

Каскад.

Комбинирование и приоритет селекторов

Цели занятия

- разобраться с тем, что такое каскад и как он работает;
- научиться комбинировать несколько селекторов;
- узнать как считается приоритет селекторов;

Полезные материалы

- [Справочник CSS](#)
- [Правила оформления CSS-кода](#)
- [Валидатор CSS](#)

Каскад

На первом занятии про CSS мы узнали, что этот термин расшифровывается как каскадные таблицы стилей. Настолько время узнать, что же такое каскад.

Представьте что мы выбрали все элементы абзаца и установили для них цвет фона orange и шрифт размером 24 пикселя. Затем в нижней части мы снова выбираем все элементы абзаца и устанавливаем для них цвет фона green:

```
p {  
  background: orange;  
  font-size: 24px;  
}
```

```
p {  
  background: green;  
}
```

Поскольку селектор абзаца, который устанавливает зелёный цвет фона располагается после селектора абзаца, который задаёт оранжевый цвет фона, он будет иметь приоритет в каскаде. Все абзацы появятся на зелёном фоне.

Каскад работает со свойствами внутри отдельных селекторов. Например, мы выбрали все элементы абзаца и установили для них цвет фона orange. Затем прямо ниже свойства background и его значения мы добавляем ещё одно свойство и значение, которое задаёт цвет фона green, как показано здесь.

```
p {  
  background: orange;  
  background: green;  
}
```

В этом случае абзацы также будут отображаться на зеленом фоне.

Приоритет селекторов (специфичность)

Каждый селектор в CSS имеет вес специфичности, он вместе с положением в каскаде определяет, как будут отображаться стили.

У селектора типа низкий вес специфичности и значение балла 0-0-1. У селектора класса средний вес специфичности и значение балла 0-1-0. Наконец, у идентификаторов высокая специфичность и значение балла 1-0-0. Как мы видим, баллы специфичности вычисляются с помощью трёх колонок. В

первой колонке количество идентификаторов, во второй классов, а третья колонка считает селекторы типа.

Чем выше вес специфичности селектора, тем больше первенства ему отдаётся при возникновении конфликта стилей. Например, если элемент абзаца выбирается с помощью селектора типа в одном месте и идентификатора в другом, то идентификатор будет иметь приоритет над селектором типа, независимо от того, где идентификатор появляется в каскаде.

HTML

```
<p id="food">...</p>
```

CSS

```
#food {  
    background: green;  
}  
  
p {  
    background: orange;  
}
```

Комбинирование селекторов

Пока мы рассмотрели как использовать разные типы селекторов индивидуально, но мы также должны знать, как использовать эти селекторы вместе. Комбинируя селекторы мы можем быть более конкретными в том, какой элемент или группу элементов мы хотели бы выбрать.

Например, мы хотим выбрать все элементы абзаца, которые находятся внутри элемента со значением атрибута класса `hotdog` и установить для них цвет фона как `brown`. Однако если один из этих абзацев, случаем, содержит значение атрибута класса `mustard`, мы хотим установить его цвет фона как `yellow`.

HTML

```
<div class="hotdog">
  <p>...</p>
  <p>...</p>
  <p class="mustard">...</p>
</div>
```

CSS

```
.hotdog p {
  background: brown;
}

.hotdog p.mustard {
  background: yellow;
}
```

Когда селекторы комбинируются они должны читаться справа налево. Самый крайний селектор справа, непосредственно перед открытой скобкой, известен как ключевой селектор. Он определяет, к каким именно элементам будут применяться стили. Любой селектор слева от ключевого будет служить уточнением.

Первый комбинированный селектор выше, `.hotdog p`, включает в себя два селектора: класс и селектор типа. Эти два селектора разделяются пробелом. Ключевым селектором выступает селектор типа, нацеленный на элементы абзаца. Поскольку этот селектор сочетается с классом `hotdog`, полный комбинированный селектор выбирает только элементы абзаца, которые находятся внутри элемента с классом `hotdog`.

Второй селектор выше, `.hotdog p.mustard`, включает в себя три селектора: два класса и один селектор типа. Единственное различие между вторым и первым селекторами является добавление класса `mustard` в конце селектора абзаца. Поскольку новый класс `mustard` находится в правой части комбинированного

селектора, то он ключевой, а все отдельные селекторы идущие перед ним теперь уточняющие.

Разделение стилей по нескольким классам

Одним из способов сохранить низкими веса специфичности наших селекторов является при возможности модульность, передача похожих стилей от элемента к элементу. Один из вариантов модульности — разделение на разные стили с помощью нескольких классов.

Элементы в HTML могут содержать более одного атрибута `class`, при этом их значения разделяются пробелами. За счёт этого мы можем применить некоторые стили ко всем элементам одного вида, а другие стили к конкретным элементам этого же вида.

Давайте взглянем на кнопки, к примеру. Мы хотим, чтобы у всех кнопок был размер шрифта 16 пикселей, но чтобы цвет фона кнопок мог меняться в зависимости от того, где кнопки применяются. Мы можем создать несколько классов и распределить их по элементам

HTML

```
<a class="btn btn_danger">...</a>
<a class="btn btn_success">...</a>
```

CSS

```
.btn {
  font-size: 16px;
}

.btn_danger {
  background: red;
}

.btn_success {
  background: green;
}
```

Здесь вы можете увидеть два элемента ссылок с несколькими значениями атрибутов класса. Первый класс, `btn`, используется для задания размера шрифта 16 пикселей каждому из элементов.

Затем, первый элемент ссылки использует дополнительный класс `btn-danger`, чтобы применить красный цвет фона, а второй элемент ссылки использует дополнительный класс `btn-success`, чтобы применять зелёный цвет фона. Наши стили чистые и модульные.

Используя несколько классов мы можем делить стили на множество классов по своему желанию, сохраняя наш код компактным с низким весом специфичности. Понимание каскада и вычисление специфичности требует практики, которая занимает время до полного постижения, но в этом мы становимся с каждым уроком всё лучше.