Селекторы

Селекторы позволяют очень точно указывать к каким элементам применять CSS-свойства. Селекторы — это наш прицел, наведенный на элементы HTML-документа, сейчас мы будем учиться им пользоваться. Начнем нашу стрелковую тренировку! В каждом задании вам нужно будет с помощью селекторов закрывать мишени.

Для начала представим, что механизма селекторов не существует. И в этом случае задавать CSS-свойства элементам приходится с помощью атрибута style. Например:

```
<p style="color: red;">...</p>
```

Задание

1. Откройте подготовленную папку с файлами и создайте в ней новый файл с именем style.css
2. Подключите его в index.html
3. Закройте все мишени, добавив к каждому элементу списка атрибут style со значением background-color: white;

Селекторы по тегам

Задавать атрибут style для каждого тега неудобно и долго. Особенно, если тот же результат можно получить с помощью единственного CSS-правила, в котором используется селектор для тега li.

С помощью селекторов по именам тегов можно задать стили для всех элементов списка, изображений, абзацев и так далее. Эти селекторы содержат имя тега без символов < и >. Например:

```
li {
  /* стили для элементов списка */
}
```

Одно правило может относиться сразу к нескольким селекторам, в таком случае селекторы перечисляются через запятую:

```
a, img {
  /* стили для ссылок и изображений */
}
```

Задание

1. В index.html уберите атрибут style у элементов списка;
2. Закройте все мишени. В файле style.css с помощью селектора по имени тега задайте всем элементам списка свойство background-color: white;

Селекторы по классам

Класс — это один из атрибутов тегов. Выглядит он вот так:

```
<li class="first"></li>
```

Этот атрибут особенный, так как в CSS существует возможность выбирать элементы по классу. Делается это с помощью такого селектора: .имя_класса. Например:

```
.first {
  /* стили для класса first */
}
```

Имена классов могут состоять из латинских символов, цифр и знаков - и _. Имя класса должно начинаться с латинской буквы

Задание

В файле index.html

1. Добавьте атрибут class со значением hit для 1,2 и 4 элемента списка
2. Добавьте атрибут miss со значением hit для 3 и 5 элемента списка Для класса

В файле style.css

1. Для класса hit задайте свойство background-color: white; (1, 2, 4 мишени закрыты)
1. Для miss задайте свойство background-color: red; (3, 5 мишени — мимо).

Отрабатываем селекторы по классам

Синтаксис CSS позволяет выбирать элементы не только по одному классу или тегу. Можно, например, выбрать элемент одновременно по тегу и по классу или же элемент с двумя классами сразу. Для этого селектор составляется просто одной строкой из всех желаемых «частей» без пробелов. Давайте рассмотрим примеры.

В селекторе по тегу и классу первым пишется название тега, а потом идёт класс:

HTML
<ul class="target">
CSS
ul.target {...} /* выбор всех тегов ul с классом target */

Если у элемента задано несколько классов, в HTML и в CSS-селекторе они могут идти в разном порядке — это не будет влиять на выборку элементов:

HTML

CSS
.text.green {...} /* выбор элементов с двумя классами: text и green */

Количество классов в селекторе может быть любым:

HTML

CSS
span.underlined.red.big.text {...} /* выбор тегов span с четырьмя классами: underlined, red, big и text */

В этой части ваша задача — раскрасить мишени в определённые цвета, применяя подходящие селекторы.

Всего будет три типа выстрелов в мишень, каждому из которых соответствует свой цвет:

1. Мишень закрыта — background-color: white;
2. Выстрел мимо — background-color: red;
3. Ошибка техники (попадание есть, но мишень не закрылась) — background-color: yellow;

Задание

1. Изменим классы у элементов списка - пронумеруем их по порядку: 1 - first, 2 - second и т.д.
2. Наш спортсмен должен поразить первую мишень, вторая дала сбой, затем два досадных промаха, но пятая мишень закрыта.

Контекстные селекторы

Селектор может состоять из нескольких частей, разделённых пробелом, например:

```
p strong { ... }
ul .hit { ... }
.footer .menu a { ... }
```

Такие селекторы называют контекстными или вложенными. Их используют для того, чтобы применить стили к элементу, только если он вложен в нужный элемент.

Например, селектор .menu a сработает для ссылки a только в том случае, если она расположена внутри элемента с классом .menu.

Читать их проще всего справа налево:

```
/* выбрать все теги strong внутри тегов p */
p strong { ... }

/* выбрать все элементы с классом .hit внутри тегов ul */
ul .hit { ... }

/* выбрать все ссылки внутри элементов с классом .menu,
   которые лежат внутри элементов с классом .footer */
.footer .menu a { ... }
```

Таким образом, можно задавать элементам различные стили в зависимости от их контекста. Если ссылка расположена внутри меню, сделать её крупнее, а если внутри основного текста, то задать ей нужный цвет.

В этом задании вы потренируетесь использовать контекстные селекторы

Задание

1. Для начала изменим стартовые условия - устроим биатлон - дуэль! В файле main-style.css у класса .main измените значение с url("bg.jpg") на url("bg2.jpg")
2. На странице index.html внутри div с классом main создайте новый div с классом shooter-1 и перенесите в него список ul
3. После этого скопируйте ниже наш shooter-1 со всем содержимым, и переименуйте класс в shooter-2
4. Настала очередь дуэли на огневом рубеже:

a. Первый биатлонист закрывает все мишени, кроме второй и пятой.

b. А второй промахнулся по первой, но затем отстрелялся чисто.

```
/* Небольшая подсказка
.shooter-1 .first {
  background-color: white;
}
*/
```

Соседние селекторы

Контекстные селекторы используются для вложенных друг в друга элементов, а соседние — для расположенных рядом.

Например, теги <li> в списке являются соседними по отношению друг к другу и вложенными в тег <ul>.

Соседние селекторы записываются с помощью знака +, например, селектор1 + селектор2. Стили применятся к элементу, подходящему под селектор2, только если сразу перед ним расположен элемент, подходящий под селектор1.

Пример. Есть два элемента списка:

```
<ul>
  <li class="hit"></li>
  <li class="miss"></li>
</ul>
```

Селектор .hit + .miss применит стили к элементу с классом miss, так как перед ним есть элемент с классом hit.

Селектор .hit + li, а также селектор li + .miss, или даже li + li тоже применит стили к элементу с классом miss, то есть ко второму элементу списка.

А вот селектор .miss + .hit не сработает, так как элемент с классом miss находится после элемента с классом hit в разметке

Еще выполняя это задание мы добавили html-тегам классы miss и hit и сейчас попробуем соседскую селективность на них

Задание

1. Очистим стили нашим целям
2. Придадим накала финалу! Пусть:

a. Первый поражает все цели, кроме той, что расположена после .miss

b. Второй поражает все, кроме .miss, расположенной после .hit

Напоминание:

- Мишень закрыта — background-color: white;
- Выстрел мимо — background-color: red;

Дочерние селекторы

Потомком называются любые элементы, расположенные внутри родительского элемента. А дочерними элементами называются ближайшие потомки. Взгляните на пример:

```
<ul>
  <li><span>...</span></li>
  <li><span>...</span></li>
</ul>
```

По отношению к списку <ul> элементы <li> являются дочерними элементами и потомками, а — потомки, но не дочерние элементы.

Контекстные селекторы влияют на всех потомков, что не всегда удобно. Иногда необходимо задать стили только для дочерних элементов. Особенно это полезно при работе с вложенными списками.

Для этого существует дочерний селектор, в котором используется символ >. Например: ul > li или ul > li > span.

Псевдоклассы

Псевдоклассы — это дополнения к обычным селекторам, которые делают их еще точнее и мощнее. Обычный селектор — это снайперский прицел, а с псевдоклассом он становится прибором ночного видения.

Псевдокласс добавляется к селектору с помощью символа :, вот так

селектор:псевдокласс.

Например:

```
a:visited { ... }
li:last-child { ... }
.alert:hover { ... }
```

Знакомство с псевдоклассами мы начнём с first-child и last-child.

Псевдокласс first-child позволяет выбрать первый дочерний элемент родителя, а last-child — последний дочерний элемент. Например:

```
li:last-child { ... }
```

Этот селектор выберет последний элемент списка. Снова вернемся к нашей паре спортсменов

Задание

1. Очистим стили нашим целям
2. Комбинируя контекстный селектор и селектор по псевдоклассу:
   a. Первый поражает все цели, кроме последней
   b. Второй поражает все, кроме .первой

Псевдокласс :nth-child

Псевдоклассы из предыдущего примера относятся к семейству псевдоклассов, помогающих выбирать элементы по их расположению.

Вспомним это задание . В нём каждому тегу <li> был задан собственный класс. Используя классы, мы могли выбрать любой из пяти тегов. Если бы тегов было десять, то пришлось бы использовать десять разных классов.

С помощью псевдокласса nth-child можно выбирать теги по порядковому номеру, не используя классы. Синтаксис псевдокласса: селектор:nth-child(выражение). Выражением может быть число или формула. Например:

```
li:nth-child(2) { ... }
li:nth-child(4) { ... }
li:nth-child(2n) { ... }
```

Первый селектор выберет второй элемент списка, второй селектор — четвёртый элемент списка, третий селектор — все чётные элементы

Задание

1. У второго стрелка зафиксирована ошибка техники (попадание есть, но мишень не закрылась). Задайте background-color: yellow для 3ей цели

Псевдокласс :hover

Некоторые псевдоклассы позволяют выбирать элементы, с которыми взаимодействует пользователь. Сначала познакомимся с псевдоклассом :hover.

Этот псевдокласс позволяет выбрать элемент, когда на него наведён курсор мыши и кнопка мыши не нажата. Примеры:

```
a:hover { ... }
tr:hover { ... }
.menu-item:hover { ... }
```

Первый селектор выбирает ссылку, второй строку таблицы, третий элемент с классом menu-item, но только в том случае, если на них наведён курсор мыши.

Благодаря этому псевдоклассу можно добавлять в интерфейс динамику и интерактивность, так как элементы начинают реагировать на действия пользователя, изменяя свой внешний вид

Задание

1. Снова чистим стили целям
2. И пробуем целиться сами - для элементов списка li при наведении меняйте цвет фона на lightgray
3. Обновив страницу наведите курсор мыши на любую цель

Селектор по id

Существует ещё один HTML-атрибут, для которого существует специальный селектор. Этот атрибут id (идентификатор), а селектор записывается с помощью символа #, например, #some-id.

На значение id распространяются те же ограничения, что и на имя класса. Также id должен быть уникальным на странице.

HTML

```
<p id="greeting">Приветствие!</p>
```

CSS

```
#greeting { ... }
```

Использование селекторов по id при оформлении считается плохой практикой. Существуют редкие исключения из этого правила, например, при оживлении слайдера на чистом CSS

Конспект «Селекторы. Знакомство»

Селекторы позволяют точно указывать к каким элементам применять CSS-свойства.

Без использования селекторов стили можно задать при помощи атрибута style.

...

Селекторы по тегам

Селекторы по именам тегов задают стили для всех элементов списка, изображений, абзацев и так далее. Эти селекторы содержат имя тега без символов < и >. Например:

```
li {
  /* стили для элементов списка */
}
```

Если правило относится сразу к нескольким селекторам, то селекторы перечисляются через запятую:

```
a, img {
  /* стили для ссылок и изображений */
}
```

Селекторы по классам

Можно задавать стили по классу элемента. Делается это с помощью такого селектора: .имя_класса. Например:

```
.first {
  /* стили для класса first */
}
```

Имена классов могут состоять из латинских символов, цифр и знаков - и _. Имя класса должно начинаться с латинской буквы.

Синтаксис CSS позволяет выбирать элементы одновременно по тегу и по классу или же элемент с двумя классами сразу. Для этого селектор составляется просто одной строкой из всех желаемых «частей» без пробелов.

В селекторе по тегу и классу первым пишется название тега, а потом идёт класс:

<ul class="target">

ul.target {...} /* выбор всех тегов ul с классом target */

Если у элемента задано несколько классов, в HTML и в CSS-селекторе они могут идти в разном порядке.

.text.green {...} /* выбор элементов с двумя классами: text и green */

Количество классов в селекторе может быть любым:

span.underlined.red.big.text {...} /* выбор тегов span с четырьмя классами: underlined, red, big и text */

Контекстные селекторы

Селектор может состоять из нескольких частей, разделённых пробелом, например:

```
p strong { ... }
ul .hit { ... }
.footer .menu a { ... }
```

Такие селекторы называют контекстными или вложенными. Их используют для того, чтобы применить стили к элементу, только если он вложен в нужный элемент.

Читать их проще всего справа налево:

```
/* выбрать все теги strong внутри тегов p */
p strong { ... }

/* выбрать все элементы с классом .hit внутри тегов ul */
ul .hit { ... }

/* выбрать все ссылки внутри элементов с классом .menu,
   которые лежат внутри элементов с классом .footer */
.footer .menu a { ... }
```

Соседние селекторы

Соседние селекторы используются для расположенных рядом элементов.

Например, теги <li> в списке являются соседними по отношению друг к другу и вложенными в тег <ul>.

Соседние селекторы записываются с помощью знака +, например, селектор1 + селектор2. Стили применятся к элементу, подходящему под селектор2, только если сразу перед ним расположен элемент, подходящий под селектор1.

Пример. Есть два элемента списка:

```
<ul>
  <li class="hit"></li>
  <li class="miss"></li>
</ul>
```

Селектор .hit + .miss применит стили к элементу с классом miss, так как перед ним есть элемент с классом hit.

Селекторы в CSS можно очень гибко комбинировать. В частности, можно комбинировать контекстные и соседние селекторы.

Дочерние селекторы

Любые элементы, расположенные внутри родительского элемента называются потомками. А дочерними элементами являются ближайшие потомки. Взгляните на пример:

```
<ul>
  <li><span>...</span></li>
  <li><span>...</span></li>
</ul>
```

По отношению к списку <ul> элементы <li> являются дочерними элементами и потомками, а — потомки, но не дочерние элементы.

Контекстные селекторы влияют на всех потомков.

Если нужно задать стили только для дочерних элементов используется дочерний селектор, в котором используется символ >. Например: ul > li или ul > li > span.

Псевдоклассы

Псевдоклассы — это дополнения к обычным селекторам, которые делают их ещё точнее и мощнее.

Псевдокласс добавляется к селектору с помощью символа :. Например:

a:visited { ... }

li:last-child { ... }

.alert:hover { ... }

Псевдокласс first-child позволяет выбрать первый дочерний элемент родителя, а last-child — последний дочерний элемент. Например:

li:last-child { ... }

С помощью псевдокласса nth-child можно выбирать теги по порядковому номеру. Синтаксис псевдокласса: селектор:nth-child(выражение). Выражением может быть число или формула. Например:

li:nth-child(2) { ... }

li:nth-child(4) { ... }

li:nth-child(2n) { ... }

Селекторы с псевдоклассами хорошо сочетаются с контекстными селекторами.

Псевдоклассы состояний

Благодаря некоторым селекторам можно добавлять в интерфейс динамику и интерактивность.

Псевдокласс :hover позволяет выбрать элемент, когда на него наведён курсор мыши и кнопка мыши не нажата. Пример:

a:hover { ... }

Существуют специальные псевдоклассы для ссылок:

- :link выбирает ещё не посещённые ссылки.
- :visited выбирает посещённые ссылки.
- :active выбирает активные ссылки (кнопка мыши зажата на ссылке).

Псевдокласс :focus позволяет выбрать элемент, который в данный момент в фокусе.

Селекторы атрибутов

Селекторы атрибутов позволяют выбирать элементы по любым атрибутам. Они записываются с использованием квадратных скобок: элемент[атрибут]. Примеры селекторов:

input[checked] { ... }

input[type="text"] { ... }

Первый селектор выберет поля формы, у которых есть атрибут checked, второй селектор выберет поля формы, у которых атрибут type имеет значение text.

Селектор по id

Для атрибута id существует специальный селектор. Он записывается с помощью символа #, например, #some-id.

На значение id распространяются те же ограничения, что и на имя класса. Также id должен быть уникальным на странице.

Приветствие!

#greeting { ... }

Испытание: интерактивное расследование

Пришла пора испытания. Но испытание это не совсем обычное: в части вы изучали селекторы, но CSS-код сейчас вам писать не придётся. Более того - CSS изменять запрещено! В его коде используются как раз те селекторы, которые вы изучили в этой части.

В HTML создана таблица с результатами соревнований четырёх спортсменов. В ней выводится имя участника, его очки в каждом из раундов и суммарный итог. В итоге среди спортсменов определились золотой, серебряный и бронзовый медалист, а один участник был дисквалифицирован.

А что же нужно сделать? Вам нужно проанализировать CSS-код и понять, какие именно правки нужно внести в разметку, чтобы все стили применялись к ней. Вот как это должно выглядеть в результате:

Результаты соревнования

Игрок	Раунд 1	Раунд 2	Раунд 3	Итог
Игорь Иванов	10	-20	30	**20**
Федор Свердлов	50	10	-30	**30**
~~Василий Смирнов~~	~~50~~	~~50~~	~~50~~	~~**150**~~
Иван Киселев	-10	-20	40	**10**

- создайте два файла - под разметку и под стили в папке task, свяжите их
- заполните их, содержимое ниже
- замените некоторые теги на другие;
- пропишите классы для некоторых элементов.

стартовый HTML

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Испытание: интерактивное расследование</title>
    <link rel="stylesheet" href="challenge.css">
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <span>Результаты соревнования</span>
    <table>
      <tr>
        <td>Игрок</td>
        <td>Раунд 1</td>
        <td>Раунд 2</td>
```

```
      <td>Раунд 3</td>
      <td>Итог</td>
    </tr>
    <tr>
      <td>Игорь Иванов</td>
      <td>10</td>
      <td>-20</td>
      <td>30</td>
      <td>20</td>
    </tr>
    <tr>
      <td>Федор Свердлов</td>
      <td>50</td>
      <td>10</td>
      <td>-30</td>
      <td>30</td>
    </tr>
    <tr>
      <td>Василий Смирнов</td>
      <td>50</td>
      <td>50</td>
      <td>50</td>
      <td>150</td>
    </tr>
    <tr>
      <td>Иван Киселев</td>
      <td>-10</td>
      <td>-20</td>
      <td>40</td>
      <td>10</td>
    </tr>
  </table>
 </body>
</html>
```

стартовый CSS

```
   /*стили не менять!*/
*{
  margin:0;
  padding:0;
  font-family: Verdana, Geneva, Tahoma, sans-serif;
  color: rgb(47, 47, 47);
}
table {
 border-collapse: collapse;
}
```

```
td,
th {
  padding: 3px 15px;
  border: 2px solid grey;
}

h1 {
  font-size: 20px;
}

h1 + .results {
  margin-top: 20px;
}

th {
  font-weight: normal;
  background-color: #dad7f4;
  text-align: left;
}

td:nth-child(5) > span {
  font-weight: bold;
}

.disqualified {
  text-decoration: line-through;
}

.player-1 .total {
  background-color: silver;
}

.player-2 .total {
  background-color: gold;
}

.player-3 .total {
  background-color: red;
}

.player-4 .total {
  background-color: chocolate;
}
```