Селекторы

В прошлых уроках и упражнениях мы использовали достаточно простые селекторы. Они позволяли выбрать элементы, к которым будут применены CSS правила. Вы уже умеете выбрать элемент по его тегу, классу, идентификатору и найти вложенный элемент.

CSS даёт намного большие возможности по выбору элементов. В этом уроке мы изучим самые популярные селекторы второго и третьего уровня спецификации W3C. В конце урока будут оставлены ссылки на спецификации, в которых вы сможете узнать и о других селекторах.

Выбор соседнего элемента

Ранее, использование селекторов указывало, какой конкретно элемент мы хотим выбрать и где он находится относительно своих родительских блоков. В большинстве случаев этого достаточно для точного указания элемента, но бывают случаи, когда нужно выбрать соседний элемент, а не вложенный. Например,

```
<section>
  <div class="time">...</div>
  <div class="timer">...</div>
</section>
```

Элемент `.timer` полностью зависит от блока `.time`. Предположим, что стиль таймера меняется в зависимости от наличия элемента с классом `.time`. Есть несколько путей решения такой задачи:

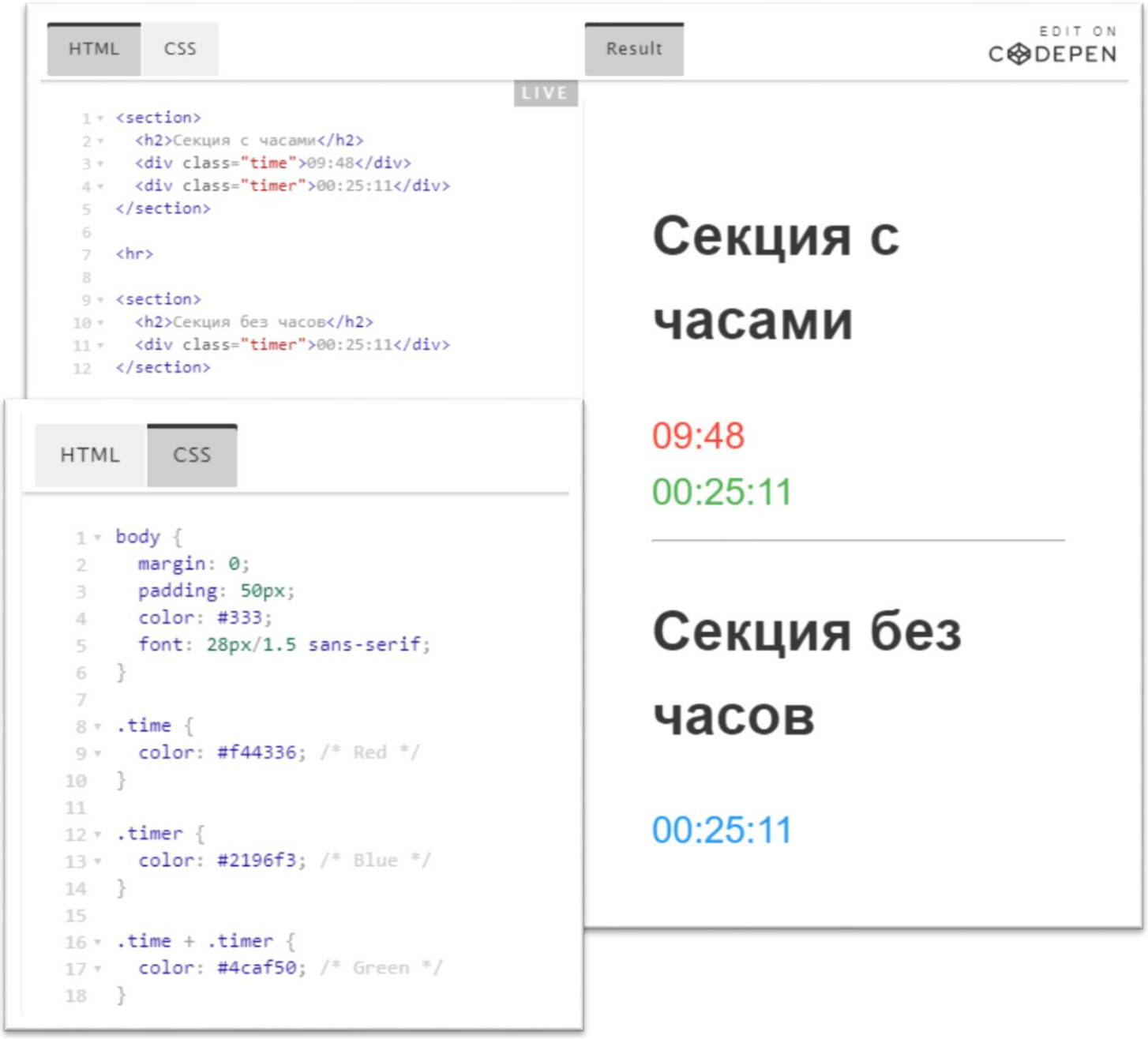
1. Положить оба элемента в единого родителя и дать им уникальные классы для разных ситуаций.
2. Воспользоваться селектором соседнего элемента.

Чаще всего, именно первый вариант будет предпочтительным. При этом вы не раз столкнётесь с ситуациями, когда это невозможно. Такое может происходить при динамическом добавлении элементов на страницу. И тут на помощь приходят селекторы.

В CSS существует два селектора для выбора элемента, который лежит рядом с другим элементом:

- `A + B` — выбор элемента *B*, который находится непосредственно после элемента *A*. Такой селектор называется *смежным* или *соседним*
- `A ~ B` — выбор элемента *B*, который находится на том же уровне вложенности, что и *A*. При этом они имеют общего родителя и элемент *B* находится после элемента *A* в HTML. Такой селектор называется *родственным*.

Для примера выше отлично подойдёт смежный селектор. Элементы `.time` и `.timer` идут друг за другом и являются дочерними элементами одного и того же родителя. Стилизуем элемент `.timer` в зависимости от существования элемента `.time`:

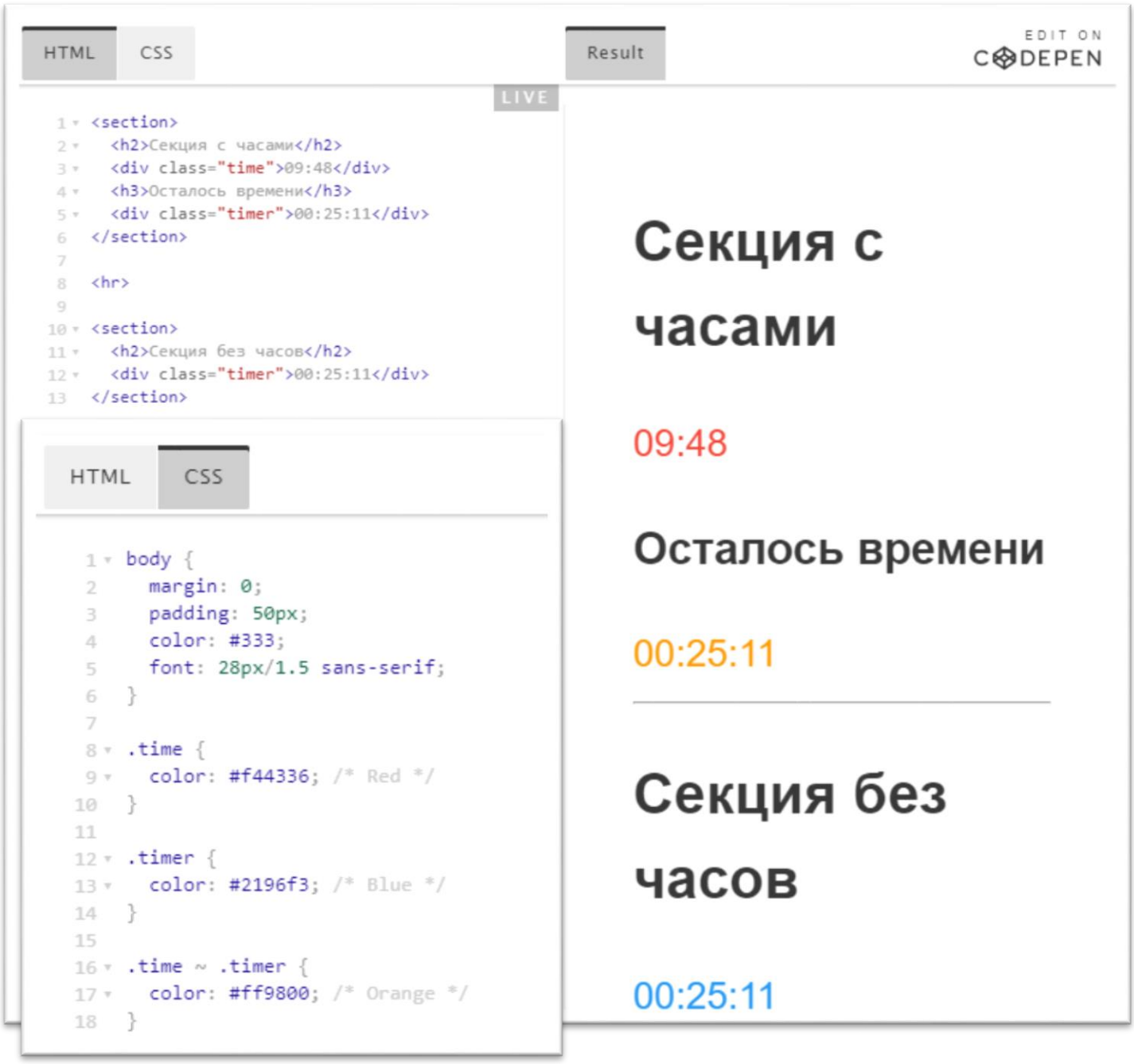


Родственный селектор позволяет немного усложнить ситуацию. Ведь теперь будет возможность не просто выбрать соседний элемент, а элемент, лежащий на том же уровне. Изменим пример, который позволит наглядно продемонстрировать возможность родственного селектора.

```
<section>
  <h2>...</h2>
  <div class="time">...</div>
  <h3>...</h3>
  <div class="timer">...</div>
</section>
```

Схематически CSS будет выглядеть следующим образом:

```
.time ~ .timer {
  /* Стили элемента */
}
```



Селекторы по атрибуту

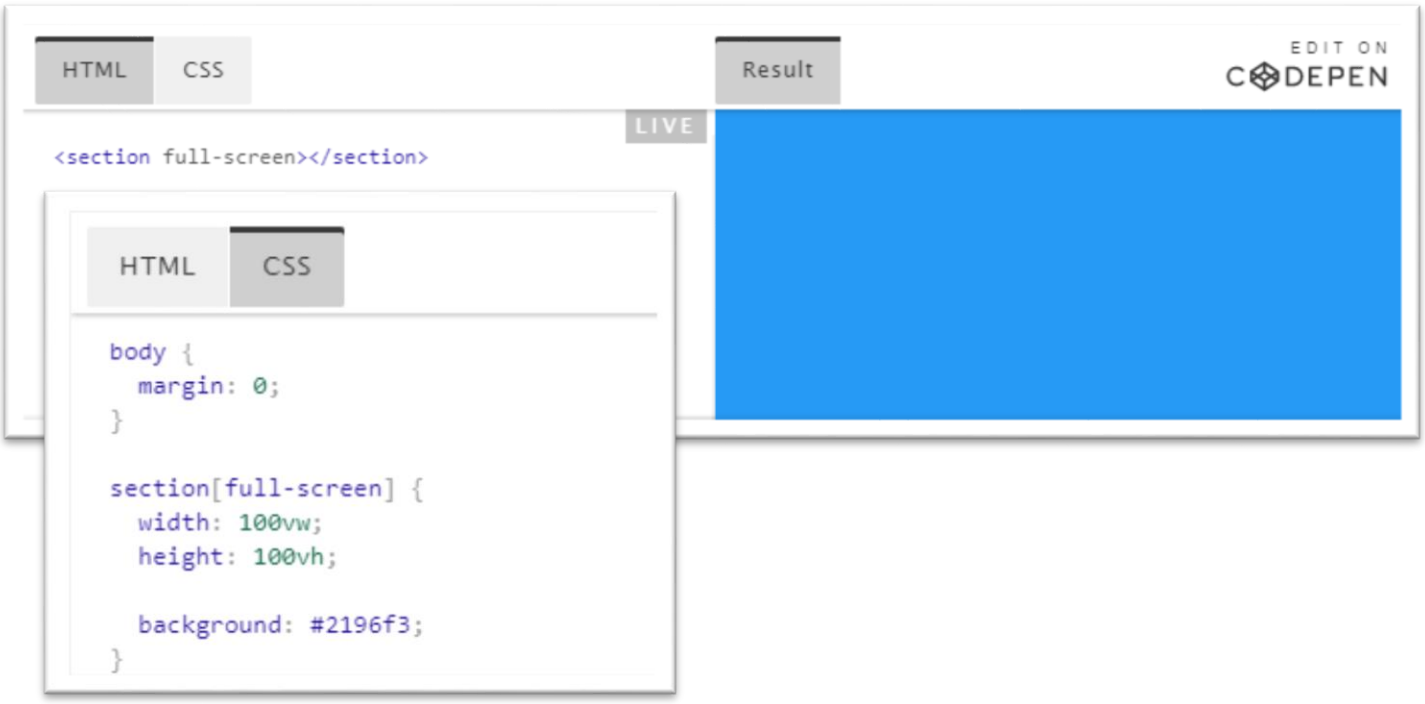
Атрибуты — неизменная часть работы верстальщика и фронтенд-разработчика. Они не всегда несут в себе семантический смысл, как например атрибут alt, title, class, id и так далее. HTML позволяет добавлять любые пользовательские атрибуты и работать с ними. Для стилизации таких элементов в CSS существуют специальные селекторы.

Самый простой селектор по атрибуту просто выбирает элемент по его атрибуту. Атрибут записывается внутри квадратных скобок.

```
<section full-screen></section>
```

```
section[full-screen] {
  width: 100vw;
  height: 100vh;

  background: #2196f3;
}
```



Можно выбирать не только по названию атрибута, но и по его значению. В этом случае рядом с именем атрибута указывается его значение в следующем синтаксисе:

```
<section full-screen="true"></section>
section[full-screen="true"] {
  width: 100vw;
  height: 100vh;

  background: #2196f3;
}
```

С опытом вы заметите, что многие JavaScript библиотеки работают именно с пользовательскими атрибутами. Это позволяет добиться изолированности компонентов и их удобного переиспользования.

Бывают ситуации, когда в HTML есть группа элементов с одинаковыми именами атрибутов, но с разными значениями. Причём некоторые из них могут быть похожи друг на друга, составляя части одного компонента. Например,

```
<section nm-section="catalog"></section>
<section nm-section="catalog-popular"></section>
<section nm-section="catalog-new"></section>
```

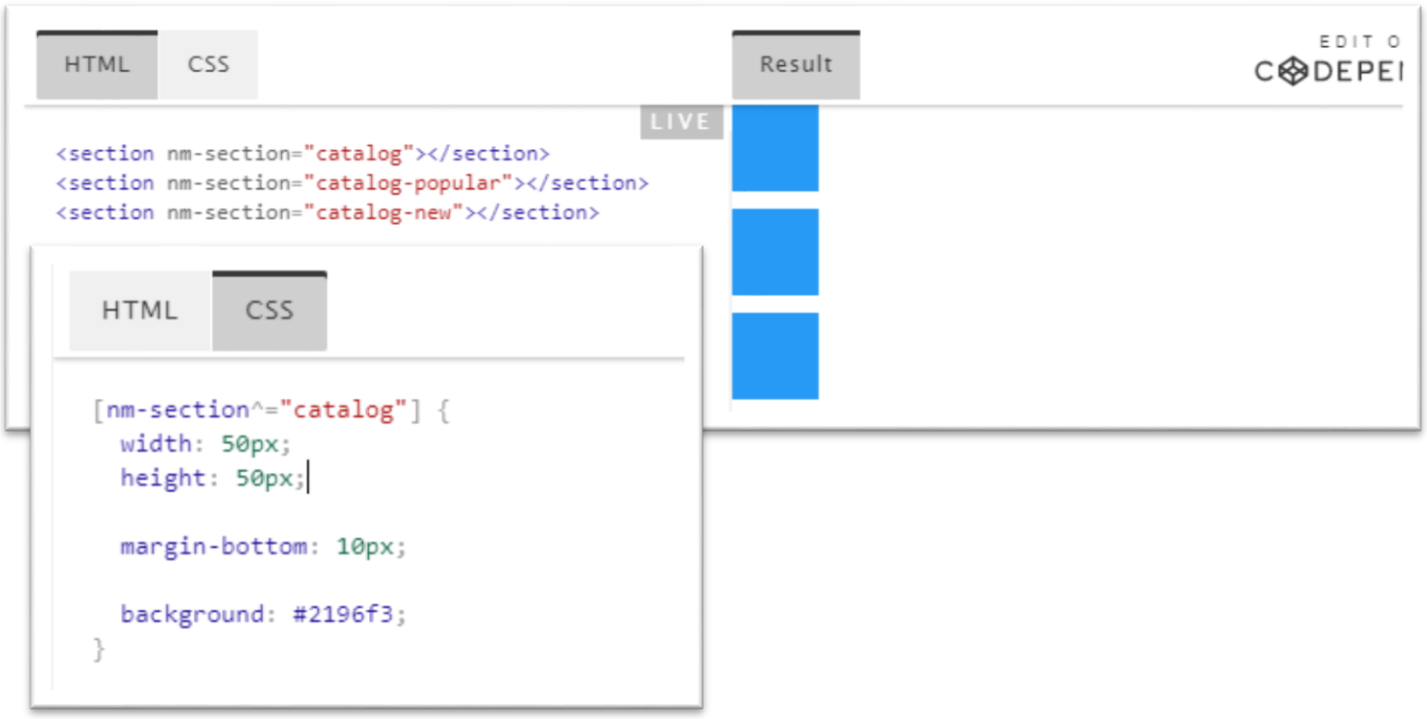
Все три секции, по своей логике, будут иметь похожее оформление. Можно добавить всем одинаковый класс, но существует одна проблема: если элементы добавляются динамически, с помощью JS, то есть вероятность существования такого же класса внутри проекта. Это приведёт к коллизии, когда один селектор перебьёт свойства другого. Поэтому и используются атрибуты.

Все три секции имеют одинаковую приставку *catalog*. Это поможет отделить их от остальных названий секций с помощью конструкции `[nm-section^="catalog"]`. Такой селектор выберет все элементы с атрибутом *nm-section* значение которого **начинается** с *catalog*.

```
[nm-section^="catalog"] {
  width: 50px;
  height: 50px;
```

```
  margin-bottom: 10px;

  background: #2196f3;
}
```



Есть ещё несколько похожих конструкций, которые ищут «вхождение» подстроки в строку:

- `[nm-section$="catalog"]` — вхождение строки в конце значения атрибута.
- `[nm-section*="catalog"]` — вхождение строки в любом месте значения атрибута.