Изображения заведомо являются одним из самых сложных аспектов адаптивного веб дизайна. Сегодня мы рассмотрим как элемент <picture>, являющийся решением проблемы адаптивных изображений, можно использовать *прямо сейчас*.

**Вначале о проблеме**

Времена попиксельного (pixel perfect) и дизайна фиксированной ширины (fixed-width) ушли в прошлое. Теперь во времена широкоформатных мониторов, интернет телевидения, планшетов и смартфонов различных размеров наши дизайны должны удовлетворять любому устройству шириной от 320px до потенциальных 7680px.

Вместе с множеством разрешений экранов приходит необходимость в сжатии или растяжении изображений для того, чтобы соответствовать различным требованиям. Это может оказаться проблей, т.к. за исключением векторной графики, у подавляющего большинства изображений есть базовая ширина в px, которую нельзя изменить.

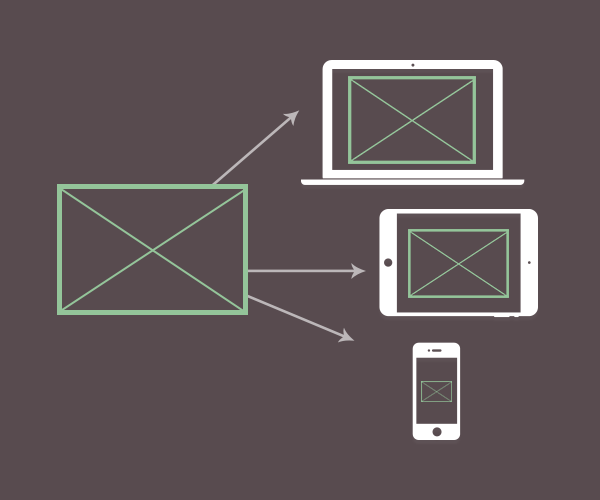
Так что же нам делать?

**Текущее общепринятое решение**

Как правило, вы найдете этот код на любом сайте с адаптивные дизайном:

|  |  |
| --- | --- |
| 1  2  3  4 | img {      max-width: 100%;      height: auto;  } |

Здесь используется max-width: 100%; для гарантии того, что изображение никогда не выйдет за пределы ширины родительского контейнера. Если родительский контейнер сжимается до ширины, меньшей чем ширина изображения - последнее сжимается вместе с контейнером. Установка height: auto; нужна для сохранения пропорций.



Одно "жидкое" изображение под все случаи

Проблема решается лишь в одном ключе, позволяя нам показывать одно и то же изображение для всех случаев. Но это не дает нам возможность устанавливать *разные* изображения для разных ситуаций.

**Новое решение: <picture>**

<picture> это новый элемент, который является частью HTML5.

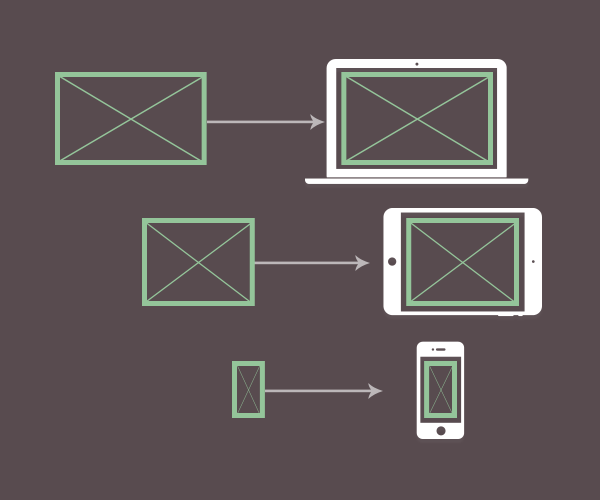
Он реализует способ описания адаптивных изображений таким же способом, как это делается в <audio> и <video>. Таким образом можно размещать несколько тегов <source>, каждый из которых содержит имена файлов различных изображений вместе с условиями, при которых те должны быть загружены.

Это позволит загружать разные изображения в зависимости от:

* Результатов media выражений, например высоты, ширины или ориентации видимой области
* Плотности пикселей

Это в свою очередь означает, что вы можете:

* Загружать файлы изображений соответствующего размера, эффективно используя пропускную способность.
* Загружать по-разному ориентированные изображения в разных пропорциях для того, чтобы соответствовать изменениям в макете при разных ширинах.
* Загружать изображения с высоким разрешением для дисплеев с повышенной плотностью пикселей.



Использование разных изображений в зависимости от условий

**Как работает <picture>?**

Основные шаги при работе с <picture>:

1. Создайте открывающий и закрывающий теги <picture></picture>.
2. Внутри создайте <source> элемент для каждого выражения, что хотите обработать.
3. Добавьте атрибут media, содержащий выражения для таких вещей как высота и ширина области просмотра, ориентация и т.д.
4. Добавьте атрибут srcset с соответствующим именем файла изображения для загрузки.
5. Добавьте дополнительные имена файлов к атрибуту srcset, если нужно поддерживать разную плотность пикселей, например для Retina дисплеев.
6. Добавьте резервный (fallback) элемент <img>.

Вот простой пример, где для случая, когда ширина видимой области меньше 768px - загружается уменьшенное (smaller) изображение:

|  |  |
| --- | --- |
| 1  2  3  4  5 | <picture>      <source srcset="smaller.jpg" media="(max-width: 768px)">      <source srcset="default.jpg">      <img srcset="default.jpg" alt="My default image">  </picture> |

Можно заметить, что синтаксис, используемый в атрибуте mediaявляется таким же, как и при использовании в CSS media queries. Вы можете использовать те же самые проверки, т.е. проверять max-width, min-width, max-height, min-height, orientation и т.д.

Эти проверки можно использовать для того, чтобы например загружать альбомную (landscape) или книжную (portrait) версию изображения в зависимости от ориентации устройства, к тому же можно одновременно проверять размеры в этих выражениях. Например:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | <picture>      <source srcset="smaller\_landscape.jpg" media="(max-width: 40em) and (orientation: landscape)">      <source srcset="smaller\_portrait.jpg" media="(max-width: 40em) and (orientation: portrait)">      <source srcset="default\_landscape.jpg" media="(min-width: 40em) and (orientation: landscape)">      <source srcset="default\_portrait.jpg" media="(min-width: 40em) and (orientation: portrait)">      <img srcset="default\_landscape.jpg" alt="My default image">  </picture> |

Этот код загружает уменьшенную альбомную (landscape) версию изображения для устройств с малым экраном и соответствующей ориентацией. И увеличенную версию того же изображения для устройств с большим экраном.

Если устройство имеет книжную (portrait) ориентацию - загружается книжная версия изображения, уменьшенная для устройств с малым и увеличенная для устройств с большим экраном.

Если вы хотите предоставлять изображения в другом разрешении для дисплеев с повышенной плотностью пикселей, это можно сделать, указав дополнительные имена файлов в атрибуте srcset. Давайте к примеру посмотрим на наш первый кусок кода с добавлением обработки для Retina 2x дисплеев:

|  |  |
| --- | --- |
| 1  2  3  4  5 | <picture>      <source srcset="smaller.jpg, smaller\_retina.jpg 2x" media="(max-width: 768px)">      <source srcset="default.jpg, default\_retina.jpg 2x">      <img srcset="default.jpg, default\_retina.jpg 2x" alt="My default image">  </picture> |

Т.к. сначала обрабатывается media query, вы можете управлять размерами изображения, с которыми оно будет показано на экране. Затем будет проверяться плотность пикселей: если дисплей поддерживает повышенную плотность и в пользовательских настройках есть на это разрешение - будут загружены соответствующие версии изображений.

**Использование <picture> сегодня**

Прямо сейчас встроенная поддержка <picture> реализована в Chrome, Firefox и Opera. В будущем, вполне вероятно, мы увидим широкую поддержку и другими браузерами. Но до этого момента еще надо дожить.

Тем временем вам не нужно ждать, если хотите использовать <picture>прямо сейчас. Просто воспользуйтесь [Picturefill 2.0](http://scottjehl.github.io/picturefill/" \t "_self); polyfill от умных людей из [Filament Group](http://filamentgroup.com/" \t "_self).



После скачивания файла **picturefill.js** в свой проект просто подключите его в шапке:

|  |  |
| --- | --- |
| 1 | <script src="picturefill.js"></script> |

Есть также опция для асинхронной загрузки скрипта, о которой можно прочитать в [документации Picturefill](http://scottjehl.github.io/picturefill/).

При использовании этого скрипта элемент <picture> будет работать так, как я и объяснял, но с несколькими ограничениями.

**Ограничения Picturefill**

**IE9**

Picturefill отлично работает с разными версиями IE, однако IE9 не поддерживает <source> элементы, которые используются внутри <picture></picture>. Чтобы обойти это, оберните source элементы в <video> теги с помощью условных комментариев; это сделает их видимыми для IE9, например:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | <picture>      <!--[if IE 9]><video style="display: none;"><![endif]-->      <source srcset="smaller.jpg" media="(max-width: 768px)">      <source srcset="default.jpg">      <!--[if IE 9]></video><![endif]-->      <img srcset="default.jpg" alt="My default image">  </picture> |

**Android 2.3**

Как и IE9, Android 2.3 не показывает <source> элементы внутри <picture>. Однако он понимает атрибут srcset при использовании в обычных <img>тегах. Убедитесь в том, что всегда включаете резервный <img> с именем файла по умолчанию в атрибут srcset для Android 2.3 и других браузеров, которые могут иметь такую же проблему.

**Требуется JavaScript и встроенная поддержка Media Query**

Соответственно требуется, чтобы JavaScript был включен в браузере. Picturefill 2.0 не предоставляет «no-js» решения, ведь если это будет сделано, то когда браузер выкатит нативную поддержку <picture> - будет показываться уже несколько изображений. Однако, вы можете использовать Picturefill 1.2, если параметр «no-js» является для вас обязательным.

Другим требованием Picturefill является встроенная поддержка media query, чтобы позволить обрабатывать выражения в атрибуте media. Все современные браузеры поддерживают media-выражения, лишь IE8 и ниже не имеют их поддержки, что соответствует лишь [малой части пользователей](http://www.w3schools.com/browsers/browsers_explorer.asp).

Advertisement

**Возможны дополнительные HTTP-запросы**

Возможно, что в браузерах, которые имеют встроенную поддержку srcset, но пока не имеют поддержки <picture>, указанный в резервном элементе <img> файл может быть запрошен до того, как будет определен подходящий вариант из <source>.

Это лишь временная проблема, и она пропадет как только выкатят встроенную поддержку <picture>.

*которые помогут вам правильно применять эти элементы.*

Тег figure обычно используется для вставки изображений:

|  |  |
| --- | --- |
| 1  2  3 | <figure>    <img src="dog.jpg" alt="Maltese Terrier">  </figure> |

Тег figure – замкнутая единица контента. Если вы переместите элемент чуть ниже на странице или в самый низ, это никак не повлияет на его значение в документе. Поэтому нужно помнить, что не все изображения можно указывать в теге figure.

### Несколько изображений в теге figure

Если изображения связаны между собой и подаются в общем контексте, то в тег figure можно прописать несколько img.

|  |  |
| --- | --- |
| 1  2  3  4  5 | <figure>    <img src="dog1.jpg" alt="Maltese Terrier">    <img src="dog2.jpg" alt="Black Labrador">    <img src="dog3.jpg" alt="Golden Retriever">  </figure> |

### Figure можно использовать и с другими тегами

Тег figure не ограничивается только лишь изображениями. Его можно использовать с:

Блоками для кода,

Видео,

Аудиофайлами или

Рекламой.

Пример вставки блока с кодом в тег figure:

|  |
| --- |
| <figure>    <pre>          p {            color: #333;            font-family: Helvetica, sans-serif;            font-size: 1rem;        } |

### Вложенность одного figure в другой

Тег figure можно поместить внутрь другого тега figure. В коде ниже для улучшения семантики был добавлен ARIA атрибут role.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | <figure role="group">    <figcaption>Dog breeds</figcaption>    <figure>      <img src="dog1.jpg" alt="Maltese Terrier">      <figcaption>Adorable Maltese Terrier</figcaption>    </figure>    <figure>      <img src="dog2.jpg" alt="Black Labrador">      <figcaption>Cute black labrador</figcaption>    </figure>  </figure> |

### Правильное применение figcaption

Тег figcaption представляет собой подпись или легенду к тегу figure. Не всегда тегу figure нужен figcaption. Однако в случаях, когда этот тег необходим, он должен быть первым или последним внутри figure:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | <figure>    <figcaption>Three different breeds of dog.</figcaption>    <img src="dog1.jpg" alt="Maltese Terrier">    <img src="dog2.jpg" alt="Black Labrador">    <img src="dog3.jpg" alt="Golden Retriever">  </figure> |

Или

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | <figure>    <img src="dog1.jpg" alt="Maltese Terrier">    <img src="dog2.jpg" alt="Black Labrador">    <img src="dog3.jpg" alt="Golden Retriever">    <figcaption>Three different breeds of dog.</figcaption>  </figure> |

Если нужно добавить больше семантики изображению, можно использовать теги h1 и p.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | <figure>    <img src="dogs.jpg" alt="Group photo of dogs">    <figcaption>      <h2>Puppy School</h2>      <p>Championship Class of 2016</p>    </figcaption>  </figure> |