GeekBrains

Профессиональная вёрстка

Урок 7

Практическое применение адаптивной верстки

Эффекты перехода. Применение трансформации.

Flexbox

CSS flexbox (Flexible Box Layout Module) – это система компоновки элементов. Она состоит из flex-контейнера – родительского контейнера и flex-элементов – дочерних блоков. Дочерние элементы могут выстраиваться в строку или столбик, а оставшееся свободное пространство распределяется между ними различными способами.

Благодаря flexbox элементы ведут себя предсказуемо на всех типах устройств и при различных размерах экрана. Модель гибкой разметки имеет преимущества перед блочной за счет отсутствия плавающих элементов (свойство float) и отсутствия схлопывания внешних отступов margin.

Дочерние flex-элементы можно располагать в любом направлении, благодаря гибким размерам они могут приспосабливаться к размерам экрана. Первоначальный порядок отображения элементов зависит от их порядка в исходном коде. Изменить направление и порядок отображения элементов можно за счет свойств flex-direction и order.

Flex-контейнер ведет себя как блочный элемент в случае display: flex; или как строчный для display: inline-flex;. В зависимости от размера экрана элементы в контейнере расширяются для заполнения свободного пространства или уменьшаются для избежания его переполнения.

В модели flexbox для внутренних блоков не работают такие CSS-свойства, как float, clear, vertical-align. На flex-контейнер не оказывают влияние свойства column-, задающие колонки в тексте.

По умолчанию flex-элементы располагаются по горизонтали – вдоль главной оси main axis, а поперечная ось cross axis пересекает главную. Задавая направление контента, можно поменять местами направления главной и поперечной осей.

Если задано flex-flow: row; или flex-flow: row-reverse;, главная ось будет расположена горизонтально, а поперечная – вертикально. Если задано flex-flow: column; или flex-flow: column-reverse;, то главная ось будет расположена вертикально, а поперечная – горизонтально.

Свойства flex-контейнера

display: flex; //для родительского элемента.

После установки данных значений свойства каждый дочерний элемент автоматически становится flex-элементом, выстраиваясь в ряд (вдоль главной оси) колонками одинаковой высоты, равной высоте блока-контейнера. При этом блочные и строчные дочерние элементы ведут себя одинаково, т.е. ширина блоков равна ширине их содержимого с учетом внутренних полей и рамок элемента.

justify-content

Свойство выравнивает flex-элементы по ширине flex-контейнера, распределяя оставшееся свободное пространство. Для выравнивания элементов по вертикали используется свойство align-content.

- **flex-start** – значение по умолчанию. Flex-элементы позиционируются от начала flex-контейнера.
- **flex-end** – flex-элементы позиционируются относительно правой границы flex-контейнера.
- **center** – flex-элементы выравниваются по центру flex-контейнера.
- **space-between** – flex-элементы выравниваются по главной оси, свободное место между ними распределяется следующим образом: первый блок располагается в начале flex-контейнера,

последний блок – в конце, все остальные блоки равномерно распределены в оставшемся пространстве, а свободное пространство равномерно распределяется между элементами.
- **space-around** – flex-элементы выравниваются по главной оси, а свободное место делится поровну, добавляя отступы справа и слева.

align-items

Свойство выравнивает flex-элементы, в том числе анонимные flex-элементы, по перпендикулярной оси (по высоте). Не наследуется.

- **stretch** – значение по умолчанию. Flex-элементы растягиваются, занимая все пространство по высоте.
- **flex-start** – flex-элементы выравниваются по левому краю flex-контейнера относительно верхнего края блока-контейнера.
- **flex-end** – flex-элементы выравниваются по левому краю flex-контейнера относительно нижнего края блока-контейнера.
- **center** – flex-элементы выравниваются по центру flex-контейнера.
- **baseline** – flex-элементы выравниваются по базовой линии.

flex-direction

Свойство определяет, каким образом flex-элементы укладываются во flex-контейнере, задавая направление главной оси flex-контейнера. Они могут располагаться в двух главных направлениях: горизонтально, как строки, и вертикально, как колонки. Главная ось по умолчанию идет слева направо. Поперечная – сверху вниз.

- **row** – значение по умолчанию, слева направо (в rtl – справа налево). Flex-элементы выкладываются в строку. Начало (main-start) и конец (main-end) направления главной оси соответствуют началу (inline-start) и концу (inline-end) инлайн оси (inline-axis).
- **row-reverse** – направление справа налево. Flex-элементы выкладываются в строку относительно правого края контейнера.
- **column** – направление сверху вниз. Flex-элементы выкладываются в колонку.
- **column-reverse** – колонка с элементами в обратном порядке, снизу вверх.

flex-wrap

Свойство управляет тем, как flex-контейнер будет выкладывать flex-элементы – в одну строку или в несколько, и направлением, в котором будут укладываться новые строки. По умолчанию flex-элементы укладываются в одну строку. При переполнении контейнера их содержимое будет выходить за границы flex-элементов. Не наследуется.

- **nowrap** – значение по умолчанию. Flex-элементы не переносятся, а располагаются в одну линию слева направо (в rtl – справа налево).
- **wrap** – flex-элементы переносятся, располагаясь в несколько горизонтальных рядов (если не помещаются в один ряд) в направлении слева направо (в rtl справа налево).
- **wrap-reverse** – flex-элементы переносятся, располагаясь в обратном порядке слева направо, при этом перенос происходит снизу вверх.

Свойства flex-элементов

Порядок отображения элементов **order.** Свойство определяет порядок, в котором flex-элементы отображаются внутри flex-контейнера. По умолчанию для всех flex-элементов задан порядок order: 0;

и они следуют друг за другом по мере добавления во flex-контейнер. Самый первый flex-элемент по умолчанию расположен слева. Чтобы поставить любой flex-элемент в начало строки, ему нужно назначить order: -1;, в конец строки – order: 1;.

align-self

Свойство отвечает за выравнивание отдельно взятого flex-элемента по высоте flex-контейнера. Переопределяет выравнивание, заданное align-items.

- **auto** – значение по умолчанию. Flex-элемент использует выравнивание, указанное в свойстве align-items flex-контейнера.
- **flex-start** – flex-элемент выравнивается по верхнему краю flex-контейнера относительно левой границы.
- **flex-end** – flex-элемент выравнивается по нижнему краю flex-контейнера относительно левой границы.
- **center** – flex-элемент выравнивается по высоте по середине flex-контейнера относительно левой границы.
- **baseline** – flex-элемент выравнивается по базовой линии flex-контейнера относительно левой границы.
- **stretch** – flex-элемент растягивается на всю высоту flex-контейнера, при этом учитываются поля и отступы.

Использование графики формата SVG

SVG (Scalable Vector Graphics – масштабируемая векторная графика) – стандарт векторной графики, разработанный консорциумом W3C.

SVG – это язык разметки для описания двумерных графических приложений и изображений, входящий в подмножество расширяемого языка разметки XML. Сюда относится также ряд связанных графических скриптов.

Достоинства SVG:

- Графика в формате SVG создается с использованием математических формул, которые при изменении размера изображения можно скорректировать. Таким образом векторные изображения масштабируются лучше, чем растровые.
- Размер векторной картинки обычно меньше, чем у сравнимых по качеству изображений в форматах JPEG, GIF или PNG.
- SVG-графика имеет текстовый формат, который можно и править в блокноте, и рисовать в графических векторных редакторах Adobe Illustrator, CorelDRAW.
- Скрипты и анимация в SVG позволяют создавать динамичную и интерактивную графику.
- Текст в графике SVG является текстом, а не изображением, поэтому он индексируется поисковыми системами (если не переведен в кривые).
- К SVG-формату можно подключать внешние таблицы стилей CSS, глобальные стили внутри контейнера <style>...</style> или добавлять внутренние стили с помощью атрибута style в тегах фигур и путей.

Использование SVG в HTML5

Использование <object> тега

Если планируется использовать более продвинутые функции SVG, такие как применение таблицы стилей CSS или внедрение скриптов, тег HTML5 <object> – лучший способ.

```
<object type="image/svg+xml" data="image.svg" width="200" height="200" >
Ваш браузер не поддерживает SVG
</object>
```

Для старых браузеров, не поддерживающих SVG, можно использовать следующий метод:

```
<object type="image/svg+xml" data="SvgImg.svg" width="200" height="200">
  <img src="SvgImg.png" width="200" height="200" alt="image format png" />
</object>
```

Браузер, не понимающий SVG, проигнорирует тег <object>, перейдет к следующему тегу <img>, обработает его, как обычный HTML-тег, и выведет картинку.

Использование <iframe> тега

Так как браузеры могут отрисовывать по своим правилам SVG-документы, это дает возможность загружать картинки внутри тегов <iframe>.

```
<iframe src="SvgImg.svg">
  <img src="SvgImg.png" width="200" height="200" alt="image format png" />
</iframe>
```

Это хороший метод, если вы хотите полностью отделить SVG-код и скрипт на главной странице.

Использование <img> тега

SVG-документ может быть добавлен на веб-страницу, как любое другое изображение:

```
<img src="image.svg" width="200" height="200" alt="image description">
```

Для браузеров, не воспринимающих SVG, есть способ замены *.svg изображением *.png:

```
<img src="image.svg" onerror="this.onerror=null; this.src='image.png'">
```

Из соображений безопасности при этом способе добавления SVG браузеры отключают скрипты, связывания и другие типы интерактивности, когда они добавляются на страницу. Кроме того, IE9, Safari, Chrome не принимают стили, если они определены в отдельном файле таблицы стилей.

Использование CSS Background Image

SVG может быть использован в качестве CSS-фона для любого элемента:

```
#AnyElement
{
    background-image: url(image.svg);
}
```

Как и при использовании тега <img>, связывание, скриптование и другие методы интерактивности будут недоступны.

Введение

Ранее приходилось работать с ограниченным набором шрифтов, пригодных для веб-страниц, о которых заведомо известно, что они работают в разных браузерах и разных операционных системах. Данная ситуация не подходит для современных веб-страниц, где шрифты играют огромную роль в создании общего впечатления о документе.

В CSS3 поддержку сложных шрифтов обеспечивает возможность @font-face, которая применяется следующим образом:

1. Требуемый шрифт (или несколько версий шрифта для поддержки разных браузеров) загружается на сайт.
2. Каждый шрифт регистрируется в таблице стилей с помощью команды @font-face.
3. Зарегистрированный шрифт используется в правилах стиля указанием его названия так же, как и обычные веб-шрифты.
4. Когда браузер обнаруживает таблицу стилей, в которой используется специальный веб-шрифт, он загружает этот шрифт с сервера в свой кеш для временного хранения страниц и изображений. После этого браузер использует этот шрифт только для данной страницы или сайта. Если этот же шрифт указывается в другой странице, он должен быть зарегистрирован на этой странице и загружен на ее сервер, откуда может быть загружен браузером в свой кеш.

Форматы веб-шрифтов

Все современные браузеры поддерживают возможность @font-face, но не все поддерживают одинаковые типы файлов шрифтов. Например, Internet Explorer, который обеспечивает использование @font-face в течение многих лет, поддерживает только файлы типа EOT (Embedded OpenType). Этот формат предоставляет ряд преимуществ: в нем используется сжатие для уменьшения объема файла шрифтов, а также применяется строгое лицензирование для веб-сайтов, чтобы шрифт нельзя было украсть с одного сайта и использовать на другом.

Но формат EOT никогда не пользовался большой популярностью и не используется никакими другими браузерами. Вместо него браузеры работают с более знакомыми стандартами шрифтов, применяемыми в компьютерных приложениях, – TTF (TrueType) и OTF (OpenType PostScript). Кроме этого, существуют еще два типа отображения шрифтов: SVG и WOFF. В таблице ниже дано краткое описание всех этих форматов шрифтов.

Форматы внедряемых шрифтов:

Формат	Описание	Используется в
TTF (TrueType), OTF (OpenType PostScript)	Распространенные форматы шрифтов настольных компьютеров	Firefox (до версии 3.6), Chrome (до версии 6), Safari и Opera
EOT (Embedded OpenType)	Формат, специфичный для продуктов корпорации Microsoft. Не завоевал популярности у браузеров, за исключением Internet Explorer	Internet Explorer (до версии 9)
SVG (Scalable Vector Graphics)	Универсальный графический формат, который можно использовать для создания шрифтов. Дает хорошие, но не отличные результаты – медленно отображается и демонстрирует текст пониженного качества	Safari Mobile (на iPhone и iPad до iOS 4.2) и мобильные устройства под управлением операционной системы Android
WOFF (Web Open Font Format)	Возможно, единый формат шрифтов будущего. Поддерживается новыми версиями браузеров	Любой поддерживающий браузер, начиная с Internet Explorer 9, Firefox 3.6 и Chrome 6

Таким образом, если вы хотите использовать возможность @font-face и поддерживать широкий диапазон браузеров, нужно предоставлять шрифт в нескольких разных форматах. Как минимум, его нужно предоставить в TTF или OTF, EOT и SVG. Хорошо (но не обязательно) также предоставить шрифт в перспективном формате WOFF, который может стать более популярным и лучше поддерживаемым в будущем. Одним из достоинств этого формата является использование сжатых файлов, что сокращает время их загрузки.

Даже если вы последуете всем приведенным правилам и предоставите все требуемые форматы шрифтов, с ними могут возникнуть следующие проблемы:

- Многие шрифты выглядят плохо в операционной системе Windows XP, т. к. в настройках многих компьютеров с этой операционной системой отключено сглаживание (anti-aliasing), а без применения сглаживания шрифты выглядят очень непривлекательно.
- От пользователей поступали жалобы о проблемах с печатью определенных внедренных шрифтов из некоторых браузеров или в операционных системах.
- Некоторые браузеры страдают проблемой FOUT (Flash of Unstyled Text, вспышка нестилизованного текста). Это явление происходит, когда внедренный шрифт не успевает загрузиться вовремя, и страница отображается сначала в резервном шрифте, а потом воспроизводится повторно на встроенном шрифте. Эта проблема особенно заметна в старых версиях браузера Firefox. Если вас это сильно беспокоит, можете воспользоваться библиотекой JavaScript от Google, позволяющей разработчику определить резервные стили, которые используются вместо незагруженных внедренных шрифтов, таким образом предоставляя ему полный контроль над воспроизведением текста в любое время.

Хотя эти небольшие проблемы иногда и возникают, большинство из них постепенно решается с выпуском новых версий браузеров. Например, браузер Firefox теперь сводит эффект FOUT к минимуму, ожидая до трех секунд, пока не загрузится внедренный шрифт, прежде чем использовать резервный шрифт.

Наборы шрифтов

На рисунке ниже показано несколько шрифтов из представленных на сайте Font Squirrel: www.fontsquirrel.com.

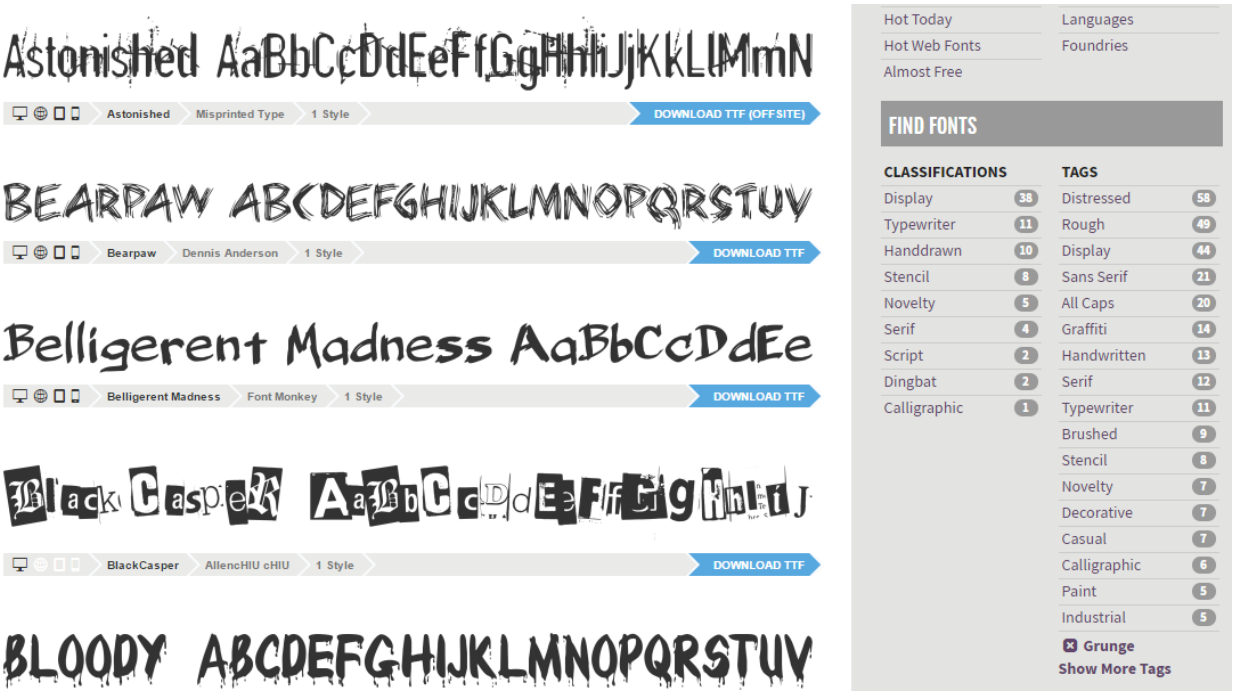

Рис. 1. Пример выбора шрифтов с сайта Font Squirrel

Сайт Font Squirrel предоставляет для загрузки несколько сотен высококачественных шрифтов, организованных в разделы по категориям (например, Calligraphic, Grunge или Retro). Все эти шрифты бесплатны для использования на персональном компьютере для создания документов или на веб-странице в Интернете.

Набор шрифтов загрузится в виде файла, сжатого в формате ZIP, который содержит несколько файлов шрифта в разных форматах. Чтобы использовать шрифт в своей веб-странице, файлы разных форматов шрифта нужно загрузить на веб-сервер в папку этой веб-страницы. После этого шрифт нужно зарегистрировать, чтобы он был доступен для использования в таблице стилей. Регистрация выполняется с помощью правила @font-face в начале таблицы стилей, которое выглядит следующим образом:

```
@font-face {
     font-family: 'MetrophobicRegular';
   src: url('Metrophobic-webfont.eot');
   src: local('Metrophobic'),
        url('Metrophobic-webfont.eot?#iefix') format('embedded-opentype'),
        url('Metrophobic-webfont.woff') format('woff'),
        url('Metrophobic-webfont.ttf') format('truetype'),
        url('Metrophobic-webfont.svg#MetrophobicRegular') format('svg');
   font-weight: normal;
   font-style: normal;
}
```

Строки приведенного кода выполняют следующие функции:

- Выражение @font-face регистрирует шрифт для его дальнейшего применения в таблице стилей.
- Шрифту можно присвоить любое название. Это название будет употреблено позже, при использовании шрифта.
- Первым должен быть указан формат EOT, т. к. дальнейшая часть правила сбивает с толку Internet Explorer и тот не обращает внимания на остальные форматы. Функция таблицы стилей url() указывает браузеру загрузить файл из обозначенного URL. Если шрифт размещен в одной папке с веб-страницей, здесь можно просто указать название файла.
- Функция local() указывает браузеру название шрифта, и если этот шрифт установлен на компьютере посетителя веб-страницы, браузер использует локальный шрифт. В редких случаях это может вызвать проблемы. Например, в зависимости от того, где установлен шрифт на компьютере посетителя, компьютеры Mac OS X могут вывести предупреждение о нарушении безопасности, или же может загрузиться другой шрифт с таким же названием, как и ваш. По этой причине веб-разработчики иногда указывают явно несуществующее имя файла, чтобы браузер не нашел локального шрифта. В качестве простого имени такого типа можно использовать какой-либо бессмысленный символ.
- Последний шаг – сообщить браузеру о других файлах шрифтов, которые он может использовать. Если имеется файл шрифта типа WOFF, укажите этот файл первым, т. к. данный формат предоставляет наилучшее качество шрифта. Следующим укажите файл формата TTF или OTF, а самым последним – файл формата SVG.

Зарегистрировав веб-шрифт с помощью функции ©font-face, вы можете использовать его в любой таблице стилей. Для этого используется свойство font-family, которому присваивается значение в виде названия семейства шрифтов, зарегистрированного с помощью функции @font-face (в строке 2). Далее приведен пример использования этого шрифта в правиле таблицы стилей:

```
body {
       font-family: Amerika;
}
```

Это правило применяет веб-шрифт ко всей странице, хотя область его применения можно было бы ограничить определенными элементами или использовать классы. Но шрифт нужно в обязательном порядке зарегистрировать до того, как использовать в правиле таблицы стилей, иначе он не будет работать должным образом.

Веб-шрифты Google

Еще один источник бесплатных веб-шрифтов – служба Google Web Fonts. Ее особенность в том, что пользователю не нужно беспокоиться о форматах шрифтов, т. к. Google определяет браузер посетителя страницы и автоматически отправляет файл шрифта нужного формата.

Чтобы использовать шрифт Google в своих страницах, выполните последовательность шагов:

1. Откройте в браузере страницу Google Web Fonts (https://fonts.google.com/). Она содержит длинный список имеющихся в наличии шрифтов.

2. Настройте опции поиска. Если вы знаете название требуемого вам шрифта, введите его в поле поиска. В противном случае нужно просматривать все шрифты, прокручивая страницу вверх, что для латинских шрифтов может занять значительное время. Чтобы ускорить поиск, можно отсортировать и отфильтровать список согласно определенным критериям.

3. Обнаружив подходящий шрифт, щелкните по знаку «+». Откроется новое окно, содержащее описание шрифта и все подробности.
4. Если на этом этапе вы решите использовать данный шрифт, вы можете скачать его себе и использовать так, как показано в предыдущем примере, или использовать код из ссылки на таблицу стилей, которую нужно вставить в разметку вашей веб-страницы, и примера правила таблицы стилей, применяющего шрифт.
5. Вставьте в свою веб-страницу ссылку на таблицу стилей. Например:

```
<link href="https://fonts.googleapis.com/css?family=Underdog"
rel="stylesheet">
```

Данная таблица стилей регистрирует шрифт посредством функции @font-face. Лучше всего то, что Google предоставляет файлы шрифтов, избавляя вас от необходимости загружать их на сайт.

Используйте шрифт в любое время, обращаясь к нему по его названию. Например, далее приводится правило для применения должным образом зарегистрированного шрифта Ceviche One в заголовках первого уровня, предоставляющее резервный шрифт на случай, если браузер не сможет загрузить основной шрифт:

```
h1 {
      font-family: 'Underdog', arial, serif;
}
```

Размещение текста в нескольких колонках

В CSS3 был добавлен абсолютно новый модуль для размещения текста в нескольких колонках. Таким образом, у разработчиков появляется гибкое средство для решения проблемы длинного текста без потерь его читаемости. Создание нескольких колонок текста не требует почти никаких усилий и может быть выполнено двумя способами. Первый подход – установить количество требуемых колонок с помощью свойства column-count, как показано в следующем коде:

```
.Content {
   text-align:justify;
   column-count:3;
}
```

Для браузеров Firefox, Chrome и Safari это свойство используется с префиксом разработчика браузера:

```
.Content {
   text-align:justify;
   -moz-column-count: 3;
   -webkit-column-count: 3;
   column-count:3;
}
```

Указание точного числа колонок лучше всего подходит для веб-страниц с компоновкой фиксированного размера. Но если размеры компоновки меняются в соответствии с размерами окна браузера, ширина колонок может оказаться слишком большой, и текст в них будет трудно читать. Во избежание такой проблемы лучше не устанавливать точное число колонок, а дать указание браузеру, каким должен быть размер каждой колонки, используя свойство column-width, как показано далее:

```
.Content {
   text-align:justify;
   -moz-column-width: 10em;
   -webkit-column-width: 10em;
   column-width: 10em;
}
```

Таким образом, браузер может создать такое количество колонок, которое необходимо для заполнения имеющегося пространства.

Размер колонок можно указывать в пикселах, но лучше использовать для этого em-единицы, т. к. em-единицы адаптируются к текущему размеру шрифта. Таким образом, если посетитель веб-страницы увеличит размер шрифта в браузере, ширина колонки возрастет пропорционально размеру шрифта. В пикселах одна em-единица приблизительно вдвое больше размера шрифта. Например, для 12-пиксельного шрифта одна em-единица будет равна 24 пикселам.

С помощью CSS3-свойства column-gap вы можете установить величину отступа между столбцами текста.

```
div {
column-count:4;
column-gap:50px;
}
```

С помощью свойства column-rule можно задать ширину, цвет и стиль оформления пространства между столбцами.

```
div {
column-count:4;
column-rule:2px dotted #7F0055;
}
```

Добавление теней к тексту

С помощью нового CSS3-свойства text-shadow вы можете добавлять к тексту элементов тени (к тексту одного элемента может быть добавлено одновременно несколько теней).

При задании тени для текста необходимо указать величину смещения тени от текста по горизонтали и вертикали (может быть отрицательной), а также радиус размытия и цвет тени.

```
#shadow1 {
text-shadow:3px 2px #FFAE00;
}
#shadow2 {
text-shadow:1px 1px 10px #FFAE00;
}
#shadow3 {
text-shadow:2px 2px 2px #FFAE00, 2px 2px 15px #1435AD;
}
```

Свойство text-overflow

В CSS3 было добавлено новое свойство text-overflow, которое позволяет указать, что должно случиться с текстом, вышедшим за пределы границ элемента.

```
#wrap1 {
text-overflow:ellipsis;
overflow:hidden;
}
#wrap2 {
text-overflow:clip;
overflow:hidden;
}
```

Свойство word-wrap

С помощью нового CSS3-свойства word-wrap вы можете указать, что длинные слова, выходящие за пределы границ элемента, должны разделяться и переноситься на новую строку.

```
#wrap2 {
word-wrap:break-word;
}
```

В таблице ниже приведены основные свойства CSS для текста.

Свойство	Описание	Введено в
@font-face	Позволяет подключить к веб-страницам произвольные шрифты	CSS3
font	Позволяет установить все свойства шрифта (font-family, font-size, font-style, font-variant, font-weight) за одно определение	CSS1
font-family	Позволяет установить шрифт текста	CSS1
font-size	Позволяет установить размер текста	CSS1
color	Позволяет установить цвет текста	CSS1
text-shadow	Позволяет привязать тень (или несколько теней) к тексту элемента	CSS3
text-decoration	Позволяет оформить текст элемента	CSS1

text-align	Позволяет определить горизонтальное выравнивание текста	CSS1
letter-spacing	Позволяет определить расстояние между символами текста	CSS1
word-spacing	Позволяет определить расстояние между словами текста	CSS1
line-height	Позволяет установить высоту строк	CSS1
font-style	Позволяет установить стиль шрифта элемента	CSS1
font-variant	Позволяет отобразить текст элемента капителью	CSS1
font-weight	Позволяет установить толщину шрифта	CSS1
text-overflow	Позволяет указать, как должен отображаться текст, вышедший за пределы границ элемента	CSS3
vertical-align	Позволяет установить вертикальное выравнивание текста	CSS1
text-transform	Позволяет управлять регистром символов в тексте	CSS1
text-indent	Позволяет установить величину отступа первого символа текста	CSS1
text-justify	Позволяет установить алгоритм выравнивания для свойства "text-align:justify"	CSS3
word-wrap	Позволяет указать, должны ли длинные слова, выходящие за пределы родительского элемента, разбиваться и переноситься на новую строку	CSS3
white-space	Позволяет установить, как должны оформляться пробелы в тексте элемента	CSS1
quotes	Позволяет установить, как должны оформляться кавычки, вставленные тэгом <q>	CSS1
direction	Позволяет установить направление письма текста	CSS1

Практическое задание

1. Доделать интернет-магазин.

Дополнительные материалы

1. @font-face – https://webref.ru/css/font-face.
2. Сервис для подбора шрифтов, конвертации, определения – https://www.fontsquirrel.com/.
3. Подключение нестандартных шрифтов – @font-face! https://habrahabr.ru/post/113136/.
4. Производительность веб-шрифтов – https://habrahabr.ru/post/159907/.

Используемая литература

Для подготовки данного методического пособия были использованы следующие ресурсы:

1. http://www.wisdomweb.ru/.
2. https://html5book.ru/.
3. Гоше Х. HTML5. Для профессионалов. – СПб.: Питер, 2013. – 496 с.: ил.
4. Хоган Б. HTML5 и CSS3. Веб-разработка по стандартам нового поколения. – СПб.: Питер, 2012. – 272 с.
5. Макфарланд Д. Большая книга CSS3. – СПб.: Питер, 2016. – 608 с.