Видео недели:

1. [Стилизация](https://www.youtube.com/watch?v=zU-odT6QzB0)
2. [Селекторы](https://www.youtube.com/watch?v=aoAG7rDgMVs)
3. [CSS-свойства](https://www.youtube.com/watch?v=ZpvmSPCy52s)
4. [Шрифты](https://www.youtube.com/watch?v=rj54s9arOAY)

**Стилизация**

Стили являются удобным, практичным и эффективным инструментом при вёрстке веб-страниц и оформления текста, ссылок, изображений и других элементов.

Вот их преимущества:

* Разграничение кода и оформления

В идеале, веб-страница должна содержать только теги логического форматирования, а вид элементов задаётся через стили. При подобном разделении работа над дизайном и версткой сайта может вестись параллельно.

* Разное оформление для разных устройств

С помощью стилей можно определить вид веб-страницы для разных устройств вывода: монитора, принтера, смартфона, планшета и др. Например, на экране монитора отображать страницу в одном оформлении, а при её печати — в другом. Эта возможность также позволяет скрывать или показывать некоторые элементы документа при отображении на разных устройствах.

* Ускорение загрузки сайта

При хранении стилей в отдельном файле, он кэшируется и при повторном обращении к нему извлекается из кэша браузера. За счёт кэширования и того, что стили хранятся в отдельном файле, уменьшается код веб-страниц и снижается время загрузки документов.

*Кэшем* называется специальное место на локальном компьютере пользователя, куда браузер сохраняет файлы при первом обращении к сайту. При следующем обращении к сайту эти файлы уже не скачиваются по сети, а берутся с локального диска. Такой подход позволяет существенно повысить скорость загрузки веб-страниц.

* Единое стилевое оформление множества документов

Сайт - это не просто набор связанных между собой документов, но и одинаковое расположение основных блоков, и их вид. Применение единого оформления заголовков, основного текста и других элементов создает общность между страницами и облегчает пользователям работу с сайтом и его восприятие в целом.

* Централизованное хранение

Стили, как правило, хранятся в одном или нескольких специальных файлах, ссылка на которые указывается во всех документах сайта. Благодаря этому удобно править стиль в одном месте, при этом оформление элементов автоматически меняется на всех страницах, которые связаны с указанным файлом. Вместо того чтобы модифицировать десятки HTML-файлов, достаточно отредактировать один файл со стилем и оформление всех нужных документов сразу же поменяется.

**Как подключаются стили**

Чтобы указать стили у нас есть несколько вариантов:

1. С помощью уже знакомого нам атрибута style - это удобно, если нужно что-то быстро протестировать или для короткого набора стилей. *Но в реальном проекте лучше этот способ не использовать, потому что его очень сложно поддерживать.*
2. Это слово будет написано <span style="color:'red'">красным</span>
3. С помощью тега <style>. В этом случае свойства CSS описываются в самом документе и располагаются в заголовке веб-страницы. Это позволяет хранить стили в одном месте, прямо на той же странице - *и этот способ гораздо более поддерживаемый, ведь так стили гораздо легче находить и при необходимости изменять*:
4. <head>
5. <style>
6. \*\*h1\*\* {
7. color: #333366;
8. }
9. \*\*img\*\* {
10. width: '100%'
11. }
12. </style>
13. </head>
14. С помощью тега <link> можно подключить стили, находящиеся в отдельном файле. *Это самый оптимальный способ, ведь тогда одни и те же стили можно использовать на разных страницах в случае, если их больше одной.*
15. <head>
16. <meta charset="utf-8">
17. <title>Стили</title>
18. <link rel="stylesheet" href="style.css">
19. </head>

**Как писать CSS**

Синтаксис CSS следующий: сначала мы указываем *селектор* (то есть, к чему именно должен относиться данный стиль), а затем в скобках { } пишем пары *свойств* и их *значений*:

\*\*селектор\*\* {

свойство: значение;

свойство: значение;

свойство: значение;

...

}

Например,

p { color: red; } /\* красный цвет шрифта у всех абзацев \*/

**Селекторы**

Селектор - это часть CSS-правила, которая сообщает браузеру, к какому элементу (или элементам) веб-страницы будет применён стиль.

\*\*p\*\* {...} /\* по тегу \*/

.\*\*title\*\* {...} /\* по классу \*/

\*\*p\*\*.\*\*warning\*\* {...} /\* тег с классом \*/

\*\*#promo\*\* {...} /\* по id \*/

\*\*h1#chapter1\*\* {...} /\* тег с id \*/

.\*\*title a\*\* {...} /\* тег внутри тега с классом \*/

.\*\*title\*\* .\*\*bage\*\* {...} /\* класс внутри тега с классом \*/

Селекторы могут определяться тегами, классами, псевдоклассами, идентификаторами и пр. и их комбинациями:

* По **тегу**
* По **идентификатору**
* По **классу**
* **Комбинирование**
* **Группировка**
* **Псевдоклассы**
* **Псевдоэлементы**
* **Каскадность и приоритеты**

**CSS-свойства**

\*\*position top left z-index

display width height margin padding

font-family font-style line-height color

text-decoration text-transform

float clear

background box-shadow text-shadow\*\*

/\* И очень много других \*/

**Классификация свойств**

Свойства делятся на следующие группы:

* Позиционирование (расположение на странице)
* Блочная модель (размеры и отступы) и поток
* Оформление текста
* Декоративные эффекты (фоны, тени и так далее)
* Анимация и динамические эффекты
* Разное

**Типы значений**

* **Абсолютные**
* **Относительные**
* **Ключевые слова**
* **Цвета**
* **Функции**
* **Произвольные строки**

**Нормализация стилей**

**Normalize.css** — это небольшой CSS-файл, который обеспечивает для HTML-элементов лучшую кроссбраузерность (поддержку разных браузеров) в стилях по умолчанию. Сайт проекта: <http://necolas.github.io/normalize.css>

Цели normalize.css:

* сохранять полезные настройки браузера, а не стирать их;
* нормализовать стили для широкого круга HTML-элементов;
* корректировать ошибки и основные несоответствия браузера;
* совершенствовать юзабилити незаметными улучшениями;
* объяснять код, используя комментарии и детальную документацию.

**Валидация CSS**

Наиболее удобно делать проверку кода через сайт <http://jigsaw.w3.org/css-validator/>, с помощью этого сервиса можно указать адрес документа, загрузить файл или проверить набранный текст.

**Принципы форматирования**

В процессе написания CSS следует придерживаться некоторых принципов, которые позволяют сократить код CSS, сделать его более удобным, наглядным и читабельным. Читабельность в данном случае означает, что разработчик спустя какое-то время может легко понять и модифицировать стиль или что в коде разберётся даже сторонний человек.

* Размещайте каскадные таблицы стилей в отдельном файле

Размещение стилей в отдельном файле позволяет ускорить загрузку веб-страниц за счёт уменьшения их кода, а также кэширования файла с описанием стиля.

* Удаляйте неиспользуемые селекторы

Большое количество селекторов создаёт путаницу в вопросе о том, кто из них за что отвечает, да и просто увеличивает объем документа. Чтобы этого не произошло, удаляйте селекторы, которые никак не применяются на сайте. К сожалению, определить точно, какой селектор используется, а какой нет, довольно сложно, поэтому добавляйте комментарий в код. Это поможет хотя бы не запутаться в большом объёме текста.

* Применяйте группировки

Достоинство и удобство группирования состоит в описании одинаковых свойств в одном месте. Тем самым, значение свойства пишется только один раз, а не повторяется многократно.

* Пользуйтесь единым стилем написания кода

Существует множество разных подходов как же писать CSS-код. Кто-то упорядочивает селекторы по блокам, другой - согласно структуре документа, третий - по алфавиту, в общем, сколько людей, столько и мнений. Не важно какой именно вариант вы выберете на данном этапе, но главное, чтобы он был всегда одинаковый.

А по свойствам хороший пример грамотного *code style* можно найти у HTML Академии, они используют группировку по смыслу:

<http://codeguide.academy/html-css.html#css-order>

**Сокращенная запись свойств**

Некоторые свойства CSS допускают использование комбинированных свойств, что устраняет необходимость в некоторых индивидуальных свойствах. Это называется сокращенная форма записи свойств CSS.

Основные свойства, которые обладают сокращённой формой записи:

* background
* margin
* padding
* border
* font

Например, свойства margin, padding позволяют объединять соответствующие индивидуальные свойства, такие как margin-top-width, margin-right-width, margin-bottom-width и т.д., в форме **свойство: верхнее правое нижнее левое;**

Стиль с таким описанием:

p {

margin-top: 1px;

margin-right: 5px;

margin-bottom: 10px;

margin-left: 20px;

}

Можно объединить следующим образом:

p {

margin: 1px 5px 10px 20px;

}

Свойства border-width, border-color и border-style также могут быть объединены в единое свойство border, например:

p {

border: 1px red solid;

}

Применение этих свойств сокращает объём кода и повышает его читабельность, но с другой стороны иногда добавляет путаницу.

* **Возможные проблемы с сокращенной записью**

**Шрифты**

Поскольку не все шрифты доступны на всех компьютерах (существуют тысячи шрифтов и большинство из них не бесплатны), CSS предусматривает резервную систему. Первым вы указываете шрифт, который хотели бы использовать. Затем следуют любые шрифты, которые вы могли бы использовать, если первый указанный шрифт не доступен. А закончить список вы должны *типовым шрифтом*, который имеет 5 видов: **serif, sans-serif, monospace, cursive** и **fantasy**.

**Базовые свойства CSS-шрифтов**

1. Семейство шрифтов: свойство font-family
2. font-family: "Times New Roman", Georgia, serif;
3. Насыщенность шрифта: свойство font-weight
4. font-weight: normal;
5. font-weight: bold;
6. font-weight: lighter;
7. font-weight: bolder;
8. font-weight: 100;
9. Начертание шрифта: свойство font-style
10. font-style: normal;
11. font-style: italic;
12. font-style: oblique;
13. Размер шрифта: свойство font-size
14. font-size: small;
15. font-size: medium;
16. font-size: large;
17. font-size: smaller;
18. font-size: larger;
19. font-size: 14px;
20. font-size: 0.8em;
21. Сокращенная запись свойств шрифта: свойство font
22. font: 12pt/14pt sans-serif;
23. font: 80% sans-serif;
24. font: x-large/110% "new century schoolbook", serif;
25. font: bold italic large Palatino, serif;
26. font: normal small-caps fantasy;
27. font: condensed oblique 12pt "Helvetica Neue", serif;

**Алгоритм работы при подключении шрифта**

1. Проверяем, стандартный ли шрифт
2. Если нестандартный, ищем на внешних сервисах
3. Если нашли, подключаем из сервиса
4. Если не нашли, подключаем локально

**Подключение с Google Fonts**

1. Находим шрифт на сайте <https://fonts.google.com/>

Для того, чтобы сервис показал шрифты с поддержкой русского языка в пункте Languages выбираем Cyrillic.

1. Открываем нужный шрифт и жмем "+" на желаемом начертании. Справа мы увидим код для вставки на страницу.

Помните, чем больше начертаний вы выберете, тем больше будет вес загружаемого файла, поэтому кликаем + только на те, которые реально нужны.

1. Прописываем шрифты у себя в коде
2. <head>
3. <link href="<https://fonts.googleapis.com/css2?family=Roboto:wght@300&display=swap>" rel="stylesheet">
4. <style>
5. p {
6. font-family: 'Roboto', sans-serif;
7. }
8. </style>
9. </head>

P.S. Шрифт - это очень важно. Просто почувствуйте разницу:

**Домашнее задание на эту неделю:**

**Практическое задание 💎**

1. Сверстайте страницу по макету из п.3 прошлого домашнего задания
   * примените минимум 3 вида селекторов
   * используйте минимум 3 псевдокласса
   * сделайте нормализацию
   * подключите нестандартный шрифт, например, Jost или любой другой на выбор
   * проверьте html и css на валидность
   * приведите код к единому стилю

Большим подспорьем к ДЗ будет являться вкладка **Inspect** в Figma, которая генерирует уже готовый CSS-код при выделении элемента.

**Тест 💎**

1. Какими способами можно подключать CSS-стили? Найдите сами еще один способ, не указанный в уроке
2. Зачем нужен Normalize.css?
3. Что такое CSS-директивы?
4. В чем разница между **margin** и **padding**?
5. Как в CSS определяются приоритеты? Какое из свойств будет приоритетнее - #link .main или span #login?
6. В чем разница между CSS1 и CSS3?
7. Что такое псевдоклассы? А псевдоэлементы?
8. Изучите статью про "плохие" теги <https://msiter.ru/tutorials/html-srednego-urovnya/plokhie-tegi> и пришлите список тегов, которые нежелательно использовать
9. Как можно подключать шрифты локально?
10. Почему не стоит использовать сокращенную запись без необходимости? И если все же использовать, как это делать правильно?
11. Разберитесь самостоятельно, как сделать анимацию через CSS

ОТВЕТЫ

1. 1 способ: <link rel="stylesheet" href="/assets/styles/style.css">

2 способ: <head>

<style>

………….

</style>

</head>

3 способ: <style>

@import url("assets/style.css");

</style>

1. Normalize.css нужен, чтобы сохранять полезные настройки браузера, улучшат кроссбраузерность и т.д.
2. Директивы — это конструкции, которая позволяет создавать в CSS инструкции для изменения отображения либо поведения элементов страницы. Директива начинается со знака @, за которым следует одно из служебных слов. Это общий синтаксис, которому следуют все директивы.
3. Margin определяет интервал между блоками, padding указывает сколько места должно быть между содержимым элемента и его границами.
4. Css определяет приоритет исходя из специфичности: чем более специфично свойство, тем оно приоритетней. Т.е. в данном случае в приоритете будет #link .main
5. CSS3 является новейшим стандартом для языка CSS, который разрабатывается с 2005 года. Он обратно совместим со старыми версиями CSS, что имеет новые свойства, которые отлаживают предыдущие особенности и расширяют возможности CSS2, и даже имеет некоторые JavaScript-подобные возможности. CSS3 также решает ряд проблем, связанных с разработкой мобильных приложений, учитывая адаптивный дизайн и устраняя проблемы, вызванные несовместимостью Adobe Flash на мобильных устройствах. В сочетании с JavaScript, CSS3 обладает множеством функциональных возможностей Flash, это с точки зрения анимации и интерактивности.
6. Псевдоклассы – это селекторы, которые определяют состояние уже существующих элементов, которое может меняться при определенных условиях. Псевдоэлементы – это селекторы, которые определяют область элементов, которая изначально отсутствует в дереве документа. Эта область создается искусственно с помощью CSS.

«Плохие» теги:

* <b>
* <em>
* <big>
* <small>
* <hr>
* <u>
* <center>
* <layer> .
* <blink>  <marquee
* <font>

Атрибуты

* text и bgcolor
* background
* link, alink, vlink
* align
* target

1. <style>

@font-face {

font-family: Pompadur; /\* Имя шрифта \*/

src: url(fonts/pompadur.ttf); /\* Путь к файлу со шрифтом \*/

}

P {

font-family: Pompadur;

}

1. Сокращенная форма записи бывает удобна потому, что сокращает длину и вес кода, но может создать путаницу.