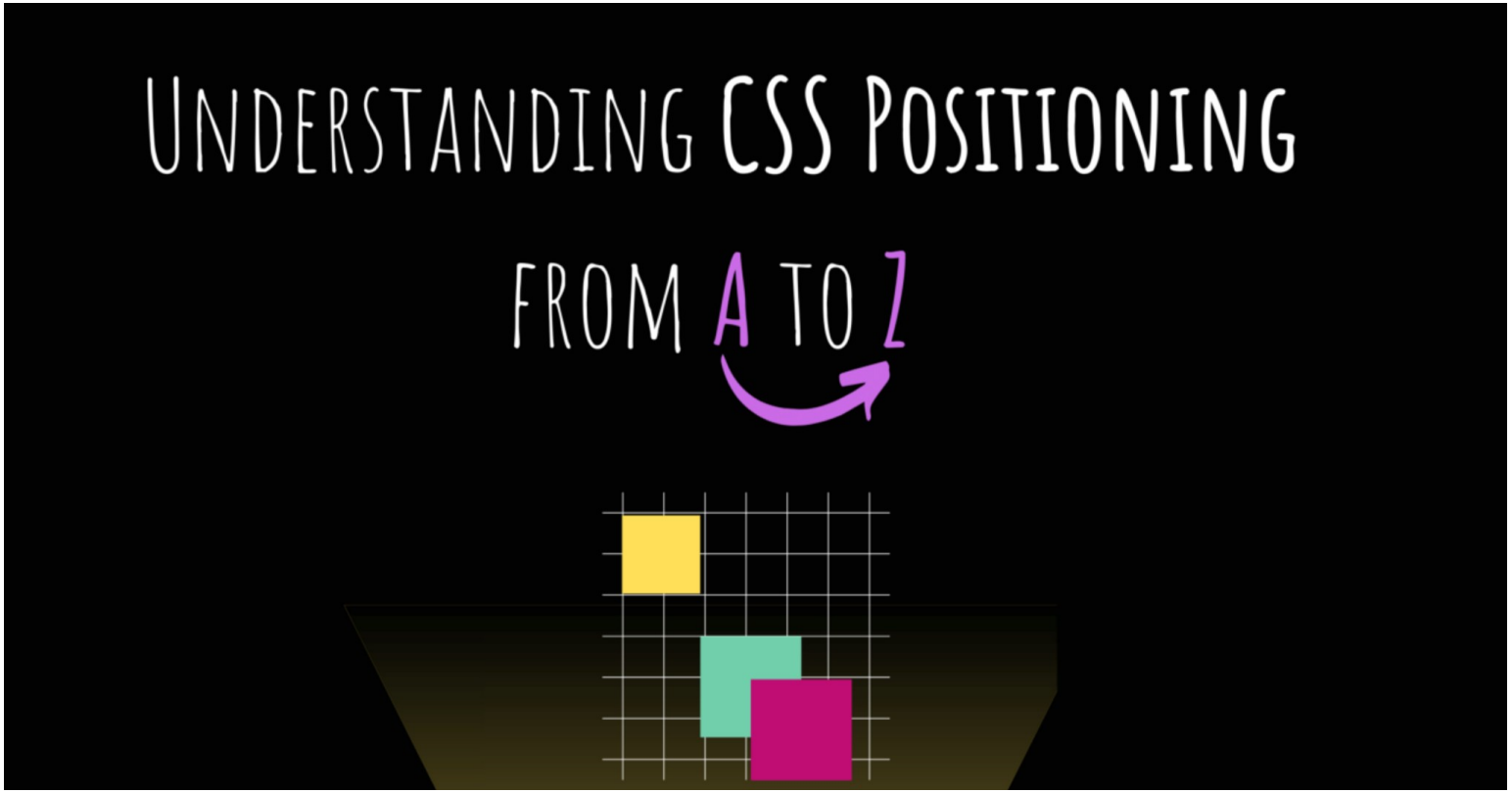
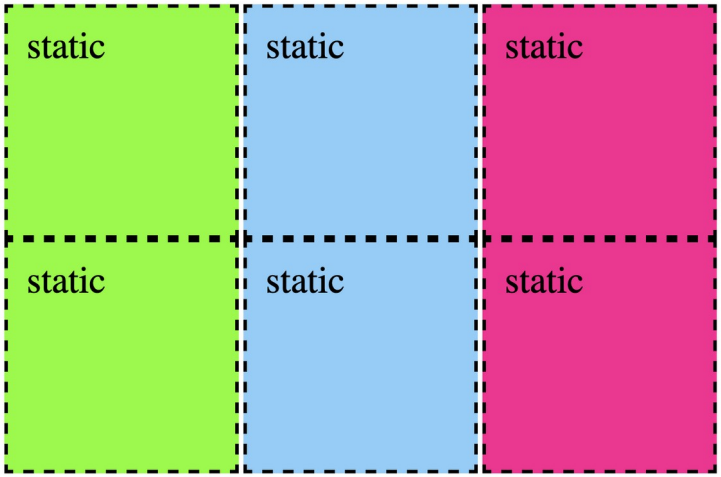
CSS Position



CSS Position - static

```
div {
  position: static

  ~~z-index: number~~ /* DOES NOT WORK */
}
```



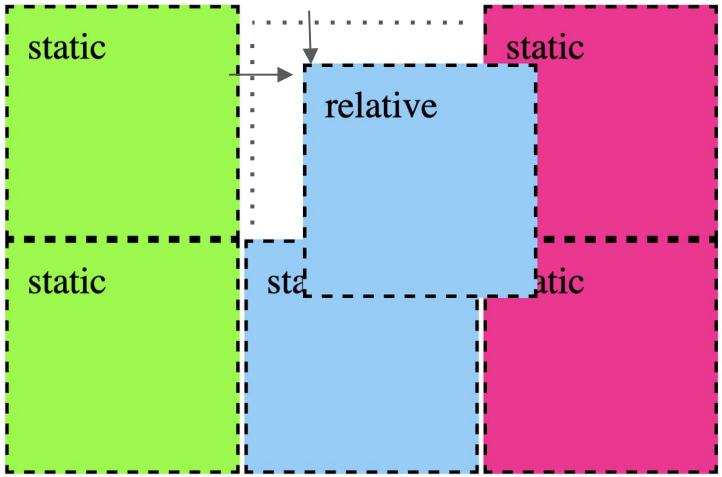
position: static (default)
Es bedeutet, dass das Element im “normalen Fluss” der Seite ist.

Der einzige Grund, ein Element auf position: static; setzen, ist, um eine Positionierung, die auf ein Element angewendet wurde, gewaltsam zu entfernen.

CSS Position - relative

```
div {
  position: relative
  top: 50px
  left: 50px

  z-index: number /* OPTIONAL */
}
```



position: relative

Das Element bleibt zwar im “normalen Fluss” aber es wird "relativ zu sich selbst" verschoben.

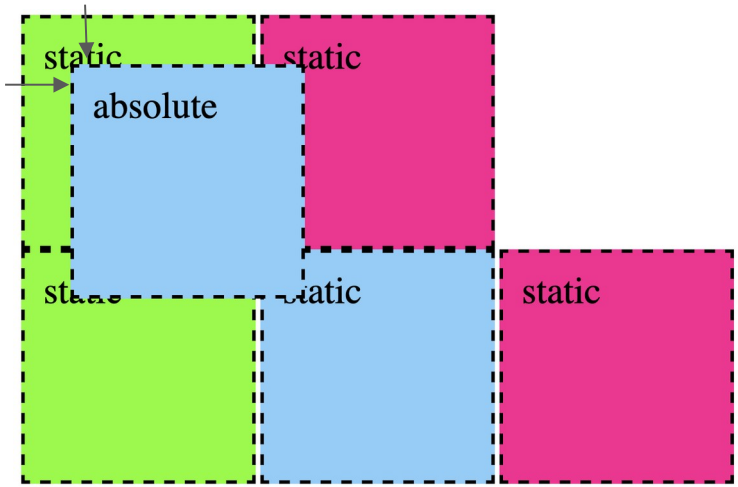
1- Der freie Platz kann von keinem anderen Element eingenommen werden!
2- Erscheint über jedem static-Element.
3- Kann z-index bekommen.
4- Absolut positionierte Kind-Elemente werden im Bezug zum relativen positionierten Parent-Element verschoben!

CSS Position - absolute

```
div {
  position: absolute
  top: 50px
  left: 50px

  z-index: number /* OPTIONAL */
}
```

hier “relative” zum viewport



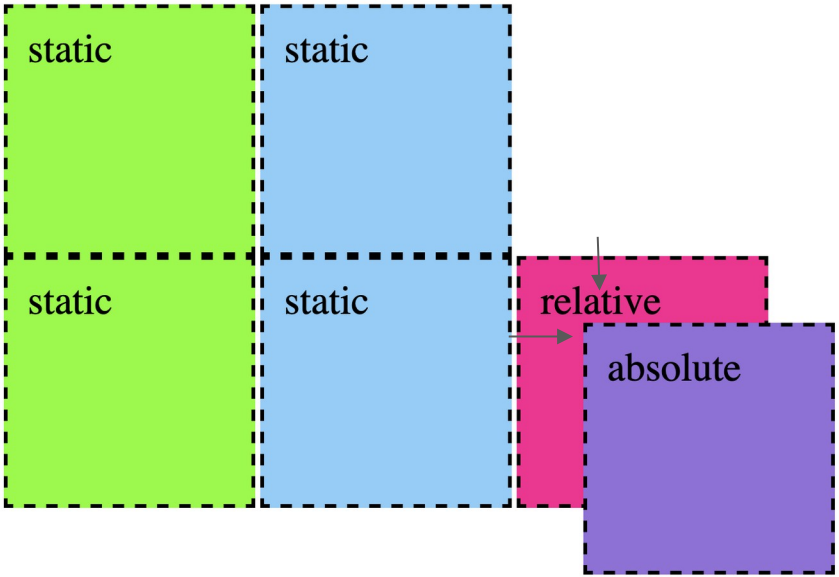
position: absolute

Das Element wird aus dem “normalen Fluss” genommen und unabhängig verschoben.

1- Andere Elemente tun so, als ob das Element nicht vorhanden wäre.
2- Erscheint über jedem static-Element.
3- Kann z-index bekommen.
4- Braucht content oder width/height um an Größe zu haben.
5- Wird zum nächsten relative positionierten Parent-Element positioniert oder falls keines gefunden wird zum Viewport (0/0).

CSS Position - absolute

hier “relative” zum nächsten relative positionierten Parent




```
.relative {
   position: relative;
}

.absolute {
   position: absolute;
   top: 50px;
   left: 50px;
}
```

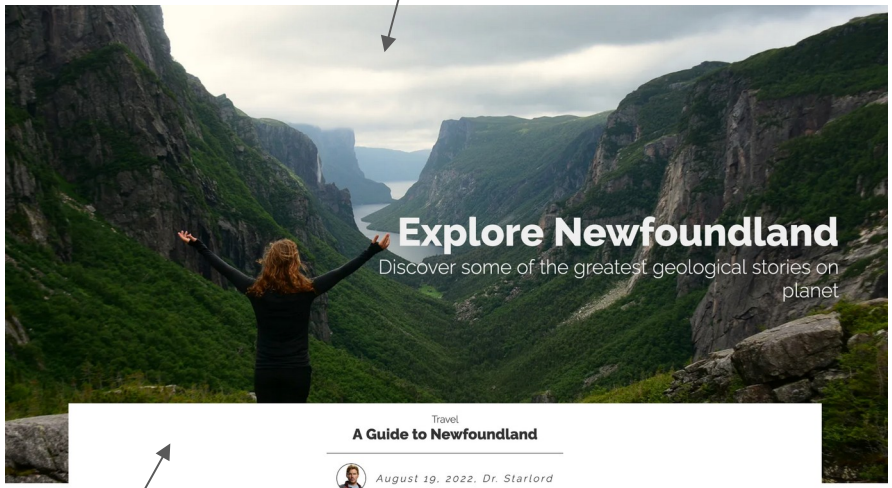
```
<div class="relative”>
  relative
  <div class="absolute">absolute</div>
<div>
```

CSS Position - fixed

```
div {
  position: fixed
  top: number
  right: number

  z-index: number /* OPTIONAL */
}
```

fixed-element with full background image



relative-element um darüber scrollen zu können.
Achtung: **static-element** würde darunter scrollen!!

position: fixed
Das Element wird aus dem “normalen Fluss” genommen und bleibt auch beim Scrollen an seiner fest definierten Position.

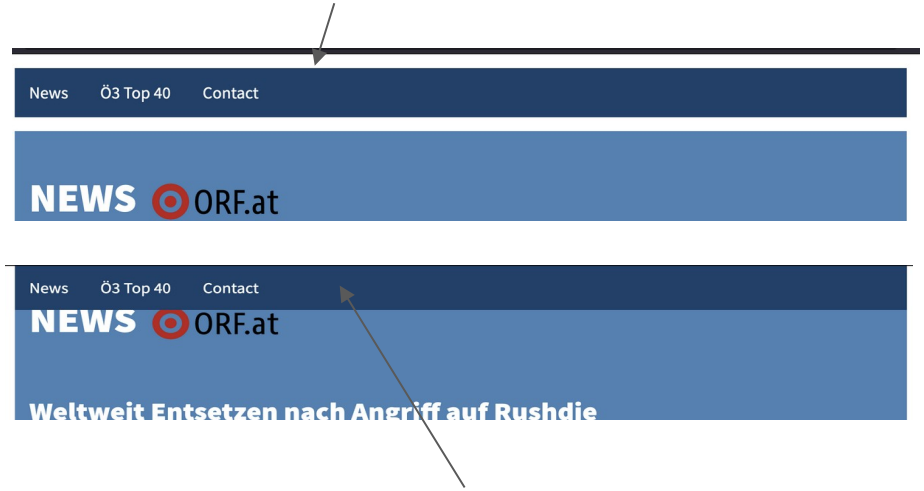
1- Wird relative zum Viewport (0/0) verschoben.
2- Erscheint über jedem static-Element.
3- Kann z-index bekommen.
4- Braucht content oder width/height um an Größe zu haben.

CSS Position - sticky

```
div {
  position: sticky
  top: number
  right: number

  z-index: number /* OPTIONAL */
}
```

Am Anfang im normalen Fluss und wird normal gescrollt.



Klebt bei erreichen von top: 0px fest!

```
.site-nav { position: sticky; top: 0px }
```

position: sticky

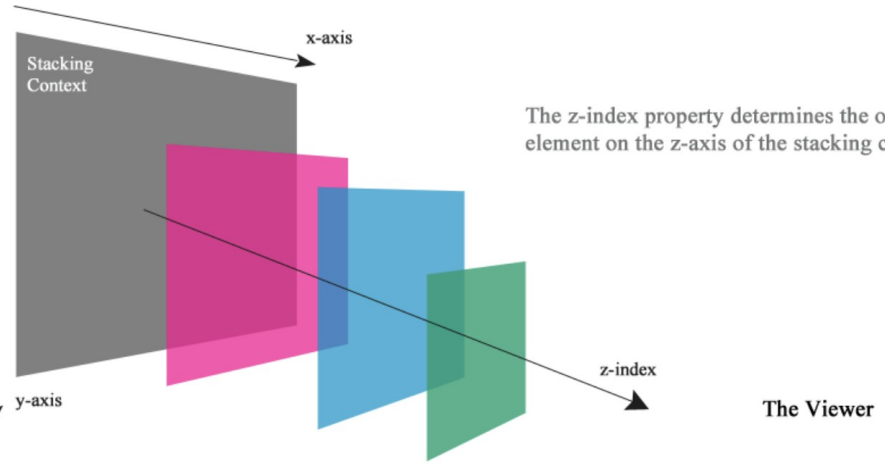
Das Element bleibt solange “normalen Fluss”, bis es die top-position beim scrollen erreicht hat und klebt dann an dieser Position fest.

1- Klebt fest bei erreichen von top-position.
2- Erscheint über jedem static-Element.
3- Kann z-index bekommen.
4- Braucht content oder width/height um an Größe zu haben.

CSS Position - z index

```
div {
  position: relative | absolute |
            fixed | sticky

  z-index: number
}
```

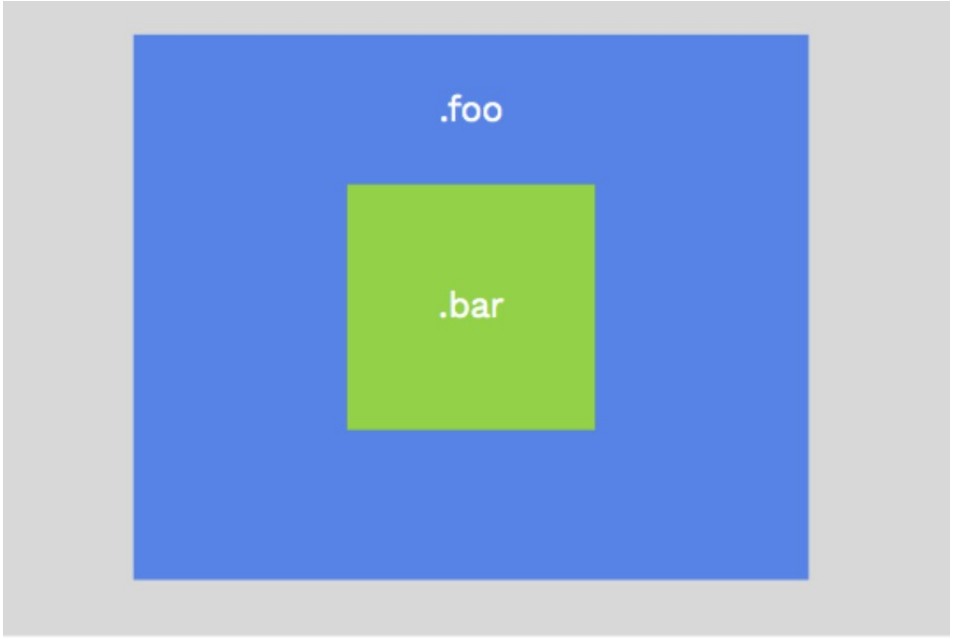


The z-index property determines the order of the element on the z-axis of the stacking context.

z-index

Der z-index legt die z-Reihenfolge des positionierten Elements (nicht position: static) und seiner Kinder fest.

CSS Position - z index



Stacking-Context

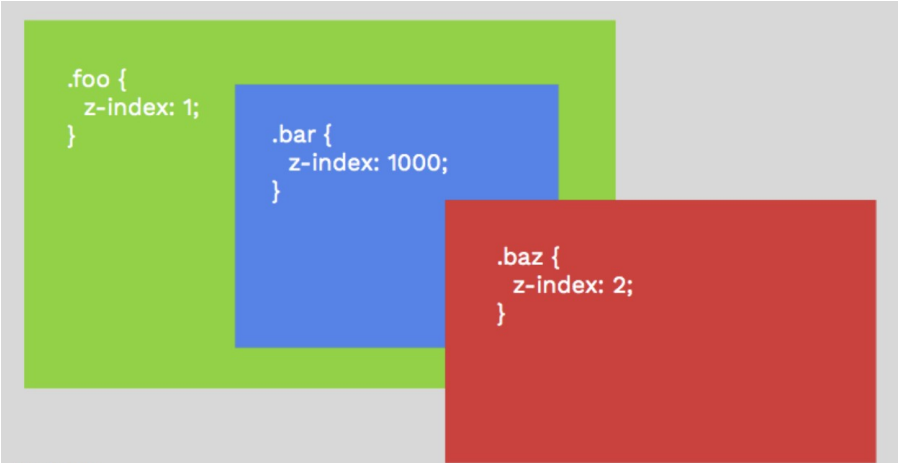
Wir erstellen einen Stacking-Context, indem wir den z-Index eines Elements auf eine beliebige ganze Zahl setzen. Zwei Dinge passieren:

1- Setzen des z-Index des aktuellen Elements.
2- Erstellen eines neuen Stacking-Context für sich und seine Kind-Elemente. Die Kinder liegen nun innerhalb des Parent-Stacking-Context.

```
<div class="foo">
   <div class="bar"></div>
</div>
```

```
.foo { position: relative; z-index:
1; }
```

CSS Position - z index



z-index

Stacking-Context

Achtung: Kind-Elemente bleiben innerhalb des Stacking-Context vom Parent-Element.
Das blaue Element liegt nicht über dem Roten!

```
<div class="foo">
  <div class="bar"></div>
</div>

<div class="baz">
</div>
```