Как работает position: sticky

По первым четырём значениям позиционирования всё интуитивно понятно и просто. Основное различие между `static` и `relative`, `absolute` и `fixed` в том, какое место они занимают в потоке документа. Элементы с позицией `static` и `relative` сохраняют своё место в потоке документа, в то время как `absolute` и `fixed` «вырываются» из потока документа и становятся плавающими.

Значение `sticky` похоже на все предыдущие значения сразу и в то же время имеет свои особенности. Разберём в деталях, как именно липкое позиционирование работает.

Сразу стоит определить три вещи: сам «липкий» элемент, его контейнер, который мы дальше будем звать липким контейнером, и позицию, начиная с которой элемент будет становиться липким.

«Липкий» элемент — это элемент, которому мы задали `position: sticky`. «Липкий» элемент может залипать только в пределах родительского контейнера.

«Липкий» контейнер — это элемент, который обрамляет «липкий» элемент. Это максимальная область, в которой может перемещаться наш элемент. Липкий контейнер автоматически становится родитель липкого элемента. Если элемент с «липким» позиционированием перестаёт залипать, это может быть связано с недостаточными размерами родительского элемента.

Элемент будет становиться «липким», как только пользователь при скролле веб-страницы достигнет определённой позиции родителя «липкого» элемента (`left`, `top`, `right` или `bottom` отличное от `auto`), например `top: 20px`. До этого момента «липкий» элемент можно рассматривать как обыкновенный позиционированный.

Если подытожить, для того чтобы элемент стал «липким», необходимо:

— задать для элемента `position: sticky`,

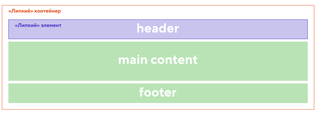
— размеры непосредственного родителя позволяют элементу становиться «липким»,

— указать для элемента одно из CSS-свойств `left`, `top`, `right` или `bottom` отличное от `auto`.

Без перечисленных трёх составляющих «липкое» позиционирование не работает.

Разберём на примерах.

Для начала просто сделаем элемент «липким». Представим, что у нас есть сайт, который состоит из шапки, основного содержимого и подвала. Шапку сделаем «липкой», причём она будет залипать, как только `body` достигнет позиции `top: 20px`. «Липкий» контейнером в этом случае будет тег `body`.



«Липкая» шапка на сайте

Разметка:

```
<body>
  <header class="header">header</header>
  <main>main content</main>
  <footer>footer</footer>
</body>
```

Стили:

```
.header {
  position: sticky;
  top: 20px;
}
```

Рассмотрим ещё один пример, неудачный.

Мы хотим сделать «липкими» элементом не всю шапку, а только список ссылок на разделы сайта. Но меню находится внутри тега `nav`. По высоте эти элементы примерно совпадают. В этом случае `position: sticky` не даст желаемого результата. При прокрутке страницы «липкий» контейнер быстро уйдёт из области видимости и увидеть за собой «липкий» элемент.

«Липкий» элемент и «липкий» контейнер совпадают по высоте

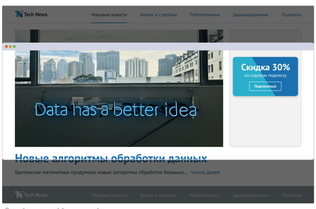
Разметка:

```
<nav class="nav">
  <a href="#">
    <img src="img/logo.svg" width="120" height="50" alt="">
  </a>
  <ul class="sticky-list">
    <li>...</li>
    <li>...</li>
    <li>...</li>
    ...
  </ul>
</nav>
```

Стили:

```
.sticky-list {
  position: sticky;
  top: 0;
}
```

На изображении ниже «липким» элементом должен быть рекламный баннер. Серой рамкой выделен «липкий» контейнер, отдельно показаны размеры экрана. При этой компоновке элементов баннер будет становиться плавающим — размеры «липкого» контейнера позволяют.



«Липкий» рекламный баннер на сайте

```
<body>
  <header>шапка</header>
  <main>
    <div class="wrapper">
      <article class="main-article">Основная статья</article>
      <aside class="aside">
        <div class="banner">баннер</div>
      </aside>
    </div>
    ...
  </main>
  ...
</body>
```

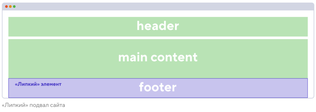
```
.wrapper {
  display: flex;
}

.main-article {
  width: 75%;
}

.aside {
  width: 25%;
}

.banner {
  position: sticky;
  top: 0;
}
```

В большинстве случаев вы будете использовать `position: sticky`, чтобы прикрепить элемент к верхнему краю страницы. Что-то вроде этого:

```
.component {
  position: sticky;
  top: 0;
}
```

Но с помощью `position: sticky` можно прикрепить элемент и к нижнему краю страницы. К примеру, подвалу сайта можно задать «липкое» позиционирование, и при скролле он всегда будет залипать к нижнего краю. И когда мы дойдём до конца «липкого» контейнера, подвал сайта остановится на своей естественной позиции. Эти особенность лучше использовать для элементов, которые находятся в самом конце контейнера.

Пример:



«Липкий» подвал сайта

HTML

```
<main class="main-container">
  <header class="main-header">header</header>
  <div class="main-content">main content</div>
  <footer class="main-footer">footer</footer>
</main>
```

CSS

```
.main-footer {
  position: sticky;
  bottom: 0;
}
```

В реальной жизни такое поведение можно использовать для сводных таблиц. Обычно объём данных в сводных таблицах большой, и когда шапка таблицы уходит из поля зрения, сложно понять, что за данные отображаются в той или иной ячейке таблицы. За счёт `position: sticky` можно зафиксировать шапку таблицы с названиями колонок, это улучшит восприятие информации.

▼ Застройщик	Юридический адрес	ИНН
ООО «НАШ РАЙОН»	г. Москва, Нахимовский проспект, дом 8, строение 23	1234567890
ООО «СТРОЙЭТАЖ»	г. Владимир, улица Горького, дом 2, корпус 7, комната 1	0987654321

Сводная таблица с «липкой» горизонтальной шапкой

```
<table class="developers">
  <tr class="developers-row developers-row-sticky">
    <th>Застройщик</th>
    <th>Юридический адрес</th>
    <th>ИНН</th>
  </tr>
  <tr class="developers-row">
    <td>ООО «НАШ РАЙОН»</td>
    <td>г. Москва, Нахимовский проспект, дом 8, строение 23</td>
    <td>1234567890</td>
  </tr>
  <tr class="developers-row">
    <td>ООО «СТРОЙЭТАЖ»</td>
    <td>г. Владимир, улица Горького, дом 2, корпус 7, комната 1</td>
    <td>0987654321</td>
  </tr>
  ...
</table>
```

```
.developers-row-sticky {
  position: sticky;
  top: 10px;
}
```

С помощью `position: sticky` можно фиксировать не только заголовочные строки сводных таблиц, но и заголовочные колонки. Это может быть удобно в случае, когда таблицу нужно скролить не только сверху вниз, но и вправо/влево. Для этого нужно обозначить, соответственно, отступ справа или слева: например, `right: 10px` или `left: 10px` для того, чтобы элемент становился липким на расстоянии в 10 пикселей от боковой стороны своего родительского контейнера.

Браузерная поддержка

До 2020 года, чтобы использовать свойство `position: sticky` и быть уверенным, что оно сработает приходилось подключать **полифилы** *(полифиллсанит. Polyfill). Самый популярный из них Stickyfill. В настоящее время Stickyfill не поддерживается и в последнем своём коммите авторы написали:*

> Unmaintained! Stickyfill did a good job while the browsers were implementing position: sticky support. You can now safely use stickies without a polyfill, all modern browsers support them natively.

*Перевод: Необслуживаемый! **Stickyfill** проделал хорошую работу, пока браузеры реализовали поддержку `position: sticky`. Теперь вы можете использовать `position: sticky` без полифила, все современные браузеры его поддерживают из коробки.*

У свойства и на самом деле хорошая браузерная поддержка, в чём можно убедиться на CanIUse.

Позиционирование
z-index