CSS3-медиазапросы

В 2001 году в HTML4 и CSS2 была введена поддержка аппаратно-зависимых таблиц стилей, позволившая создавать стили и таблицы стилей для определенных типов устройств. В качестве медиа-типов были определены следующие: `aural`, `braille`, `handheld`, `print`, `projection`, `screen`, `tty`, `tv`. Таким образом, браузер применял таблицу стилей только в случае, когда активизировался данный тип устройства.

Кроме того, было введено ключевое слово `all`, которое использовалось, чтобы указать, что таблица стилей применяется ко всем типам носителей.

В HTML4 медиа-запрос записывался следующим образом:

HTML

```
<link rel="stylesheet" type="text/css" media="screen" href="sans-serif.css">
<link rel="stylesheet" type="text/css" media="print" href="serif.css">
```

Внутри таблицы стилей также можно было объявить, что блоки объявлений должны применяться к определенным типам носителей:

CSS

```
@media screen {
  * {font-family: sans-serif;}
}
```

Предусматривая возможность введения новых значений и значений с параметрами в будущем, для браузеров была реализована поддержка значений атрибута медиа-носителя, указанных следующим образом:

HTML

```
<link rel="stylesheet" media="screen, 3d-glasses, print and resolution > 90dpi" href="...">
```

Текущий синтаксис HTML5 и CSS3 напрямую ссылается на первую спецификацию Media Queries, обновляя

правила для HTML. Также был расширен список характеристик медиа-носителей.

Поддержка браузерами

> ***IE:*** *9.0 (кроме вложенных медиазапросов)*
> ***Edge:*** *12.0*
> ***Firefox:*** *3.5*
> ***Chrome:*** *26.0*
> ***Safari:*** *6.1*
> ***Opera:*** *10.1*
> ***iOS Safari:*** *7.1*
> ***Android:*** *4.4*
> ***Chrome for Android:*** *55.0*

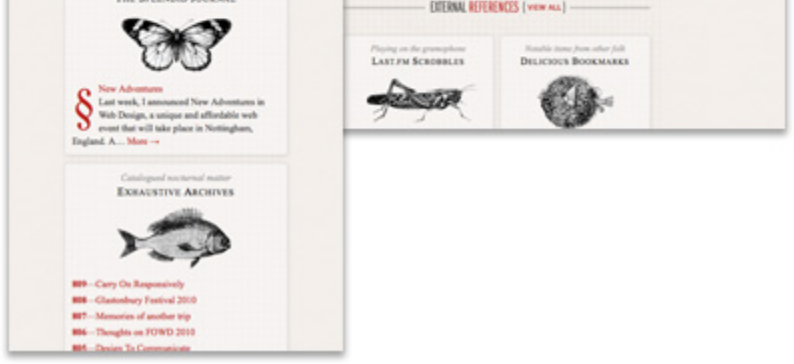

РИС. 1. ПРИМЕР АДАПТИВНОЙ ВЕРСТКИ НА ОСНОВЕ МЕДИАЗАПРОСОВ

1. Что такое медиа-запрос

В общем случае медиа-запрос состоит из ключевого слова, описывающего тип устройства (необязательный параметр) и выражения, проверяющего характеристики данного устройства. Из всех характеристик чаще всего проверяется ширина устройства `width`. Медиа-запрос является логическим выражением, которое возвращает истину или ложь.

Медиа-запросы могут быть добавлены следующими способами:

1) С помощью HTML:

```CSS
<link rel="stylesheet" media="screen and (color)" href="example.css">
```

2) С помощью правила @import внутри элемента `<style>` или внешней таблицы стилей:

```CSS
@import url(color.css) screen and (color);
```

3) Непосредственно в коде страницы:

```HTML
<style>
```

```
@media (max-width: 600px) {
  #sidebar {display: none;}
}
</style>
```

4) Внутри таблицы стилей style.css:

```css
@media (max-width: 600px) {
  #sidebar {display: none;}
}
```

Таблица стилей, прикрепленная через тег `<link>`, будет загружаться вместе с документом, даже если её медиа-запрос вернет ложь.

Для поддержки медиа-запросов в старых браузерах можно воспользоваться JavaScript-библиотекой `css3-mediaqueries.js`, доступную по адресу https://code.google.com/archive/p/css3-mediaqueries-js/.

2. Логические операторы

С помощью логических операторов можно создавать комбинированные медиазапросы, в которых будет проверяться соответствие нескольким условиям.

2.1. Оператор and

Оператор `and` связывает друг с другом разные условия:

```css
@media screen and (max-width: 600px) {
/* CSS-стили */;
}
```

Стили этого запроса будут применяться только для экранных устройств с шириной области просмотра не более

`600px` .

```css
@media (min-width: 600px) and (max-width: 800px) {
/* CSS-стили */;
}
```

Стили этого запроса будут применяться для всех устройств при ширине области просмотра от `600px` до `800px` включительно.

Правило `@media all and (max-width: 600px) {...}` равнозначно правилу `@media (max-width: 600px) {...}` .

2.2. Оператор запятая

Оператор запятая работает по аналогии с логическим оператором `or` .

```css
@media screen, projection {
/* CSS-стили */;
}
```

В данном случае CSS-стили, заключенные в фигурные скобки, сработают только для экранных или проекционных устройств.

2.3. Оператор not

Оператор `not` позволяет сработать медиазапросу в противоположном случае. Ключевое слово `not` добавляется в начало медиазапроса и применяется ко всему запросу целиком, т.е. запрос

```css
@media not all and (monochrome) {...}
```

будет эквивалентен запросу

```css
@media not (all and (monochrome)) {...}
```

Если медиазапрос составлен с использованием оператора запятая, то отрицание будет распространяться только на ту часть, которая идет до запятой, т.е. запрос

```css
@media not screen and (color), print and (color)
```

будет эквивалентен запросу

```css
@media (not (screen and (color))), print and (color)
```

2.4. Оператор only

Оператор `only` используется, чтобы скрыть стили от старых браузеров (поддерживающих синтаксис медиа-запросов CSS2).

```html
media="only screen and (min-width: 401px) and (max-width: 600px)"
```

Эти браузеры ожидают список медиа-типов, разделённых запятыми. И, согласно спецификации, они должны отсекать каждое значение непосредственно перед первым неалфавитно-цифровым символом, который не является дефисом. Таким образом, старый браузер должен интерпретировать предыдущий пример как `media="only"` . Поскольку данного типа медиа-типа не существует, то и таблицы стилей будут игнорироваться.

3. Тип носителя

Тип носителя представляет собой тип устройства, например, принтеры, экраны.

ТАБЛИЦА 1. ТИП НОСИТЕЛЯ

Значение	Описание
`all`	Подходит для всех типов устройств.
`print`	Предназначен для страничных материалов и документов, просматриваемых на экране в режиме предварительного просмотра печати.
`screen`	Предназначен в первую очередь для экранов цветных компьютерных мониторов.
`speech`	Предназначен для синтезаторов речи.

CSS2.1 и Media Queries 3 определяли несколько дополнительных типов, таких как `aural`, `braille`, `embossed`, `projection`, `tty`, `tv` и `handheld`, но они приняты устаревшими в Media Queries 4 и не будут использоваться.

4. Характеристики носителя

К характеристикам медианосителя относятся проверяемые параметры устройства. Значения, которые используются при задании характеристик, являются контрольными точками.

ТАБЛИЦА 2. ХАРАКТЕРИСТИКИ НОСИТЕЛЯ

Параметр	Описание
`width`	Проверяет ширину области просмотра. Значения задаются в единицах длины, `px`, `em` и т.д., например, `(width: 800px)`. Обычно для проверки используются минимальные и максимальные значения ширины. `min-width` применяет правило если ширина области просмотра больше значения, указанного в запросе, `max-width` — ширина области просмотра меньше значения, указанного в запросе.
`height`	Проверяет высоту области просмотра. Значения задаются в единицах длины, `px`, `em` и т.д., например, `(height: 500px)`. Обычно для проверки используются минимальные и максимальные значения высоты. `min-height` применяет правило если высота области просмотра больше значения, указанного в запросе, `max-height` — высота области просмотра которого меньше значения, указанного в запросе.
`aspect-ratio`	Проверяет соотношение ширины к высоте области просмотра. Широкоэкранный дисплей с соотношением сторон 16:9 может быть помечен как `(aspect-ratio: 16/9)`.

	`min-aspect-ratio` проверяет минимальное соотношение, `max-aspect-ratio` — максимальное соотношение ширины к высоте области просмотра.
`orientation`	Проверяет ориентацию области просмотра. Принимает два значения: `(orientation: portrait)` и `(orientation: landscape)`.
`resolution`	Проверяет разрешение экрана (количество пикселей). Значения также могут проверять количество точек на дюйм (dpi) или количество точек на сантиметр (dpcm), например, `(resolution: 300dpi)`. `min-resolution` проверяет минимальное разрешение экрана, `max-resolution` — максимальное.
`color`	Проверяет количество бит на каждый из цветовых компонентов устройства вывода. Например, `(min-color: 4)` означает, что экран конкретного устройства должен иметь 4-битную глубину цвета. `min-color` проверяет минимальное количество бит, `max-color` — максимальное количество бит.
`color-index`	Проверяет количество записей в таблице подстановки цветов. В качестве значения указывается положительное число, например, `(color-index: 256)`. `min-color-index` проверяет минимальное количество записей, `max-color-index` — максимальное количество записей.
`monochrome`	Проверяет количество битов на пиксель монохромного устройства. Значение задается целым положительным числом, например, `(min-monochrome: 8)`. `min-monochrome` проверяет минимальное количество битов, `max-monochrome` — максимальное количество битов.
`-webkit-device-pixel-ratio`	Задаёт количество физических пикселей устройства на каждый CSS-пиксель.

`device-width`, `device-height`, `device-aspect-ratio` являются устаревшими API, они удалены из Media Queries Level 4.

5. Метатег viewport

Для управления разметкой в мобильных браузерах используется метатег `viewport`. Изначально данный тег был представлен разработчиками Apple для браузера Safari на iOS. Мобильные браузеры отображают страницы в виртуальном окне просмотра, которое обычно шире, чем экран устройства. С помощью метатега `viewport` можно контролировать размер окна просмотра и масштаб.

Страницы, адаптированные для просмотра на разных типах устройств, должны содержать в разделе `<head>` метатег `viewport`.

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

Свойство `width` определяет виртуальную ширину окна просмотра, значение `device-width` — физическую ширину устройства. Другими словами, `width` отражает значение `document.documentElement.clientWidth`, а `device-width` — `screen.width`.

При первой загрузке страницы свойство `initial-scale` управляет начальным уровнем масштабирования, `initial-scale=1` означает, что 1 пиксель окна просмотра = 1 пиксель CSS.

РИС. 2. РАЗНИЦА МЕЖДУ WIDTH И DEVICE-WIDTH

6. На какие размеры экрана нужно ориентироваться

При составлении медиазапросов нужно ориентироваться на так называемые **переломные (контрольные) точки дизайна**, т.е. такие значения ширины области просмотра, в которых дизайн сайта существенно меняется, например, появляется горизонтальная полоса прокрутки. Чтобы определить эти точки, нужно открыть сайт в браузере и постепенно уменьшать область просмотра.

CSS

```
/* Smartphones (вертикальная и горизонтальная ориентация) ----------- */
@media only screen and (min-width : 320px) and (max-width : 480px) {
/* стили */
}

/* Smartphones (горизонтальная) ----------- */
@media only screen and (min-width: 321px) {
/* стили */
}

/* Smartphones (вертикальная) ----------- */
@media only screen and (max-width: 320px) {
/* стили */
}

/* iPads (вертикальная и горизонтальная) ----------- */
@media only screen and (min-width: 768px) and (max-width: 1024px) {
/* стили */
}

/* iPads (горизонтальная) ----------- */
@media only screen and (min-width: 768px) and
(max-width: 1024px) and (orientation: landscape) {
/* стили */
}

/* iPads (вертикальная) ----------- */
@media only screen and (min-width: 768px) and (max-width: 1024px) and
(orientation: portrait) {
/* стили */
```

```
}

/* iPad 3**********/
@media only screen and (min-width: 768px) and (max-width: 1024px) and
(orientation: landscape) and (-webkit-min-device-pixel-ratio: 2) {
/* стили */
}
@media only screen and (min-width: 768px) and (max-width: 1024px) and
(orientation: portrait) and (-webkit-min-device-pixel-ratio: 2) {
/* стили */
}

/* Настольные компьютеры и ноутбуки ----------- */
@media only screen  and (min-width: 1224px) {
/* стили */
}

/* Большие экраны ----------- */
@media only screen  and (min-width: 1824px) {
/* стили */
}

/* iPhone 4 ----------- */
@media only screen and (min-width: 320px) and (max-width: 480px) and
(orientation: landscape) and (-webkit-min-device-pixel-ratio: 2) {
/* стили */
}
@media only screen and (min-width: 320px) and (max-width: 480px) and
(orientation: portrait) and (-webkit-min-device-pixel-ratio: 2) {
/* стили */
}

/* iPhone 5 ----------- */
@media only screen and (min-width: 320px) and (max-height: 568px) and
(orientation: landscape) and (-webkit-device-pixel-ratio: 2){
```

```
/* стили */
}
@media only screen and (min-width: 320px) and (max-height: 568px) and
(orientation: portrait) and (-webkit-device-pixel-ratio: 2){
/* стили */
}

/* iPhone 6 ---------- */
@media only screen and (min-width: 375px) and (max-height: 667px) and
(orientation: landscape) and (-webkit-device-pixel-ratio: 2){
/* стили */
}
@media only screen and (min-width: 375px) and (max-height: 667px) and
(orientation: portrait) and (-webkit-device-pixel-ratio: 2){
/* стили */
}

/* iPhone 6+ ---------- */
@media only screen and (min-width: 414px) and (max-height: 736px) and
(orientation: landscape) and (-webkit-device-pixel-ratio: 2){
/* стили */
}
@media only screen and (min-width: 414px) and (max-height: 736px) and
(orientation: portrait) and (-webkit-device-pixel-ratio: 2){
/* стили */
}

/* Samsung Galaxy S3 ---------- */
@media only screen and (min-width: 320px) and (max-height: 640px) and
(orientation: landscape) and (-webkit-device-pixel-ratio: 2){
/* стили */
}
@media only screen and (min-width: 320px) and (max-height: 640px) and
(orientation: portrait) and (-webkit-device-pixel-ratio: 2){
/* стили */
```

```
}

/* Samsung Galaxy S4 ----------- */
@media only screen and (min-width: 320px) and (max-height: 640px) and
(orientation: landscape) and (-webkit-device-pixel-ratio: 3){
/* стили */
}
@media only screen and (min-width: 320px) and (max-height: 640px) and
(orientation: portrait) and (-webkit-device-pixel-ratio: 3){
/* стили */
}

/* Samsung Galaxy S5 ----------- */
@media only screen and (min-width: 360px) and (max-height: 640px) and
(orientation: landscape) and (-webkit-device-pixel-ratio: 3){
/* стили */
}
@media only screen and (min-width: 360px) and (max-height: 640px) and
(orientation: portrait) and (-webkit-device-pixel-ratio: 3){
/* стили */
}
```