
      }
    </style>
</head>
<body>
    <div class="icon-container">
       <svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 576 512">
          <!--!Font Awesome Free 6.6.0 by @fontawesome - https://
fontawesome.com License - https://fontawesome.com/license/free Copyright
2024 Fonticons, Inc.-->
          <path fill="currentcolor"
             d="M320 192l17.1 0c22.1 38.3 63.5 64 110.9 64c11 0 21.8-1.4 32-4l0
4 0 32 0 192c0 17.7-14.3 32-32 32s-32-14.3-32-32l0-140.8L280 448l56
0c17.7 0 32 14.3 32 32s-14.3 32-32 32l-144 0c-53
0-96-43-96-96l0-223.5c0-16.1-12-29.8-28-31.8l-7.9-1c-17.5-2.2-30-18.2-27.
8-35.7s18.2-30 35.7-27.8l7.9 1c48 6 84.1 46.8 84.1 95.3l0 85.3c34.4-51.7
93.2-85.8 160-85.8zm160 26.5s0 0 0 0c-10 3.5-20.8 5.5-32 5.5c-28.4
0-54-12.4-71.6-32c0 0 0 0 0 0c-3.7-4.1-7-8.5-9.9-13.2C357.3 164 352 146.6
352 128c0 0 0 0 0 0l0-96 0-20 0-1.3C352 4.8 356.7 .1 362.6 0l.2 0c3.3 0 6.4
1.6 8.4 4.2c0 0 0 0 0 .1L384 21.3l27.2 36.3L416 64l64 0 4.8-6.4L512 21.3
524.8 4.3c0 0 0 0 0-.1c2-2.6 5.1-4.2 8.4-4.2l.2 0C539.3 .1 544 4.8 544 10.7l0
1.3 0 20 0 96c0 17.3-4.6 33.6-12.6 47.6c-11.3 19.8-29.6 35.2-51.4 42.9zM432
128a16 16 0 1 0 -32 0 16 16 0 1 0 32 0zm48 16a16 16 0 1 0 0-32 16 16 0 1 0 0
32z" />
       </svg>
    </div>

</body>
```

36-) Animations kullanımı :

Örnek : üzerine gelince renk değiştirme

```
       .container{
          width: 150px;
          height: 150px;
          font-size: 2rem;
          background-color: orange;
          transition: 4s cubic-bezier(0, 0.86, 0.36, 1) 1s;
          color: white;
       }
       .container:hover{
          width: 150px;
```

```
        height: 150px;
        background-color: green;
        color: black;
    }
```

37-) ANİMATİON -transition
***Transition yaptığı 2 farklı state arasında geçiş yapıyor renk geçişi veya boyut geçişi vs.**
Örnek:

```
  <title>Animasyonlar - transition</title>
  <style>
    .container{
      width: 150px;
      height: 150px;
      font-size: 2rem;
      background-color: orange;
      transition: 4s cubic-bezier(0, 0.86, 0.36, 1) 1s;
      color: white;
    }
    .container:hover{
      width: 150px;
      height: 150px;
      background-color: green;
      color: black;
    }

  </style>
</head>
<body>
  <div class="container">
    div
  </div>
</body>
```

Örnek 2-

```
    /* Üzerine gelince hem renk hem boyut değiştirme */
    .container{
      width: 150px;
      height: 150px;
      font-size: 2rem;
      background-color: orange;
      /* Her birine verilen 2. parametreler(1s,4s) animasyonu bekletme süresidir */
      transition: height 2s 1s, width 2s 4s;

    }
    .container:hover{
```

```
    width: 300px;
    height: 300px;
    background-color: teal;
    color: black;
}
```

Örnek 3- belli zamanlarda yüzdelik olarak kontrol etmek:

```
.container{
    width: 150px;
    height: 150px;
    font-size: 2rem;
    background-color: orange;
}
.container:hover{
    animation: renk_degistir 6s;
}
@keyframes renk_degistir {
    from{
        background-color: blue;
    }
    50%{
        background-color: red;
    }
    to{
        background-color: aquamarine;
    }
}
```

Örnek 4: sonsuz şekilde animasyon dönmesi için

```
.container{
    width: 150px;
    height: 150px;
    font-size: 2rem;
    background-color: orange;
}
.container:hover{
    /*  cubic-bezier(0.99,-0.17, 0.57, 1) browser dan ayarlayıp koyuyoruz ve 1s yazan kısım animasyon 1 sn sonra başlasın infinite ise her defasında çalışsın demek */
    animation: renk_degistir 4s cubic-bezier(0.99,-0.17, 0.57, 1) 1s infinite;
}
@keyframes renk_degistir {
    from{
        background-color: blue;
    }
    50%{
        background-color: red;
```

```
        }
        to{
            background-color: aquamarine;
        }
    }
```

Örnek 5: animasyonun başlayıp boyut büyütmesi ve geriye dönmesi 2 defa en az:

```
    .container{
        width: 150px;
        height: 150px;
        font-size: 2rem;
        background-color: orange;



    }
    .container:hover{
        animation: renk_degistir 4s cubic-bezier(0.99,-0.17, 0.57, 1) 1s 2
alternate,
        boyut_degistir 4s 2s 2 alternate;
     /* 2 animasyon ard arda çağrılması 4 saniye sürmesi ve boyut_degistir
için konuşmak gerekirse: 2s saniye beklemesi ve geriye gitmesi için
alternate*/
    }
    @keyframes renk_degistir {
        from{
            background-color: blue;
        }
        50%{
            background-color: red;
        }
        to{
            background-color: aquamarine;
        }
    }

    @keyframes boyut_degistir {
        to{
            width: 300px;
            height: 300px;
        }
    }
```

Not animasyonu üzerine gelince durdurmak için:

animation-play-state:paused

Örnek 6: animasyon aynı değerleri taşıyorsa şöyle tanımlanabilir:

```
.container{
    width: 150px;
    height: 150px;
    font-size: 2rem;
    background-color: orange;

    animation-name: renk_degistir, boyut_degistir;
    animation-duration: 2s; /*Kaç sn kalacağı */
    animation-timing-function: ease-in; /*Görünme efect şekli */
    animation-delay: .1s; /* Bekleme süresi */
    animation-iteration-count: 2; /* Kaç defa çalışacağı*/
    animation-fill-mode: both; /* Doldurma şekli beraber vs*/
    animation-direction: alternate; /* Başladıktan sonraki yönü*/
}
.container:hover{
   animation-play-state: paused; /* üzerine gelince durdurmak */
}
@keyframes renk_degistir {
    from{
        background-color: blue;
    }
    50%{
        background-color: red;
    }
    to{
        background-color: aquamarine;
    }
}

@keyframes boyut_degistir {
    to{
        width: 300px;
        height: 300px;
    }
}
```

9-) CONTAINER İÇİN

****

1-) Eğer flex-direction:row ise yatay sıralar eğer cloumn ise dikey yukardan aşağıya sıralar.

justify-content:

Satır olarak ortalamak için kullanılır.

justify-content: flex-start

justify-content: flex-end

justify-content: center

justify-content: space-**between** // s**adece ortalarda Boşluk verir** ve yatay sıralar
justify-content: space-**around** // b**aşta ve sonda biraz boşluk verir ortalarda kalan değeri eşit verir.**
justify-content: space-**evenly**. // **başlardan ve ortalarda eşit boşluk** verir

2-) align-items:
Y ekseninde items ları **Dikey** sıralar
justify-content: flex-start
justify-content: flex-end
justify-content: center

justify-content: stretch. // ekranı doldurur ve sıralar
justify-content: basealine. // ekranda x- eksende yatay sıralar

3-) align-content:
Birden fazla satır varsa bunları justify-content gibi bu defa satır olarak işlem yapar
Örneğin itemlar 2 satar kadar oluyorsa bu kullanılır.
****

10-)ITEM LAR İÇİN FLEX Kullanımı

****
1-) flex-grow:

İtemlardan bazılarını daha büyük göstermek için kullanırız.

Flex-grow: 4 dersek 4. İndexteki elemanı bulunduğu satırda diğerlerinin 4 katı büyük yapar. Flex gibi çalışır kısaca
Sdsdsdsd

2-) flex-shrink:
Bir elemanın diğerlerinden daha küçük yapmak için kullanırız.
Flex-shrink:4

3-)sadece flex vermek:

Varsayılan bazı özellikleri vermek için 2 farklı özellikten biri
<fex-grow> < flex-shrink>? || <fex-basis>

Dfdf
4-) align-self:
Sadece tek bir elemanın konumunu değiştirmek için kullandırılır

align-self: flex-start
align-self: flex-end
justify-content: center

justify-content: stretch. // ekranı doldurur ve sıralar
align-self: basealine. // ekranda x- eksende yatay sıralar

5-)

"format": "compressed", // css oluştururken boşlukları sil yer kaplamasın demek
"savePath": null, // css ve map dosyalarını varsayılan ilgili klasörde yap demek

)

1-) Sayfanın tüm ekranını sıfırlasması için:

```
*,*::before, *,*::after{
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}
```

2-)Değişken tanımlama ve kullanma:

```
$main-color: orange;
```

3-)Değişken lerin import edilmesi:

```
@import 'reset';
@import 'variables';

.fullscreen-header{
    background-color: red;
    height: 100vh;
}
```

4-) varsayılan satır alıyor ama biz sütün yapalım:

```
    display: flex;
    flex-direction: column;
```

5-) Tekrarlayan yapıları mixing içine koymak:

```
   justify-content: center;
    align-items: center;
    flex-direction: column;
```

Örnek:

```
@import 'reset';
@import 'variables';
@import 'mixins';

.fullscreen-header{
    background-color: red;
    height: 100vh;
    flex-direction: column;
    @include flexCenter;
}
```

6-) Bazı yerlerde rem bazı yerlerde vh ve bazı yerlerde em kullanılır

1-) Layoutlarda **rem** kullanılır

2-) Body vb büyük ekranlar **vh** kullanılır

3-) Componenet button vs de içinde bulunduğu layouttun boyutuna göre küçülüp büyümesi için **em** kullanılır

Burada bir de **px** var **sabit** olan değerleri px olarak verilir diğer dinamik değerler rem veya em verilir.
Border, border radius ,margin, padding => **em** olarak verilir.
Font-size: **inherit** verilir.

7-)