**Физический пиксель** – самая маленькая точка цвета, которая может быть отображена на устройстве.

**Пиксель в *CSS*** – физическая величина, которая определяет сколько места на экране будет занимать элемент. Может соответствовать х1, х2 и х3 физическому пикселю.

В зависимости от плотности пикселей на экране нужно подгружать картинки в разных размерах.

**Векторная графика** – это изображения формата *svg*, которые состоят из координат, по которым строятся линии изображения. Их можно растягивать без потери качества, а также менять цвет, толщину линий и размер отдельных элементов.

Для оптимизации и снижения веса векторной графики рекомендуется использовать [*SVGOMG*](https://jakearchibald.github.io/svgomg/).

**Спрайт** – это файл с несколькими *svg* изображениями.

**Растровая графика** – это изображения формата *jpeg*, *png*, которые состоят из пикселей. При растягивании качество теряется, становятся видны пиксели.

В изображениях в формате *png* присутствует альфа-канал (может быть прозрачный фон или элементы).

Формат *gif* используется для коротких зацикленных анимаций. Много весит.

Для ускорения загрузок растровых изображений особое внимание нужно уделить их оптимизации:

* Использовать подходящий формат. Для простых картинок это *png*, а не *jpeg*.
* Использовать нужный размер. Не нужно отдавать картинку размером 2000х2000 в блок 500х500.
* Использовать для разных разрешений экранов разные изображения.
* Оптимизировать вес изображения через сжатие в онлайн-инструментах [TinyJPG](https://tinyjpg.com/) или [Sqoosh](https://squoosh.app/).

Пока картинка грузится используют цветной фон при помощи *background-color*.

***<svg></svg>*** – теги векторного изображения. Из плюсов – позволяет отредактировать картинку, из минусов – засоряется *html* разметка, сложно переиспользовать в других местах.

*<svg width="100" height="100" viewBox="0 0 390 390" fill="none" xmlns="http://www.w3.org/2000/svg">*

*<path d="M389 195c0 107.143-86.857 194-194 194S1 302.143 1 15 87.857" stroke="#000" stroke-width="2"/>*

*<path d="M358 135.5c0 66.55-53.95 120.5-120.5 120.5S117 202.05 135.5Z" stroke="#000" stroke-width="2"/>*

*<path d="M339 172c0 41.421-33.579 75-75 75s-75-33.579-75-75 75 75Z" stroke="#000" stroke-width="2"/>*

*<path d="M286 180c0 26.51-21.49 48s-48-21.49-48-48 48 21.49 48 48Z" stroke="#000" stroke-width="2"/>*

*</svg>*

С помощью специальных [утилит](https://yoksel.github.io/url-encoder/ru/) можно конвертировать *svg*-файл в ***data*-формат** – длинная строка в *html* разметке. Это позволяет не тратить время на загрузку изображения, но не позволяет закешировать её.

*background-image:*

*url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' fill='none' stroke='currentColor' stroke-linecap='round' stroke-linejoin='round' stroke-width='2' class='lucide lucide-heart' viewBox='0 0 24 24'%3E%3Cpath d='M19 14c1.49-1.46 3-3.21 3-5.5A5.5 5.5 0 0 0 16.5 3c-1.76 0-3 .5-4.5 2-1.5-1.5-2.74-2-4.5-2A5.5 5.5 0 0 0 2 8.5c0 2.3 1.5 4.05 3 5.5l7 7Z'/%3E%3C/svg%3E");*

*}*

***<use>*** – одиночный тег для использования изображения в формате *svg* через ссылку на его *id*. Позволяет сохранить интерактивность и возможность редактирования *svg* изображения.

*<svg viewBox="0 0 24 24">*

*<use href="#apple-1"> // ссылка на код в этом же файле*

*<use href="svg/apple.svg#apple-2"> // ссылка на код в другом файле*

*</svg>*

*<svg xmlns="http://www.w3.org/2000/svg">*

*<symbol id="apple-1" stroke="currentColor" fill="currentColor">*

*<path d="M389 107.143-86.857 194-194 194S1 302.143 1 15 87.857" stroke="#000" stroke-width="2"/>*

*<path d="M358 66.55-53.95 120.5-120.5 120.5S117 202.05 135.5Z" stroke="#000" stroke-width="2"/>*

*</symbol>*

*</svg>*

***<img>*** – одиночный тег для универсального подключения растрового или векторного изображения. Из минусов – нельзя отредактировать изображение.

Обязательный атрибут ***src*** указывает местоположение картинки.

*<img src=”images/mobile.jpg”>*

Необязательный атрибут ***alt*** отображает текст, который пользователь видит, когда картинка не загружается:

*<img src=”images/mobile.jpg” alt=”здесь должна быть картинка, но что-то пошло не так…”>*

Необязательный атрибут ***width*** позволяет задать ширину картинки в пикселях. Это также позволит избежать рывков страницы при медленной загрузке изображений.

*<img src=”images/mobile.jpg” width=230>*

Необязательный атрибут ***srcset*** позволяет задать картинки с разной плотностью пикселей.

Необязательный атрибут ***loading=”lazy”*** позволяет подгружать картинки только при попадании на видимый экран. Пока не поддерживается всеми браузерами.

*<img src="flower.jpg"*

*srcset= "flower.jpg 1x,*

*flower-2x.jpg 2x,*

*flower-3x.jpg 3x"*

*alt="Букет ярких роз крупным планом"*

*loading=”lazy”*

*>*

***<picture></picture>*** - теги всех вариантов картинки, заданных при помощи одиночных тегов *source* и *img*. Один вариант картинки – один тег *source*. В теге *img* указываем железобетонный вариант, который точно поддерживается браузером.

*<picture>*

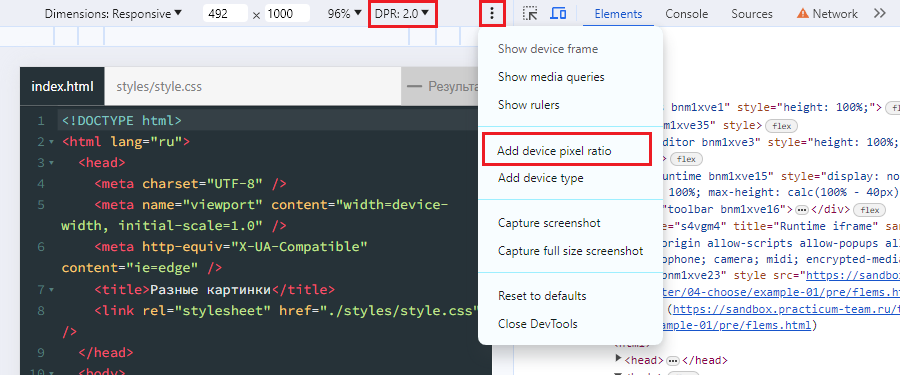
*<source srcset="flowers.avif" type="image/avif">*

*<source srcset="flowers.webp" type="image/webp">*

*<img src="flowers.jpeg" alt="Букет ярких роз крупным планом">*

*</picture>*

При тестировании кода, можно указать плотность пикселей экрана ***DRP*** в инструментах разработчика.



***<figure></figure>*** - теги контентного изображения с подписями.

*<figure>*

*<img src="" alt="">*

*<figcaption>*

*Описание изображения*

*</figcaption>*

*</figure>*

Функция ***image-set()*** позволяет загружать разные варианты фоновых изображений для *CSS*-свойства *background-image*. Пока не поддерживается всеми браузерами, поэтому пишется после гарантированного варианта.

*div {*

*background-image: url(flowers.jpeg);*

*background-image: -webkit-image-set( // вендорный префикс включающий*

*url("flowers.avif") type("image/avif"), // экспериментальное свойство*

*url("flowers.jpg") type("image/jpeg"));*

*background-image: image-set(*

*url("flowers.avif") type("image/avif"),*

*url("flowers.jpg") type("image/jpeg"));*

*}*

**Фрейм** – это окошко, в котором видно другую страницу сайта. Для его создания нужны теги ***iframe***. Технически фреймы ведут себя в потоке как *div*.

*<iframe src=”https://ya.ru/” frameborder=”0”></iframe>*

Атрибут ***frameborder*** со значением 0 позволяет отключить серую рамку фрейма.

Свойство ***boarder*** позволяет модифицировать или отключить серую рамку фрейма.

***Application Programming Interface*** (или ***API***) – это интерфейс взаимодействия между сайтом и сторонними программами и серверами.

Рассмотрим пример с подключением *API Youtube* через фрейм. Для этого используется ссылка вида *https://www.yotube.com/embed/идентификатор\_ролика*.

*<iframe*

*src="https://www.youtube.com/embed/RhlQvbvMg-0?start=30&mute=true" // без звука и с 30й секунды*

*title="YouTube video player"*

*frameborder="0"*

*allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share"*

*allowfullscreen*

*></iframe*>

Рассмотрим пример с подключением Яндекс.Карт через фрейм. Для этого воспользуемся специальным конструктором *https://yandex.ru/map-constructor/*.

*<iframe*

*src="https://yandex.ru/map-widget/v1/?um=constructor%3A6fc6359a9fea00ada5a4032946d1df7ac9ad308d11f5e45b29cf78e2bdf6a738&amp;source=constructor"*

*width="100%"*

*height="500"*

*frameborder="0"*

*></iframe>*

Рассмотрим пример подключения своего видео без использования сторонних сервисов. Для этого нам нужны теги ***video***. Не рекомендуется позиционировать видео относительно элемента html, т.к. на некоторых размерах экранов будет неправильно высчитывать ширину в 100%, и вырвать видео из потока ***position:relative***.

Атрибут ***controls*** позволяет отобразить кнопки управления видео.

Атрибут ***muted*** позволяет выключить звук видео.

Атрибут ***autoplay*** позволяет включить автовоспроизведение видео. Поддерживается не всеми браузерами и работает в связке с ***muted***.

Атрибут ***loop*** позволяет включить зациклить видео по кругу.

Атрибут ***poster*** позволяет установить картинку, которую будет видеть пользователь, пока не запустит воспроизведение или пока видео загружается.

*<video src="video.mp4" controls muted autoplay loop poster=”poster.png”></video>*

Для подключения нескольких видео используется одиночный тег ***source***. В первую очередь нужно подключать более современные типы файлов, которые указываются в атрибуте ***type***. Браузер возьмет первый файл, который сможет проиграть, и не будет скачивать остальные файлы.

*<video controls muted loop poster=”poster.png”>*

*<source src="video.webm" type="video/webm" />*

*<source src="video.mp4" type="video/mp4" />*

*</video>*

Свойство ***object-fit*** позволяет задать как видео будет подстраиваться под заданные размеры.

*object-fit: fill; // по умолчанию, вписывается без сохранения пропорций*

*object-fit: contain; // вписывается под заданные размеры без нарушения пропорций*

*object-fit: cover; // вписывается без нарушения пропорций, обрезая все лишнее*

*object-fit: none; // отображается без изменений пропорций или размеров*

*object-fit: scale-down; // браузер сам выбирает между none и contain*

Свойство ***object-position*** позволяет выровнять изображение по осям *X* и *Y* после подгонки по размерам.

*object-position: top right; // сначала по X, потом по Y*

Рассмотрим пример подключения своего аудио без использования сторонних сервисов. Для этого нам нужны теги ***audio***. Они имеют такие же атрибуты как у тегов ***video***.

*<audio>*

*<source src="sound.mp3" type="audio/mp3" />*

*<source src="sound.ogg" type="audio/ogg" />*

*</audio>*

Как сделать свой видеоплеер можно почитать [тут](https://habr.com/ru/company/microsoft/blog/127295/).