**Css**

<s>Стилем или CSS (Cascading Style Sheets, каскадные таблицы стилей) называется набор параметров форматирования, который применяется к элементам документа, чтобы изменить их внешний вид.

Я считаю, что наступил самый сложный этап обучения. Здесь много кто отказывается от прежней мечты быть веб разработчиком. Начались ягодки. Тяжело уложить все получаемые знания, ведь знакомясь с css вы будите изучать целую кучу свойств, которые еще надо правильно между собой комбинировать. Вы будите учиться разделять сложный объект на кучу свойств. Возможно вы будите чувствовать себя необучаемым, но потерпите до первого результата.

Иными словами, с помощью этой технологии элементам можно задать любой внешний вид. С помощью css можно менять цвет, размер, форму, тень, обводку и добавить анимацию. В общем все, что может придумать дизайнер.

Css сегодня это неотъемлемая часть сайтостроения, ведь нынешний пользователь не станет посещать сайт, если информацию оттуда тяжело воспринимать. Тем более конкуренция не позволяет выглядеть современным сайтам плохо.   
И так мы уже разобрались с тем как наполнить страницу нужными нам элементами, но признаемся честно, выглядит все это не очень. Css это тот инструмент, который позволит вашему сайту выглядеть хорошо. Все что касается отображения элемента будь то размер, цвет, форма, расположение и т.п. прописывается с помощью css.

**Способы добавления стилей на страницу**

Для добавления стилей на веб-страницу существует несколько способов, которые различаются своими возможностями и назначением. Далее рассмотрим их подробнее

При использовании связанных стилей описание селекторов и их значений располагается в отдельном файле, как правило, с расширением css, а для связывания документа с этим файлом применяется тег <link>. Данный тег помещается в контейнер <head>. Это основной способ добавления стилей на страницу.

img/2/2.1.jpg big

Значение href задаёт путь к CSS-файлу, он может быть задан как относительно, так и абсолютно. Заметьте, что таким образом можно подключать таблицу стилей, которая находится на другом сайте.

При использовании глобальных стилей свойства CSS описываются в самом документе и располагаются в заголовке веб-страницы. По своей гибкости и возможностям этот способ добавления стиля уступает предыдущему, но также позволяет хранить стили в одном месте, в данном случае прямо на той же странице с помощью контейнера <style>.

img/2/2.2.jpg big

Внутренний или встроенный стиль является по существу расширением для одиночного тега, используемого на текущей веб-странице. Для определения стиля используется атрибут style, а его значением выступает набор стилевых правил.

img/2/2.3.jpg big

**Базовый синтаксис CSS**

Cтилевые правила записываются в своём формате, отличном от HTML. Основным понятием выступает селектор — это некоторое имя стиля, для которого добавляются параметры форматирования. В качестве селектора выступают теги, классы и идентификаторы. Общий способ записи имеет следующий вид.

img/2/2.4.png

На данном изображении мы обращаемся к тегу body и задаем ему цвет заднего фона. Вначале пишется имя селектора, например, TABLE, это означает, что все стилевые параметры будут применяться к тегу <table>, затем идут фигурные скобки, в которых записывается стилевое свойство, а его значение указывается после двоеточия. Стилевые свойства разделяются между собой точкой с запятой, в конце этот символ можно опустить.

Вот пример записи нескольких свойств элементу.

img/2/2.5.jpg big

Комментарии в css пишутся так

img/2/2.6.jpg big

Попробуйте прямо сейчас к созданному ранее html файлу подключить css, неважно каким из способов и попытайтесь поменять фон страницы со стандартного белого на красный. Как пишется свойство меняющее фон элемента было выше. В том примере цвет определяется через решётку, но вы можете написать просто 'red'.

**Обращение к элементу.**

Так ват вы уже могли понять, что чтобы описать свойства элементу, нужно к этому элементу как-то обратиться. Почему бы не обращаться к нему по тегу как показано в примере выше? Проблема заключается в том, что одинаковых тегов на странице может быть очень много и если вашей целью не является определить для них одни и те же стили, то тут приходят на помощь классы и id.

То есть в css можно обращается к элементу по его классу или id. Делается это так. Есть у нас допустим элемент с классом ‘box’ и мы хотим поменять ему цвет фона. Для этого мы в подключенных стилях пишем точку и название класса, а после задаем ему нужное нам свойство.

img/2/2.7.jpg big

Обращение по id то же самое, но вместо точка будет решётка

img/2/2.8.jpg big

Важно понимать, что один id мы можем присвоить только одному элементу, а определенный класс можно подключать к нескольким элементам. Чаще всего для обращения к элементу используют class, так как таким образом можно задать нескольким элементам одни и те же стили. Например, нам нужно застилизовать пункты меню, у них будут одинаковый шрифт, размер, цвет и т.п. Вот как будет выглядеть html

img/2/2.9.jpg big

А вот так будет выглядеть css стили

img/2/2.10.jpg big

Попробуйте реализовать нечто подобное.

Так же сюда можете запихать и тему контекстных селекторов. Это когда у нас допустим в элементе div с классом menu есть единственный тег span, к которому надо обратиться, но придумывать и писать для этого отдельный класс как-то не хочется. Тогда мы можем обратиться к этому тегу, но не в рамках целого документа (то есть не ко всем span`ам), а в рамках определенного элемента

img/2/2.12.jpg big

так же можно обращаться к элементам через запятую и писать для них общие стили.

img/2/2.13.jpg big

**Свойства css**

Тут будет сложнее чем с тегами в html. Придется знать кучу свойств, чтобы уметь верстать более-менее серьезные страницы. Гугл в этом случае ваш лучший друг. Для того чтобы найти нужное вам свойство задайтесь вопросом ‘что конкретно мне нужно? ’. Может быть вам нужно передвинуть элемент или поменять цвет его фона или поменять цвет и размер шрифта так далее. То есть вбиваем конкретный вопрос, например, ‘как увеличить межстрочный интервал css ’ или 'как выcтроить элементы в ряд по горизонтали' по мере необходимости вы будите знакомиться с новыми свойствами css.

Существует много на первый взгляд похожих свойств, но применять их нужно в разных ситуациях, например, margin и padding. Margin используется для отступов снаружи а, padding отступы внутри, применять их надо в разных ситуациях и понять эту разницу вы сможете только достаточно раз использовав, то есть с опытом.

Некоторые свойства разбивают на отдельные подсвойства, например, есть margin, но также есть margin-left, margin-right, margin-top и margin-bottom.

Еще нужно иметь в виду и то что некоторые свойства могут работать по-разному в разных браузерах, либо вместо этого свойства придумали аналог, который удобнее использовать, например, использовать float моветон, когда flex уже добрался до всех браузеров.

Попробуйте создать страницу с заголовком и абзацем. Попытайтесь изменить размер заголовка и его цвет. А абзац оттолкнуть от заголовка на 20 px и чтобы еще все это было по середине страницы.

img/2/2.10.jpg smoll

**единицы измерения**

В CSS существует множество единиц измерения. Обычно мы используем единицы размеров, но также есть единицы для углов — например, deg и turn, единицы времени — s и ms, единицы плотности экрана — например, dpi и dppx и другие.

Здесь будут рассматриваться только единицы размеров, которыми мы пользуемся чаще всего.

Общим для всех единиц длины будет то, что для значения 0, единицы можно не указывать: height: 0px и height: 0 будут работать одинаково, так что единицы измерения можно отбросить. Это позволяет немного быстрее писать код и считывать значения свойств.

Существует большое количество единиц измерения, одной самых популярных является пиксели

Пиксель — это базовая, абсолютная и окончательная единица измерения. <s>

Почему бы во всех свойствах, требующих единицу измерения длины не использовать пиксели? Все дело в том, что разных случаях есть более удобные, например, нам нужно сказать элементу, чтобы он занимал половину от своего родителя и мы конечно можем узнать размер родительского элемента (нажмите f12 и наведитесь на нужный вам элемент), посчитать его половину и присвоить, но это не целесообразно, когда есть проценты. То есть мы стразу можем сказать элементу: занимай 50% от родительского элемента. Это пример использования относительной единицы измерения.

Относительные единицы измерения вычисляются на основе каких-то других величин: размера шрифта или размера экрана, и могут динамически меняться вместе с ними. <s>

Но не стоит, например, задавать высоту элемента в процентах, если точно не знаешь высоту родительского блока. Высота блока зависит от его заполненности и отталкиваться от таких неопределённых данных, при использовании процентов, не стоит.

Изначально блочный элемент имеет нулевую высоту и 100% ширину и когда в этом блочном элементе появляется содержимое то высота блока подстраивается под это содержимое. Но иногда нам нужно чтобы высота была как минимум в пол экрана (часто нужен для приветствующих экранов или слайдеров). Для этого мы, во-первых, пишем свойство min-height которое в отличии от обычного height позволит увеличивать высоту элемента по мере поступления туда содержимого, а во-вторых пишем значение не в px или в %, а в vh(зависит от высоты экрана). Чтобы такой элемент занял пол экрана нужно присвоить значение 50vh.

Размер текстам и некоторым блокам можно задавать в rem. Они равны значению font-size для тега html (изначально 16px), а значит это значение можно будет перезадать, просто изменив значения для этого свойства у тега html. В будущем, при необходимости пропорционально уменьшить все что вы задавали с помощью rem, можно будет в одну строчку кода. Это очень помогает в процессе адаптивности под другие устройства (тема адаптивности будут следующей).

На начальных этапах можете смело писать все в пикселях. И через 2, 3 сверстанных макета, когда вы более-менее привыкните к написанию кода, можете задуматься над тем как можно сделать то же самое проще и гибче. Не заморачивайтесь над тем что не используете весь потенциал разных единиц измерения. Изучайте все по мере необходимости.

https://www.youtube.com/watch?v=gSr7DW\_89xU

https://www.youtube.com/watch?v=dPzoBD2Z68A

**адаптивность**

Сразу говорю, что данный этап не обязателен для новичков, которые верстают свой первый макет. Просто имейте ввиду то что front-end разработчик должен продумать то как его сайт будет отображаться на гаджетах у пользователя.

Вкратце адаптивность это любые действие разработчика, направленные на то чтобы сайт отображался на разных экранах адекватно и им удобно было пользоваться. <s>

Первое правило адаптивности: отсутствие горизонтальной полосы прокрутки на всех устройствах. Чтобы этого добиться используют относительные единицы измерения для задания ширины элементам. Полученный результат в ходе таких действий называют резиновой версткой.

Чаще всего наполнения любого блока является текст и если вы делайте резиновый блок, то текст подстроится под его ширину. Но когда экран пользователя очень маленький (мобильный телефон), то блок уменьшается в ширину, но в высоту очень сильно увеличивается и сами буквы будут выглядят большими. Так что было бы неплохо задать условие: если экран этого пользователя меньше, допустим 700px, то текст меньше на 1px, чем изначально заданный. Такие условия можно задавать через так называемые медиа запросы. Используйте rem для задания любой величины которую нужно будет поменять при изменении экрана и тогда в условии медиа запроса вы сможете поменять их буквально одной строчкой, вместо того чтобы прописывать это вручную для каждого элемента.

Так же очень советую в случаях, когда нужно поставить элементы в ряд по горизонтали использовать flex-box. И в связке с медиа запросами это позволит легко перестроить элементы в вертикальное положение, когда элементы не смогут отображаться адекватно по горизонтали из-за маленькой ширины экрана пользователя.

Так же адаптивность подразумевает то что вашим сайтом будет удобно пользоваться с помощью сенсорных экранов. Это значит, что, допустим слайдеры должны листаться не только с помощью кнопки, но и с помощью свайпа влево или в право. О слайдерах поговорим позже, когда мы зайдем на территорию js (JavaScript привыкайте к сокращениям).

Итак, эта информация в основном пригодиться тем, кто уже умеет верстать простенькие макеты и гуглить непонятные ему вещи. А значит вы и сами сможете найти источники, где будет описано применение медиа запросов и создание резиновых блоков. На youtube полно видео на эту тему. Но советую посмотреть, как прикручивают адаптивность в реальных проектах в полном цикле верстки одного сайта.

Например, как здесь

https://www.youtube.com/watch?v=D8DqoTjd\_dg