**Как сделать страницу отзывчивой**

Как мы узнали из прошлой лекции, элементы адаптивной (или отзывчивой) веб-страницы подстраиваются под размер экрана и могу менять свои свойства для разной ширины окна браузера. Но как это происходит?

Инструменты, позволяющие сделать страницу адаптивной:

* резиновые блоки
* гибкие медиафайлы
* медиазапросы (контрольные точки)
* max- и min-значения
* работа с viewport
* растровые и векторные изображения

Теперь рассмотрим как нужно правильно использовать эти инструменты.

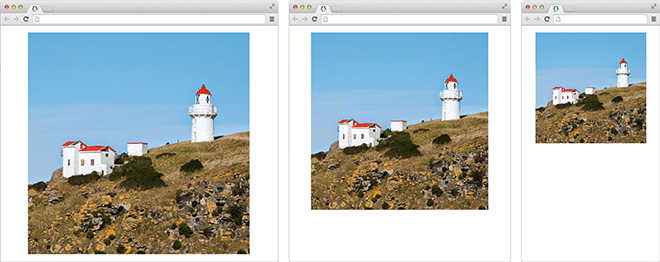
## **Резиновые блоки**

Как мы уже говорили в предыдущей лекции, в основе отзывчивой верстки лежат резиновые блоки. Поэтому блоки с информацией (контейнеры) задаются не в пикселях, а в процентах.

Из любого правила бывают исключения, и вполне вероятна ситуация, в которой по дизайну какой-то блок будет иметь фиксированную ширину. В этом нет ничего страшного, тут все зависит от дизайна и закладываемого поведения блоков.

## **Гибкие медиафайлы**

По умолчанию медиафайлы (например, изображения) не подстраиваются по ширине под размер контейнера. То есть если мы поставили картинку размером в 300px, а при каком-то разрешении резиновый контейнер, в котором лежит эта картинка, вдруг станет шириной в 200px - картинка выйдет за пределы контейнера. Скорее всего это cломает верстку.



Чтобы размер картинки не превышал ширину родительского контейнера - зададим ей ограничение по размеру:

|  |
| --- |
| .img  {  max-width: 100%;  height: auto;  } |

После этого картинка не будет выезжать за пределы контейнера, а высота будет подобрана автоматически.

К медиафайлам также относятся такие элементы, как <vide>, <embed> или <object>. Их также нужно ограничивать по ширине.

## **Медиазапросы (ключевые точки)**

Медиазапросы позволяют использовать разные стили для разного разрешения экрана.

Медиазапросы прописываются в CSS-файлах, из синтаксис выглядит следующим образом:

|  |
| --- |
| @media screen and (max-width: **1024px**)  {  p  {  color: red;  }  } |

Все медиазапросы начинаются с ключевого слова **@media**, за которым следует условие. Условия можно объединять с помощью логического оператора **and**, или исключать с помощью **not**.

После слова **@media** следует тип устройства, с которого используется сайт. Основные типы медиа:

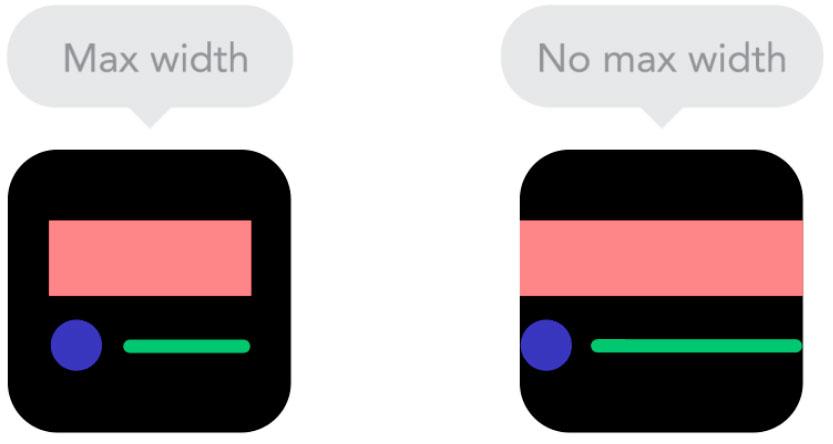
* all (любой тип)
* screen (экран монитора/устройства)
* print (печатающее устройство)

Затем в условиях мы можем указать максимальную или минимальную ширину, для которых будут применяться стили, а также ориентацию устройства (альбомная или портретная и т.д.).

Внутри фигурных скобок мы прописываем стили, которые хотим применить, если условие сработает. Например, пример выше сделает цвет всех абзацев красным при разрешении экрана меньше 1024px.

## **Min- и max-значения**

Контент, занимающий всю ширину экрана — это здорово, если он отображается на мобильном. А если вы откроете страницу через ваш телевизор? Вряд ли увиденная картина обрадует вас.

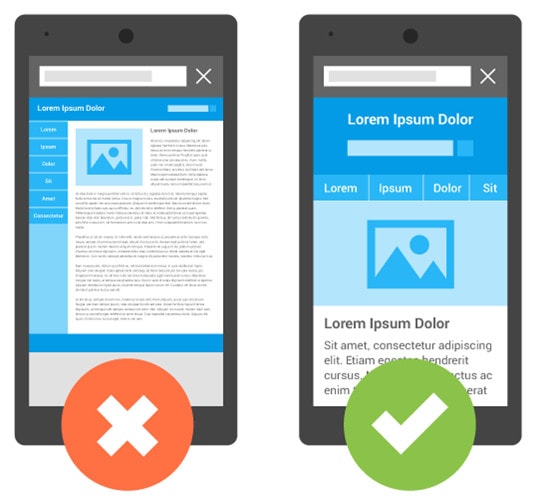


Поэтому здравым решением будет использование минимальных и максимальных значений. Например, если задать блоку свойства `width: 100%` и `max-width: 1000px`, то он будет отображаться на весь экран, если ширина экрана меньше 1000 пикселей; в противном случае, блок будет занимать 1000 пикселей.

## **Работа с viewport**

Viewport - это видимая пользователю область веб-страницы. Т. е. это то, что может увидеть пользователь, не прибегая к прокрутке. У телефонов область просмотра меньше, у монитора или телевизора - больше.

Если не указать viewport, браузер не сможет узнать об области просмотра вашего устройства. В случае с адаптивной страницей это приведет к тому, что она отобразится полностью, как это произошло бы на обычном мониторе. Только уменьшилась бы в масштабе.



Помимо этого есть еще одна проблема с отображением на мобильных устройствах. Есть такое понятие как плотность пикселей. У двух устройств с одинаковой диагональю экрана может быть разное разрешение. Например, iPhone 3 с шириной в 320px и iPhone 4 с шириной 640px.



Текст размером в 16px смотрится нормально на iPhone 3, но его сложно прочитать на iPhone 4, так как он в два раза меньше.

Viewport помогает нам решить эту проблему. Благодаря ему пиксель считается не абсолютной, а относительной величиной, таким образом один пиксель на iPhone 3 будет представлен четырьмя на iPhone 4 (2 на 2).

Пример ниже говорит браузеру о ширине устройства и отключает масштабирование для страницы, чтобы отобразить ее в подготовленном для мобильного устройства виде.

|  |
| --- |
| <meta name="viewport" content="width=device-width, initial-scale=1.0"> |

**Параметры viewport:**

* width - определяет ширину viewport
* height - определяет высоту viewport
* initial-scale – определяет начальный масштаб страницы
* user-scalable - определяет, может ли пользователь изменять масштаб в окне.
* minimum-scale - определяет минимальный масштаб viewport. maximum-scale - определяет максимальный масштаб viewport.

Подробнее о том как рассчитывается коэффициент для относительных пикселей и что такое физическая ширина и CSS-ширина можно почитать здесь:

<https://itchief.ru/lessons/html-and-css/meta-viewport-how-it-works>

## **Растровые и векторные изображения**

Графика на сайтах бывает двух типов: растровая и векторная.

Растровые изображения состоят из пикселей. Как правило это форматы jpg, png, bmp, gif. С помощью растровой графики можно создать рисунок любой сложности. Но у нее есть свои недостатки - при увеличении картинки больше ее оригинального размера теряется качество. Кроме этого, растровые изображения как правило весят больше.

Векторная графика состоит из разных фигур, заданных формулами (линии, круги, кривые). Как правило это форматы cdr, ai, svg, eps. При увеличении размера их качество не уменьшается, а размер простых изображений в векторе в разы меньше растра.



**Как это можно использовать на странице?**

На веб-странице как правило растровые изображения используются в качестве фона, когда нужно вставить живую фотографию или сложный рисунок.

Если требуется поставить простую иконку, то предпочтительнее использовать svg - она будет меньше весить и лучше выглядеть на мобильных устройствах с большой плотностью пикселей.