

# Шрифты и работа с ними

Страшным сном дизайнера и верстальщика является фраза заказчика «Поиграйся со шрифтами». Многие эту фразу слышали, но, а как же поиграть со шрифтами? CSS предоставляет много правил для стилизации шрифтов. С некоторыми из таких свойств вы уже знакомы — это свойство `font-weight` и `font-size`.

Помимо насыщенности, CSS позволяет управлять следующими настройками:

- Семейство шрифтов;
- Стиль шрифта;
- Стиль строчных символов;
- Размер шрифта;
- Межстрочный интервал.

Не пугайтесь такому количеству различных свойств, их необязательно использовать все сразу. Некоторые используются достаточно редко.

## Семейство шрифтов

CSS позволяет указывать шрифты, которые будут использованы на сайте. Для этого используется свойство `font-family`. Оно принимает список шрифтов, которые могут быть загружены на сайте. Это может быть одно семейство шрифтов или сразу несколько. Если указано несколько шрифтов, то браузер будет считывать их слева направо до первого шрифта, который он сможет использовать. Остальные шрифты будут игнорироваться. Например:

```
.font {  
  font-family: Georgia, "Times New Roman";  
}
```

Если у пользователя в системе есть шрифт *Georgia*, то будет применён он. В противном случае браузер будет искать шрифт *Times New Roman*. Теперь возможны две ситуации:

- У пользователя в системе установлен шрифт *Times New Roman* — шрифт применится к странице.
- У пользователя в системе **не установлен** шрифт *Times New Roman* — браузер подставит шрифт из основных настроек браузера. Это необходимо для возможности отобразить контент на странице.

А какой шрифт именно выберет браузер? Тут всё зависит от настроек. Может случиться так, что стандартный шрифт будет из другого типа. Это может «сломать» визуальную часть сайта. Все шрифты можно разделить на три большие группы:

- Антикwa или шрифты с засечками. Такие шрифты чаще всего используется в книгах и новостных сайтах. В CSS обозначается `serif`.
- Гротеск или шрифты без засечек. Основной тип шрифтов на сайтах. Прямо сейчас вы читаете именно такой шрифт. В CSS обозначается `sans-serif`.

- Моноширинный шрифт. У этого семейства все символы имеют одинаковую ширину. Вы можете увидеть такой шрифт в терминалах или редакторах кода. В CSS обозначается **monospace**.

При указании шрифтов также возможно указать и основное семейство шрифтов. Если ни один из указанных шрифтов не был найден, то браузер подберёт системный шрифт из того семейства, которое было указано. Дополним пример семейством шрифтов по умолчанию. В примере указаны шрифты из семейства антиква, поэтому стоит добавить соответствующее значение.

```
.font {  
  font-family: Georgia, "Times New Roman", serif;  
}
```

Всегда указывайте семейство шрифтов по умолчанию. Это считается хорошей практикой, так как не все шрифты могут присутствовать в системе пользователя.

## Установка новых шрифтов

Необязательно полагаться только на системные шрифты. В проекте могут использоваться совершенно разные шрифты и верстальщик должен уметь их подключать. Сам по себе шрифт представляет собой файл. Форматов таких файлов может быть несколько и важно знать, какими браузерами они поддерживаются.

- **.woff/.woff2** — шрифт формата Web Open Font Format. Распространённый формат, который поддерживается большинством современных браузеров. В настоящих условиях вы можете использовать только этот формат, если нет альтернатив.
- **.ttf** — шрифт формата TrueType. Данный формат был придуман в 80-е года 20 века и сейчас используется для поддержки старых браузеров. Совместим с современными браузерами. Золотая середина форматов.
- **.eot** — шрифт формата Embedded OpenType. Это наиболее старый формат. Его использование необходимо только в случае поддержки старых браузеров, например, Internet Explorer 6.0. Ситуация редкая, поэтому его использование почти не встречается.

Для установки шрифта используется конструкция **@font-face**. Она позволяет подключить шрифт в различных расширениях, определить имя и путь к файлу. На примере шрифта *Roboto* воспользуемся такой конструкцией.

Пусть наш проект имеет директорию *fonts*, внутри которой находятся файлы шрифтов.

```
project/  
├── css/  
│   └── style.css  
├── fonts/  
│   ├── Roboto-Regular.ttf  
│   ├── Roboto-Bold.ttf  
│   └── Roboto-Light.ttf
```

Хорошим тоном является указание **@font-face** в самом начале CSS файла, а не перед первым использованием шрифта. Это позволит правильнее структурировать CSS. Есть два основных свойства, которые принимает **@font-face**:

- **font-family** — Имя подключаемого шрифта. Именно по нему можно обратиться после подключения.
- **src** — Путь к файлу со шрифтом.

После указания этих свойств уже можно пользоваться шрифтом. Подключим *Roboto-Regular*.

```
@font-face {
  font-family: 'Roboto Regular';
  src: url('../fonts/Roboto-Regular.ttf');
}

body {
  font-family: 'Roboto Regular', sans-serif;
}
```

Следующий шаг — подключить другие два начертания. Можно воспользоваться примером выше. Тогда наш CSS приобретёт следующий вид:

```
@font-face {
  font-family: 'Roboto Regular';
  src: url('../fonts/Roboto-Regular.ttf');
}

@font-face {
  font-family: 'Roboto Bold';
  src: url('../fonts/Roboto-Bold.ttf');
}

@font-face {
  font-family: 'Roboto Light';
  src: url('../fonts/Roboto-Light.ttf');
}

body {
  font-family: 'Roboto Regular', sans-serif;
}

h1 {
  font-family: 'Roboto Bold', sans-serif;
}

h2 {
  font-family: 'Roboto Light', sans-serif;
}
```

Способ хоть и рабочий, но немного сложный. Вместо одного названия шрифта и управления его насыщенностью свойством **font-weight** приходится указывать три разных названия для каждого начертания в отдельности.

**@font-face** позволяет указывать насыщенность шрифта с помощью **font-weight**. Это позволит подключить все начертания с использованием всего одного имени. В остальном запись не будет отличаться от того, что было в прошлом примере.

```
@font-face {
  font-weight: 400; /* Соответствует значению normal */
  font-family: 'Roboto';
  src: url('../fonts/Roboto-Regular.ttf');
}

@font-face {
  font-weight: 700; /* Соответствует значению bold */
```

```

font-family: 'Roboto';
src: url('../fonts/Roboto-Bold.ttf');
}

@font-face {
font-weight: 300;
font-family: 'Roboto';
src: url('../fonts/Roboto-Light.ttf');
}

body {
font-family: 'Roboto', sans-serif;
}

h1 {
font-weight: bold;
}

h2 {
font-weight: 300;
}

```

## Стиль шрифта

Помимо обычного начертания, CSS позволяет задать и другой вариант отображения шрифта — курсивный. Для этого используется свойство `font-style`, которое может принимать одно из трёх значений:

- `normal` — стандартное значение. Соответствует нормальному стилю отображению шрифта. Именно такой стиль вы читаете прямо сейчас.
- `italic` — курсивное начертание. Данное начертание имеет наклонные буквы, в отличие от нормального стиля. *Вот так выглядит курсивный шрифт.*
- `oblique` — тоже курсивное начертание. Зачастую оно не отличается от значения `italic`.

Но почему существует два похожих значения для курсива? На самом деле они не очень похожие. Курсив, который задаётся значением `italic`, является самостоятельным шрифтом, для которого есть отдельный файл в системе или на сервере. Он относится больше к рукописному тексту, тогда как `oblique` искусственно наклоняет символы текущего шрифта.

```

.italic {
font-style: italic;
}

```

## Строчные символы

С помощью CSS можно задать такой вид строчных символов как «Капитель».

Капитель — вид строчных букв, размер которых совпадает (или приближен) к размеру заглавных букв. Такой приём используется в дизайне в виде стилистического оформления. Посмотрите на этот параграф:

По своей высоте эта фраза не отличается от простого набора строчными символами, но стилистически они подстраиваются под заглавные символы.

Капитель устанавливается с помощью свойства `font-variant`. У него возможны два значения:

- `normal` — стандартная стилистика строчных символов.
- `small-caps` — капитель.

```
p {  
  font-variant: small-caps;  
}
```

## Межстрочный интервал

Межстрочный интервал, или, как его ещё называют интерлиньяж — это расстояние по вертикали между базовыми линиями одного и другого символа в соседних строках. Так создаётся расстояние между этими строками. Важным элементом при создании дизайна является использование достаточного межстрочного интервала. Чаще всего это 150% от значения размера шрифта. Например, если шрифт имеет размер 16 пикселей, то межстрочный интервал должен быть не менее 24 пикселей. Такое значение не является необходимым правилом и от него всегда можно отступить.

Для указания межстрочного интервала используется свойство `line-height`. Оно может принимать значения с различными единицами измерениями. Чаще всего используют число, указывающее, во сколько раз интервал больше размера шрифта.

```
p {  
  font-size: 16px;  
  line-height: 1.5;  
}
```

## Обобщённое правило

Теперь, для работы со шрифтами, вы знаете все основные правила. Это:

- `font-style`
- `font-variant`
- `font-weight`
- `font-size`
- `line-height`
- `font-family`

Их можно указывать не только по отдельности, но и все вместе с помощью одного CSS правила `font`. Шесть разных правил внутри одного! Это может быть удобно, если действительно нужны все значения. При этом вы не обязаны указывать все значения. Единственное ограничение — наличие значений для `font-size` и `font-family`. Остальные значения можно не указывать. Укажем значения для всех этих свойств внутри правила `font`:

```
p {  
  font: italic small-caps bold 16px/24px Roboto, sans-serif;  
}
```

Важно обратить внимание на запись *16px/24px*. Внутри правила **font** так обозначаются свойства **font-size** и **line-height**.

## Использование одного правила или нескольких

Этот раздел относится не только к правилу **font**, но и ко всем обобщённым правилам, которые вы изучите в процессе прохождения курсов. С одной стороны кажется, что использование одного правила сокращает количество строк, которые используются в CSS. Это действительно так, но есть две основные проблемы использования таких свойств:

1. Запоминание правильного порядка значений. Используя обобщённые свойства вам всегда стоит держать в голове верный порядок значений свойств. В этом легко можно ошибиться на первых этапах изучения. Хорошим вариантом будет использование отдельных свойств, но в том порядке, в котором они идут в обобщённом свойстве. С опытом вы сможете переключиться на одно правило.
2. Обобщённые свойства перебивают отдельные. Если в коде вы указали **font-variant: small-caps;**, а потом для этого же элемента применили **font: 16px/24px sans-serif;**, то капитель будет сброшена в значение по умолчанию.

Используйте обобщённые свойства только один раз при задании стандартных стилей. Например, свойство **font** отлично подойдёт для селектора **body**, а модификации текста будут производиться одиночными свойствами.