Урок 2.3. Приоритет селекторов. Отступы. Шрифты

Приоритет подключаемых стилей

Существует три основных способа подключения стилей: файл css, тег `<style>` и инлайн (с использованием атрибута `style`).

Приоритетнее всего инлайн-стили - они будут применятся и перекрывать все остальные

На втором месте по приоритетам стили из тега `<style>`, они будут перекрывать файловые стили

На третьем месте - стили в файле.

Приоритет селекторов

Каждый селектор имеет свой числовой приоритет (вес):

Селектор тега:	1
Селектор класса:	10
Селектор ID:	100
Inline-стиль:	1000

При комбинировании селекторов приоритеты складываются:

Селектор	ID	Класс	Тег	Общий вес
p	0	0	1	1
.your_class	0	1	0	10
p.your_class	0	1	1	11
#your_id	1	0	0	100
#your_id p	1	0	1	101
#your_id .your_class	1	1	0	110
p a	0	0	2	2
#your_id #my_id .your_class p a	2	1	2	212

В итоге к тегу применятся стили селектора с наибольшим весом, поэтому чтобы переопределить уже заданный стиль, можно, например, добавить ещё один класс к тегу и использовать более приоритетный селектор.

Если же вес селекторов совпадает, то применяется тот стиль, который находится ниже в CSS коде.

Если одинаковые по весу селекторы находятся в двух разных файлах, то применится тот, который подключён ниже в html (с помощью тега `<link>`).

box-sizing

Специальное свойство CSS, определяющее, будут ли граница (`border`) и внутренние отступы (`padding`) включаться в общий размер элемента, или будут увеличивать размер элемента.

По умолчанию значение этого свойства – `content-box`. Если задано это значение, то внутренние отступы и граница будут увеличивать результирующую ширину и высоту. То есть, если ширина элемента `120px`, а `border` – `10px`, то итоговая ширина элемента будет `140px` (граница по `10px` слева и справа).

Другое значение этого свойства – `border-box`. Если задано это значение, то ширина и высота элемента останется та же, а `border` и `padding` будут использовать имеющуюся ширину и высоту элемента (находиться внутри). То есть, если ширина элемента `120px`, а `border` – `10px`, то итоговая ширина останется `120px`, при этом визуально ширина самого элемента уменьшится на ширину границы (граница как бы “съест” часть ширины элемента).

Я рекомендую для всех элементов использовать значение `border-box`.

Для этого в начале CSS файла нужно написать селектор всех элементов – `*`:

```
* {
    box-sizing: border-box;
}
```

padding

`padding` – это внутренний отступ элемента. Существует для того, чтобы визуально добавить пространство между краями элемента и внутренними дочерними элементами.

`padding` – составное свойство. Тут может быть указано до 4х значений через пробел:

```
.class {
    padding: 10px 20px 30px 40px; /* 10px верх, 20px право, 30px низ, 40px лево */
}

.class-2 {
    padding: 10px 20px 30px; /* 10px верх, 20px право, 30px низ, 20px лево */
}

.class-3 {
    padding: 10px 20px; /* 10px верх, 20px право, 10px низ, 20px лево */
}

.class-4 {
    padding: 10px; /* со всех сторон 10px */
}
```

Так же каждое из значений может задаваться отдельно:

```
.class {
    padding-top: 10px; /* отступ сверху */
    padding-right: 20px; /* отступ справа */
    padding-bottom: 30px; /* отступ снизу */
    padding-left: 40px; /* отступ слева */
}
```

margin

`margin` – это внешние отступы. В отличие от `border` и `padding`, значение `box-sizing` на него не влияет.

`margin` – составное свойство, так же как и `padding`, может содержать до 4х значений через пробел.

```
.class {
    margin: 10px 20px 30px 40px; /* 10px верх, 20px право, 30px низ, 40px лево */
}

.class-2 {
    margin: 10px 20px 30px; /* 10px верх, 20px право, 30px низ, 20px лево */
}

.class-3 {
    margin: 10px 20px; /* 10px верх, 20px право, 10px низ, 20px лево */
}

.class-4 {
    margin: 10px; /* со всех сторон 10px */
}
```

Так же как и у `padding`, у `margin` значения можно задать отдельными свойствами:

```
.class {
    margin-top: 10px; /* отступ сверху */
    margin-right: 20px; /* отступ справа */
    margin-bottom: 30px; /* отступ снизу */
    margin-left: 40px; /* отступ слева */
}
```

`margin` существует для того, чтобы создавать отступы *между* блоками.

Дополнительные CSS свойства

object-fit

Свойство используется для картинок и видео, чтобы правильно разместить их в контейнере.

Свойство указывается непосредственно для тега `<img>` или `<video>`.

```
/* картинка не изменит пропорции, займёт всю площадь контейнера
   (если нужно – обрежется) */
.image {
    object-fit: cover;
}

/* картинка не изменит пропорции, будет показана вся картинка
   (часть контейнера может остаться пустой) */
.image-2 {
    object-fit: contain;
}

/* картинка изменит пропорции: полностью влезет в контейнер
   и заполнит его целиком */
.image-3 {
    object-fit: fill;
}
```

font-size

Свойство задаёт размер шрифта. Могут использоваться любые единицы измерения, например, пиксели.

По умолчанию большинство браузеров делает размер шрифта равным 16px (16 пикселей)

```
.text {
    font-size: 24px;
}
```

font-weight

Задаёт жирность шрифта. Значения могут быть зарезервированными словами: `normal`, `medium`, `bold` или цифрами от 100 до 900 (`100`, `200`, `300`, `400`, `500`, `600`, `700`, `800`, `900`). Чем больше цифра – тем жирнее шрифт.

Некоторые шрифты сами по себе могут не содержать в себе различных начертаний, поэтому данное свойство для таких шрифтов либо не будет работать, либо будет работать неверно.

```
.text {
    font-weight: 700; /* жирный шрифт */
}
```

font-family

Служит для задания семейства шрифтов (проще – для задания самого шрифта).

Значения, которые рекомендуется использовать БЕЗ подключения дополнительных шрифтов – `sans-serif` (без засечек), `serif` (с засечками), `cursive` (курсив), `fantasy` (жирный для заголовков), `monospace` (моноширинный). Перечисленные 5 значений – базовые. Для использования других шрифтов (например, `Times New Roman`, который по умолчанию установлен в Windows) необходимо их подключить в CSS.

В самом свойстве может задаваться несколько значений через запятую. Если будет недоступно первое значение, браузер попытается применить второе, и так далее.

```
/* Так можно писать, только если шрифт 'Arial' подключён в CSS,
   потому что несмотря на то, что на вашем компьютере он может быть
   установлен, на компьютерах других пользователей он может быть отсутствовать,
   и шрифт будет отображаться неверно */
.text {
    font-family: 'Arial', sans-serif;
}

.text-2 {
    font-family: serif; /* шрифт с засечками */
}
```

@font-face

Используется для подключения шрифтов в CSS.

```
@font-face {
    /* название шрифта, которое потом будет использоваться в
       свойстве font-family. Здесь можно задать любое значение,
       но рекомендуется указывать здесь именно название самого шрифта */
    font-family: 'MyFontFamily';
```

```
    /* здесь задаётся список путей до самих файлов шрифтов. Используются
       шрифты в формате woff и woff2 */
    src: url('fonts/MyFontFamily.woff') format('woff'),
         url('fonts/MyFontFamily.woff2') format('woff2');

    /* значение жирности шрифта, при котором будет срабатывать именно
       данный файл шрифта */
    font-weight: 700;
}
```

Скачанные в формате `otf` или `ttf` шрифты можно сконвертировать в `woff` и `woff2` с помощью сайта https://transfonter.org