МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Тульский государственный университет»

Интернет-институт

КОНТРОЛЬНАЯ РАБОТА

по дисциплине

«Интернет-технологии»

на тему

«Нововведения CSS3»

Семестр 5

Вариант 3

Выполнил: студент гр. ИБ262521-ф

Артемов Александр Евгеньевич

Проверил: канд. техн. наук, доц.

Французова Юлия Вячеславовна

Тула 2024

Оглавление

Введение....3
Модули CSS3....6
Селекторы....6
Блоковая модель....11
Фоны и границы....13
Текстовые эффекты....16
Преобразования 2D/3D....17
Анимация....20
Расположение нескольких столбцов....23
Пользовательский интерфейс....25
Flexbox....27
Заключение....29
Используемые источники....30

Введение

CSS (англ. Cascading Style Sheets, каскадные таблицы стилей) – это простой язык дизайна, предназначенный для упрощения процесса презентации веб-страниц.

CSS обрабатывает внешний вид веб-страницы. Используя CSS, вы можете контролировать цвет текста, стиль шрифтов, расстояние между параграфами, размеры и расположение колонок, используемые фоновые изображения и цвета, макеты дизайна, варианты отображения на разных устройствах и размерах экрана. А также множество других эффектов.

CSS легко освоить и понять, но он обеспечивает мощный контроль над представлением HTML-документа. Чаще всего CSS комбинируется с языками разметки HTML или XHTML.

CSS был предложен Хоконом Виумом Ли 10 октября 1994 года и поддерживается через группу людей из W3C, называемой рабочей группой CSS. Рабочая группа CSS создает документы, называемые спецификациями. Когда спецификация обсуждается и официально утверждается членами W3C, она становится рекомендацией. Эти ратифицированные спецификации называются рекомендациями, поскольку W3C не контролирует фактическую реализацию языка. Независимые компании и организации создают это программное обеспечение. Примечание. Консорциум Всемирной паутины (The World Wide Web Consortium) или W3C – это группа, которая дает рекомендации о том, как работает Интернет и как он должен развиваться.

Каскадные таблицы стилей, уровень 1 (CSS1), вышли из W3C в качестве рекомендации в декабре 1996 года. В этой версии описывается язык CSS, а также простая модель визуального форматирования для всех HTML-тегов.

CSS2 стал рекомендацией W3C в мае 1998 года и основывается на CSS1. Эта версия добавляет поддержку для конкретных таблиц стилей,

например, принтеров и звуковых устройств, загружаемых шрифтов, элементов позиционирования и таблиц.

CSS3 стал рекомендацией W3C в июне 1999 года и основывается на более старых версиях CSS. Он разделен на документацию, которая называется «Модулями» и здесь каждый модуль имеет новые расширенные функции, определенные в CSS2.

CSS обладает рядом преимуществ:

– экономит время. Вы можете написать CSS один раз, а затем использовать одну и ту же таблицу на нескольких HTML-страницах. Вы можете определить стиль для каждого HTML-элемента и применить его ко многим веб-страницам;

– страницы загружаются быстрее. Если вы используете CSS, вам не нужно каждый раз писать атрибуты HTML-тегов. Просто напишите одно CSS правило для тега и примените его ко всем вхождениям этого тега. Таким образом, меньшее количество кода означает более быстрое время загрузки;

– простота обслуживания. Чтобы внести глобальные изменения, просто измените стиль, и все элементы на всех веб-страницах будут обновляться автоматически;

– улучшенные стили для HTML. CSS имеет гораздо более широкий набор атрибутов, чем HTML, поэтому вы можете сделать гораздо лучший вид своей HTML-страницы по сравнению с атрибутами HTML;

– совместимость нескольких устройств. Таблицы стилей позволяют оптимизировать контент для более чем одного типа устройств. Используя один и тот же HTML-документ, можно представить различные версии веб-сайта для карманных устройств, таких как PDA и сотовые телефоны, или для печати;

– глобальные веб-стандарты. Теперь атрибуты HTML устарели, и рекомендуется использовать CSS. Поэтому неплохо было бы начать исполь-

зовать CSS во всех HTML-страницах, чтобы сделать их совместимыми с будущими браузерами;

– оффлайн-просмотр. Веб-приложения могут хранить CSS локально с помощью оффлайн кэша. Используя это, мы можем просматривать сайты находясь оффлайн. Кэш также обеспечивает быструю загрузку и лучшую общую производительность веб-сайта;

– независимость от платформы. Скрипт обеспечивает независимость от платформы и поддерживает новейшие браузеры.

Модули CSS3

Селекторы.

Селектор (от англ. select — выбирать) — это шаблон, который позволяет обратиться к элементу или группе элементов веб-страницы, чтобы применить к ним стили CSS. Его указывают перед блоком со свойствами:

```
a {
    text-decoration: none;
}
```

В примере выше селектор указывает на тег <a> (гиперссылка). Так мы говорим браузеру отключить подчёркивание у всех ссылок на странице, устанавливая для свойства text-decoration значение none.

Виды селекторов.

Универсальный селектор

Он применяет стили ко всем элементам страницы и обозначается символом * (звёздочка). С его помощью удобно сбрасывать отступы и задавать значение box-sizing для всех блочных элементов:

```
* {
    margin: 0;
    box-sizing: border-box;
}
```

Селектор по тегу (элементу)

Этот селектор CSS применяет стили ко всем элементам с одинаковым тегом. Например, для всех , <h2>, <p> и так далее.

Мы уже познакомились с ним, когда убирали подчёркивание у ссылок:

```
a {
    text-decoration: none;
}
```

Селектор по идентификатору (id)

Селектор по идентификатору обозначается символом # (решётка) и применяет стили к элементу, для которого задан атрибут id с соответствующим значением. При этом у элемента может быть только один id, и этот id должен быть уникальным в пределах веб-страницы.

```
#intro{
    color: red;
    font-weight: bold;
}
```

Селектор по классу (class)

CSS-селектор по классу выбирает элементы, для которых назначен атрибут class с соответствующим значением. При этом один элемент может принадлежать нескольким классам — в таком случае их перечисляют через пробел:

```
.plain_text{
    font-size: 20px;
}
.article{
    font-family: "Montserrat";
}
```

Группа селекторов

CSS-селекторы можно сгруппировать, чтобы применить стили к нескольким группам и/или классам элементов. Для этого достаточно перечислить их через запятую:

```
.plain_text, p, h1, figure, div {
        margin-top: 0;
        margin-left: 0;
}
```

Выбор элементов по отношению и расположению

Есть группа селекторов, которые позволяют выбрать элемент по его отношению к другим элементами (родитель — потомок) и по расположению в DOM (Document Object Model).

Выбрать всех потомков: чтобы обратиться ко всем потомкам B элемента A, независимо от уровня их вложенности, используют конструкцию A B (селекторы разделяют пробелом).

Выбрать потомков первого уровня: если нужно применить CSS-стили к потомкам B элемента A только на первом уровне вложенности, то вместо пробела пишут символ >.

Выбрать все следующие элементы: селектор A ~ B выбирает все элементы B, которые идут после A. Обратите внимание: «идут после», а не вложены в него.

Выбрать первый следующий элемент: селектор A + B выбирает только первый элемент B, который следует за A.

CSS-селекторы по атрибуту

[attr] — применяет стили к элементам, для которых задан этот атрибут;

[attr=value] — работает по имени и значению атрибута;

[attr^=value] — находит элементы с заданным атрибутом, значение которого начинается с value;

[attr|=value] - ищет по названию атрибута и значению, которое равно или начинается с value;

[attr$=value] — применяет CSS-стили к элементам, у которых значение заданного атрибута оканчивается на value;

[attr*=value] — селектор по названию атрибута и значению, которое должно содержать value;

[attr~=value] - этот шаблон выбирает элементы с атрибутом attr, значение которого состоит из нескольких слов, разделённых пробелом, одно из которых — value;

Псевдоклассы и псевдоэлементы

Псевдокласс выбирает элементы, находящиеся в определённом состоянии или положении в иерархии DOM. Вот несколько примеров таких состояний: на кнопку наведён курсор мыши; пользователь перешёл или не перешёл по ссылке; курсор установлен на поле ввода.

Например, так с помощью CSS можно увеличить размер ссылок, на которые пользователь навёл курсор:

```
a:hover {
        font-size: 20px;
}
```

Список основных псевдоклассов:

Название	**Состояние элемента**
:hover	Наведён курсор
:focus	Элемент находится в фокусе (например, по нему кликнули мышью или его выбрали клавишей Tab)
:visited	Ссылка, которая была посещена
:active	Активный элемент (в промежутке времени между

Название	Состояние элемента
	нажатием и отпусканием кнопки мыши)
:checked	Элементы radio, checkbox или option, которые были выбраны
:first-child	Первый потомок элемента
:last-child	Последний потомок элемента
:nth-child()	Каждый n-й потомок — число n передаётся в качестве аргумента
:last-nth-child()	Последние n потомков — число n передаётся в качестве аргумента
:read-write	Элементы, доступные для редактирования

Вес CSS-селектора, или специфичность

Для одного и того же элемента веб-страницы можно прописать сколько угодно стилей. Если в разных местах CSS-файла какому-то его свойству заданы разные значения, то браузер должен выбрать одно из них. Обычно подключается правило, которое определено последним, но так происходит не всегда. Дело в том, что одни селекторы обладают более высокой специфичностью, чем другие. Специфичность — это показатель, по которому браузер определяет, какие стили применить к элементу. Её можно представить в виде четырёх чисел 0.0.0.0, где каждый разряд — это вес, определяемый специальными правилами:

Наивысший приоритет — у стилей, прописанных в атрибуте style .

На втором месте — селекторы по идентификатору.

Затем идут три равноправные группы: селекторы по классу, атрибуту и псевдоклассы.

На четвёртом месте — селекторы по тегу и псевдоэлементы.

Комбинаторы ~, >, + и универсальный селектор * веса не добавляют.

Вес псевдоклассов :is(), :has() и :not() равен весу самого специфичного селектора внутри скобок.

Чтобы определить самый «тяжёлый» селектор, браузер сначала взвешивает каждый, а затем сравнивает их поразрядно.

Блоковая модель.

В HTML-документе каждому элементу на странице соответствует прямоугольная область (бокс или блок). Движок рендеринга в браузере определяет размеры и положение боксов на странице, а также их свойства вроде цвета, фоновой картинки для того, чтобы отобразить их на экране.

В языке CSS есть специальная боксовая модель (также блоковая модель или блочная модель, англ. box model), которая описывает, из чего состоит бокс и какие свойства влияют на его размеры. В ней у каждого бокса есть 4 области: margin (внешние отступы), border (рамка), padding (внутренние поля), и content (контент или содержимое).

Внутренняя область элемента (content area) содержит текст и другие элементы, расположенные внутри (контент или содержимое). У неё часто бывает фон, цвет или изображение (в таком порядке: фоновый цвет скрывается под непрозрачным изображением), и она находится внутри content edge; её размеры называются ширина контента (content width или content-box width), и высота контента (content height или content-box height). Иногда ещё говорят «внутренняя ширина/высота элемента»

По умолчанию, если CSS-свойство box-sizing не задано, размер внутренней области с содержимым задаётся свойствами width, min-width, max-width, height, min-height и max-height. Если же свойство box-sizing задано, то оно определяет, для какой области указаны размеры.

Поля элемента (padding area) — это пустая область, окружающая контент. Она может быть залита каким-то цветом, покрыта фоновый картинкой, а её границы называются края полей (padding edge).

Размеры полей задаются по отдельности с разных сторон свойствами padding-top, padding-right, padding-bottom, padding-left или общим свойством padding.

Область рамки (border area) окружает поля элемента, а её граница называется края рамки (border edge). Ширина рамки задаётся отдельным свойством border-width или в составе свойства border. Размеры элемента с учётом полей и рамки иногда называют внешней шириной/высотой элемента.

Отступы (margin area) добавляют пустое пространство вокруг элемента и определяют расстояние до соседних элементов.

Величина отступов задаётся по отдельности в разных направлениях свойствами margin-top, margin-right, margin-bottom, margin-left или общим свойством margin.

Отступы двух соседних элементов, расположенных друг над другом или вложенных друг в друга, могут накладываться. Это называется схлопывание границ (margin collapsing). Схлопываются только вертикальные отступы.

Для элементов с display: inline (или inline-block, inline-table) на занимаемое по высоте место также влияет значение свойства line-height.

Фоны и границы.

Свойство background-color определяет цвет фона для любого элемента в CSS. Свойство принимает любой допустимый цвет <color>. background-color распространяется на сам контент и отступы от него (padding).

Свойство background-image позволяет отображать изображение в качестве фона элемента. По умолчанию большое изображение не масштабируется до размера блока.

Свойство background-repeat используется для управления повторениями фонового изображения. Доступные значения:

no-repeat — останавливает повторение фонового изображения во всех направлениях;

repeat-x — повторение фонового изображения по горизонтали;

repeat-y — повторение фонового изображения по вертикали;

repeat — повторение фонового изображения в обоих направлениях, по умолчанию.

Для изменения размеров фонового изображения используется свойство background-size, которое может принимать значения длины или в процентах, чтобы размер изображения соответствовал размеру фона. Также можете использовать ключевые слова:

cover — браузер сделает изображение достаточно большим, чтобы оно полностью заполнило блок, сохраняя при этом соотношение сторон. В этом случае часть изображения, скорее всего, окажется за пределами блока.

contain — браузер сделает изображение нужного размера, чтобы поместиться в блоке. В этом случае могут появиться пробелы с обеих сторон или сверху и снизу изображения, если соотношение сторон изображения отличается от соотношения сторон блока.

Свойство background-position позволяет вам изменять позицию, в которой фоновое изображение появляется в блоке. При этом используется систе-

ма координат, в которой левый верхний угол блока равен (0,0), а сам блок располагается вдоль горизонтальной (x) и вертикальной (y) осей. По умолчанию значение background-position равно (0,0).

Обычно свойство background-position задают в виде двух последовательных значений — значение по горизонтали, за которым следует значение по вертикали.

Другая опция, которую можно применить к фону, - это указать, как он будет прокручиваться при прокрутке содержимого. Это контролируется с помощью свойства background-attachment, которое может принимать следующие значения:

scroll: Заставляет элементы фона прокручиваться при прокрутке страницы. Если содержимое элемента прокручивается, фон не перемещается. Фактически, фон фиксируется в той же позиции на странице, поэтому он прокручивается по мере прокрутки страницы.

fixed: Фиксирует элементы фона в области просмотра, чтобы он не прокручивался при прокрутке страницы или содержимого элемента. Фон всегда будет оставаться на одном и том же месте на экране.

local: Значение local фиксирует фон для элемента, к которому он применён, поэтому, когда вы прокручиваете элемент, фон прокручивается вместе с ним.

Свойство background-attachment действует только тогда, когда есть контент для прокрутки.

Установить границу для всех четырёх сторон блока с помощью свойства border: 1px solid black. Для указания конкретных границ используем border-top: 1px solid black, или по отдельности: border-top-width, border-top-style, border-top-color.

Закругление углов блока достигается с помощью свойства border-radius и связанных свойств, которые относятся к каждому углу блока. В качестве значения могут использоваться два значения длины или процента: первое

значение определяет горизонтальный радиус, а второе - вертикальный радиус. Чаще задают только одно значение, которое используется для обоих. Например, чтобы сделать все четыре угла блока радиусом 10px: border-radius: 10px. Или, чтобы верхний правый угол имел горизонтальный радиус 1em и вертикальный радиус 10%: border-top-right-radius: 1em 10%.

Текстовые эффекты.

Переполнение текста. CSS свойство text-overflow указывает, как переполняется содержимое, которое не должно быть доведено до пользователя. Принимает значения clip — обрезает текст или ellipsis — заменяется многоточием.

Перенос слов. CSS свойство word-wrap позволяет разбивать длинные слова и переносить их на следующую строку. Если слово слишком длинное, чтобы поместиться в пределах области, оно расширяется снаружи.

Разбиение слов. CSS свойство word-break задает правила разрыва строк.

Режим записи. CSS Свойство writing-mode определяет, будут ли строки текста располагаться горизонтально или вертикально.

Преобразования 2D/3D.

Модуль 3D Transforms расширяет спецификацию CSS 2D Transforms, позволяя преобразовывать элементы в трехмерном пространстве. Новые функции преобразования для свойства transform выполняют трехмерные преобразования, расширяя координатное пространство до трех измерений, добавляя ось Z, перпендикулярную плоскости экрана, которая увеличивается по направлению к зрителю, а дополнительные свойства позволяют контролировать взаимодействие вложенных трехмерных преобразованных элементов.

Хотя некоторые значения свойства transform позволяют преобразовывать элемент в трехмерной системе координат, сами элементы не являются трехмерными объектами. Они существуют в двумерной плоскости (плоская поверхность) и не имеют глубины.

Свойство transform-style. По умолчанию преобразованные элементы создают плоское представление своего содержимого. Свойство transform-style позволяет преобразованным 3D-элементам и их 3D-потомкам использовать общее трехмерное пространство, выстраивая иерархии трехмерных объектов. Отображение 3D-потомков определяется моделью — так называемым контекстом 3D-рендеринга. Отображение зависит от z-позиции элементов в трехмерном пространстве, и если 3D-преобразования этих элементов вызывают пересечение, то они отображаются с пересечением.

Свойство perspective. В нормальном потоке элементы отображаются плоскими и в той же плоскости, что и блок, содержащий их. Двумерные функции преобразования могут изменять внешний вид элемента, но этот элемент по-прежнему отображается в той же плоскости, что и содержащий его блок.

Свойство perspective-origin. Обычно предполагаемое положение глаза зрителя находится в центре рисунка. Свойство perspective-origin управляет точкой начала координат, позволяя изменять направление трансформации до-

чернего 3D-элемента. Свойство должно использоваться вместе со свойством perspective для родительского элемента и свойством transform для дочернего элемента.

Свойства perspective и perspective-origin можно использовать для добавления ощущения глубины в сцену, делая элементы выше по оси Z (ближе к зрителю) и кажущимися большими, а те, которые находятся дальше — меньшими. Масштаб пропорционален d / (d - Z), где d — значение перспективы, является расстоянием от плоскости рисования до предполагаемого положения глаза зрителя.

Свойство backface-visibility. Используя трехмерные преобразования, можно преобразовать элемент так, чтобы его обратная сторона была видна. 3D-преобразованные элементы отображают одинаковое содержимое с обеих сторон, поэтому обратная сторона выглядит как зеркальное отображение лицевой стороны. Свойство backface-visibility позволяет делать элемент невидимым, когда его обратная сторона обращена к зрителю.

Функции 3D-трансформации. Свойство задает вид как 2D, так и 3D-преобразований элемента. 3D-преобразования описываются с помощью функций трансформации, перечисленных в таблице ниже:

Функция	**Описание**
matrix3d (n,n,n,n, n,n,n,n, n,n,n,n, n,n,n,n)	Функция задает трехмерное преобразование как однородную матрицу размером 4×4 с шестнадцатью значениями в столбцах. Все другие функции преобразований основаны на данной функции.
translate3d(x,y,z)	Функция задает перемещение элемента в 3D-пространстве. Движение происходит по вектору [tx, ty, tz], где tx перемещение вдоль оси X, ty — перемещение вдоль оси Y, а tz — вдоль оси Z. Значения могут задаваться в единицах длины или в %. Отрицательные значения будут перемещать элемент в противоположном направлении.

translateZ(z)	Функция задает перемещение элемента на заданное расстояние в направлении оси Z. Значения могут задаваться в единицах длины или в %. Отрицательные значения будут перемещать элемент в противоположном направлении.
scale3d(x,y,z)	Функция задает операцию трехмерного масштабирования по вектору масштабирования [sx,sy,sz], описываемому тремя параметрами. Отрицательные значения отображают элемент зеркально вдоль трех осей.
scaleZ(z)	Функция масштабирует элемент в направлении оси Z, делая его больше или меньше. В качестве значения задается число. Результат функции наиболее выражен при совместном использовании с такими функциями, как rotate() и perspective().
rotate3d(x,y,z,угол)	Функция вращает элемент по часовой стрелке относительно трех осей. Элемент поворачивается под углом, задаваемым последним параметром относительно вектора направления [x,y,z]. Отрицательные значения поворачивают элемент против часовой стрелки.
rotateX(угол)	Функция задает поворот по часовой стрелке под заданным углом относительно оси X. Функция rotateX(180deg) эквивалентна rotate3d(1,0,0,180deg).
rotateY(угол)	Функция задает поворот по часовой стрелке под заданным углом относительно оси Y. Функция rotateY(180deg) эквивалентна rotate3d(0,1,0,180deg).
rotateZ(угол)	Функция задает поворот по часовой стрелке под заданным углом относительно оси Z. Функция rotateZ(180deg) эквивалентна rotate3d(0,0,1,180deg).
perspective(n)	Функция меняет перспективу обзора элемента, создавая иллюзию глубины. Чем больше значение функции перспективы, тем дальше от смотрящего расположен элемент. Значение должно быть больше нуля.

initial	Устанавливает значение свойства в значение по умолчанию.
inherit	Наследует значение свойства от родительского элемента.

Анимация.

CSS3-анимация придаёт сайтам динамичность. Она оживляет веб-страницы, улучшая взаимодействие с пользователем. В отличие от CSS3-переходов, создание анимации базируется на ключевых кадрах, которые позволяют автоматически воспроизводить и повторять эффекты на протяжении заданного времени, а также останавливать анимацию внутри цикла.

CSS3-анимация может применяться практически для всех html-элементов, а также для псевдоэлементов :before и :after. При создании анимации не стоит забывать о возможных проблемах с производительностью, так как на изменение некоторых свойств требуется много ресурсов.

Ключевые кадры используются для указания значений свойств анимации в различных точках анимации. Ключевые кадры определяют поведение одного цикла анимации; анимация может повторяться ноль или более раз. Ключевые кадры указываются с помощью правила @keyframes. Создание анимации начинается с установки ключевых кадров правила @keyframes. Кадры определяют, какие свойства на каком шаге будут анимированы. Каждый кадр может включать один или более блоков объявления из одного или более пар свойств и значений. Правило @keyframes содержит имя анимации элемента, которое связывает правило и блок объявления элемента.

Свойство animation-name определяет список применяемых к элементу анимаций. Каждое имя используется для выбора ключевого кадра в правиле, которое предоставляет значения свойств для анимации. Если имя не соответствует ни одному ключевому кадру в правиле, нет свойств для анимации, отсутствует имя анимации, анимация не будет выполняться.

Если несколько анимаций пытаются изменить одно и то же свойство, то выполнится анимация, ближайшая к концу списка имен. Имя анимации чувствительно к регистру, не допускается использование ключевого слова none. Рекомендуется использовать название, отражающее суть анимации, при

этом можно использовать одно или несколько слов, перечисленных через дефис - или символ нижнего подчеркивания _.

Свойство animation-duration определяет продолжительность одного цикла анимации. Задаётся в секундах s или миллисекундах ms. Если для элемента задано более одной анимации, то можно установить разное время для каждой, перечислив значения через запятую.

Свойство animation-timing-function описывает, как будет развиваться анимация между каждой парой ключевых кадров. Во время задержки анимации временные функции не применяются.

Свойство animation-iteration-count указывает, сколько раз проигрывается цикл анимации. Начальное значение 1 означает, что анимация будет воспроизводиться от начала до конца один раз. Это свойство часто используется в сочетании со значением alternate свойства animation-direction, которое заставляет анимацию воспроизводиться в обратном порядке в альтернативных циклах.

Свойство animation-direction определяет, должна ли анимация воспроизводиться в обратном порядке в некоторых или во всех циклах. Когда анимация воспроизводится в обратном порядке, временные функции также меняются местами. Например, при воспроизведении в обратном порядке функция ease-in будет вести себя как ease-out.

Свойство animation-play-state определяет, будет ли анимация запущена или приостановлена. Остановка анимации внутри цикла возможна через использование этого свойства в скрипте JavaScript. Также можно останавливать анимацию при наведении курсора мыши на объект — состояние :hover.

Свойство animation-delay определяет, когда анимация начнется. Задается в секундах s или миллисекундах ms.

Свойство animation-fill-mode определяет, какие значения применяются анимацией вне времени ее выполнения. Когда анимация завершается, элемент возвращается к своим исходным стилям. По умолчанию анимация не

влияет на значения свойств animation-name и animation-delay, когда анимация применяется к элементу. Кроме того, по умолчанию анимация не влияет на значения свойств animation-duration и animation-iteration-count после ее завершения. Свойство animation-fill-mode может переопределить это поведение.

Все параметры воспроизведения анимации можно объединить в одном свойстве — animation, перечислив их через пробел: animation: animation-name animation-duration animation-timing-function animation-delay animation-iteration-count animation-direction;

Для воспроизведения анимации достаточно указать только два свойства — animation-name и animation-duration, остальные свойства примут значения по умолчанию. Порядок перечисления свойств не имеет значения, единственное, время выполнения анимации animation-duration обязательно должно стоять перед задержкой animation-delay.

Для одного элемента можно задавать несколько анимаций, перечислив их названия через запятую.

Расположение нескольких столбцов.

Спецификация макет с несколькими столбцами даёт вам метод вёрстки контента по столбцам, точно также как вы можете видеть в газете, часто называемый multicol.

Макет multicol использует одно из двух свойств column-count или column-width. Какое значение задано свойству column-count столько столбцов он и создаст. Колонки имеют гибкую ширину — браузер решает какое пространство назначить каждому столбцу.

Столбцы, созданные при помощи multicol не могут быть стилизованы по одному. Нет способа сделать один столбец больше, чем другие, или изменить фон или цвет текста одного столбца. Есть две возможности изменить способ отображения столбцов: изменение размера отступов между столбцами используя column-gap и добавление линейки между столбцами при помощи column-rule.

Можно заставить элемент растянуться через все столбцы. В этом случае контент разрывается, когда сталкивается со spanning элементом и продолжается ниже, создавая новый набор блоков столбцов. Чтобы растянуть элемент через все столбцы используется свойство column-span установленное на значение all.

Содержимое макета нескольких столбцов является фрагментированным. По сути, он ведёт себя так же, как контент в постраничных медиа — так же, как когда печатаются веб-страницы. При переводе контента в multicol контейнер он фрагментируется на столбцы и контент разбивается чтобы позволить этому произойти. Порой это разрывание происходит в местах, мешающих чтению.

Для того чтобы управлять этим поведением можно использовать свойства из спецификации CSS Фрагментации. Эта спецификация даёт свойства для управления разрывами контента в multicol и постраничных медиа.

Например, свойство break-inside со значением avoid. В настоящее время также стоит добавлять старое свойство page-break-inside: avoid для лучшей поддержки старых браузеров.

Пользовательский интерфейс.

Свойства внешнего контура.

Контуры позволяют выделять активные элементы интерфейса, такие как, кнопки, поля формы, карты изображений и т. п. Браузеры часто отображают контуры элементов в состоянии :focus, поэтому не рекомендуется делать контур невидимым для таких элементов без альтернативного механизма выделения.

Контур всегда находится сверху и не влияет на положение или размер блока или любых других блоков. Поэтому отображение или скрытие контуров не вызывает перекомпоновку. Части контура не обязательно должны быть прямоугольными, например, он повторяет закругленные углы элемента.

Краткая запись внешнего контура: свойство outline. Свойство представляет краткую запись свойств outline-color outline-style outline-width. Значение по умолчанию outline: invert none medium.

Толщина внешнего контура: свойство outline-width. Свойство outline-width задает толщину внешнего контура, принимает те же значения, что и свойство border-width.

Узор внешнего контура: свойство outline-style. Свойство outline-style принимает те же значения, что и border-style, за исключением значения hidden.

Цвет внешнего контура: свойство outline-color. Свойство outline-color позволяет установить цвет внешнего контура с помощью значений цвета и ключевого свойства invert.

Смещение внешнего контура: свойство outline-offset. По умолчанию контур рисуется начиная с края рамки элемента. Свойство outline-offset позволяет сместить контур от края границы на указанную величину.

Изменение размера блоков: свойство resize. Свойство resize позволяет указать, может ли пользователь изменять размер элемента, и если да, то

вдоль какой оси/осей. Свойство применяется к элементам, чье вычисленное значение overflow отличается от visible.

Если для элемента установлено изменение размеров, в правом нижнем углу появляется треугольник, с помощью которого элемент можно растягивать в обеих направлениях. Уменьшение первоначальных размеров элемента не предусмотрено.

Браузер должен позволять пользователю изменять размер элемента без каких-либо других ограничений, кроме ограничений, накладываемых свойствами min-width, max-width, min-height и max-height.

Стилизация курсора: свойство cursor.

Свойство cursor указывает тип курсора, который будет отображаться для устройства, когда точка доступа курсора находится в пределах границ элемента. Браузеры могут игнорировать свойство над собственными элементами управления, например, полосами прокрутки. Браузеры также могут игнорировать свойство cursor и отображать его по своему выбору, чтобы указать различные состояния пользовательского интерфейса, например, когда страница не отвечает или когда пользователь выделяет текст.

Цвет каретки вставки: свойство caret-color.

Символ каретки является видимым индикатором точки вставки в элементе, в который пользователь вставляет текст (и, возможно, другой контент). Свойство caret-color контролирует цвет этого видимого индикатора.

Flexbox.

Это новая технология, которая уже имеет достаточно широкую поддержку браузеров. Flexbox предоставляет инструменты для быстрого создания сложных, гибких макетов, и функции, которые были сложны в традиционных методах CSS.

Долгое время единственными надёжными инструментами CSS вёрстки были такие способы как Float (обтекание) и позиционирование. С их помощью сложно или невозможно достичь следующих простых требований к макету:

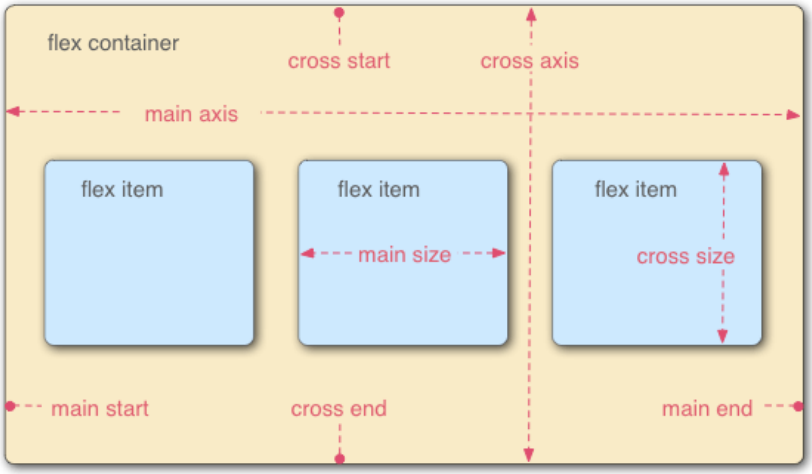
вертикального выравнивания блока внутри родителя;

оформления всех детей контейнера так, чтобы они распределили между собой доступную ширину/высоту, независимо от того, сколько ширины/высоты доступно;

сделать все колонки в макете одинаковой высоты, даже если наполнение в них различно.

Для начала нам нужно выбрать, какие элементы следует выкладывать в виде flex блоков. Для этого необходимо установить специальное значение display в родительском элементе тех элементов, которые нужно оформить.

Когда элементы выложены как flex блоки, они располагаются вдоль двух осей:



Главная ось (main axis) проходит в том направлении, вдоль которого расположены Flex элементы (например, в строку слева направо или вдоль колонок вниз.) Начало и конец этой оси называются main start и main end.

Поперечная ось (cross axis) проходит перпендикулярно Flex элементам. Начало и конец этой оси называются cross start and cross end.

Родительский элемент, на который назначено свойство display: flex называется flex container.

Элементы, размещённые в нём как Flex блоки называются flex items.

В Flexbox есть свойство под названием flex-direction, которое определяет направление главной оси (в каком направлении располагаются flexbox-дочерние элементы) — по умолчанию ему присваивается значение row, т.е. располагать дочерние элементы в ряд слева направо (для большинства языков) или справа налево (для арабских языков).

Заключение.

CSS прошел долгий путь с момента своего появления в 1996 году. В начале 2000-х годов в CSS были добавлены такие функции, как границы, градиенты и переходы для лучшего представления контента. По мере того как веб становился все более сложным с динамичными пользовательскими интерфейсами и интерактивными элементами, появились новые функции, такие как анимация и flexbox, чтобы предоставить разработчикам расширенные инструменты верстки, которые можно было использовать для создания сложных пользовательских интерфейсов. Анимация позволяет разработчикам добавлять движение в дизайн, что делает его более привлекательным, а Flexbox обеспечивает эффективное средство организации контента в аккуратные ряды или колонки, по которым пользователям легко ориентироваться. Переходы также появились в это время, позволяя разработчикам легче переходить от одного состояния элемента к другому на странице без использования кода JavaScript. Все вместе эти функции кардинально изменили наши сегодняшние представления о дизайне веб-сайтов, позволив нам контролировать макеты как никогда раньше.

Влияние HTML и CSS на современный веб огромно. Эти два языка позволили любому человеку, имеющему доступ к компьютеру, создавать потрясающие веб-сайты, приложения и другие цифровые продукты. HTML обеспечивает структуру и содержание веб-страницы, а CSS позволяет разработчикам оформлять страницу любым удобным для них способом. Вместе они образуют мощную комбинацию, позволяющую пользователям быстро создавать красивые веб-сайты с минимальными усилиями. Это произвело революцию в том, как мы взаимодействуем с Интернетом, а также открыло новые возможности для предприятий, организаций и частных лиц. Поскольку HTML и CSS продолжают развиваться вместе с технологическим прогрессом, можно с уверенностью сказать, что их влияние будет только расти в будущем!

Используемые источники.

1. https://elementarika.ru/istoriya-css/
2. https://developer.mozilla.org/ru/docs/Learn/CSS/Building_blocks/Backgrounds_and_borders
3. https://developer.mozilla.org/ru/docs/Learn/CSS/CSS_layout/Multiple-column_Layout
4. https://developer.mozilla.org/ru/docs/Learn/CSS/CSS_layout/Flexbox
5. http://css.yoksel.ru/text-effects/
6. https://html5book.ru/css3-animation/
7. https://html5book.ru/3d-transform/
8. https://html5book.ru/css3-ui/
9. https://skillbox.ru/media/code/selektory-v-css-chto-eto-takoe-kak-oni-rabotayut-i-kakie-byvayut/