

Pozycjonowanie elementów

Opływanie elementów

float

Atrybut `float` pozwala na opływanie elementów przez tekst a także ustawiać elementy blokowe w jednej linii.

Domyślnie grafika w tekście nie jest opływana, tak jak pokazuje poniższy przykład:



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Architecto aspernatur consequatur deserunt dolores exercitationem inventore laborum libero necessitatibus odit, quae, quas, quos totam ullam unde voluptates. Amet dolore eos voluptatibus?

Właściwość float może przyjąć trzy wartości:

- `left` - element trafia na lewo, reszta elementów opływa go z prawej strony
- `right` - element trafia na prawo, reszta elementów opływa go z lewej strony
- `none` - wyłączenie opływania (przywrócenie do stanu domyślnego)

Opływanie elementów

```
img {  
  float:left;  
}
```



Tytuł tekstu

Lorem ipsum dolor sit amet,
consectetur adipisicing elit.
Architecto aspernatur
consequatur deserunt dolores
exercitationem inventore
laborum libero necessitatibus
odit, quae, quas, quos totam
ullam unde voluptates. Amet dolore eos voluptatibus?

Jak widzisz powyżej, po zastosowaniu tej właściwości dla grafiki, reszta elementów (w tym blokowy tytuł `h2`) zaczyna opływać daną grafikę z prawej strony.

Podobne zachowanie możemy wymusić na elementach blokowych. Dzięki właściwości `float` elementy blokowe zaczną układać się obok siebie:

```
div {  
  border:1px solid dodgerblue;  
  width: 100px;  
  height: 100px;  
  background: rgba(30,144,255, 0.5);  
  
  float:left;  
}
```



Niepożądane efekty

Właściwość `float` nigdy nie została stworzona do budowania layoutów, przez co podejście z poprzedniego slajdu bardzo często stwarza niepożądane efekty.

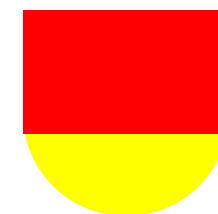
Pierwszym z nich jest fakt, że jeżeli wszystkie dzieci danego elementu mają nadany `float`, wtedy rodzic traci swoją wysokość, a kolejne elementy wchodzą na jego miejsce.

Przykład

```
<div class="container">
  <div class="box-left"></div>
  <div class="box-right"></div>
</div>
<div class="circle"></div>
```

```
.box-left {
  float: left;
}
.box-right {
  float: right;
}
.circle {
  width: 100px;
  height: 100px;
  background: yellow;
  border-radius: 50%;
}
```

Efekt



clearfix

Aby go naprawić musimy na końcu rodzica z floatowanymi elementami dodać element z właściwością `clear: both`.

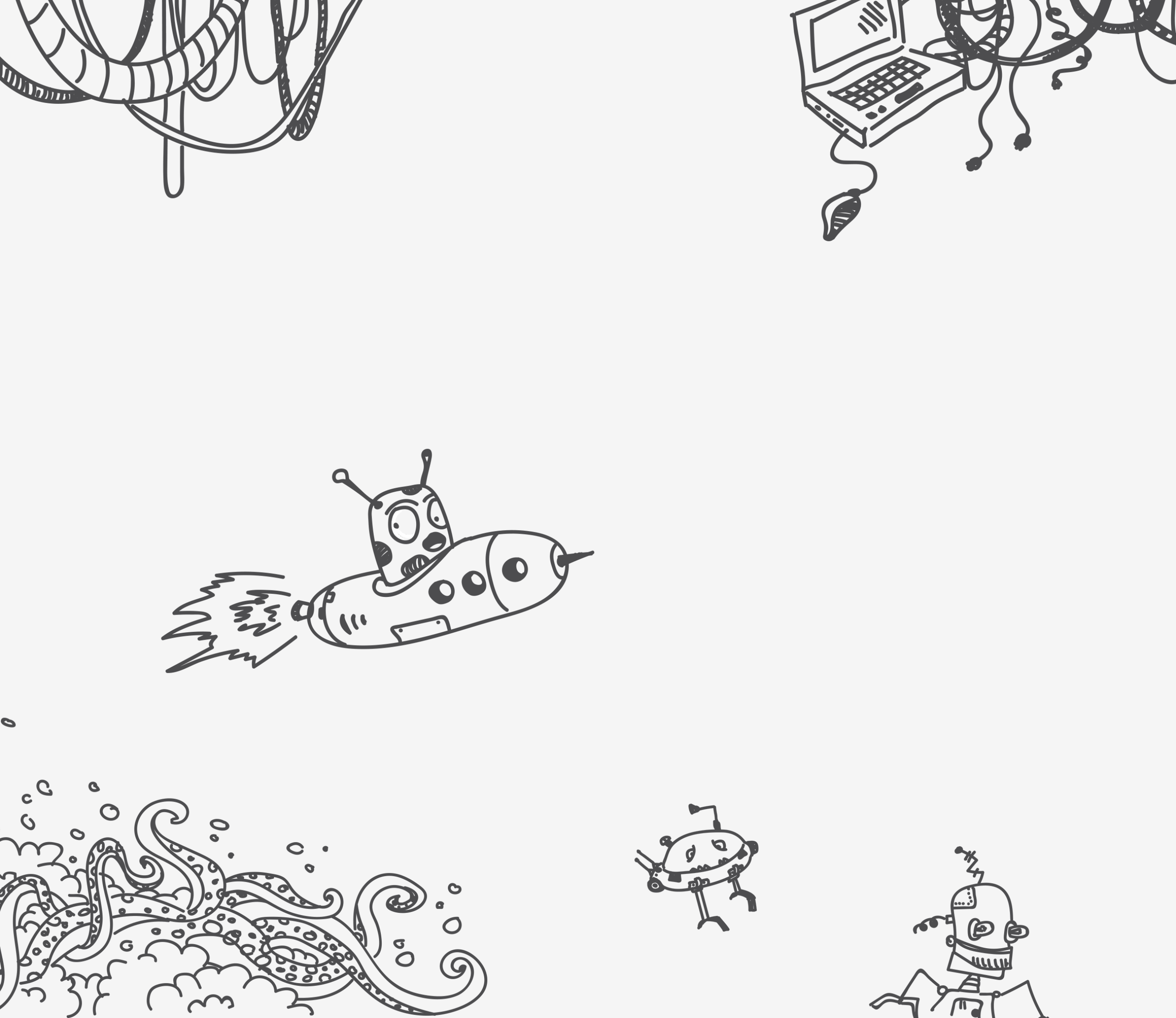
Aby nie dodawać niepotrzebnego elementu do HTML możemy skorzystać z pseudoelementu. Dodajemy dodatkową klasę do rodzica elementów, które korzystają z float.

```
<div class="container clearfix">
  <div class="box-left"></div>
  <div class="box-right"></div>
</div>
<div class="circle"></div>
```

```
.clearfix::after{
  content: "";
  display: block;
  clear: both;
}
```

Efekt





position

position

Domyślnie każdy element ma ustawioną pozycję statyczną – `static`.

Jest to naturalne ułożenie elementów na stronie, czyli od lewej do prawej (`inline`) i od góry do dołu (`block`).

Inne wartości position to:

- `relative`
- `absolute`
- `fixed`
- `sticky`

position: relative

Jest to podobne ułożenie jak `static`, z tym że możemy przesuwąć elementy z ich oryginalnej pozycji. Korzystamy przy tym z właściwości:

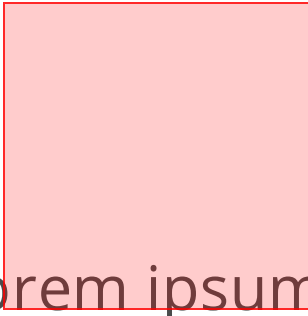
- `top`
- `right`
- `bottom`
- `left`

Przykład

```
<div class="container">
  <div class="child"></div>
  <p>Lorem ipsum dolor sit amet, consectetur
  adipisicing elit. Earum enim ex provident
  quos repudiandae totam?</p>
</div>
```

```
.child {
  width: 150px;
  height: 150px;
  border: 1px solid rgba(255, 0, 0, 0.8);
  background: rgba(255, 0, 0, 0.2);
  position: relative;
  left: 30px;
  top: 30px;
}
```

Efekt



>Lorem ipsum dolor sit amet, consectetur adipisicing elit.
Earum enim ex provident quos repudiandae totam?

position: absolute

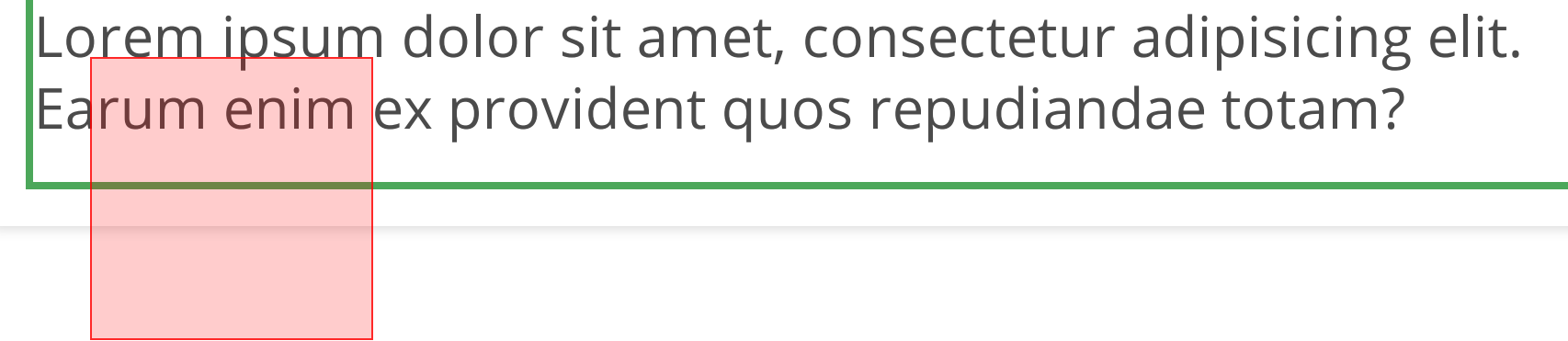
Elementy z ustawionym `position: absolute` są wyjęte z biegu dokumentu i akceptują przesunięcie tak jak relative:

- `top`
- `right`
- `bottom`
- `left`

Element z `position: absolute` jest pozycjonowany względem najbliższego rodzica (zielona ramka) z pozycjonowaniem innym niż static.

```
.child {  
  position: absolute;  
}
```

Efekt



Lorem ipsum dolor sit amet, consectetur adipisicing elit.
Earum enim ex provident quos repudiandae totam?

position: fixed

Elementy z ustawionym `position: fixed` działają podobnie jak `absolute`, z tym że są pozycjonowane względem okna przeglądarki.

Podczas przewijania strony element z `position: fixed` jest „przyczepiony” do okna.

Lorem, ipsum dolor sit amet consectetur adipisicing elit. Voluptatem dolores minus necessitatibus nostrum rem? Neque aperiam unde autem nam odit beatae cum id vero aut fugit, dolorum adipisci suscipit esse!

Bacon ipsum dolor amet ham hock landjaeger tail hamburger kielbasa. Chuck hamburger short ribs, cupim frankfurter tri-tip drumstick tongue rump brisket jerky ground round fatback corned beef. Short loin short ribs spare ribs porchetta, sausage alcatra ball tip turkey pork loin. Alcatra sirloin prosciutto, ribeye bresaola flank fatback. Beef ribs prosciutto alcatra spare ribs porchetta sirloin corned beef drumstick cow pig ham hock biltong short loin short ribs. Chicken tri-tip rump filet mignon short loin salami.

Alcatra landjaeger andouille chuck brisket. Pork belly ham capicola, tri-tip picanha venison hamburger beef ribs beef pastrami short ribs tongue jowl. Picanha shankle pig sausage ball tip pancetta. Picanha beef ribs pastrami jerky, biltong doner sirloin tri-tip tenderloin leberkas shankle sausage.

Rump capicola pork, beef chuck ham doner pork loin. Chicken boudin landjaeger, prosciutto sausage jowl ham doner ground round kevin fatback cow drumstick beef ribs. Ham tri-tip picanha, quince short ribs jerky filet mignon leberkas strip steak veni-

Element zostanie w tym miejscu, niezależnie od scrollowania strony.

Podobnie obrazek pod spodem.



Powyższy przykład live:

<https://codepen.io/Coders-Lab/full/XWaLvww>

position: sticky

Pozycjonowanie `sticky` jest pośrednim pomiędzy `relative` i `fixed`.

Element ma swoją domyślną pozycję do czasu, aż dojdzie do krawędzi ekranu. Wtedy zostaje do niej przyklejony.

```
nav {  
  position: sticky;  
}
```



Powyższy przykład live:

<https://codepen.io/Coders-Lab/full/ZEJdgdZ>

position absolute a punkt zaczepienia

Element, któremu nadamy inną właściwość niż `position: static` staje się kontenerem zawierającym. Dzięki temu elementy w nim zawarte, którym nadaliśmy `position: absolute`, będą ustawiane względem jego krawędzi a nie krawędzi elementów nadrzędnych (np. body).

Najczęściej używa się `position: absolute` wraz z `position: relative` w elemencie nadrzędnym.

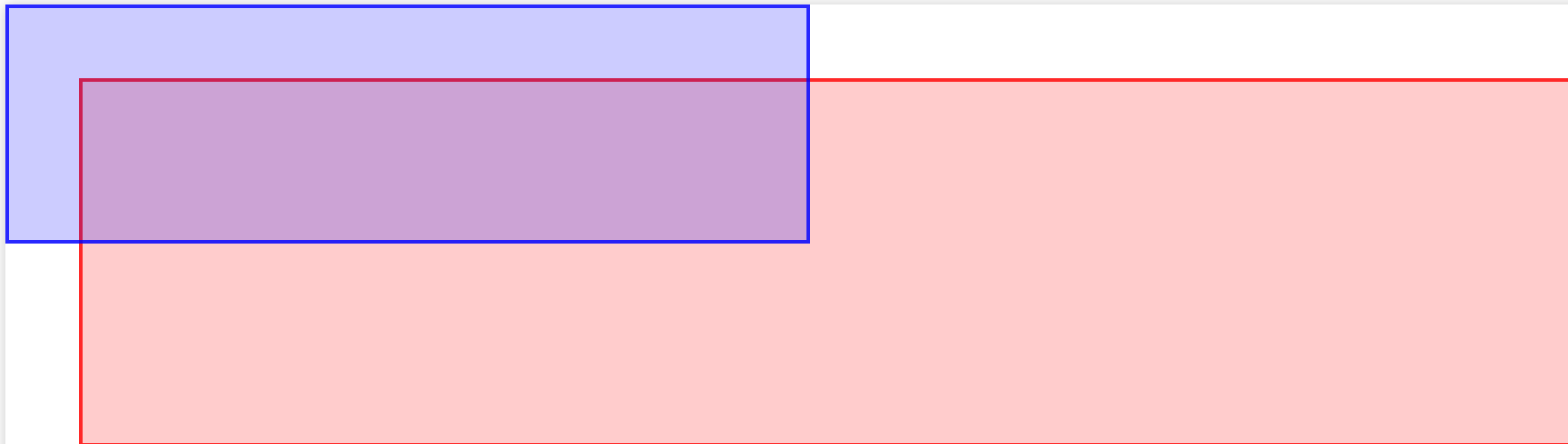
Przykład na kolejnym slajdzie.

position absolute a punkt zaczepienia

static

```
.child {  
  position: absolute;  
  top: 0;  
  left: 0;  
}
```

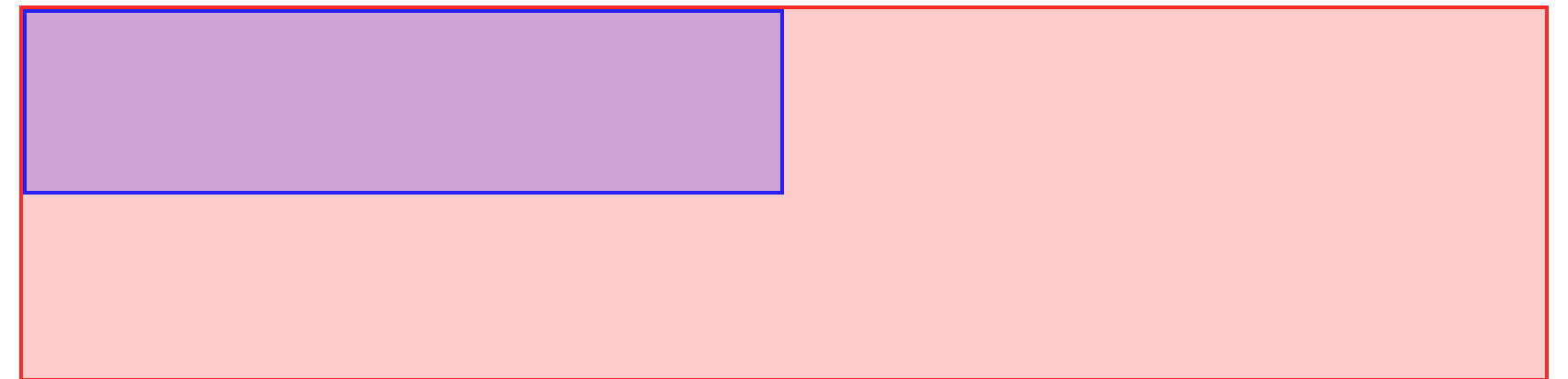
```
.parent {  
  position: static;  
}
```



relative

```
.child {  
  position: absolute;  
  top: 0;  
  left: 0;  
}
```

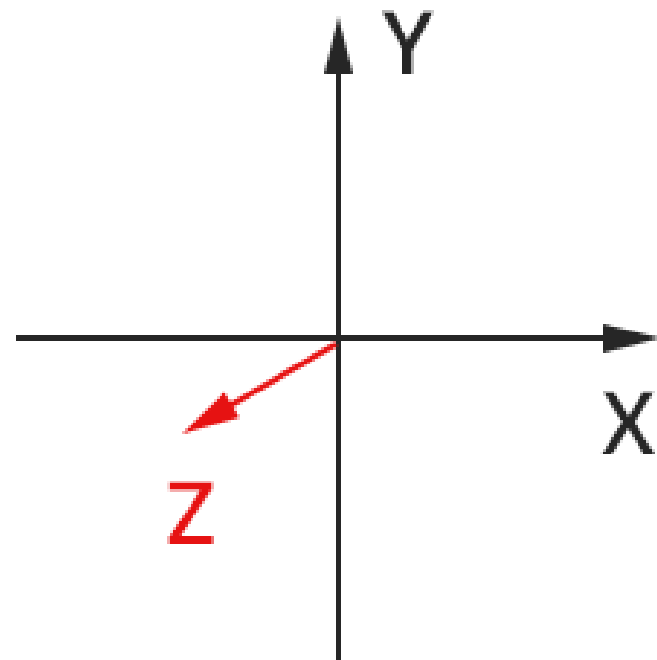
```
.parent {  
  position: relative;  
}
```



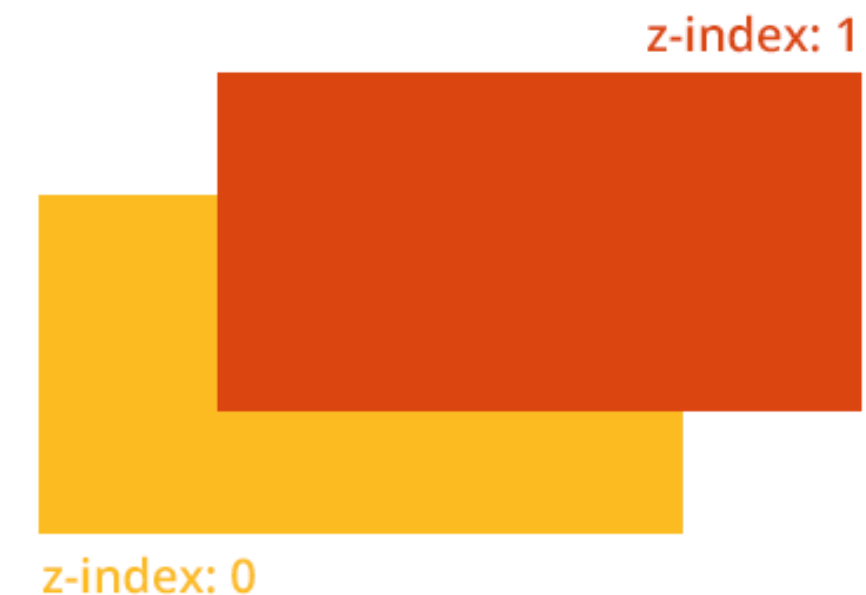
z-index

Trzeci wymiar

Do tej pory przesuwaliśmy elementy w lewo, prawo (oś X) oraz góra, dół (oś Y). Możemy przesuwać elementy również po osi Z.



Dzięki `z-index` możemy układać elementy warstwowo. `z-index` działa dla elementów, które mają ustawioną własność `position` na inną niż `static`.



Interaktywny przykład:

<https://codepen.io/Coders-Lab/full/dyzBxxp>