Htlml5test.com

Dom Inspector

# Introduction

## New Semantic

Canvas demos

http://padolsey.net/p/Sonic/repo/demo/demo.html

### Doctype

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<title>Директива DOCTYPE</title>

</head>

<body>

<p>

Директива DOCTYPE отвечает за определение типа документа, то есть

сообщает браузерам и валидаторам, какая версия HTML использовалась

при написании страницы.

В HTML5 версия в DOCTYPE не указывается, что с точки зрения синтак­сиса обеспечивает для языка обратную совместимость.

Допустим, у вас есть сайт, соответствующий стандарту HTML 4.0, но вы

хотите перевести его на HTML5. Для этого достаточно изменить DOCTYPE.

Все браузеры поддерживают сокращенное объявление DOCTYPE и ото­бражают страницу в строгом соответствии стандартам.

</p>

### Charset

<meta charset="utf-8" />

<title>Указание кодировки</title>

</head>

<body>

<p>

Кодировка сообщает браузерам и валидаторам, какой набор символов исполь­зовать при отображении веб-страницы. Если в HTML-документе кодировка

не объявлена, то браузер сначала попытается определить ее через заголовок ответа HTTP-сервера (заголовок Content-type).

Кодировка, заданная в заголовке ответа, предпочтительнее, нежели ука­занная в документе, поэтому заголовки — основной способ предоставления

такой информации.

</p>

</body>

### Script and Link

<link rel="stylesheet" href="styles.css" />

<script src="scripts.js"></script>

</head>

<body>

<p>

В предыдущих версиях HTML как в CSS-ссылках

link, так и в элементе script для определения языка приходилось задавать

атрибут type. Сейчас, даже если вы забыли объявить type, в большинстве

браузеров уже предусмотрено правильное значение.

</p>

### Mark

<head>

<title>Элемент mark</title>

</head>

<body>

<p>Вы искали слово <mark>HTML 5</mark>. <mark>HTML 5</mark> - это язык для структурирования и представления содержимого для всемирной паутины.</p>

</body>

### Abbr

<!-- abbr - указывает, что последовательность символов является аббревиатурой -->

<!DOCTYPE html>

<html>

<head>

<title>Элемент abbr</title>

</head>

<body>

<abbr title="язык.">HTML 5</abbr>

</body>

### Link

<head>

<title>Использование ссылок</title>

</head>

<body>

<a href="http://edu.cbsystematics.com">

<h1>HTML5 для работы &amp; отдыха</h1>

<img src="img/logo.png" alt="HTML5" />

</a>

### Details

<details open>

<summary>Предстоящие обсуждения</summary>

<p>В новому году у нас появится огромное количество статей</p>

<ul>

<li>Понимание структуры HTML документа</li>

<li>Когда использовать <code>hgroup</code></li>

<li>Машинная семантика микроданных</li>

</ul>

</details>

### Order list

<!--

Атрибут start устанавливает номер, с которого будет начинаться список

«Упорядочивание» через start работает во всех браузерах.

-->

<ol start="3">

<li>HTML и CSS</li>

<li>Javascript</li>

<li>jQuery</li>

</ol>

<!--

Атрибут reversed применяется для управления нумерацией упорядоченных

списков и отображает список в обратном порядке

-->

<ol reversed>

<!-- Не работает в IE-->

<li>HTML и CSS</li>

<li>Javascript</li>

<li>jQuery</li>

</ol>

<ul reversed>

<!-- Не работает в IE-->

<li>HTML и CSS</li>

<li>Javascript</li>

<li>jQuery</li>

</ul>

<!-- Все современные браузеры поддерживают для li атрибут value.-->

<ol>

Результаты голосования на самую популярную web - технологию. Первое место разделили Javascript и HTML и CSS

<li value="1">HTML и CSS</li>

<li value="1">Javascript</li>

<li value="2">jQuery</li>

</ol>

### Contenteditable

<!--Атрибут contenteditable обеспечивает возможность редактировать стра­ницу непосредственно через браузер-->

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<title>Атрибут contenteditable</title>

</head>

<body>

<!--Атрибут contenteditable обеспечивает возможность редактировать стра­ницу непосредственно через браузер-->

<div contenteditable>

<h2>He злоупотребляйте <code>section</code>!</h2>

<p>

Основной структурный элемент HTML5 <code>section</code> зачастую

используется как обычный контейнер для стиля, хотя для этого

больше подходит <code>div</code>.

</p>

</div>

### Modernizer

<!--

Modernizr http://modernizr.com/— это библиотека JavaScript. Она определяет поддержку различных HTML5

и СSS3 функций в браузере и позволяет создавать запасные варианты, если в настоящее время необходимая функция не поддержива­ется.

-->

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<title>Modernizer 2.5.3 - Проверка технологий HTML5</title>

<script src="Modernizr-2.5.3.js" type="text/javascript"></script>

</head>

<body>

<script type="text/javascript">

if (Modernizr.canvas) {

document.write("Поддержка элемента canvas <br />");

}

if (Modernizr.video.h264) {

document.write("Поддержка формата h264 <br />");

}

if (Modernizr.geolocation) {

document.write("Поддержка Geolocation API<br />");

}

if (Modernizr.inputtypes.color) {

document.write("Поддержка поля ввода типа color<br />");

}

</script>

### Modernizer load

--

С помощью Modernizr можно проверить поддержку необходимой функции и определить какой JavaScript сценарии необходим клиенту.

Тот, который основан на возможностях браузера и использует новые возможности HTML5 и CSS3 или тот скрипт, который программно имитирует

возможноси HTML5 и CSS3

-->

<!DOCTYPE html>

<html>

<head>

<title>Modernizer load test</title>

<script src="Modernizr-2.5.3.js" type="text/javascript"></script>

<script type="text/javascript">

Modernizr.load({

test: Modernizr.input.placeholder,

yep: 'Yep.js',

nope: 'No.js'

});

</script>

</head>

<body>

Modernizr.load() test page

</body>

### Time

<!--

Time - помечает текст внутри тега <time> как дата, время или оба значения.

-->

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>time</title>

<style>

time {

background: #f0f0f0;

}

</style>

</head>

<body>

<article>

<p>Опубликовано: <time datetime="2012-12-23T08:23:11+07:00">сегодня</time></p>

<p><time>1957-10-04</time> запущен первый искусственный спутник Земли.</p>

<p><time>1960-08-19</time> первый полет собак в космос.</p>

<p><time>1961-04-12</time> первый полет человека в космос.</p>

<p><time>1963-06-16</time> первый полет женщины-космонавта.</p>

<p><time>1969-07-21</time> высадка человека на Луну.</p>

</article>

### Hidden

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<title>Атрибут hidden</title>

</head>

<body>

<div hidden="hidden">Текст, который не отобразится</div>

<div>Hello World</div>

</body>

</html>

# HTML5 верстка

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<title>HTML5</title>

<link href="html5.css" rel="stylesheet" />

</head>

<body>

<header>

<h1>Header</h1>

<h2>Subtitle</h2>

<!--Элемент будет поврнут на 45 градусов с помощью CSS стилей-->

<h4>Learn HTML5!</h4>

</header>

<div>

<nav>

<h3>Nav</h3>

<a href="#">Link 1</a> <a href="#">Link 2</a> <a href="#">Link 3</a> <a href="#">Link 4</a> <a href="#">Link 5</a>

</nav>

<section>

<article>

<header>

<h1>Article Header 1</h1>

</header>

<p>Для того чтобы определить, какие из новых элементов необходимы на вашей странице, проанализируйте контент и структуру документа.</p>

<ul>

<li>header &mdash; используется для хранения заголовков страницы и/или разде&shy;лов section. Может содержать дополнительную информацию, например логотипы и навигацию.</li>

<li>footer &mdash; содержит информацию о странице и/или разделе section, на&shy;пример информацию об авторе, ссылки на использованные источники а также сведения об авторских правах.</li>

<li>nav &mdash; включает основные навигационные ссылки на страницы и зачастую содержится в header.</li>

<li>aside &mdash; хранит информацию, связанную с окружающим контентом, но существующую независимо, например на боковой панели.</li>

<li>section &mdash; основной из новых структурных элементов, контент которых может быть сгруппирован тематически или связан между собой.</li>

<li>article &mdash; определяет автономный контент, который может использоваться независимо от страницы в целом, подобно записи в блоге.</li>

</ul>

</article>

<article>

<header>

<h1>Article Header 2</h1>

</header>

<p>

Знаете ли вы, когда следует применять эти новые элементы? Опять же,

</p>

</article>

</section>

<aside>

<h3>Aside</h3>

<p>

blah, blah, blah...

</p>

<p>

blah, blah, blah...

</p>

</aside>

<footer>

Footer

</footer>

</div>

</body>

</html>

body

{

background-color:#cccccc;

font-family: Arial, Sans-Serif;

margin:0px auto;

max-width:900px;

border:solid 4px white;

}

header

{

display:block;

background-color:Orange;

color:White;

text-align:center;

}

h3

{

color:Orange;

text-align:center;

}

h4

{

color:Orange;

background-color:White;

-webkit-transform: rotate(-45deg);

-moz-transform:rotate(-45deg);

text-align:left;

padding-left:20px;

position:relative;

left:-140px;

top:-350px;

}

nav

{

width:25%;

float:left;

}

nav a:link, nav a:visited

{

display:block;

font-weight:bold;

padding:10px;

text-decoration:none;

margin:5px;

}

nav a:hover

{

color:White;

background-color:Orange;

}

section

{

display:block;

width:50%;

float:left;

}

article header

{

/\*-webkit - для Chrome и Safari\*/

-webkit-border-radius:10px;

/\*-moz- для Mozilla\*/

-moz-border-radius: 10px;

border-radius:10px;

padding:5px;

}

article p

{

margin:10px;

}

aside

{

float:left;

width:25%;

}

aside p

{

margin:15px;

}

footer

{

clear:both;

color:white;

background-color:Orange;

text-align:center;

padding:15px;

}

# Forms video and audio

## Forms

### Form elements

<form>

<!--Кнопка, основное событие onclick-->

<input type="button" name="button1" value="Button1" />

<br />

<!--Флажок, основное событие onclick-->

<input type="checkbox" checked="checked" name="checbox1" />

Check Box 1

<br />

<!--Поле для ввода файла загружаймого на сервер, основное событие onchange-->

<input type="file" name="file1" value="" />

<br />

<!--Данные сохраняемые вместе с формой но невидимые пользователю.-->

<input type="hidden" name="hidden1" value="" />

<br />

<!--Поле для ввода пароля-->

<input type="password" name="password1" value=" " />

<br />

<!--Переключатель-->

Radio 1<input type="radio" checked="checked" name="group1" />

Radio 2<input type="radio" name="group1" />

<br />

<!--Кнопка очищающая значения формы.-->

<input type="reset" name="reset1" value="Reset" />

<br />

<br />

<!--Список в котором может быть выбран один элемент-->

<select name="select1">

<option value="value1">text 1</option>

<option value="value2">text 2</option>

<option value="value3">text 3</option>

</select>

<br />

<br />

<!--Список со множественным выбором-->

<select multiple="multiple" name="multipleSelect1">

<option value="value">text</option>

<option value="value">text</option>

<option value="value">text</option>

</select>

<br />

<br />

<!--Кнопка для передачи данных формы-->

<input type="submit" name="submit1" value="Submit" />

<br />

<!--Однострочное поле ввода-->

<input type="text" name="text1" value=" " />

<br />

<!--Многострочное поле ввода-->

<textarea name="textArea1" rows="4" cols="10"></textarea>

</form>

### Label

<form>

<!--

Тег <label> устанавливает связь между определенной меткой, в качестве которой обычно выступает текст,

и элементом формы (<input>, <select>, <textarea>).Такая связь необходима, чтобы изменять значения элементов формы при нажатии курсором мыши на текст.

Кроме того, с помощью <label> можно устанавливать горячие клавиши на клавиатуре и переходить на активный элемент подобно ссылкам.

Существует два способа связывания объекта и метки. Первый заключается в использовании идентификатора id внутри элемента формы и указании его имени

в качестве атрибута for тега <label>.

При втором способе элемент формы помещается внутрь контейнера <label>.

-->

<p>

<label for="search" accesskey="S">

Найти

</label>

<input id="search" type="search" name="query" />

</p>

<button type="submit">Отправить </button>

</form>

### Contact

<form>

<fieldset>

<legend>Контактная информация</legend>

<p>

<!--

email - браузер проводит валидацию корректности введенного email адреса.

При работе в мобильном браузере будет отображаться специальная экранная клавиатура с символом @ и .com

-->

<label>

E-mail

<input type="email" name="email" title="Формат данных не правельный. Только в Chrome" />

</label>

</p>

<p>

<!--

url - браузер проводит валидацию корректности введенного адреса.

-->

<label>

Сайт

<input type="url" name="website" title="Абсолютный URL. Обязательно с протоколом HTTP. Пример - http://html5readiness.com/" /></label>

</p>

<p>

<!--

tel - мобильный браузер будет отображать экранную клавиатуру с цифрами

-->

<label>

Телефон

<input type="tel" name="phone" /></label>

</p>

</fieldset>

<br>

<p>

<input type="submit" value="Отправить" />

</p>

</form>

### Numbers

<form>

<p>

<label>

Количество

<input type="number" name="quantity" /></label>

</p>

<p>

<input type="submit" value="Отправить" />

</p>

</form>

<form>

<p>

<label>

Количество

<input type="number" name="quantity" min="2" max="20" step="2" /></label>

</p>

<p>

<input type="submit" value="Отправить" />

</p>

</form>

### Range

<form>

<p>

<label>

Громкость

<input type="range"

name="volume"

min="2"

max="10"

step="1"

value="5" /></label>

</p>

</form>

### ColorPicker

<form>

<p>

<label>

Цвет фона

<input type="color" name="bg" /></label>

</p>

<p>

<label>

Цвет шрифта

<input type="color" name="fg" /></label>

</p>

<p>

<input type="submit" value="Отправить" />

</p>

</form>

### Select list

<form>

<p>

<label>

Сумма пожертвования

<!--Элемент, который связанный с источником данных через атрибут list-->

<input type="text" name="donation" list="donations" /></label>

</p>

<!--Источник данных-->

<datalist id="donations">

<option value="10.00">10.00 $</option>

<option value="25.00">25.00 $</option>

<option value="50.00">50.00 $</option>

</datalist>

<p>

<input type="submit" value="Отправить" />

</p>

</form>

### Required field

<form>

<fieldset>

<legend>Aвторизация</legend>

<p>

<label>

Имя пользователя

<!-- Атрибут required указывает, что поле обязательное для заполнения -->

<input type="text" name="username" required="required" /></label>

</p>

<p>

<label>

Пароль

<input type="password" name="pwd" required="required" /></label>

</p>

</fieldset>

<p>

<input type="submit" value="Отправить" />

</p>

</form>

### Autofocus

<form>

<p>

<label>Первое поле<input type="search" name="query" /></label>

</p>

<p>

<label>

Второе поле с афтофокусом

<!--

Атрибут autofocus определяет элемент управления, который получит фокус, когда загрузится страница

-->

<input type="search" name="query" autofocus="autofocus" /></label>

</p>

<p>

<input type="submit" value="Отправить" />

</p>

</form>

### Placeholder

<form>

<!--

Атрибут placeholder используется для того что бы определить подсказку отображаемую в поле ввода до того как пользователь начнет вводить текст

-->

<fieldset>

<legend>Контактная инфомациж</legend>

<p>

<label>

Адpec электронной почты

<input type="email" name="email" placeholder="user@domain.com" /></label>

</p>

<p>

<label>

Сайт<input type="url" name="website" placeholder="http://www.domain.com/" />

</label>

</p>

<p>

<label>

Телефон

<input type="tel" name="phone" placeholder="123-12-34" />

</label>

</p>

</fieldset>

<p>

<input type="submit" value="Отправить" />

</p>

### Autocomplete

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<title>Атрибут autocomplete</title>

</head>

<body>

<form>

<fieldset>

<legend>Авторизация </legend>

<p>

<label>

Имя пользователя

<input type="text" name="username" autocomplete="off" /></label>

</p>

<p>

<label>

Пароль

<!-- autocomplete="off" - атрибут используется для того, чтобы браузер не сохранял данные введенные в поле ввода, для обеспечения безопасности для критических данных. -->

<input type="password" name="pwd" autocomplete="off" /></label>

</p>

</fieldset>

<p>

<input type="submit" value="Отправить" />

</p>

</form>

<a href="http://www.miketaylr.com/code/input-type-attr.html">Проверка работа атрибутов формы</a>

<p>Перейдите по ссылки, после чего нажмите на кнопку "Back". Обратите внимание что браузер не запомнил значение в поле ввода пароля</p>

</body>

</html>

### Validation pattern

<form>

<p>

<label>

Телефон

<!--

Атрибут pattern позволяет указать регулярное выражения с помощью, которого будт производиться валидация данных

значение введенное в атрибут title будет использоваться для подсказки если пользователь допустит ошибку.

-->

<input type="tel" name="phone" pattern="\+38 \(\d{3}\) \d{3}-\d{2}-\d{2}"

title="+38 (063) 223-23-23" />

</label>

</p>

<input type="submit" value="Отправить" />

</form>

### Modernizr

<!--Код формы, использующей атрибут autofocus:-->

<form>

<p>

<label>

Search

<input type="search" name="query" id="query" autofocus="autofocus" /></label>

</p>

<p>

<input type="submit" value="Отправить" />

</p>

</form>

<script>

// Если браузер не поддерживает атрибут autofocus вызываем метод focus на необходимом элементе.

if (!Modernizr.input.autofocus) {

document.getElementById("query").focus();

}

</script>

### Modernizr2

<form id="search">

<p>

Найти

<input id="query1" type="text" value="" placeholder="Введите запрос" />

<input id="query2" type="text" value="" placeholder="Введите запрос" />

</p>

<input type="submit" value="Отправить" />

</form>

<script>

// Если браузер не поддерживает атрибут placeholder

if (!Modernizr.input.placeholder) {

$("input[placeholder]").focus(function () {

if ($(this).val() == $(this).attr('placeholder')) {

$(this).val("");

}

});

$("input[placeholder]").blur(function () {

if ($(this).val() == "") {

$(this).val($(this).attr('placeholder'));

}

});

}

</script>

### Modernizr3

<head>

<title>Использование Modernizr</title>

<script src="modernizr.custom.52481.js"></script>

<script src="http://code.jquery.com/jquery-1.9.0.js"></script>

<script src="http://code.jquery.com/ui/1.10.0/jquery-ui.js"></script>

<link href="http://code.jquery.com/ui/1.10.0/themes/base/jquery-ui.css" rel="stylesheet" />

</head>

<body>

<form>

<p>

<label>

Дата рождения

<input type="date" name="dob" id="dob" /></label>

</p>

<p>

<input type="submit" value="Отправить" />

</p>

</form>

<script>

if (!Modernizr.inputtypes.date) {

$("#dob").datepicker();

}

</script>

</body>

## Video and Audio

### Audio control play

<!--

controls - добавить элементы управления воспроизведением

autoplay - начать воспроизведение после загрузки страницы

-->

<audio src="audio/woodkid\_-\_iron.mp3" controls autoplay>

<a href="http://htmlbook.ru/html/audio">Браузер не поддерживает воспроизведение звуковых медиафайлов</a>

</audio>

Популярные мультимедиа контейнеры:

<ul>

<li><strong>ААС </strong>&mdash; формат сжатия с потерями, дает более высокое качество чем MP3</li>

<li><strong>MP3</strong> &mdash; лицензируемый формат файла для хранения аудиоинформации</li>

<li><strong>OGG</strong> &mdash; открытый стандарт мультимедиа контейнера </li>

<li><strong>WAV</strong> &mdash; проприетарный аудиоформат без какого-либо сжатия;</li>

<li><strong>WEBM</strong> &mdash; открытый бесплатный медиаформат Google, основанный на аудиокодеке Vorbis</li>

</ul>

### Check if browser can play

<!--

autoplay - сообщает браузеру о том, что необходимо начать воспроизведение аудио, как только будет загружена страница

loop - повторно запустить аудио после завершения

canplaytype(тип) - возвращает строку, указывающую, может ли браузер воспроизводить данный тип медиафайла

currentTime - определяет текущую позицию воспроизведения в секундах

duration - содержит длину аудиофайла в секундах

play() - функция, начинает воспроизведение с текущей позиции

pause() - функция, приостанавливает воспроизведение аудио

-->

<audio id="audio" controls autoplay src="audio/Vivaldi.ogg">

Браузер не поддерживает воспроизведение звуковых медиафайлов

</audio>

<script type="text/javascript">

if (!Modernizr.audio) {

alert("Браузер не поддерживает воспроизведение звуковых файлов");

}

else {

var audio = document.getElementById("audio");

var res = audio.canPlayType('audio/ogg; codecs="vorbis"');

if (res) alert(res + " can play ogg vorbis");

else alert("no support for ogg vorbis");

}

</script>

### Play if you can

<body>

<audio controls>

<!-- Указание файлов, которые будут использоваться для воспроизведения браузером.

Браузер начнет воспроизводить первый файл, формат которого поддерживает.

-->

<source src="audio/Vivaldi.mp3">

<source src="audio/Vivaldi.ogg">

<a href="http://html5accessibility.com/">Браузер не поддерживает воспроизведение звуковых медиафайлов</a>

</audio>

</body>

### Audio with JS

<input id="playButton" type="button" value="play" />

<audio id="audioElement">

<source src="audio/Vivaldi.mp3">

<source src="audio/Vivaldi.ogg">

</audio>

<script type="text/javascript">

document.getElementById("playButton").onclick = playSound;

function playSound() {

var music = document.getElementById("audioElement");

var button = document.getElementById("playButton");

// если аудиоплеер на паузе

if (music.paused) {

music.play(); // начать воспроизведение

button.value = "stop";

}

else { // иначе

music.pause(); // остановить воспроизведение

button.value = "play";

}

}

</script>

### Audio PlayAt

<audio>

<source src="audio/Vivaldi.mp3" />

<source src="audio/Vivaldi.ogg" />

</audio>

<button title="Bocпроизвести с 30 секунды" onclick="playAt(30);">30 секунда</button>

<script>

function playAt(seconds) {

var audio = document.getElementsByTagName("audio")[0];

audio.currentTime = seconds;

audio.play();

}

</script>

<button title="Остановить воспроизведение" onclick="stopAudio();">

STOP</button>

<script>

function stopAudio() {

var audio = document.getElementsByTagName("audio")[0];

audio.currentTime = 0;

audio.pause();

}

</script>

### Video

<video controls loop poster="poster.png">

<source src="video/vsvideo.mp4" />

<source src="video/vsvideo.ogg" />

<source src="video/vsvideo.webm" />

</video>

<ul>

<li><strong>MP4</strong> &mdash; мультимедиа-контейнер для проприетарного кодека Н.264.</li>

<li><strong>OGV</strong> &mdash; свободный мультимедиа-контейнер с открытым исходным кодом для кодека Theora. Результаты сжатия имеют более низкое качество, чем Н.264.</li>

<li><strong>WEBM </strong>&mdash; еще один мультимедиа-контейнер с открытым исходным кодом, использующий новый бесплатный кодек VP8 от Google.</li>

</ul>

### Video with JS

<video id="video" style="border: 2px solid gray; cursor: pointer;">

<source src="video/vsvideo.mp4" />

<source src="video/vsvideo.ogg" />

<source src="video/vsvideo.webm" />

</video>

<script type="text/javascript">

window.onload = function () {

var videoElem = document.getElementById("video");

// устанавливаем размер элемента на странце как половина высоты и ширины видеозаписи

videoElem.width = videoElem.videoWidth / 2;

videoElem.height = videoElem.videoHeight / 2;

// запускаем видео при наведении курсора

videoElem.onmouseover = function () {

this.play();

}

// останавливаем видео при выходе курсора

videoElem.onmouseout = function () {

this.pause();

}

}

</script>

### Video preview

<button title="Создать превью" onclick="generatePreview();">Создать превью</button>

<br />

<video id="origVideo" width="640" height="360" style="border: 1px solid grey">

<source src="video/vsvideo.mp4" />

<source src="video/vsvideo.ogg" />

<source src="video/vsvideo.webm" />

</video>

<br />

<canvas id="previewMed" style="border: 1px solid grey"></canvas>

<canvas id="previewSm" style="border: 1px solid grey"></canvas>

<script>

function generatePreview() {

var video = document.getElementById('origVideo');

// получение контекста для работы с первым canvas

var canvas1 = document.getElementById('previewMed');

var context1 = canvas1.getContext('2d');

// получение контекста для работы со вторым canvas

var canvas2 = document.getElementById('previewSm');

var context2 = canvas2.getContext('2d');

// задаем размеры для обеих canvas

canvas1.width = 320;

canvas1.height = 180;

canvas2.width = 160;

canvas2.height = 90;

// добавляем обработчик на событие начала воспроизведения видео

video.addEventListener('play', function () {

drawVideo(this, context1, context2);

}, false);

// заускаем видео

video.play();

}

function drawVideo(video, canvas1, canvas2) {

// если видео на паузе или завершено - не выполняем данную функцию

if (video.paused || video.ended) {

return false;

}

// рисуем на пером canvas видео размером 320 \* 180

canvas1.drawImage(video, 0, 0, 320, 180);

// рисуем на втором canvas видео размером 160 \* 90

canvas2.drawImage(video, 0, 0, 160, 90);

// планируем запуск функции drawVideo через 25 миллисекунд передавая текущие значения параметров функции

setTimeout(drawVideo, 25, video, canvas1, canvas2);

}

</script>

### Video duration

<video id="video" style="border: 2px solid gray;" height="200" width="200">

<source src="video/vsvideo.mp4" />

<source src="video/vsvideo.ogg" />

<source src="video/vsvideo.webm" />

</video>

<br />

<button id="part1Button">

1

</button>

<button id="part2Button">

2

</button>

<button id="part3Button">

3

</button>

<button id="playButton">play</button>

<button id="pauseButton">pause</button>

<script type="text/javascript">

window.onload = function () {

var v = document.getElementById("video");

var part = v.duration / 3;

document.getElementById("part1Button").onclick = function () {

v.currentTime = 0;

};

document.getElementById("part2Button").onclick = function () {

v.currentTime = part;

};

document.getElementById("part3Button").onclick = function () {

v.currentTime = part \* 2;

};

document.getElementById("playButton").onclick = function () {

v.play();

};

document.getElementById("pauseButton").onclick = function () {

v.pause();

};

}

</script>

### MicroData and Geolocation

## MicroData

### ItemScope

<!--

MicroData - способ семантически размечать сведенья о разнообразных сущностях на веб страницах.

Атрибут itemscope применяется для определения области действия микро­данных — набора пар «имя/значение».

Значения атрибута itemprop устанав­ливают имена свойств и связанную с ними информацию, в данном случае содержимое тегов span.

http://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D0%BA%D1%80%D0%BE%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5 - микроданные на Википедии

-->

<p itemscope>

<span itemprop="inovator">Тим Бернерс-Ли</span>

<span itemprop="inovation">Всемирную паутину</span>

</p>

<!--

http://www.google.com/webmasters/tools/richsnippets - инструмент проверки структуры данных

Перейдите по указанной выше ссылке и скопируйте HTML код текущей страницы в поле ввода инструменат проверки.

-->

### Item Scope Example

<!-- schema.org — Официальный сайт словаря семантической разметки -->

<!-- itemtype="http://schema.org/Person" - указывает что section содержит информацию о человеке -->

<section itemscope itemtype="http://schema.org/Person">

<!-- itemprop="name" - указывает что, это не просто h1, а элемент в котором находиться имя человека-->

<h1 itemprop="name">Tим Бернерс-Ли</h1>

<img itemprop="image" src="http://www.w3.org/Press/Stock/Berners-Lee/2001-europaeum-eighth.jpg">

<!-- itemtype="http://schema.org/Organization" - Элемент содержит данные о организации -->

<p itemprop="affiliation" itemscope itemtype="http://schema.org/Organization">

<span itemprop="name">Консорциум Всемирной паутины</span>,

<span itemprop="url">http://www.w3.org</span>

</p>

<!-- itemtype="http://schema.org/PostalAddress" - элемент содержит данные о почтовом адресе -->

<p itemprop="address" itemscope itemtype="http://schema.org/PostalAddress">

<span itemprop="addressLocality">Кембридж</span>,

<span itemprop="addressRegion">Maccaчусетс</span>

</p>

<a itemprop="url" href="http://www.w3.org/People/Berners- Lee/">Сайт W3C</a>

</section>

### Data Attribute

<ul>

<!-- data-\* универсальный атрибут, позволяет добавить пользовательсие данные к любым DOM элементам-->

<li data-year="1996" data-соlог="белый" data-engine="VR6">Cabrio</li>

<li data-уеаг="1993" data-color="фиолетовый" data-engine="VR6">Corrado</li>

<li data-year="2008" data-color="красный" data-engine="2.0T">Eos</li>

<li data-year="2003" data-color="синий" data-engine="W8">Passat</li>

</ul>

### Read Attribute

<script>

window.onload = function () {

var cars = document.getElementsByTagName("li");

var output = "Цвета автомобилей: <br />";

for (var i = 0; i < cars.length; i++) {

// dataset - получение доступа ко всем атрибутам data-\*

// color - свойство определяет обращение к атрибуту data-color

output += cars[i].dataset.color + "<br />";

// Если атрибут содержит несколько дефисов после первого дефиса, например, data-year-production,

// то для получения доступа к такому атрибуту из JavaScript кода необходимо использовать имя yearProduction -

// удалить дефис, а следующий после него символ перевести в верхний регистр.

output += cars[i].dataset.yearProduction + "<br />";

}

document.getElementsByTagName("p")[0].innerHTML = output;

}

</script>

</head>

<body>

<ul>

<li data-year-production="1996" data-color="белый" data-engine="VR6">Cabrio</li>

<li data-year-production="1993" data-color="фиолетовый" data-engine="VR6">Corrado</li>

<li data-year-production="2008" data-color="красный" data-engine="2.0T">Eos</li>

<li data-year-production="2003" data-color="синий" data-engine="W8">Passat</li>

</ul>

<p></p>

<a href="http://www.w3.org/html/wg/drafts/html/master/dom.html#dom-dataset">Ссылка на документацию</a>

### Write Attribute

window.onload = function () {

var cars = document.getElementsByTagName("li");

for (var i = 0; i < cars.length; i++) {

if (cars[i].dataset) {

cars[i].dataset.color = "желтый";

cars[i].dataset.rating = "удивительный";

} else {

cars[i].setAttribute("data-color", "желтый");

cars[i].setAttribute("data-rating", "удивительный");

}

}

var output = "Цвета автомобилей: <br />";

for (var i = 0; i < cars.length; i++) {

if (cars[i].dataset) {

output += cars[i].dataset.color + "<br />";

}

else {

output += cars[i].getAttribute("data-color") + "<br />";

}

}

document.getElementsByTagName("p")[0].innerHTML = output;

}

</script>

</head>

<body>

<ul>

<li data-year="1996" data-color="белый" data-engine="VR6">Cabrio</li>

<li data-уеаг="1993" data-color="фиолетовый" data-engine="VR6">Corrado</li>

<li data-year="2008" data-color="красный" data-engine="2.0T">Eos</li>

<li data-year="2003" data-color="синий" data-engine="W8">Passat</li>

</ul>

<p></p>

<a href="https://addons.mozilla.org/ru/firefox/addon/dom-inspector-6622/">DOM Инспектор (расширение для Firefox)</a>

### Drag and Drop

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<title>Drag & Drop</title>

<style>

#source {

background-color: yellow;

width: 100px;

height: 100px;

cursor: move;

color: black;

}

#target {

background-color: blue;

width: 150px;

height: 150px;

padding: 5px;

margin-bottom: 5px;

color: white;

}

</style>

<script>

// dragstart - вызывается в самом начале переноса перетаскиваемого элемента.

// dragend - вызывается в конце события перетаскивания - как успешного, так и отмененного.

// dragenter - происходит в момент когда перетаскиваемый объект попадает в область целевого элемента.

// dragleave - происходит когда перетаскиваемый элемент покидает область целевого элемента.

// dragover - происходит когда перетаскиваемый элемент находиться над целевым элементом.

// drop - вызывается, когда событие перетаскивания завершается отпусканием элемента над целевым элементом.

window.onload = function () {

var source = document.getElementById("source");

// начало операции drag

source.addEventListener('dragstart', function (evt) {

this.style.border = "3px dotted #000"; // меняем стиль в начале операции drag & drop

// Свойство effectAllowed управляет визуальным эффектом (чаще всего это вид указателя мыши), который браузер создает в ответ

// на тип происходящего события перетаскивания (перемещение, копирование и т. п.).

// http://msdn.microsoft.com/en-us/library/ie/ms533743%28v=vs.85%29.aspx

evt.dataTransfer.effectAllowed = "move";

// Метод setData(...) сообщает механизму перетаскивания в браузере, какие данные из перетаскиваемого объекта должен «поймать»

// целевой элемент, также называемый зоной приема. Здесь мы указываем, что передаваемые данные это id элемента

evt.dataTransfer.setData("Text", this.id);

}, false);

// конец операции drag

source.addEventListener("dragend", function (evt) {

this.style.border = ""; // удаляем стили добавленные в начале операции drag & drop

}, false);

var target = document.getElementById("target");

// перетаскиваемый объект попадает в область целевого элемента

target.addEventListener("dragenter", function (evt) {

this.style.border = "3px solid red";

}, false);

// перетаскиваемый элемент покидает область целевого элемента

target.addEventListener("dragleave", function (evt) {

this.style.border = "";

}, false);

target.addEventListener("dragover", function (evt) {

// отменяем стандартное обработчик события dragover.

// реализация данного обработчика по умолчанию не позволяет перетаскивать данные на целевой элемент, так как большая

// часть веб страницы не может принимать данные.

// Для того что бы элемент мог принять перетаскиваемые данные необходимо отменить стандартный обработчик.

if (evt.preventDefault) evt.preventDefault();

return false;

}, false);

// перетаскиваемый элемент отпущен над целевым элементом

target.addEventListener("drop", function (evt) {

// прекращаем дальнейшее распространение события по дереву DOM и отменяем возможный стандартный обработчик установленный браузером.

if (evt.preventDefault) evt.preventDefault();

if (evt.stopPropagation) evt.stopPropagation();

this.style.border = "";

var id = evt.dataTransfer.getData("Text"); // получаем информации которая передавалась через операцию drag & drop

var elem = document.getElementById(id);

// добавляем элемент в целевой элемент. Так как в DOM не может быть два идентичных элемента - элемент удаляется со своей старой позиции.

this.appendChild(elem);

return false;

}, false);

}

</script>

</head>

<body>

<!-- Для того что бы активировать возможность переносить данные необходимо установить атрибут draggable и создать обработчик на событие dragstart -->

<div id="source" draggable="true">Этот элемент</div>

<div id="target">Перенести в эту область</div>

</body>

</html>

## Geolocation

### Check Geolocation

<script type="text/javascript">

function checkSupport() {

if (navigator.geolocation) {

document.body.innerHTML = "Ваш браузер поддерживает HTML5 Geolocation API!";

}

else {

document.body.innerHTML = "Ваш браузер НЕ поддерживает HTML5 Geolocation API!";

}

}

window.addEventListener("load", checkSupport, true);

</script>

### Navigator Geolocation

<table border="1" cellpadding="10" cellspacing="0">

<tr>

<td>

широта

</td>

<td>

<p id="latitude">

</p>

</td>

</tr>

<tr>

<td>

долгота

</td>

<td>

<p id="longitude">

</p>

</td>

</tr>

<tr>

<td>

точность (м)

</td>

<td>

<p id="accuracy">

</p>

</td>

</tr>

<tr>

<td>

высота

</td>

<td>

<p id="altitude">

</p>

</td>

</tr>

<tr>

<td>

точность (высота)

</td>

<td>

<p id="altitudeAccuracy">

</p>

</td>

</tr>

<tr>

<td>

курс

</td>

<td>

<p id="heading">

</p>

</td>

</tr>

<tr>

<td>

скорость

</td>

<td>

<p id="speed">

</p>

</td>

</tr>

</table>

<p id="status">

</p>

<script type="text/javascript">

// Разовое получение данных о местоположении.

// updateLocation - метод, который запуститься после получения данных о местоположении.

// handleError - метод запуститься в случае сбоя при получении местоположения.

navigator.geolocation.getCurrentPosition(updateLocation, handleError);

function updateLocation(position) {

var $ = function (selector) {

return document.getElementById(selector);

}

$("latitude").innerHTML = position.coords.latitude;

$("longitude").innerHTML = position.coords.longitude;

$("accuracy").innerHTML = position.coords.accuracy;

$("altitude").innerHTML = position.coords.altitude;

$("altitudeAccuracy").innerHTML = position.coords.altitudeAccuracy;

$("heading").innerHTML = position.coords.heading;

$("speed").innerHTML = position.coords.speed;

}

function handleError(error) {

switch (error.code) {

case 0:

updateStatus("При попытке определить местоположение возникала ошибка: " + error.message);

break;

case 1:

updateStatus("Пользователь запретил получение данных о местоположении.");

break;

case 2:

updateStatus("Браузеру не удалось определить местоположение: " + error.message);

break;

case 3:

updateStatus("Истекло доступное время ожидания.");

break;

}

}

function updateStatus(message) {

document.getElementById("status").innerHTML = message;

}

</script>

### Geolocation Options

<button id="get">Получить текущее расположение</button>

<table border="1" cellpadding="10" cellspacing="0">

<tr>

<td>широта

</td>

<td>

<p id="latitude">

</p>

</td>

</tr>

<tr>

<td>долгота

</td>

<td>

<p id="longitude">

</p>

</td>

</tr>

<tr>

<td>точность (м)

</td>

<td>

<p id="accuracy">

</p>

</td>

</tr>

</table>

<p id="status">

</p>

<img id="loader" src="loader.gif" style="visibility:hidden" />

<script type="text/javascript">

document.getElementById("get").onclick = function () {

showLoadStatus();

// Разовое получение данных о местоположении.

// updateLocation - метод, который запуститься после получения данных о местоположении.

// handleError - метод запуститься в случае сбоя при получении местоположения.

// {

// enableHighAccuracy: true, - попытаться получить наиболее точные данные

// timeout: 10000, - данные должны быть получены в течении 10 секунд.

// maximumAge: 30000 - если данные были получены в течении последних 30 секунд их можно не запрашивать заново.

// }

navigator.geolocation.getCurrentPosition(updateLocation, handleError, { enableHighAccuracy: true, timeout: 10000, maximumAge: 30000 });

function updateLocation(position) {

var $ = function (selector) {

return document.getElementById(selector);

}

$("latitude").innerHTML = position.coords.latitude;

$("longitude").innerHTML = position.coords.longitude;

$("accuracy").innerHTML = position.coords.accuracy;

hideLoadStatus();

}

function handleError(error) {

switch (error.code) {

case 0:

updateStatus("При попытке определить местоположение возникала ошибка: " + error.message);

break;

case 1:

updateStatus("Пользователь запретил получение данных о местоположении.");

break;

case 2:

updateStatus("Браузеру не удалось определить местоположение: " + error.message);

break;

case 3:

updateStatus("Истекло доступное время ожидания.");

break;

}

hideLoadStatus();

}

function updateStatus(message) {

document.getElementById("status").innerHTML = message;

}

function showLoadStatus() {

document.getElementById("loader").style.visibility = "visible";

}

function hideLoadStatus() {

document.getElementById("loader").style.visibility = "hidden";

}

}

</script>

### Watch Position

<script type="text/javascript">

if (window.addEventListener)

window.addEventListener("load", function () {

document.getElementById("start").onclick = startWatch;

document.getElementById("stop").onclick = stopWatch;

}, true);

var watchId;

function startWatch() {

watchId = navigator.geolocation.watchPosition(updateLocation, handleError, { maximumAge: 1000 });

}

function stopWatch() {

navigator.geolocation.clearWatch(watchId);

}

function updateLocation(position) {

var $ = function (selector) {

return document.getElementById(selector);

}

$("latitude").innerHTML = position.coords.latitude;

$("longitude").innerHTML = position.coords.longitude;

$("accuracy").innerHTML = position.coords.accuracy;

$("altitude").innerHTML = position.coords.altitude;

$("altitudeAccuracy").innerHTML = position.coords.altitudeAccuracy;

$("heading").innerHTML = position.coords.heading;

$("speed").innerHTML = position.coords.speed;

}

function handleError(error) {

switch (error.code) {

case 0:

updateStatus("При попытке определить местоположение возникала ошибка: " + error.message);

break;

case 1:

updateStatus("Пользователь запретил получение данных о местоположении.");

break;

case 2:

updateStatus("Браузеру не удалось определить местоположение: " + error.message);

break;

case 3:

updateStatus("Истекло доступное время ожидания.");

break;

}

}

function updateStatus(message) {

document.getElementById("status").innerHTML = message;

}

</script>

### Show On Map Position

<head>

<title>Geolocation API</title>

<script type="text/javascript" src="http://maps.google.com/maps/api/js?sensor=true"></script>

</head>

<body>

<div id="map" style="height: 300px; width: 600px;">

</div>

<p id="message">

</p>

<script type="text/javascript">

navigator.geolocation.getCurrentPosition(updatePosition);

function updatePosition(pos) {

// задание координат.

var position = new google.maps.LatLng(pos.coords.latitude, pos.coords.longitude);

document.getElementById("message").innerHTML = "Позиция определена с точностью в " + pos.coords.accuracy + " метров.";

// параметры для карты.

var options = {

zoom: 15,

center: position, // позиция на карте

mapTypeId: google.maps.MapTypeId.ROADMAP // тип карты - ROADMAP, SATELLITE, HYBRID and TERRAIN

};

// объект карты.

var map = new google.maps.Map(document.getElementById("map"), options);

// маркер на карте

var marker = new google.maps.Marker({

position: position,

map: map,

title: "Это Вы!"

});

// объект всплывающей подсказки.

var infowindow = new google.maps.InfoWindow({

content: "Скорее всего, это Ваше текущее положение"

});

// присвоение обработчика на нажатие по маркеру.

google.maps.event.addListener(marker, 'click', function () {

infowindow.open(map, marker);

});

}

</script>

### Position By Adress

<script src="http://maps.google.com/maps/api/js?sensor=false"></script>

<script>

window.onload = function () {

// проверяем нажатие кнопки

document.getElementById('go').onclick = function () {

// получаем введенный пользователем адрес

var address = document.getElementById('address').value;

if (address) {

// используем API Google Maps для геокодирования адреса

// настраиваем объект Geocoder

var geocoder = new google.maps.Geocoder();

// возвращаем координаты

geocoder.geocode({ 'address': address }, function (results, status) {

if (status == google.maps.GeocoderStatus.OK) {

if (results[0]) {

// выводим результаты на экран

printLatLong(

results[0].geometry.location.lat(),

results[0].geometry.location.lng());

} else {

error('Google не возвратил результатов.');

}

} else {

error("Геокодирование завершилось ошибкой - " + status);

}

});

}

else {

error('Пожалуйста, введите адрес');

}

};

};

// выводим значения широты и долготы

function printLatLong(lat, long) {

document.getElementById("latOutput").innerHTML = lat;

document.getElementById("longOutput").innerHTML = long;

}

function error(msg) {

alert(msg);

}

</script>

</head>

<body>

<div>

<label for="address">

Введите адрес:

<input type="text" id="address" /></label>

<div>

<input type="button" id="go" value="Определить широту и долготу" />

</div>

<br />

<table>

<tr>

<td>Широта:</td>

<td id="latOutput"></td>

</tr>

<tr>

<td>Долгота:</td>

<td id="longOutput"></td>

</tr>

</table>

</div>

# Canvas

## Canvas

### Line

<title>Canvas</title>

<script src="Modernizr-2.0.6.js" type="text/javascript"></script>

<script type="text/javascript">

function draw() {

// проверка поддержки элемента canvas в браузере.

if (!Modernizr.canvas) return;

// получение элемента и его графического контекста.

var canvas = document.getElementById("canvas");

var context = canvas.getContext("2d");

// создание пути.

context.beginPath();

context.moveTo(100, 100);

context.lineTo(200, 200);

// черчение линии на холсте.

context.stroke();

}

if (window.addEventListener)

window.addEventListener("load", draw, true);

</script>

</head>

<body>

<canvas id="canvas" height="300" width="300" style="border: solid 1px black;">

Ваш браузер не поддерживает элемент <b>Canvas</b>

</canvas>

</body>

### Transform

<script src="Modernizr-2.0.6.js" type="text/javascript"></script>

<script type="text/javascript">

function draw() {

// проверка поддержки элемента canvas в браузере.

if (!Modernizr.canvas) return;

// получение элемента и его графического контекста.

var canvas = document.getElementById("canvas");

var context = canvas.getContext("2d");

// сохраняем копию текущего состояния контекста.

context.save();

// переносим графический контекст вправо и вниз.

context.translate(100, 100);

// рисуем линию.

context.beginPath();

context.moveTo(0, 0);

context.lineTo(100, 100);

context.stroke();

// восстанавливаем прежнее состояние контекста.

context.restore();

}

if (window.addEventListener)

window.addEventListener("load", draw, true);

</script>

</head>

<body>

<canvas id="canvas" height="300" width="300" style="border: solid 1px black;">

Ваш браузер не поддерживает элемент <b>Canvas</b>

</canvas>

</body>

### Draw Path

<script src="Modernizr-2.0.6.js" type="text/javascript"></script>

<script type="text/javascript">

function draw() {

// проверка поддержки элемента canvas в браузере.

if (!Modernizr.canvas) return;

var canvas = document.getElementById("canvas");

var context = canvas.getContext("2d");

context.save();

context.translate(100, 100);

// начало пути.

context.beginPath();

context.moveTo(0, 0);

context.lineTo(100, 0);

context.lineTo(100, 100);

context.lineTo(0, 100);

// прорисовка линни от точки (0, 100) к началу пути.

context.closePath();

// прорисовка контура полученного пути.

context.stroke();

context.restore();

}

if (window.addEventListener)

window.addEventListener("load", draw, true);

</script>

</head>

<body>

<canvas id="canvas" height="300" width="300" style="border: solid 1px black;">

Ваш браузер не поддерживает элемент <b>Canvas</b>

</canvas>

### Line style

<script src="Modernizr-2.0.6.js" type="text/javascript"></script>

<script type="text/javascript">

function draw() {

// проверка поддержки элемента canvas в браузере.

if (!Modernizr.canvas) return;

// получение элемента и его графического контекста.

var canvas = document.getElementById("canvas");

var context = canvas.getContext("2d");

context.save();

context.translate(100, 100);

// начало пути.

context.beginPath();

context.moveTo(0, 0);

context.lineTo(100, 0);

context.lineTo(100, 100);

context.lineTo(0, 100);

context.closePath();

// настройка стилей линии.

context.lineWidth = 8;

context.strokeStyle = "green";

context.lineJoin = "bevel";

context.stroke();

// восстанавливаем прежнее состояние контекста.

context.restore();

}

if (window.addEventListener)

window.addEventListener("load", draw, true);

</script>

### Fill style

<script src="Modernizr-2.0.6.js" type="text/javascript"></script>

<script type="text/javascript">

function draw() {

// проверка поддержки элемента canvas в браузере.

if (!Modernizr.canvas) return;

// получение элемента и его графического контекста.

var canvas = document.getElementById("canvas");

var context = canvas.getContext("2d");

context.save();

context.translate(100, 100);

// начало пути.

context.beginPath();

context.moveTo(0, 0);

context.lineTo(100, 0);

context.lineTo(100, 100);

context.lineTo(0, 100);

context.closePath();

// настройка стиля заливки

context.fillStyle = "blue";

context.fill();

// настройка стилей линии.

context.lineWidth = 8;

context.strokeStyle = "rgba(100, 100, 100, 1)";

context.lineJoin = "round";

context.stroke();

// восстанавливаем прежнее состояние контекста.

context.restore();

}

if (window.addEventListener)

window.addEventListener("load", draw, true);

</script>

### Fill rect

<script src="Modernizr-2.0.6.js" type="text/javascript"></script>

<script type="text/javascript">

function draw() {

if (!Modernizr.canvas) return;

var canvas = document.getElementById("canvas");

var context = canvas.getContext("2d");

// заполнение прямоугольной области синим цветом.

context.fillStyle = "blue";

context.fillRect(100, 100, 50, 50);

// вычерчивание красной границы толщиной в 10 пикселей

context.strokeStyle = "red";

context.lineWidth = 10;

context.strokeRect(100, 100, 50, 50);

// очистка прямоугольной области от контента.

context.clearRect(125, 125, 35, 35);

}

if (window.addEventListener)

window.addEventListener("load", draw, true);

</script>

### Curves

<title>quadraticCurveTo()</title>

<script src="Modernizr-2.0.6.js" type="text/javascript"></script>

<script type="text/javascript">

function draw() {

if (!Modernizr.canvas) return;

var canvas = document.getElementById("canvas");

var context = canvas.getContext("2d");

// первая точка

context.moveTo(0, 100);

context.quadraticCurveTo(/\*управляющая точка\*/50, 0, /\*вторая точка\*/100, 100);

context.lineWidth = 3;

context.strokeStyle = "red";

context.stroke();

}

if (window.addEventListener)

window.addEventListener("load", draw, true);

</script>

### BezierCurves

<script type="text/javascript">

function draw() {

if (!Modernizr.canvas) return;

var canvas = document.getElementById("canvas");

var context = canvas.getContext("2d");

// первая точка

context.moveTo(0, 100);

context.bezierCurveTo(/\*управляющая точка 1\*/50, 70, /\*управляющая точка 2\*/90, 10, /\*вторая точка\*/100, 100);

context.lineWidth = 3;

context.strokeStyle = "red";

context.stroke();

}

if (window.addEventListener)

window.addEventListener("load", draw, true);

</script>

</head>

<body>

<canvas id="canvas" height="100" width="100" style="border: solid 1px black;">

Ваш браузер не поддерживает элемент <b>Canvas</b>

</canvas>

### Load image

<script src="Modernizr-2.0.6.js" type="text/javascript"></script>

<script type="text/javascript">

function draw() {

if (!Modernizr.canvas) return;

var canvas = document.getElementById("canvas");

var context = canvas.getContext("2d");

var logo = new Image();

logo.onload = function () {

context.drawImage(logo, 22, 22);

};

logo.src = "Logo.png";

}

if (window.addEventListener)

window.addEventListener("load", draw, true);

</script>

</head>

<body>

<canvas id="canvas" height="300" width="300" style="border: solid 1px black;">

Ваш браузер не поддерживает элемент <b>Canvas</b>

</canvas>

</body>

### Repeat image

<script src="Modernizr-2.0.6.js" type="text/javascript"></script>

<script type="text/javascript">

function draw() {

if (!Modernizr.canvas) return;

var canvas = document.getElementById("canvas");

var context = canvas.getContext("2d");

var pattern = new Image();

pattern.src = "tile.jpg";

pattern.onload = function () {

context.fillStyle = context.createPattern(pattern, 'repeat');

context.fillRect(0, 0, 300, 300);

}

}

if (window.addEventListener)

window.addEventListener("load", draw, true);

</script>

</head>

<body>

<canvas id="canvas" height="300" width="300" style="border: solid 1px black;">

Ваш браузер не поддерживает элемент <b>Canvas</b>

</canvas>

### Gradient

<script src="Modernizr-2.0.6.js" type="text/javascript"></script>

<script type="text/javascript">

function draw() {

if (!Modernizr.canvas) return;

var canvas = document.getElementById("canvas");

var context = canvas.getContext("2d");

// Создание объекта градиента и указание начальной и конечной точки оси градиента.

var gradient = context.createLinearGradient(0, 0, 0, 300);

// Добавление цвета и смещения позиции остановки цвета.

gradient.addColorStop(0, "Blue");

// Добавление цвета и смещения позиции остановки цвета.

gradient.addColorStop(0.5, "LightBlue");

// Добавление цвета и смещения позиции остановки цвета.

gradient.addColorStop(1, "White");

// Установка объекта градиента в качестве цвета заливки контекста.

context.fillStyle = gradient;

context.fillRect(0, 0, 300, 300);

}

if (window.addEventListener)

window.addEventListener("load", draw, true);

</script>

</head>

<body>

<canvas id="canvas" height="300" width="300" style="border: solid 1px black;">

Ваш браузер не поддерживает элемент <b>Canvas</b>

</canvas>

### Gradient2

<script src="Modernizr-2.0.6.js" type="text/javascript"></script>

<script type="text/javascript">

function draw() {

if (!Modernizr.canvas) return;

var canvas = document.getElementById("canvas");

var context = canvas.getContext("2d");

// Настройка градиента.

var gradient = context.createLinearGradient(100, 100, 200, 200);

gradient.addColorStop(0, "Red");

gradient.addColorStop(1, "Orange");

// Установка объекта градиента в качестве цвета заливки контекста.

context.fillStyle = gradient;

context.fillRect(100, 100, 100, 100);

var gradient2 = context.createLinearGradient(150, 150, 250, 250);

// rgba(0, 0, 0, 0.5) - черный цвет со значением альфа-канала 50%

gradient2.addColorStop(0, 'rgba(0, 0, 0, 0.5)');

gradient2.addColorStop(0.5, "rgba(100, 100, 100, 0.5)");

gradient2.addColorStop(1, "rgba(190, 190, 190, 0.5)");

context.fillStyle = gradient2;

context.fillRect(150, 150, 100, 100);

}

if (window.addEventListener)

window.addEventListener("load", draw, true);

</script>

Radial gradient

<script src="Modernizr-2.0.6.js" type="text/javascript"></script>

<script type="text/javascript">

function draw() {

if (!Modernizr.canvas) return;

var canvas = document.getElementById("canvas");

var context = canvas.getContext("2d");

// Создание радиального градиента.

var gradient = context.createRadialGradient(250, 250,200, 250, 250, 400);

gradient.addColorStop(0, "Orange");

gradient.addColorStop(1, "Red");

context.fillStyle = gradient;

context.fillRect(0, 0, 600, 600);

}

if (window.addEventListener)

window.addEventListener("load", draw, true);

</script>

</head>

<body>

<canvas id="canvas" height="600" width="600" style="border: solid 1px black;">

Ваш браузер не поддерживает элемент <b>Canvas</b>

</canvas>

</body>

### Canvas translate

<title>translate transformation</title>

<script type="text/javascript">

function loadHandler() {

// функция для получения элементов по id

var $ = function (id) {

return document.getElementById(id);

}

// контекст для canvas

var context = $("canvas").getContext("2d");

$("btn").onclick = function () {

// очистка всего полотна.

context.clearRect(0, 0, context.canvas.width, context.canvas.height);

var x = $("X").value;

var y = $("Y").value;

// сохранение состояния полотна и начала системы координат.

context.save();

// трансформация перемещения.

context.translate(x, y);

context.fillStyle = "green";

context.fillRect(0, 0, 100, 100);

// восстановление сохранённого состояния полотна.

context.restore();

}

}

if (window.addEventListener)

window.addEventListener("load", loadHandler, true);

</script>

</head>

<body>

<canvas id="canvas" width="400" height="300">

</canvas>

<div style="border: 1px solid gray; padding: 8px; margin: 8px;">

X координата

<input type="text" id="X" value="0" />

Y координата

<input type="text" id="Y" value="0" />

<input type="button" id="btn" value="translate" />

</div>

</body>

### Scale

<title>scale transformation</title>

<script type="text/javascript">

function loadHandler() {

// функция для получения элементов по id

var $ = function (id) {

return document.getElementById(id);

}

// контекст для canvas

var context = $("canvas").getContext("2d");

var img = new Image();

img.src = "logo.png";

img.onload = draw;

$("btn").onclick = draw;

function draw() {

// очистка всего полотна.

context.clearRect(0, 0, context.canvas.width, context.canvas.height);

var x = $("X").value;

var y = $("Y").value;

// сохранение состояния полотна и начала системы координат.

context.save();

// трансформация масштабирования.

context.scale(x, y);

context.drawImage(img, 0, 0, 100, 100);

// восстановление сохранённого состояния полотна.

context.restore();

}

}

if (window.addEventListener)

window.addEventListener("load", loadHandler, true);

</script>

### Rotate

<title>rotate transformation</title>

<script type="text/javascript">

function loadHandler() {

// функция для получения элементов по id

var $ = function (id) {

return document.getElementById(id);

}

// контекст для canvas

var context = $("canvas").getContext("2d");

var img = new Image();

img.src = "logo.png";

img.onload = draw;

$("btn").onclick = draw;

function draw() {

// очистка всего полотна.

context.clearRect(0, 0, context.canvas.width, context.canvas.height);

var angle = $("angle").value;

angle = angle \* Math.PI / 180;

// сохранение состояния полотна и начала системы координат.

context.save();

// трансформация вращения.

context.rotate(angle);

context.drawImage(img, 100, 100, 200, 200);

// восстановление сохранённого состояния полотна.

context.restore();

}

}

if (window.addEventListener)

window.addEventListener("load", loadHandler, true);

</script>

</head>

<body>

<canvas id="canvas" width="300" height="300">

</canvas>

<div style="border: 1px solid gray; padding: 8px; margin: 8px;">

Угол поворота

<input type="text" id="angle" value="0" />

<input type="button" id="btn" value="rotate" />

</div>

</body>

### Canvas shadows

<title>Shadows</title>

<script type="text/javascript">

function draw() {

var context = document.getElementById("canvas").getContext("2d");

context.font = "bold 40px Segoe UI";

context.fillStyle = "Orange";

context.textBaseline = "alphabetic";

context.shadowColor = "rgba(0, 0, 0, 0.2)";

context.shadowOffsetX = 5;

context.shadowOffsetY = 5;

context.shadowBlur = 10;

var text = "HTML5 Rocks!";

var w = context.measureText(text);

context.fillText(text, (context.canvas.width - w.width) / 2, 100);

}

if (window.addEventListener)

window.addEventListener("load", draw, true);

</script>

</head>

<body>

<canvas id="canvas" width="300" height="300" style="border: solid 1px black;">

</canvas>

## Canvas examples

### Paint

<title>Canvas Paint</title>

<script type="text/javascript">

window.addEventListener("load", init, false);

var started = false, canvas, context;

function init() {

canvas = document.getElementById("canvas");

context = canvas.getContext("2d");

canvas.addEventListener("mousemove", moveHandler, false);

canvas.addEventListener("mousedown", downHandler, false);

canvas.addEventListener("mouseup", upHandler, false);

}

// Получение координат мыши относительно элемента canvas

function getCoords(e) {

var x, y;

if (e.layerX || e.layerX == 0) { // Firefox

x = e.layerX;

y = e.layerY;

} else if (e.offsetX || e.offsetX == 0) {

x = e.offsetX;

y = e.offsetY;

}

return { x: x, y: y };

}

// Начало рисования.

function downHandler(e) {

context.beginPath();

context.moveTo(getCoords(e).x, getCoords(e).y);

started = true;

}

// Прекращение рисования.

function upHandler(e) {

started = false;

}

// обработка движения указателя по элементу canvas

function moveHandler(e) {

if (started) {

context.lineTo(getCoords(e).x, getCoords(e).y);

context.stroke();

}

}

</script>

</head>

<body>

<canvas id="canvas" width="400" height="200" style="border: 1px solid black;">

Браузер не поддерживает canvas

</canvas>

### Canvas animation

<head>

<title>Canvas Loader</title>

<script src="http://padolsey.net/p/Sonic/repo/src/sonic.js"></script>

</head>

<body>

<script>

// Создание объекта Sonic для отображения анимированных элементов в canvas

var square = new Sonic({

width: 100,

height: 100,

fillColor: '#000',

// Массив определяющий движение анимации

path: [['line', 10, 10, 90, 10], ['line', 90, 10, 90, 90], ['line', 90, 90, 10, 90], ['line', 10, 90, 10, 10]]

});

// запуск анимации

square.play();

// добавление анимации в документ

document.body.appendChild(square.canvas);

</script>

<br />

<a href="http://padolsey.net/p/Sonic/repo/demo/demo.html">Дополнительные примеры</a><br />

<a href="https://github.com/padolsey/Sonic">Документация</a>

</body>

### Canvas animation2

<title>Анимация Canvas</title>

<style>

body {

margin: 0px;

padding: 0px;

}

canvas {

border: 1px solid #ccc;

}

</style>

</head>

<body>

<canvas id="myCanvas" width="578" height="200"></canvas>

<script>

// https://developer.mozilla.org/en-US/docs/Web/API/window.requestAnimationFrame

// requestAnimationFrame - метод, который указывает браузеру что наш код хочет выполнить анимацию

// и запрашивает вызов callback функции перед следующей перерисовкой.

window.requestFrame = function (callback) {

// переменная f будет содержать ту функцию, для запроса отрисовки одного кадра, которую поддерживает текущий браузер.

var f = window.mozRequestAnimationFrame ||

window.requestAnimationFrame ||

window.webkitRequestAnimationFrame ||

function (callback) {

window.setTimeout(callback, 500);

}

f(callback);

};

// функция которая рисует прямоугольник

// первый параметр - объект определяющий размеры прямоугольника

// второй параметр - контекст канваса на котором будет произведено рисование

function drawRectangle(rect, context) {

context.beginPath();

context.rect(rect.x, rect.y, rect.width, rect.height);

context.fillStyle = 'green';

context.fill();

context.lineWidth = rect.borderWidth;

context.strokeStyle = 'black';

context.stroke();

}

function animate(rect, canvas, context, startTime) {

// определяем разницу между текущим временем и временем начала анимации.

var time = (new Date()).getTime() - startTime;

var linearSpeed = 100;

// pixels / second

var newX = linearSpeed \* time / 1000;

// меняем свойство x объекта прямоугольника до тех пор пока он не дойдет до правого края канваса.

if (newX < canvas.width - rect.width - rect.borderWidth / 2) {

rect.x = newX;

}

// очищаем канвас

context.clearRect(0, 0, canvas.width, canvas.height);

// рисуем прямоугольник с учетом измененной координаты x

drawRectangle(rect, context);

// запрашиваем новый фрейм

requestFrame(function () {

animate(rect, canvas, context, startTime);

});

}

var canvas = document.getElementById('myCanvas');

var context = canvas.getContext('2d');

var rect = {

x: 0,

y: 75,

width: 100,

height: 50,

borderWidth: 5

};

drawRectangle(rect, context);

animate(rect, canvas, context, new Date().getTime());

</script>