Адаптивность и кроссбраузерность

Зачем нужен адаптивный дизайн

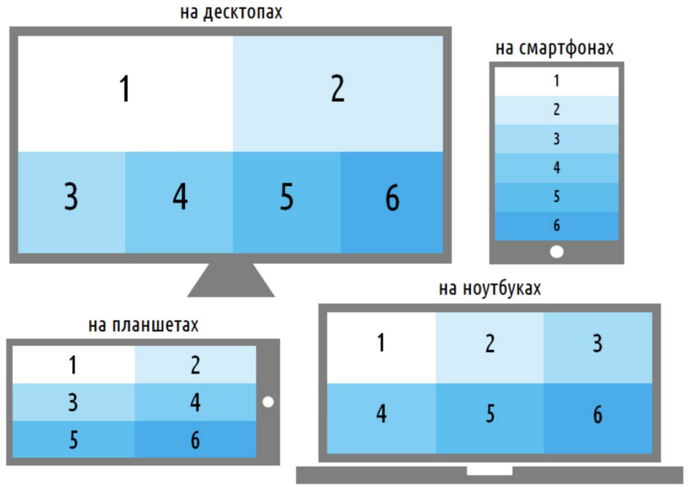
В последнее время люди все чаще пользуются мобильными устройствами и посещение сайтов с мобильных устройств неуклонно растет.
При разработке сайтов важно учитывать то, что сайт должен красиво выглядеть как на компьютере, планшете и на мобильных устройствах с разными разрешениями экранов.

Адаптивный макет сайта - это макет, который может «приспосабливаться» под различные устройства (ширину рабочей области окна браузера). Т.е. на одних устройствах он может иметь одну структуру, а на других - другую.

Зачем нужен адаптивный дизайн

Основные виды макетов:

- **Фиксированный макет**. Имеет строго определённую ширину (в пикселях).
- **Резиновый** (**отзывчивый**, **гибкий**) **макет**. Изменяется в зависимости от ширины рабочей области окна (вкладки) браузера
- **Адаптивный макет**. Может «приспосабливаться» под различные устройства (ширину рабочей области окна браузера). На разных устройствах сайт с таким макетом будет выглядеть по разному



Зачем нужен адаптивный дизайн

В чем разница между отзывчивым (гибким) макетом и адаптивным?

Принципы отзывчивого дизайна (реализуется посредством HTML+CSS):

- Резиновый макет
- Медиазапросы
- Резиновые изображения и видео

Принципы адаптивного дизайна (на базе HTML+CSS, некоторый функционал реализуется с помощью CSS + JavaScript):

- Принципы отзывчивого дизайна
- Mobile first концепция (разработка начинается с создания мобильного, самого простого сайта, который постепенно усложняется)

Медиа-запросы

Медиа-запросы — специальные условные конструкции, которые позволяют применять стили только для определённых устройств.

Медиа-запросы добавляются в CSS-стили аналогично селекторам, записываются следующим образом:

```
@media (условие) {
    …
}
```

Медиа-запросы

Типы медиа-запросов:

- Ориентация экрана (ключ - orientation). Значения:
  - landscape. Условие выполнится для устройств с *горизонтальной* ориентацией экрана (когда ширина больше высоты)
  - portrait. Условие выполнится для устройств с *вертикальной* ориентацией экрана (когда высота больше ширины)

Пример:

```
@media (orientation: landscape) {
        background: #fff;
}
```

Медиа-запросы

Типы медиа-запросов:

- Размер окна просмотра (ключ - width, height, min-width, min-height, max-width, max-height).

  Значение - в единицах размера (px, %)

Пример:

```
@media (max-width: 1024px) {
        padding: 10px;
}
```

Медиа-запросы

Типы медиа-запросов:

- Отношение ширины устройства к высоте (ключ - aspect-ratio, min-aspect-ratio, max-aspect-ratio).

  Значение - соотношение сторон (целые числа)

Пример:

```
@media (aspect-ratio: 16/9) {
        padding: 10px;
}
```

Медиа-запросы

Типы медиа-запросов:

- Разрешение (плотность пикселей) устройства (ключ - resolution, min-resolution, max-resolution). Значение - числа в единицах dpi(количество точек на дюйм), dpcm (количество точек на сантиметр), dppx (количество точек на пиксель)

Пример:

```
@media (min-resolution: 2dppx) {
        …
}
```

Медиа-запросы

Типы медиа-запросов:

- Основные типы устройств:
  - all - все устройства (по умолчанию)
  - print - принтеры и режим предварительного просмотра страницы перед печатью
  - screen - устройства с дисплеями

Пример:

```
@media screen {
    …
}
```

Медиа-запросы

Типы медиа-запросов:

- Логические операторы:
  - and - требует обязательного выполнения всех указанных условий. Пример:

```
@media screen and (min-width: 1200px) {
    …
}
```

  - **запятая** (,) - требует обязательного выполнения хотя бы одного из указанных условий в медиа запросе. Пример:

```
@media (min-width: 544px), (orientation: landscape) {
    …
}
```

Медиа-запросы

Типы медиа-запросов:

- Логические операторы:

  - not - предназначен для отрицания указанного условия (имеет по отношению к оператору and меньший приоритет, т.е. оператор not всегда выполняется после and). Пример:

```
@media not screen and (orientation: portrait) {
    …
}
```

Адаптивность изображений для дисплеев с высоким разрешением

Viewport - видимая пользователю область страницы, которая доступна без прокрутки

Мета-тег *viewport* позволяет указать, какая область просмотра необходима для страницы. Чаще всего его используют для указания ширины области просмотра. Ширина устанавливается через атрибут width:

```
<meta name="viewport" content=«width=700">
```

После установки такого мета-тега, ширина области просмотра сайта станет 700 пикселей.

Метатег viewport был разработан компанией Apple для того, чтобы указывать браузерам на то, в каком масштабе необходимо отображать пользователю видимую область веб-страницы. Другими словами meta viewport предназначен для корректного отображения веб-страницы на смартфонах, планшетах и других устройствах с высокой плотностью пикселей (>200ppi)

Адаптивность изображений для дисплеев с высоким разрешением

Появление экранов с высокой плотностью пикселей поставило перед разработчиками новую проблему — при одинаковых физических размерах у смартфонов может быть разное разрешение. Из-за этого текст, который на одном экране отображается нормально, на другом выглядит заметно меньше.

Метатег viewport решает эту проблему адаптивного дизайна с помощью двух параметров: width и initial-scale.

Параметр width=device-width приравнивает ширину viewport к CSS-ширине устройства.

Адаптивность изображений для дисплеев с высоким разрешением

CSS-разрешение зависит от плотности пикселей.

- Если плотность пикселей меньше 200ppi, то коэффициент будет 1 (у экрана с физическим разрешением 320x480 пикселей будет CSS-разрешение 320x480 пикселей).
- Плотность пикселей 200-300ppi — коэффициент 1,5.
- Плотность пикселей больше 300ppi — коэффициент рассчитывается по формуле плотность/150, а полученное значение округляется (2, 2.5, 3 и так далее)

Что такое кроссбраузерность

Раньше: В 90е годы существовало всего два ведущих браузера - Internet Explorer от Microsoft и Netscape Navigator.

Сейчас: Огромный выбор браузеров, самые популярные в России - Chrome, Mozilla, Safari, Яндекс браузер, Opera, Edge, Internet Explorer.

Разработчики вынуждены учитывать различия в отображении сайта различными браузерами и бороться с багами.

Что такое кроссбраузерность

Кроссбраузерность - способность веб-ресурса отображаться одинаково хорошо во всех популярных браузерах без перебоев в функционировании и ошибок в верстке, с одинаково корректной читабельностью контента.

Нет необходимости добиваться совершенно одинакового отображения визуальной и текстовой информации сайта во всех без исключения версиях браузеров. Достаточно, чтобы:

- ресурс обладал адаптивностью ко всем возможным устройствам просмотра вне зависимости от размеров экрана
- не должно быть проблем с версткой, когда картинки, баннеры, логотипы «наезжают» друг на друга или не отображаются
- текстовая информация должна быть читабельной

Что такое кроссбраузерность

Кроссбраузерными считаются сайты, которые хорошо работают на:

- Google Chrome и производных обозревателях (типа Яндекс.Браузера, Chromium, Opera с 15 версии и т.д.)
- Microsoft Edge (сейчас он тоже работает на движке Blink, как и Chrome)
- Mozilla Firefox
- Safari (для iOS и macOS).

Что такое кроссбраузерность

Параметры, определяющие кроссбраузерность сайта:

- **Расположение элементов**. Если сайт не адаптирован под конкретный браузер, его элементы могут съезжать за пределы экрана, накладываться друг на друга или не отображаться.
- **Текст**. Текст не должен наслаиваться, съезжать или отображаться в виде нечитаемых символов.
- **Скорость загрузки**. Если сайт очень тяжелый, страницы грузятся медленно и зависают, то пользователь очень быстро покинет такой ресурс.
- **Адекватная работа всех кнопок**, **сайдбаров и других функционально активных элементов**.
- **Адаптивность под все устройства**. Ресурс одинаково хорошо должен отображаться и работать на всех устройствах – компьютерах, планшетах, смартфонах.

Что такое кроссбраузерность

Как проверить кроссбраузерность сайта?

- Вручную открыть страницы сайта на самых популярных браузерах.
- Проверить кроссбраузерность сайта автоматически с помощью специальных ресурсов (например, CrossBrowserTesting или Browsershots)

Как обеспечить кроссбраузерность сайта

- Использование CSS-хаков.

CSS-хаки - это фрагменты CSS-кода, которые понимает только определенный браузер, другими он игнорируется.

Хаки - не самый лучший метод устранения ошибок. т.к. они усложняют и делают громоздким код страниц

Как обеспечить кроссбраузерность сайта

- Использование вендорных префиксов.

Префиксы — приставки к названиям CSS-свойств, используемые определенными браузерами. Они позволяют изменять отображение в конкретном обозревателе.

Примеры префиксов:

- -moz- для Mozilla Firefox
- -ms- в Internet Explorer и Microsoft Edge
- -webkit- для Safari, а также браузеров на движках WebKit и Blink
- -o- для старых версий Opera на движке Presto (с 2013 Opera использует Blink)

```
* {
-webkit-box-sizing: border-box;
-moz-box-sizing: border-box;
box-sizing: border-box;
}
```

Как обеспечить кроссбраузерность сайта

- Использование универсальных элементов

Старайтесь использовать только универсальные элементы, которые одинаково работают в большинстве браузеров. Работа только с ними сделает код коротким, чистым и понятным.

Перед использованием какого-нибудь CSS-правила стоит проверить его поддержку.

Это можно сделать на сайте https://caniuse.com/ (там можно проверить реализовано ли данное в конкретном браузере. Кроме этого на сайте показывается количество пользователей в процентах, которые пользуется этой версией браузера.)