




CSS Selectors



GitHub 'dan beni takip etmeyi unutma  [@burakkrt](#)

:active

Kullanıcı bir elementin üzerine tıkladığında çalışır.

```
button:active {  
  color: red;  
}
```

:any-link

Bir öğede herhangi bir yönlendirici (href) var ise aktifleşir. Genellikle a etiketleri için kullanılır.

```
<a class="link" href="#">Burası bir link</a>  
  
/* href özelliği olan tüm a etiketlerine stil verdik */  
a:any-link {  
  color: blue;  
  text-decoration: underline;  
}
```

:autofill

Input elementlerinde tarayıcıda kayıtlı kullanıcı bilgilerinin otomatik doldurulması durumunda çalışır. Not: Kullanıcı input 'da bir değişiklik yaparsa stiller kaldırılır.

```
<input type="text" class="input" placeholder="isim giriniz">  
  
/* Eğer kullanıcı bilgileri tarayıcıda kayıtlı ise ve kullanıcı bu alanları otomatik  
doldurdu ise ilgili input elementlerinde stil uygulanır */  
input:autofill {  
  color: gray;  
  background-color: green;  
}  
/* bu özelliği tarayıcı destekli kullanmak daha faydalı olacaktır */
```

```
input:-webkit-autofill {  
  color: gray;  
  background-color: green;  
}
```

:checked

Checkbox veya radio gibi input elementleri eğer işaretli ise (checked) , ilgili stiller uygulanır.

```
<input type="checkbox" name="active" id="check">  
  
/* checkbox her stil 'i kabul etmez, bu checkbox ile ilgili bir durumdur */  
input:checked {  
  outline: 2px solid black;  
}
```

:default

Form elementlerinde default olarak tanımlanan değeri olan elementleri seçer.

```
<input type="checkbox" name="active" id="check" checked>  
  
/* inputlarda default olarak tanımlanmış değeri olan tüm elemanları seçer */  
input:default {  
  border: none;  
  outline: 2px solid deeppink;  
}
```

:disabled

Bir form öğesi devre dışı bırakılmış, seçilemiyorsa, üzerine tıklanamıyorsa, içine yazılamıyorsa veya odak kabul edemiyorsa devre dışı kabul edilir ve devre dışı edilmiş elementleri seçer.

```
<input type="checkbox" name="active" id="check" disabled>  
  
input:disabled {  
  outline: 2px solid red;  
}
```

:enabled

Bir form öğesi etkin ise , seçilebiliyor, üzerine tıklanıyor, içine yazılıyor veya odak kabul ediyor ise etkin kabul edilir ve etkin elementleri seçer. (`:disable` nin tersi)

```
<input type="checkbox" name="active" id="check">

input:enabled{
  outline: 2px solid blue;
}
```

:empty

İçeriği boş olan elementleri seçer. Bir elementin altında boşta olsa bir element var ise o elementi dolu kabul eder. Boşluk karakteri olan elementleri de dolu kabul eder.

```
div:empty {
  height: 20px;
  width: 400px;
  border: 2px solid red;
}

<div></div> /* empty element */

<div>
  <p></p> /* no empty element */
</div>

<div>   </div> /* no empty element (spaces) */
```

:first-child

Bir grup kardeş öğe arasından ilk öğeyi seçer. Önemli : bu seçicide kardeş öğelerin hepsinin aynı etikete sahip olması gerekir.

```
<ul class="list-items">
  <li class="list-item">test 1</li>
  <li class="list-item">test 2</li>
  <li class="list-item">test 3</li>
</ul>

/* list-items class 'ının altındaki tüm li etiketlerinden ilk öğeyi seçiyorum */
.list-items > .list-item:first-child {
  color: red;
}

<ul class="list-items">
  <span >bu span etiketi</span>
  <p>bu p etiketi</p>
```

```

<li class="list-item">test 1</li>
<li class="list-item">test 2</li>
<li class="list-item">test 3</li>
</ul>

/* bu çalışmaz çünkü alt elemanların hepsi li ögesi olmalıdır */
.list-items > li:first-child {
  color: red;
}

```

:first-of-type

`:first-child` 'daki alt öğelerin hepsinin aynı olması sorununu ortadan kaldırarak sadece ilgili elementlerin ilk elemanını seçer. Alt öğeler farklı olsa bile.

```

<ul class="list-items">
  <span>bu span etiketi</span>
  <p>bu p etiketi</p>
  <li class="list-item">test 1</li>
  <li class="list-item">test 2</li>
  <li class="list-item">test 3</li>
</ul>

/* .list-items 'in altındaki sadece li elementlerindeki ilk öğeyi seçiyorum */
.list-items > li:first-of-type {
  color: red;
}

```

:last-child

Bir grup kardeş öğe arasından son öğeyi seçer. Önemli : bu seçicide kardeş öğelerin hepsinin aynı etikete sahip olması gerekir.

```

<ul class="list-items">
  <li class="list-item">test 1</li>
  <li class="list-item">test 2</li>
  <li class="list-item">test 3</li>
</ul>

/* list-items class 'ının altındaki tüm li etiketlerinden ilk öğeyi seçiyorum */
.list-items > .list-item:last-child {
  color: red;
}

<ul class="list-items">
  <span>bu span etiketi</span>
  <p>bu p etiketi</p>
  <li class="list-item">test 1</li>
  <li class="list-item">test 2</li>

```

```
<li class="list-item">test 3</li>
</ul>

/* bu çalışmaz çünkü alt elemanların hepsi li ögesi olmalıdır */
.list-items > li:last-child {
  color: red;
}
```

:last-of-type

:last-child 'daki alt öğelerin hepsinin aynı olması sorununu ortadan kaldırarak sadece ilgili elementlerin son elemanını seçer. Alt öğeler farklı olsa bile.

```
<ul class="list-items">
  <span>bu span etiketi</span>
  <p>bu p etiketi</p>
  <li class="list-item">test 1</li>
  <li class="list-item">test 2</li>
  <li class="list-item">test 3</li>
</ul>

/* .list-items 'in altındaki sadece li elementlerindeki ilk öğeyi seçiyorum */
.list-items > li:last-of-type {
  color: red;
}
```

:nth-child

İlgili elementin kardeş elementlerini baştan başlayarak index sırasına göre seçer.

Not: Altında birbirinden farklı etikette öğeler var ise doğru seçim için **:nth-of-type** kullan.

```
<ul class="custom-list">
  <li>İtem 1</li>
  <li>İtem 2</li>
  <li>İtem 3</li>
  <li>İtem 4</li>
  <li>İtem 5</li>
  <li>İtem 6</li>
  <li>İtem 7</li>
  <li>İtem 8</li>
  <li>İtem 9</li>
  <li>İtem 10</li>
</ul>

/* li öğelerinden 2. sıradaki */
.custom-list > li:nth-child(2) {
  background-color: #EBB02D;
}
```

```

/* li öğelerinden çift sayıdakiler (item 2,4,6,8,10, bu seçim index numarasına göre)*/
.custom-list > li:nth-child(event) {
    background-color: #EBB02D;
}

/* li öğelerinden tek sayıdakiler (item 1,3,5,7,9 bu seçim index numarasına göre)*/
.custom-list > li:nth-child(odd) {
    background-color: #EBB02D;
}

/* sayısal işlemler ise kardeş öge seçimi */
/* n değeri 0 dan başlayarak sonsuza kadar tüm pozitif tam sayıları içerir (0,1,2...)*
.custom-list > li:nth-child(n) {
    background-color: #EBB02D;
}

/* [2*0 = 0],[2*1=2],[2*2=4] .. şeklinde çift sayıları temsil eder */
.custom-list > li:nth-child(2n) {
    background-color: #EBB02D;
}

/* [5*0 = 0],[5*1=5],[5*2=10] .. şeklinde 5 in katlarını temsil eder */
.custom-list > li:nth-child(5n) {
    background-color: #EBB02D;
}

```

:nth-last-child

İlgili elementin kardeş elementlerini sondan başlayarak index sırasına göre seçer. Ters.

```

<ul class="custom-list">
  <li>İtem 1</li>
  <li>İtem 2</li>
  <li>İtem 3</li>
  <li>İtem 4</li>
  <li>İtem 5</li>
  <li>İtem 6</li>
  <li>İtem 7</li>
  <li>İtem 8</li>
  <li>İtem 9</li>
  <li>İtem 10</li>
</ul>

/* li öğelerinden sondan 2. sıradaki (İtem 9) */
.custom-list > li:nth-last-child(2) {
    background-color: #EBB02D;
}

```

:nth-of-type

Eğer bir elementin altındaki elementler farklı etiketlerde ise `:nth-child` doğru çalışmaz. Bu soruna çözüm olarak `nth-of-type` ile sadece ilgili etiketler arasında sıralamaya göre seçim yapar.

```
<ul class="custom-list">
  <li>İtem 1</li>
  <li>İtem 2</li>
  <li>İtem 3</li>
  <li>İtem 4</li>
  <ul>
    <li>İtem 4.1</li>
    <li>İtem 4.2</li>
    <li>İtem 4.3</li>
    <li>İtem 4.4</li>
  </ul>
  <li>İtem 5</li>
  <li>İtem 6</li>
  <li>İtem 7</li>
  <li>İtem 8</li>
  <li>İtem 9</li>
  <li>İtem 10</li>
</ul>

/* İtem 4 ile İtem 8 'i seçecektir.
.custom-list > li:nth-of-type(4n) {
  background-color: #EBB02D;
}
```

:only-child

İlgili elementin altında sadece 1 çocuk öge var ise seçer.

```
<ul>
  <li>İtem 1</li>
</ul>

<ul>
  <li>İtem 3</li>
  <li>İtem 4</li>
</ul>

/* ul etiketlerinden sadece 1 çocuğu olanın, çoğuna (li) ilgili stili uygular,
yani burada İtem 1 'in arkaplanını değiştirir.*/
ul > :only-child {
  background-color: #EBB02D;
}
```

:focus

Kullanıcı bir öğeye odaklandığında çalışır. Form elementlerinde kullanışlıdır.

```
<input type="text" class="input" placeholder="isim giriniz" >

input:focus {
  background-color: red;
}
```

:focus-visible

Fare veya işaretçi ile değil de klavye ile gezinirken bir öğeye odaklandığında çalışır.

```
<input type="text" class="input" placeholder="isim giriniz" >

input:focus-visible {
  background-color: blue;
}
```

:focus-within

Kullanışlı ! Bir kapsayıcı öğenin alt öğelerinden birine odaklandığında kapsayıcı öğeyi seçmeyi sağlar.

```
<div class="container">
  <p>ileri seviye css öğreniyorum</p>
  <span>github 'dan burakrt 'yi takip etmeyi unutma :)</span>
  <input type="text">
</div>

/* container class 'ının altındaki herhangi bir öğeye odaklanma durumunda
.container 'e stil özellikleri uygular */

.container:focus-within {
  background-color: yellow;
}
```

:hover

İlgili öğenin herhangi bir işaretçi ile üzerine gelindiğinde stil uygular.

```
<p>burası bir p etiketidir</p>

p:hover{
```



```
color: red;
}
```

:in-range

Form elementlerinde belirtilen değer tanımlanan min ve max 'değer aralığında ise yani karşıyor ise stil uygular.

```
<input type="number" min="1" max="100" />

input {
  background-color: red;
}
/* eğer girilen değe 1 ile 100 arasında ise ilgili stil uygulanır */
input:in-range {
  background-color: green;
}
```

:out-of-range

Form elementlerinde belirtilen değer tanımlanan min ve max 'değer aralığının dışında ise stil uygular.

```
<input type="number" min="1" max="100" />

input {
  background-color: red;
}
/* eğer girilen değe 1 ile 100 arasında değil ise ilgili stil uygulanır */
input:out-of-range {
  background-color: green;
}
```

:invalid

Form elementlerinde belirtilen değer tanımlanan min ve max 'değer aralığında değilse yani karşılamıyor ise stil uygular. `:in-range` 'nin tersi.

```
<input name="age" type="number" value="5" min="18" />
<input name="secret" type="text" value="test" pattern="[a-z]+" />

input:invalid {
  outline: 2px solid red;
}
```

:is

Belirli bir öğenin altındaki öğeleri seçmemizi sağlar. :first-of-type 'dan daha kullanışlı ve yazımı rahattır.

```
<ul class="list-items">
  <span >bu span etiketi</span>
  <p>bu p etiketi</p>
  <li class="list-item">test 1</li>
  <li class="list-item">test 2</li>
  <li class="list-item">test 3</li>
  <input type="text" class="input" placeholder="isim giriniz" >
</ul>

/* list-items altındaki elemanlardan etiketi li veya p olan elementleri seçer */
.list-items > :is(li,p) {
  border: 1px solid red;
}

<div class="container">
  <div>
    <p>beni github 'dan takip etmeyi unutma</p>
  </div>
  <ul class="list-items">
    <span >bu span etiketi</span>
    <p>bu p etiketi</p>
    <li class="list-item">test 1</li>
    <li class="list-item">test 2</li>
    <li class="list-item">test 3</li>
    <input type="text" class="input" placeholder="isim giriniz" >
  </ul>
</div>

/* container altındaki elementlerden class 'ı list-items olanın altındaki
tüm p ve li etiketlerini seçer */
.container > :is(.list-items) > :is(p, li) {
  border: 1px solid red;
}
```

:where

:is ile aynı özelliktedir, birden çok seçiciyi bir arada tanımlar.

```
<ul class="list-items">
  <span >bu span etiketi</span>
  <p>bu p etiketi</p>
  <li class="list-item">test 1</li>
  <li class="list-item">test 2</li>
  <li class="list-item">test 3</li>
  <input type="text" class="input" placeholder="isim giriniz" >
</ul>
```

```

/* list-items in altındaki tüm span ve p etiketlerini seçer ve stil uygular */
.list-items > :where(p,span){
    color: red;
}

```

:not

İlgili element ile eşleşmeyen diğer tüm elementleri seçer.

```

<ul class="list-items">
  <span>bu span etiketi</span>
  <p>bu p etiketi</p>
  <li class="list-item">test 1</li>
  <li class="list-item">test 2</li>
  <li class="list-item">test 3</li>
  <input type="text" class="input" placeholder="isim giriniz" >
</ul>

/* list-items altındaki li elementi hariç tüm öğeleri seçer */
.list-items > :not(li) {
    color: yellow;
}

/* birden çok etiket, class, id belirtilebilir */
.list-items > :not(span, .list-item) {
    color: yellow;
}

/* ayrı ayrı da belirtilebilir */
.list-items > :not(span):not(li) {
    color: yellow;
}

```

:lang

Html etiketlerinde belirtilmiş dilleri olan elementlere seçmemizi sağlar. Not: <html lang="tr-TR"> tanımlanmış ise altındaki tüm öğelere stil uygular çünkü tüm sayfa tr-TR olarak tanımlanmıştır. Fakat sadece belirli etiketlere farklı lang etiketi uygulanmış ise sadece o elementlere stil uygular.

```

<p lang="en-US">Hello, my name is Burak. I am a developer.</p>
<p lang="tr-TR">Selam, benim adım Burak. Ben bir geliştiriciyim.</p>

*:lang(tr-TR){
    color: red;
}

*:lang(en-US){

```

```
color: blue;
}
```

:link

Henüz ziyaret edilmemiş bağlantıları seçer.

```
<a href="#">Click me</a>

/* henüz tıklanmadı ise uygular */
a:link {
  color: blue;
}
```

:visited

Eğer bağlantı ziyaret edilmiş (açılmış) ise.

```
<a href="#">Click me</a>

/* ziyaret edildikten sonra */
a:link {
  color: grey;
}
```

:optional

Form elementlerinde bir form elementine içerik girmesini zorunlu kılmak için **required** (zorunlu) terimini etiket içerisinde belirtiriz. Zorunlu olmayan öğeleri seçer.

```
<form>
  <input name="name" type="text" required />
  <input name="birth" type="date" />
  <select name="origin" required>
    <option>Google</option>
    <option>Facebook</option>
    <option>Advertisement</option>
  </select>
</form>

/* form öğesinin altındaki zorunlu olmayan elementeri seçer */
form > :optional {
  border: 2px solid grey;
}
```

:required

Form elementlerinde bir form elementine içerik girmesini zorunlu kılmak için `required` (zorunlu) terimini etiket içerisinde belirtiriz. Sadece zorunlu öğeleri seçer.

```
<form>
  <input name="name" type="text" required />
  <input name="birth" type="date" />
  <select name="origin" required>
    <option>Google</option>
    <option>Facebook</option>
    <option>Advertisement</option>
  </select>
</form>

/* form öğesinin altındaki zorunlu elementeri seçer */
form > :required{
  border: 2px solid grey;
}
```

:placeholder-shown

Form elementlerinde placeholder öğesini görüntüleyen öğeleri seçer. Kullanıcı değer girmeye başladığı an stiller kaldırılır.

```
<input type="number" placeholder="Bir sayı girin" >

/* placeholder bulunan öğelere ilgili stili uygular. Eğer kullanıcı inputa değer
girmeye başlarsa placeholder text 'i kaybolacağı için stiller de kaldırılır.
Input içeriğini temizler&siler ise placeholder tekrar gözükeceğinden stillerde
tekrar yüklenir */
:input:placeholder-shown{
  outline: 2px solid orange;
}
```

:ready-only

Sadece okunabilen, değer girilmeyen, değiştirilmeyen öğeleri seçer.

```
<div class="tt2">
  <a class="a1">dasdasdas</a>
  <a class="a2">dasdasdas</a>
  <select>
    <option>item 1</option>
    <option>item 2</option>
    <option>item 3</option>
  </select>
  <a class="a4">dasdasdas</a>
```

```

<input type="text">
<input type="checkbox">
</div>

/* input text ve checkbox değer girilebilir veya değiştirelebilir bunun haricinde
a etiketi ve select 'ler sadece okunabilir olduğu için, okunabilir öğeleri seçer */
*:ready-only {
    color: red;
}

```

:ready-write

Sadece değer girelebilen ve seçilen öğeleri seçer.

```

<div class="tt2">
  <a class="a1">dasdasdas</a>
  <a class="a2">dasdasdas</a>
  <select>
    <option>item 1</option>
    <option>item 2</option>
    <option>item 3</option>
  </select>
  <a class="a4">dasdasdas</a>
  <input type="text">
  <input type="checkbox">
</div>

/* input text ve checkbox değer girilebilir veya değiştirelebilir */
*:ready-write {
    color: red;
}

```

:root

En üst kapsayıcı elemanı belirtir. Buda html elementine denktir. html elementine css değişkenlerini tanımlamak için kullanılır.

```

/* html etiketine backgorund verebiliriz ve tüm sayfanın arkaplanı değişir */
:root {
    background-color: orange;
}

/* değişken tanımlayabiliriz ve alt öğelerde çağırabiliriz, dinamik yapı oluşur */
:root {
    --main-color: black;
    --main-background: yellow;
}
/* color 'a --main-color adındaki değişkeni tanımladık yani black */
.box {

```

```
color: var(--main-color)  
}
```