Htlml5test.com

WebEssentilas

Html5rocks

Dom Inspector

Canvas demos

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

Leaking data

При роботе с несколькими вкладками, каждая вкладка может перезаписывать данные в cookies. Чтобы решить данную проблему следует применить Session storage, который сохраняет данные только для поточной вкладки.

# Introduction

## New Semantic

Google.com/fonts

### 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>

# Web Storage Web Workers Offline Apps

## Query selector

### Query selector

<script>

if (window.addEventListener) window.addEventListener("load", loadHandler, false);

function loadHandler() {

// Получение первого элемента на странице, который соответствует одному из селекторов указанных в параметрах.

// в данном случае возвращаем элемент со значением test в атрибуте class

var e = document.querySelector(".test"); // $(".test") - эквивалент в jQuery

var e = document.querySelector("#helloWorld", ".test2"); // получение элемента с id helloWold или элемента с классом test

var e = document.querySelector(".menu > li"); // первый li вложенный в элемент с классом menu

e.style.fontSize = "30px";

e.style.color = "red";

}

</script>

<style type="text/css">

.test {

color: Green;

}

</style>

</head>

<body>

<p id="helloWorld">

Item 1

</p>

<p class="test">

Item 2

</p>

<p>

Item 3

</p>

<p class="test">

Item 4

</p>

<p>

Item 5

</p>

<ul class="menu">

<li>menu item 1</li>

<li>menu item 2</li>

<li>menu item 3</li>

</ul>

### Query selectorAll

<script type="text/javascript">

if (window.addEventListener) window.addEventListener("load", loadHandler, false);

function loadHandler() {

var pArray = document.querySelectorAll(".testClass p");

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

pArray[i].style.border = "solid 1px red";

pArray[i].style.margin = "5px";

pArray[i].style.padding = "5px";

}

}

</script>

<style type="text/css">

.testClass {

border: 3px solid green;

}

</style>

</head>

<body>

<div class="testClass">

<p>

First

</p>

<p>

Second

</p>

<p>

Third

</p>

</div>

<div>

<p>

First

</p>

<p>

Second

</p>

<p>

Third

</p>

</div>

## Web Storage

### Check support

<script type="text/javascript">

function init() {

var e = document.getElementById("support");

if (window.sessionStorage) {

e.innerHTML = "Этот браузер поддерживает sessionStorage";

} else {

e.innerHTML = "Этот браузер НЕ поддерживает sessionStorage";

}

e.innerHTML += "<br />";

if (window.localStorage) {

e.innerHTML += "Этот браузер поддерживает localStorage";

} else {

e.innerHTML += "Этот браузер НЕ поддерживает localStorage";

}

}

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

</script>

</head>

<body>

<div id="support">

</div>

### Session storage

<title>Сохранение и извлечение данных sessionStorage</title>

<script type="text/javascript">

function init() {

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

var btn1 = document.getElementById("btn1");

var btn2 = document.getElementById("btn2");

btn1.onclick = function () {

window.sessionStorage.setItem("key1", "Test value");

// или

window.sessionStorage.key2 = "Test value 2";

// или

window.sessionStorage["key3"] = "Test value 3";

}

btn2.onclick = function () {

output.innerHTML = window.sessionStorage.getItem("key1");

output.innerHTML += "<br />";

output.innerHTML += window.sessionStorage.key2;

output.innerHTML += "<br />";

output.innerHTML += window.sessionStorage["key3"];

}

}

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

</script>

</head>

<body>

<p>

Значения существуют до тех пор, пока открыто окно или вкладка, в котором они сохранены

</p>

<p>

Значения видимы только в окне или вкладке, в которой они были созданы.

</p>

<input id="btn1" type="button" value="Запись в sessionStorage" />

<input id="btn2" type="button" value="Чтение из sessionStorage" />

<div id="output">

</div>

### Local storage

<script type="text/javascript">

function init() {

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

var btn1 = document.getElementById("btn1");

var btn2 = document.getElementById("btn2");

btn1.onclick = function () {

window.localStorage.setItem("key1", "Test value");

// или

window.localStorage["key2"] = "Test value 2";

}

btn2.onclick = function () {

output.innerHTML = window.localStorage.getItem("key1");

output.innerHTML += "<br />";

output.innerHTML += window.localStorage.key2;

}

}

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

</script>

</head>

<body>

<p>

Значения в localStorage существуют не зависимо от закрытия вкладки или браузера

</p>

<p>

Значения совместно используются всеми окнами и вкладками, выполняющимися на одном

и том же источнике.

</p>

<input id="btn1" type="button" value="Запись в localStorage" />

<input id="btn2" type="button" value="Чтение из localStorage" />

<div id="output">

</div>

### Local storage props

<script type="text/javascript">

window.localStorage.setItem("key1", "test1");

window.localStorage.setItem("key2", "test2");

window.localStorage.setItem("key3", "test3");

window.localStorage.setItem("key4", "test4");

window.localStorage.setItem("key5", "test5");

showStorageContent();

// удаление элемента по ключу

window.localStorage.removeItem("key1");

showStorageContent();

function showStorageContent() {

document.write("Количество пар ключ-значение " + window.localStorage.length + "<br />");

document.write("Все ключи в localStorage:<br />")

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

// key - получение элемента по индексу

document.write(window.localStorage[localStorage.key(i)] + "<br />");

}

document.write("<br />")

}

document.getElementById("btn1").onclick = function () {

// удаление всех записей в хранилище

window.localStorage.clear();

showStorageContent();

}

</script>

### Local storage quota

<script type="text/javascript">

try {

// Браузер может сохранить до 5 MB данных в хранилище.

for (var i = 0; i < 10000; i++) {

window.localStorage.setItem(i, "DATA DATA DATA DATA DATA DATA");

}

} catch (e) {

document.write(e);

}

</script>

### Storage events

// Событие storage происходит при внесении изменений в localStorage (при добавлении, удалении или изменении значений).

// Для того что бы увидить работу этого примера, откройте в браузере две вкладки

// в первой вносите изменения в localStorage, а на второй вкладке отслеживайте изменения.

window.addEventListener("storage", displayEvent, true);

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

function init() {

var key = document.getElementById("keyInput");

var val = document.getElementById("valueInput");

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

window.localStorage.setItem(key.value, val.value);

}

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

window.localStorage.removeItem(key.value);

}

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

alert(window.localStorage.getItem(key.value));

}

}

function displayEvent(e) {

var d = document.getElementById("output");

var message = "key: " + e.key + "<br />"; // измененный ключ

message += "newValue: " + e.newValue + "<br />"; // новое значение ключа

message += "oldValue: " + e.oldValue + "<br />"; // старое значение ключа

message += "url: " + e.url + "<br />"; // адрес страницы на которой произошли изменения

message += "storageArea: " + e.storageArea + "<br />"; // ссылка на объект хранилища

d.innerHTML = message;

}

</script>

</head>

<body>

Ключ <input id="keyInput" />

Значение <input id="valueInput" />

<br />

<button id="btnSave">Сохранить</button>

<button id="btnGet">Прочитать</button>

<button id="btnRemove">Удалить по ключу</button>

<div id="output">

</div>

### JSON Serialize

<script type="text/javascript">

var person = {};

person.firstName = "Jhon";

person.lastName = "Doe";

person.age = 30;

function save() {

window.localStorage["data"] = JSON.stringify(person); // сериализация данных и сохранение их в локальном хранилище

}

function read() {

var data = JSON.parse(window.localStorage["data"]); // десериализация объекта из локального хранилища.

alert("First Name: " + data.firstName +

"\r\nLast Name: " + data.lastName +

"\r\nAge: " + data.age);

}

function clearStorage() {

window.localStorage.clear(); // удаление данных из локального хранилища

}

</script>

</head>

<body>

<button onclick="save()">Save</button>

<button onclick="read()">Read</button>

<button onclick="clearStorage()">Clear</button>

</body>

## Web Workers

### Check support

<title>Web Workers Support Testing</title>

<script type="text/javascript">

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

if (typeof Worker !== "undefined") {

document.getElementById("support").innerHTML = "Браузер поддерживает HTML 5 Web Workers";

}

else {

document.getElementById("support").innerHTML = "Браузер НЕ поддерживает HTML 5 Web Workers";

}

}, false);

</script>

</head>

<body>

<div id="support">

</div>

### Simple worker

<title>Web Worker</title>

<script type="text/javascript">

// создание потока (Web Worker).

var worker = new Worker("task.js");

// Привязка обработчика на событие message, которое будет возникать в случае если поток присылает данные.

worker.addEventListener("message", function (e) {

document.getElementById("output").innerHTML += "Ответ потока: " + e.data;

}, true);

// Отправка данных потоку.

// Когда воркер получит сообщение в нем произойдет message

worker.postMessage('Hello world');

</script>

</head>

<body>

<div id="output"></div>

</body>

Task.js

// Получив сообщение поток делает ответ.

addEventListener("message", function (e) {

// продолжительные вычисления

postMessage(e.data); // Отправка данных основному потоку. На странице будет вызвано событие message

}, true);

### Worker JSON

<title>Web Worker</title>

<script type="text/javascript">

var worker = new Worker("task2.js"); // создание потока Web Worker.

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

document.getElementById("btnStart").addEventListener("click", startCmd, true);

document.getElementById("btnUnknown").addEventListener("click", unknownCmd, true);

document.getElementById("btnStop").addEventListener("click", stopCmd, true);

worker.addEventListener("message", messageHandler, true);

}, true);

function startCmd() {

// Потоку отправляется объект-команда.

worker.postMessage({ cmd: 'start', msg: 'Hi' });

}

function unknownCmd() {

worker.postMessage({ cmd: 'unknown', msg: '???' });

}

function stopCmd() {

worker.postMessage({ cmd: 'stop', msg: 'Bye' });

}

function messageHandler(e) {

document.getElementById("output").innerHTML = e.data;

}

</script>

</head>

<body>

<button id="btnStart">

Команда Старт</button>

<button id="btnUnknown">

Команда Unknown</button>

<button id="btnStop">

Команда Стоп</button>

<div id="output">

</div>

</body>

Task2.js

addEventListener("message", function (e) {

var data = e.data;

switch (data.cmd) {

case 'start':

postMessage("ПОТОК ЗАПУЩЕН : " + data.msg);

break;

case 'stop':

postMessage("ПОТОК ОСТАНОВЛЕН : " + data.msg);

close();

break;

default:

postMessage("НЕИВЕСТНАЯ КОМАНДА : " + data.msg);

}

}, true);

### Fibonacci web worker

<script>

window.onload = function () {

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

// Создание нового потока

var worker = new Worker("fibonacciWorker.js");

// Создание обработчика для получения сообщений от потока

worker.addEventListener("message", function (e) {

document.getElementById("output").innerHTML += e.data + "<br />";

}, false);

// запуск потока

worker.postMessage("");

}

}

</script>

</head>

<body>

<button id="startButton">Start</button>

<p id="output"></p>

</body>

FibonacciWorker.js

function calcFib(x) {

if (x > 1) {

return calcFib(x - 1) + calcFib(x - 2);

}

else {

return x;

}

}

addEventListener("message", function () {

for (var i = 0; i < 50; i++) {

postMessage(i + " = " + calcFib(i) + "<br />");

}

}, false);

Meomize

<script>

// Меомизация - прием кэширования результатов функции.

function calcFib(x) {

// проверка наличия результата для парамтра x.

if (!calcFib.cache[x]) { // если результат не найден то производим вычесления.

if (x > 1) {

calcFib.cache[x] = calcFib(x - 1) + calcFib(x - 2);

}

else {

calcFib.cache[x] = x;

}

}

return calcFib.cache[x];

}

// создание свойства в функции с пустым объектом.

calcFib.cache = {};

for (var i = 0; i < 500; i++) {

document.write(i + " = " + calcFib(i) + "<br />");

}

</script>

### Meomize

<script>

// Меомизация - прием кэширования результатов функции.

function calcFib(x) {

// проверка наличия результата для парамтра x.

if (!calcFib.cache[x]) { // если результат не найден то производим вычесления.

if (x > 1) {

calcFib.cache[x] = calcFib(x - 1) + calcFib(x - 2);

}

else {

calcFib.cache[x] = x;

}

}

return calcFib.cache[x];

}

// создание свойства в функции с пустым объектом.

calcFib.cache = {};

for (var i = 0; i < 500; i++) {

document.write(i + " = " + calcFib(i) + "<br />");

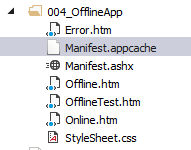
}

</script>

## Offline App

### Offline app

Чтобы произвести изменение в странице нужно сменить версию манифеста.



Manifest.apache

CACHE MANIFEST

# v3 18-05-2012 - версия для того что бы можно было указать что на сайте были внесены изменения в страницы, которые уже закешированы.

# браузер обновляет кэш в том случае если обновляется файл манифеста.

# CACHE - Кэшируемые страницы

CACHE:

Offline.htm

StyleSheet.css

OfflineTest.htm

Error.htm

# NETWORK - страницы всегда запрашиваются с сервера.

NETWORK:

Online.htm

### Manifest.ashx

<%@ WebHandler Language="C#" Class="Manifest" %>

using System;

using System.Web;

public class Manifest : IHttpHandler {

public void ProcessRequest(HttpContext context)

{

// Контент файла манифеста должен возвращаться с сервера с ContentType text/cache-manifest.

context.Response.ContentType = "text/cache-manifest";

context.Response.WriteFile(context.Server.MapPath("Manifest.appcache"));

}

public bool IsReusable {

get {

return false;

}

Offline.htm

<!DOCTYPE html>

<!-- С помощью файла manifest="Manifest.ashx" браузер определяет, какие файлы должны быть скачаны локально и сохранены -->

<html manifest="Manifest.ashx">

<head>

<title>Offline Page</title>

<link href="StyleSheet.css" rel="stylesheet" type="text/css" />

<script type="text/javascript">

var support;

function init() {

support = document.getElementById("support");

// Проверка поддержки в браузере.

if (window.applicationCache) {

support.innerHTML = "Автономные приложения поддерживаются данным браузером";

support.style.backgroundColor = "green";

}

else {

support.innerHTML = "Автономные приложения не поддерживаются данным браузером";

support.style.backgroundColor = "red";

return;

}

// Отображение событий запускаемых объектом applicationCache

function writeLog(msg) {

var logDiv = document.getElementById("log");

logDiv.innerHTML = msg + "<br />" + logDiv.innerHTML;

}

window.applicationCache.onchecking = function () {

writeLog("Проверка обновлений.");

}

window.applicationCache.onnoupdate = function () {

writeLog("Обновления не найдены.");

}

window.applicationCache.onupdateready = function () {

writeLog("Обновления загружены.");

window.applicationCache.swapCache(); // поменяет старый кеш на новый

if (confirm("Доступна новая версия этой страницы. Загрузить ее?")) {

window.location.reload();

}

}

window.applicationCache.onobsolete = function () {

writeLog("Данные приложения устарели.");

}

window.applicationCache.ondownloading = function () {

writeLog("Загрузка обновления.");

}

window.applicationCache.oncached = function () {

writeLog("Данные помещены в кэш.");

}

window.applicationCache.onerror = function (e) {

writeLog("Ошибка при кэшировании.");

}

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

messages = ["Не кэшировано", "Бездействующий", "Проверка", "Загрузка", "Готовый к обновлению", "Кэш устарел"];

writeLog(messages[window.applicationCache.status]);

}

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

try {

window.applicationCache.update();

} catch (e) {

window.applicationCache.onerror();

}

}

}

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

</script>

</head>

<body>

<div id="support">

</div>

<button id="updateCache">

Обновить кэш</button>

<button id="showStatus">

Показать текущий статус</button>

<a href="OfflineTest.htm">Offline Test</a>

<a href="Online.htm">Online Test</a>

<br />

<br />

<div id="log">

</div>

</body>

</html>

# CSS3 Indroduction

## Fundamentals

### Article > div

<style>

/\*

Child Combinator http://www.w3.org/TR/selectors/#child-combinators

Получение прямых потомков div элемента article.

\*/

article > div {

margin:10px;

border:1px solid green;

}

</style>

</head>

<body>

<article>

<div> <!--Данный элемент будет выбран селектором-->

Paragraph 1

<div>Inner Paragraph 1</div>

</div>

<div> <!--Данный элемент будет выбран селектором-->

Paragraph 2

<div>Inner Paragraph 2</div>

</div>

<div> <!--Данный элемент будет выбран селектором-->

Paragraph 3

<div>Inner Paragraph 3</div>

</div>

</article>

### H2+p

<style>

/\*

Adjacent sibling combinator http://www.w3.org/TR/selectors/#adjacent-sibling-combinators

получение p, который сразу следует за h2.

\*/

h2 + p {

margin:10px;

border:1px solid green;

}

</style>

</head>

<body>

<h2>Hello</h2>

<p>world</p><!--Данный элемент будет выбран селектором-->

<h2>Hello</h2>

<p>world</p><!--Данный элемент будет выбран селектором-->

<p>HELLO WORLD</p>

<h2>hello world</h2>

### H2~p

<style>

/\*

General sibling combinator http://www.w3.org/TR/selectors/#general-sibling-combinators

Элементы имеют общего родителья в документе и элемент p находиться за элементом h2.

\*/

h2 ~ p {

margin: 10px;

border: 1px solid green;

}

</style>

</head>

<body>

<div>

<h2>Hello</h2>

<ul>

<li>Item 1</li>

<li>Item 2</li>

<li>Item 3</li>

</ul>

<p>world</p> <!--Данный элемент будет выбран селектором-->

</div>

<div>

<h2>Hello</h2>

<a href="#">Link</a>

<p>world</p> <!--Данный элемент будет выбран селектором-->

</div>

<p>HELLO WORLD</p>

<h2>hello world</h2>

### Input[atrribute]

<style>

/\* правило для всех input \*/

input {

margin:10px;

}

/\* правило для input с атрибутом required \*/

input[required] {

border: 1px solid red;

}

/\* правило для input с атрибутом type=submit \*/

input[type=submit] {

border: 1px solid green;

}

</style>

</head>

<body>

<form>

<input type="text" required="required" />

<br />

<input type="password" />

<br />

<input type="submit" />

</form>

### Input[pattern]

<style>

/\* атрибут href начинаеться со значения http:// \*/

a[href^="http://"] {

background: url('external.png') 100% 50% no-repeat;

padding-right: 15px;

}

/\* атрибут href заканчивается значением pdf \*/

a[href$=".pdf"] {

background: url('pdf.png') 100% 50% no-repeat;

padding-right: 20px;

}

/\* атрибут href содержит значение twitter \*/

a[href\*="twitter"] {

background: url('twitter.png') 100% 50% no-repeat;

padding-right: 15px;

}

</style>

</head>

<body>

<a href="http://google.com">Search</a>

<br />

<a href="http://twitter.com/profile">Some Twitter User</a>

<br />

<a href="/resources/test.pdf">Manual</a>

<br />

</body>

## Backgrounds

### Backgrounds-size

<style>

div {

width: 600px;

height: 400px;

margin: 10px;

border: 1px solid black;

background-repeat: no-repeat;

}

#id1 {

background-image: url("bg.jpg");

/\*Размер фона установлен явно в пикселах\*/

-moz-background-size: 500px 342px;

-o-background-size: 500px 342px;

-webkit-background-size: 500px 342px;

background-size: 500px 342px;

}

#id2 {

background-image: url("bg.jpg");

/\*фон занимает половину высоты и ширины элемента\*/

-moz-background-size: 50% 50%;

-o-background-size: 50% 50%;

-webkit-background-size: 50% 50%;

background-size: 50% 50%;

}

</style>

### Cover Contain

<style>

div {

width: 800px;

height: 400px;

border: 1px solid black;

background-repeat: no-repeat;

}

#id1 {

background-image: url("bg.jpg");

/\* contain - масштабирование фона таким образом, чтобы картинка увеличивала размер сохраняя пропорции до тех пор, пока ее высота или ширина не будет равна высоте или ширине блока \*/

-moz-background-size: contain;

-o-background-size: contain;

-webkit-background-size: contain;

background-size: contain;

}

#id2 {

background-image: url("bg.jpg");

/\* cover - Масштабирование фона таким образом, чтобы картинка увеличивала размеры сохраняя пропорции полностью заполнила весь блок \*/

-moz-background-size: cover;

-o-background-size: cover;

-webkit-background-size: cover;

background-size: cover;

}

### Background-image

<style>

#id1 {

/\*

backgound-image может использоваться для задания нескольких фонов для одного блока.

Фоны должны быть определены через запятую во всех CSS свойствах относящихся к настройкам фоновых изображений.

Фон определенный в списке первым будет отображаться поверх остальных

\*/

background-image: url(logo.png), url(bg.jpg);

background-position: bottom right, center;

background-repeat: no-repeat, no-repeat;

-moz-background-size: 135px 41px, 100% 100%;

-o-background-size: 135px 41px, 100% 100%;

-webkit-background-size: 135px 41px, 100% 100%;

background-size: 135px 41px, 100% 100%;

border: 1px #808080 solid;

height: 270px;

width: 350px;

}

</style>

### Background-origin

<style>

#id1, #id2, #id3 {

padding: 40px;

border: 40px #b5e1ae solid;

height: 150px;

width: 200px;

margin: 10px;

background-repeat: no-repeat;

font-weight: bold;

}

#id1 {

background-image: url("bg.jpg");

/\* border-box - фон позиционируется относительно границы, при этом рамка элемента может перекрывать фон \*/

-moz-background-origin: border-box;

-webkit-background-origin: border-box;

background-origin: border-box;

}

#id2 {

background-image: url("bg.jpg");

/\* padding-box - фон позиционируется относительно края элемента с учетом толщины border\*/

-moz-background-origin: padding-box;

-webkit-background-origin: padding-box;

background-origin: padding-box;

}

#id3 {

background-image: url("bg.jpg");

/\* content-box - фон позиционируется относительно содержимого элемента\*/

-moz-background-origin: content-box;

-webkit-background-origin: content-box;

background-origin: content-box;

}

</style>

## Colors

### Rgba

<style>

body {

background-image: url(bg.jpg);

background-repeat: no-repeat;

-moz-background-size: 500px 370px;

-o-background-size: 500px 370px;

-webkit-background-size: 500px 370px;

background-size: 500px 370px;

}

div {

width: 200px;

height: 120px;

margin: 10px;

float: left;

padding: 10px;

font-size: 1.5em;

background-color: black;

color: orange;

z-index: 100;

}

#id1 {

/\* rgba(red, green, blue, alpha)

значение каналов red, green, blue - от 0 до 255

значение alpha-канала 0 - полностью прозрачный, 1 - полностью не прозрачный

При использовании rgba прозрачность задаётся только элементу, дочерние элементы остаются без изменений.

\*/

background-color: rgba(0, 0, 0, 0.1);

}

#id2 {

/\* opacity - устанавливает прозрачность для элемента и всех дочерних элементов \*/

opacity: 0.1;

}

</style>

### Hsl

<style>

div {

padding: 10px;

margin: 10px;

font-weight:bold;

color:white;

}

#id1 {

background-color: hsl(0, 20%, 50%);

}

#id2 {

background-color: hsl(120, 100%, 80%);

}

#id3 {

background-color: hsl(240, 100%, 50%);

}

</style>

### Hsla

<style>

body {

background-image: url(bg.jpg);

background-repeat: no-repeat;

-moz-background-size: 500px 370px;

-o-background-size: 500px 370px;

-webkit-background-size: 500px 370px;

background-size: 500px 370px;

}

div {

width: 200px;

height: 120px;

margin: 10px;

float: left;

padding: 10px;

font-size: 1.5em;

background-color: black;

color: orange;

z-index: 100;

}

#id1 {

/\* Использование цветовой модели SHL с alpha каналом\*/

background-color: hsla(0, 100%,0%, 0.1);

}

#id2 {

opacity: 0.1;

}

</style>

## Border

### Border-radius

<style>

#id1, #id2, #id3, #id4 {

float: left;

border: 2px #000000 solid;

padding: 30px;

width: 300px;

height: 100px;

margin: 20px;

background-color: #41dade;

color: #FFF;

font-weight: bold;

}

#id1 {

/\* border-radius задает радиус скругления углов рамки. Также действует на фон\*/

border-radius: 5px;

}

#id2 {

border-radius: 10px;

}

#id3 {

border-radius: 20px;

}

#id4 {

border-radius: 80px;

}

span {

text-decoration: underline;

}

</style>

### Border-radius

<style>

#id1, #id2, #id3, #id4 {

float: left;

border: 2px #000000 solid;

padding: 10px;

width: 300px;

height: 100px;

margin: 20px;

background-color: #ccc;

color: #FFF;

font-weight: bold;

}

#id1 {

border-top-left-radius: 20px;

}

#id2 {

border-top-right-radius: 20px;

}

#id3 {

border-bottom-left-radius: 20px;

}

#id4 {

border-bottom-right-radius: 20px;

}

span {

text-decoration: underline;

}

</style>

### Border-radius

<style>

#id1, #id2, #id3, #id4 {

float: left;

border: 2px #000000 solid;

padding: 30px;

width: 300px;

height: 100px;

margin: 20px;

background-color: #41dade;

color: #FFF;

font-weight: bold;

}

#id1 {

/\* border-radius с двумя значениями через слэш позволяет создать эллиптические углы\*/

border-radius: 10px/20px;

}

#id2 {

/\* первое значение скруглённые верхнего левого угла и нижнего правого, второе значение - верхний правый и нижний левый угол\*/

border-radius: 10px 30px;

}

#id3 {

/\* первое значение - верхний левый угол, второе значение - правый верхний и нижний левый, третье значение - правый нижний угол\*/

border-radius: 10px 20px 30px;

}

#id4 {

/\* значения задаются по часовой стрелке начиная с левого верхнего угла\*/

border-radius: 10px 20px 30px 40px;

}

</style>

### Border-image

<style>

#id {

border: 15px solid transparent;

width: 250px;

padding: 10px 20px;

/\* Отображение рисованной рамки вокруг элемента

Возможные значения: repeat, round, stretch

\*/

-o-border-image: url(border.png) 30 stretch;

-moz-border-image: url(border.png) 30 stretch;

-webkit-border-image: url(border.png) 30 stretch;

border-image: url(border.png) 30 stretch;

}

</style>

</head>

<body>

<div id="id">Элемент с рисованной рамкой.</div>

<a href="http://htmlbook.ru/css/border-image">border-image</a>

</body>

### Box-shadow

<style>

#id1, #id2, #id3 {

margin: 50px;

padding: 10px;

border: 1px solid;

width: 400px;

height: 130px;

}

#id1 {

/\*

box-shadow - добавляет тень к элементу

сдвиг по X | сдвиг по Y | цвет

\*/

-webkit-box-shadow: 4px 4px black;

box-shadow: 4px 4px black;

}

#id2 {

/\*

сдвиг по X | сдвиг по Y | радиус размытия | растяжение | цвет

\*/

-webkit-box-shadow: 0px 0px 16px 1px black;

box-shadow: 0px 0px 16px 1px black;

}

#id3 {

/\*

сдвиг по X | сдвиг по Y | радиус размытия | растяжение | цвет

inset - тень выводиться внутри элемента

\*/

-webkit-box-shadow: 0px 0px 6px 2px black inset;

box-shadow: 0px 0px 6px 2px black inset;

}

</style>

</head>

<body>

<div id="id1">

<span style="text-decoration: underline;">box-shadow:4px 4px black;</span>

</div>

<div id='id2'>

2. Также данное свойство может иметь значения указывающие радиус разброса тени (третее значение) и размер тени (четвертое значение).

<span style="text-decoration: underline;">box-shadow:0px 0px 16px 12px black;</span>

</div>

<div id='id3'>

3. С помощью значения inset Вы можете указать, что тень должна быть не внешней, а внутренней. Элементы с внутренними тенями кажутся 'вдавленными'.

<span style="text-decoration: underline;">box-shadow:0px 0px 6px 2px black inset;</span>

</div>

</body>

## Box

### Flexbox

<style type="text/css">

.grid {

/\* flexbox - указывает, что содержимое контейнер выводиться по горизонтали или по вертикали в зависимости от параметра box-orient \*/

display: -moz-box;

display: -webkit-box;

display: flexbox;

-moz-box-orient: horizontal;

-webkit-box-orient: horizontal;

box-orient: horizontal; /\* vertical \*/

}

.block {

width:400px;

background: #0094ff;

padding: 20px;

margin:10px;

}

</style>

</head>

<body>

<div class="grid">

<div class="block">Block 1</div>

<div class="block">Block 2</div>

<div class="block">Block 3</div>

<div class="block">Block 4</div>

</div>

</body>

### Box-group

<style>

.box {

display: -moz-box;

display: -webkit-box;

display: flexbox;

border: 1px solid black;

}

div{

margin: 5px;

padding:10px;

}

.class1 {

/\* Определяет порядок отображения элементов. Элементы с меньшим значением отображаются раньше тех, значения которых больше \*/

-moz-box-ordinal-group: 3;

-webkit-box-ordinal-group: 3;

box-ordinal-group: 3;

background-color: red;

}

.class2 {

-moz-box-ordinal-group: 2;

-webkit-box-ordinal-group: 2;

box-ordinal-group: 2;

background: green;

}

.class3 {

-moz-box-ordinal-group: 1;

-webkit-box-ordinal-group: 1;

box-ordinal-group: 1;

background: blue;

}

</style>

</head>

<body>

<div class="box">

<div class="class1">First</div>

<div class="class2">Second</div>

<div class="class3">Third</div>

</div>

</body>

### Box-flex

<style>

div {

display: -moz-box;

display: -webkit-box;

display: flexbox;

border: 1px solid black;

width:100%;

}

p{

padding:20px;

}

#p1 {

/\*

box-flex - определяет является ли дочерний элемент box гибким или нет

гибий элемент может увеличивать или уменьшать размер в зависимости от изменений размеров box элемента.

Значение свойства - относительное значение определяющее гибкость вложенного элемента.

\*/

-moz-box-flex: 1.0;

-webkit-box-flex: 1.0;

box-flex: 1.0;

border: 1px solid red;

}

#p2 {

-moz-box-flex: 1.0;

-webkit-box-flex: 1.0;

box-flex: 1.0;

border: 1px solid blue;

}

</style>

</head>

<body>

<div>

<p id="p1">Left Column</p>

<p id="p2">Right Column</p>

</div>

</body>

### Box-align

<style>

#grid {

height: 600px;

border: 1px solid black;

width: 300px;

display: -moz-box;

display: -webkit-box;

display: flexbox;

}

div {

width: 50px;

height: 25px;

/\*

box-align - определяет как выравниваются дочерние элементы box элемента

start — контент выравнивается по левому или верхнему краю контейнера (т.е. горизонтальный контент выравнивается по верху, а вертикальный по левому краю)

end — контент выравнивается по нижнему или по правому краю

center - контент выравнивается по центру

baseline — контент выравнивается по базовой линии текста (применяется только к горизонтальному контенту)

stretch — контент равномерно растягивается так, чтобы не было пустого пространства в блоке-->

\*/

-moz-box-align: start;

-webkit-box-align: start;

box-align: start;

/\*-moz-box-orient: vertical;

-webkit-box-orient: vertical;

box-orient: vertical;\*/

}

</style>

</head>

<body>

<div id="grid">

<div style="background-color: #00ff90;">

</div>

<div style="background-color: #0094ff">

</div>

<div style="background-color: #b200ff">

</div>

<div style="background-color: #ffd800">

</div>

</div>

### Box-direction

<style type="text/css">

.grid {

display: -moz-box;

display: -webkit-box;

display: flexbox;

-moz-box-orient: horizontal;

-webkit-box-orient: horizontal;

box-orient: horizontal; /\* vertical \*/

/\* box-direction - определяет направление, в котором дочерние элементы будут размещены в box контейнере \*/

-moz-box-direction: reverse;

-webkit-box-direction: reverse;

box-direction: reverse; /\* normal \*/

}

.block {

width:400px;

background: #0094ff;

padding: 20px;

margin:10px;

}

</style>

</head>

<body>

<div class="grid">

<div class="block">Block 1</div>

<div class="block">Block 2</div>

<div class="block">Block 3</div>

<div class="block">Block 4</div>

</div>

</body>

### Box-pack

<style type="text/css">

.grid {

width: 800px;

border: 1px solid black;

display: -moz-box;

display: -webkit-box;

display: flexbox;

-moz-box-orient: horizontal;

-webkit-box-orient: horizontal;

box-orient: horizontal;

/\*

box-pack - определяет как дочерние элементы должны размещаться в box элементе, когда он больше по размеру чем дочерние элементы

start|end|center|justify

\*/

-moz-box-pack: justify;

-webkit-box-pack: justify;

box-pack: justify;

}

.block {

width: 30px;

background: #0094ff;

padding: 20px;

margin: 10px;

}

</style>

</head>

<body>

<div class="grid">

<div class="block">Block 1</div>

<div class="block">Block 2</div>

<div class="block">Block 3</div>

<div class="block">Block 4</div>

</div>

### Box-sizing

<style>

.box1 {

/\*

Согласно спецификации CSS ширина блока складывается из ширины контента (width), значений отступов (margin), полей (padding) и границ (border).

Аналогично обстоит и с высотой блока. Свойство box-sizing позволяет изменить этот алгоритм, чтобы свойства width и height задавали размеры не контента, а размеры блока.

\*/

background: #f0f0f0;

width: 400px;

padding: 10px;

margin: 10px;

border: 10px solid #000;

/\*

content-box - width и height задают ширину и высоту контента и не включают в себя значения отступов, полей и границ.

\*/

-moz-box-sizing: content-box;

-webkit-box-sizing: content-box;

box-sizing: content-box;

}

.box2 {

background: #fc0;

width: 400px;

padding: 10px;

margin: 10px;

border: 10px solid #000;

/\*

border-box - Свойства width и height включают в себя значения полей и границ, но не отступов (margin).

\*/

-moz-box-sizing: border-box;

-webkit-box-sizing: border-box;

box-sizing: border-box;

}

</style>

</head>

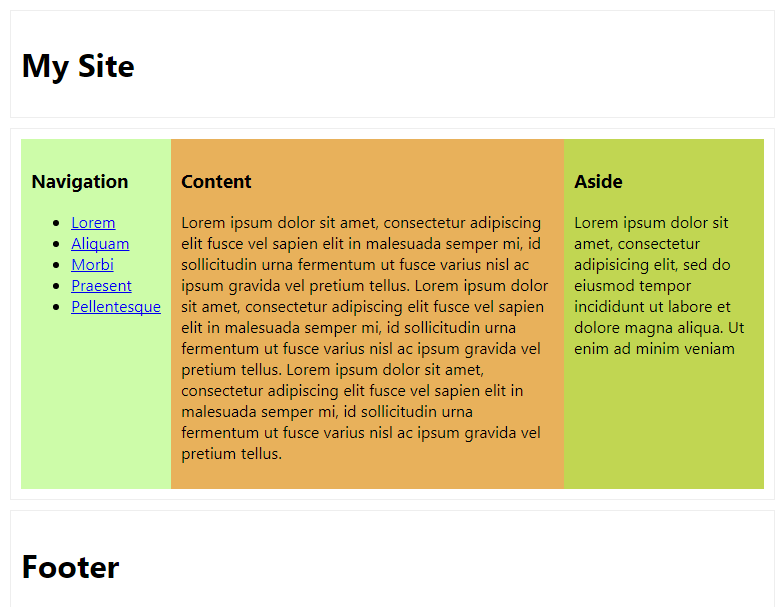
<body>

<div class="box1">content-box</div>

<div class="box2">border-box</div>

</body>

### Верстка



<!DOCTYPE html>

<html>

<head>

<title>Пример</title>

<style type="text/css">

body{

font-family:'Segoe UI', Tahoma;

}

#main, header, footer {

margin: 10px;

padding: 10px;

min-width: 640px;

border: 1px solid #eee;

}

#main {

display: -moz-box;

display: -webkit-box;

display: -ms-flexbox;

display: flexbox;

}

#content {

height: auto;

padding:10px;

-moz-box-flex: 1;

-webkit-box-flex: 1;

-ms-box-flex: 1;

box-flex: 1;

background-color: #E8B15B;

}

aside {

height: auto;

width: 180px;

padding:10px;

background-color: #C1D652;

}

nav {

padding:10px;

background-color: #CDFCA9;

}

</style>

</head>

<body>

<header>

<h1>My Site</h1>

</header>

<section id="main">

<nav>

<h1>Navigation</h1>

<ul>

<li><a href="#" title="Lorum ipsum dolor sit amet">Lorem</a></li>

<li><a href="#" title="Aliquam tincidunt mauris eu risus">Aliquam</a></li>

<li><a href="#" title="Morbi in sem quis dui placerat ornare">Morbi</a></li>

<li><a href="#" title="Praesent dapibus, neque id cursus faucibus">Praesent</a></li>

<li><a href="#" title="Pellentesque fermentum dolor">Pellentesque</a></li>

</ul>

</nav>

<section id="content">

<h1>Content</h1>

<p>

Lorem ipsum dolor sit amet, consectetur adipiscing elit fusce vel sapien elit in malesuada semper mi, id sollicitudin urna fermentum ut fusce varius nisl ac ipsum gravida vel pretium tellus.

Lorem ipsum dolor sit amet, consectetur adipiscing elit fusce vel sapien elit in malesuada semper mi, id sollicitudin urna fermentum ut fusce varius nisl ac ipsum gravida vel pretium tellus.

Lorem ipsum dolor sit amet, consectetur adipiscing elit fusce vel sapien elit in malesuada semper mi, id sollicitudin urna fermentum ut fusce varius nisl ac ipsum gravida vel pretium tellus.

</p>

</section>

<aside>

<h1>Aside</h1>

<p>

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam

</p>

</aside>

</section>

<footer>

<h1>Footer</h1>

</footer>

</body>

</html>

# Web fonts

## Fonts

### Font face

<style>

/\* @font-face - позволяет определить настройки шрифта и в случае необходимости загрузить их на машину пользователя \*/

@font-face {

font-family: heinrich; /\* Имя шрифта \*/

src: url('heinrich.ttf'), /\* Местонахождение шрифта в формате .ttf \*/

url('heinrich.eot'); /\* Местонахождение шрифта в формате .eot \*/

}

div {

font-family: heinrich;

font-size: 1.5em;

}

</style>

### Font face italic

<style type='text/css'>

/\* Подключим шрифт \*/

@font-face {

font-family: heinrich;

src: url('heinrich.ttf'), url('heinrich.eot');

}

/\* Подключим шрифт для курсивного начертания \*/

@font-face {

font-family: heinrich;

font-style: italic;

src: url('heinrichitalic.ttf'), url('heinrichitalic.eot');

}

/\* Указываем, что все параграфы должны использовать подключенные шрифты \*/

p {

font-family: heinrich;

font-size: 1.5em;

}

</style>

### Google fonts

<!--Подключение шрифта через сервис Google Fonts -->

<link href='http://fonts.googleapis.com/css?family=Kelly+Slab&subset=latin,cyrillic' rel='stylesheet' type='text/css' />

<style>

body {

font-family: 'Kelly Slab', sans-serif;

}

</style>

</head>

<body>

<p>Сервис Google Web Fonts</p>

<p>

Lorem ipsum dolor sit amet, consectetur adipiscing elit fusce vel sapien elit in malesuada semper mi, id sollicitudin urna fermentum ut fusce varius nisl ac ipsum gravida vel pretium tellus.

Lorem ipsum dolor sit amet, consectetur adipiscing elit fusce vel sapien elit in malesuada semper mi, id sollicitudin urna fermentum ut fusce varius nisl ac ipsum gravida vel pretium tellus.

</p>

<a href="http://www.google.com/fonts/"><i>http://www.google.com/fonts/</i></a>

</body>

### Text shadow

<style type='text/css'>

#id1, #id2, #id3 {

font-size: 2.5em;

font-family: Arial;

}

#id1 {

/\*

text-shadow определение тени

смещение по X | смещение по Y | цвет

\*/

-webkit-text-shadow: 3px 2px #FFAE00;

text-shadow: 3px 2px #FFAE00;

}

#id2 {

/\*

смещение по X | смещение по Y | размытие | цвет

\*/

-webkit-text-shadow: 1px 1px 10px #FFAE00;

text-shadow: 1px 1px 10px #FFAE00;

}

#id3 {

/\*

определение нескольких теней для текста

\*/

-webkit-text-shadow: 2px 2px 2px #FFAE00, 2px 2px 15px #1435AD;

text-shadow: 2px 2px 2px #FFAE00, 2px 2px 15px #1435AD;

}

</style>

</head>

<body>

<p id="id1">Пример простой тени</p>

<p id="id2">Пример тени с размытием</p>

<p id="id3">Несколько теней одновременно</p>

</body>

### Text overflow

style type='text/css'>

#id1 {

width: 450px;

border: 1px #000 solid;

-ms-text-overflow: ellipsis;

-o-text-overflow: ellipsis;

text-overflow: ellipsis; /\* Текст обрезается и к концу строки добавляется многоточие. \*/

overflow: hidden;

white-space: nowrap;

}

#id2 {

width: 450px;

border: 1px #000 solid;

-ms-text-overflow: clip;

-o-text-overflow: clip;

text-overflow: clip; /\* Текст обрезается по размеру области. \*/

overflow: hidden;

white-space: nowrap;

}

</style>

### Word wrap

<style type='text/css'>

#id1 {

width: 150px;

height: 170px;

padding: 10px;

border: 1px #000 solid;

}

#id2 {

width: 150px;

height: 170px;

padding: 10px;

border: 1px #000 solid;

/\* Перенос строк добавляется автоматически, чтобы слово поместилось в заданную ширину блока. \*/

-ms-word-wrap: break-word;

word-wrap: break-word;

}

</style>

## Column layout

### Column count

<style>

h1 {

border-bottom: 1px solid black;

}

div {

/\* количество столбцов, на которые необходимо разбить текст выбранного элемента \*/

-moz-column-count: 3;

-webkit-column-count: 3;

column-count: 3;

}

</style>

### Columns

<style>

.newspaper {

-webkit-columns: 150px auto;

-moz-columns: 150px auto;

-ms-columns: 150px auto;

columns: 150px auto; /\* определение ширины колонки и количества столбцов \*/

}

</style>

### Column gap

<style type='text/css'>

h1 {

border-bottom: 1px solid;

}

div {

-moz-column-count: 4;

-webkit-column-count: 4;

column-count: 4;

-moz-column-gap: 150px;

-webkit-column-gap: 150px;

column-gap: 150px; /\* отступ между столбцами \*/

}

</style>

### Column rule

<style type='text/css'>

h1 {

border-bottom: 1px #7F0055 solid;

}

div {

-moz-column-count: 4;

-webkit-column-count: 4;

column-count: 4;

-moz-column-rule: 2px dotted #7F0055;

-webkit-column-rule: 2px dotted #7F0055;

column-rule: 2px dotted #7F0055; /\* оформление пространства между столбцами \*/

}

</style>

### Column with

<style type='text/css'>

h1 {

border-bottom: 1px #7F0055 solid;

}

div {

-moz-column-width: 150px;

-webkit-column-width: 150px;

column-width: 150px; /\* определение ширины столбца с текстом \*/

}

</style>

## Generated content

### After

<style>

/\*

Псевдо элемент, который используется для вывода желаемого текста после содержимого элемента, к которому он добавляется.

Псевдо элемент ::after работает совместно со свойством content.

\*/

p::after {

content: ' Иван';

}

</style>

</head>

<body>

<p>любит учится</p>

<p>любит работать</p>

<p>любит путешествовать</p>

<p>любит отдыхать</p>

</body>

### Before

<style>

p {

width: 400px;

height: 200px;

border: 1px solid black;

}

/\*

Псевдо элемент ::before применяется для отображения желаемого контента до содержимого элемента, к которому он добавляется.

Работает совместно со свойством content.

\*/

p::before {

content: 'Иван ';

}

</style>

### Content counter

<style>

body {

counter-reset: myIndex; /\* начальное значение переменной myIndex\*/

}

p {

counter-increment: myIndex; /\* переменная myIndex будет увеличиваться на 1 каждый раз когда в разметке встречается параграф \*/

}

p:before {

/\* перед каждым параграфом будет установлен текст "Урок №.\*/

content: "Урок " counter(myIndex) ". ";

font-weight: bold;

}

</style>

</head>

<body>

<p>Введение</p>

<p>Семантическая разметка</p>

<p>Элементы video и audio</p>

<p>Geolocation API</p>

<p>Формы HTML5</p>

</body>

### Content increment

<style>

body {

/\* создание переменной section, значение по умолчанию 0 \*/

counter-reset: section;

}

/\* создание переменной subsection (переменная создается каждый раз, когда в разметке встречается элемент h1) \*/

h1 {

counter-reset: subsection;

}

h1::before {

counter-increment: section;

content: "Раздел " counter(section) ". ";

}

h2::before {

counter-increment: subsection;

content: counter(section) "." counter(subsection) " ";

}

</style>

</head>

<body>

<h1>HTML курсы</h1>

<h2>HTML Курсы</h2>

<h2>XHTML Курсы</h2>

<h2>CSS Курсы</h2>

<h1>Jscripting курсы</h1>

<h2>JavaScript</h2>

<h2>VBScript</h2>

<h1>XML курсы</h1>

<h2>XML</h2>

<h2>XSL</h2>

</body>

### Content attr

<style>

a {

text-decoration: none;

}

a::after {

/\*

content: attr - возвращает строку, которая является значением параметра тега указанного в скобках.

Например, a:after {content:attr(href)} добавит после ссылки её адрес, т.е. значение атрибута href.

Если указанного атрибута нет, вернется пустая строка.

\*/

content: " (" attr(href) ")";

}

</style>

</head>

<body>

<a href="http://edu.cbsystematics.com/video">Video Lessons</a> <br />

<a href="http://www.w3schools.com">W3Schools</a><br />

<a href="http://www.google.com">Google search</a><br />

</body>

### Content url

<style>

p:before {

/\* url получение изображение, если изображение не доступно - значение игнорируется \*/

content: url(homer.gif);

}

</style>

</head>

<body>

<p>My name is Homer</p>

<p>I live in Springfield</p>

</body>

### Content quotes

<style>

p:before {

/\* открывающая кавычка перед параграфом \*/

content: open-quote;

}

p:after {

/\* закрывающая кавычка после параграфа \*/

content: close-quote;

}

p.hometown:before {

/\* перед параграфом с классом hometown не устанавливать кавычку \*/

content: no-open-quote;

}

p.hometown:after {

/\* после параграфа с классом hometown не устанавливать кавычку \*/

content: no-close-quote;

}

</style>

### Blockqoute

<style>

section {

width: 700px;

height: 70px;

margin: 100px auto 0;

}

p {

margin: 10px;

padding: 0 40px;

text-align: center;

}

blockquote:before {

content: " ";

position:relative;

bottom:-25px;

background: url(quotationmark.png) -3px -3px #ddd;

border-radius: 25px;

display: block;

height: 25px;

width: 25px;

/\* устанавливаем что при смене значения свойства background-color в течении 350 ms должен произойти плавный переход значения от текущего к измененному \*/

-moz-transition: background-color 350ms;

-o-transition: background-color 350ms;

-webkit-transition: background-color 350ms;

transition: background-color 350ms;

}

blockquote:after {

content: " ";

float:right;

position:relative;

top:-25px;

border-radius: 25px;

background: url(quotationmark.png) -1px -32px #ddd;

display: block;

height: 25px;

width: 25px;

/\* устанавливаем что при смене значения свойства background-color в течении 350 ms должен произойти плавный переход значения от текущего к измененному \*/

-moz-transition: background-color 350ms;

-o-transition: background-color 350ms;

-webkit-transition: background-color 350ms;

transition: background-color 350ms;

}

blockquote:hover:after, blockquote:hover:before {

/\* запуск анимации \*/

background-color: #555;

}

</style>

</head>

<body>

<section>

<blockquote>

<p>

Есть два способа создания дизайна программы. Один из них, это сделать его настолько простым, что в нем, очевидно, не будет недостатков.

Другой способ — сделать его настолько запутанным, что в нем не будет очевидных недостатков.

<br />

— C.A. R. Hoare

</p>

</blockquote>

</section>

First latter

<style>

p {

font-family: Arial, Helvetica, sans-serif;

font-size: 16px;

}

/\*

Псевдоэлемент ::first-letter определяет стиль первого символа в тексте элемента, к которому добавляется.

\*/

p::first-letter {

font-family: "Times New Roman", Times, serif;

font-size: 32px;

color: red;

}

</style>

### First line

<style>

/\*

Псевдоэлемент ::first-line задает стиль первой строки форматированного текста.

Длина этой строки зависит от многих факторов, таких как используемый шрифт, размер окна браузера, ширина блока, языка и т.д.

В правилах стиля допустимо использовать только свойства, относящиеся к шрифту, изменению цвета текста и фона.

\*/

p::first-line {

color: red;

font-style: italic;

font-weight: bold;

}

</style>

### Selection

<style>

/\*

Псевдо элемент ::selection применяет стиль к выделенному пользователем тексту.

В правилах стилей допускается использовать следующие свойства: color, background и background-color.

\*/

p::selection {

color: #ff0;

background: #000;

}

</style>

# Pseudo classes

## Pseudo classes

### Checked

<style>

/\*

:checked псевдокласс, который применяется к элементам интерфейса (checkbox или radio) когда они находиться в выбранном состоянии

input:checked + span - найти все span, которые следуют за выбранным input

\*/

input:checked + span {

background: #fc0;

}

</style>

</head>

<body>

<p><strong>Какие курсы Вы прошли в Учебном центре</strong></p>

<p>

<input type="checkbox" name="a1" checked="checked"><span>HTML&CSS</span><br>

<input type="checkbox" name="a2"><span>JAVASCRIPT</span><br>

<input type="checkbox" name="a3"><span>jQuery</span><br>

<input type="checkbox" name="a4"><span>C# Starter</span><br>

<p>

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

</p>

</body>

### Default

<style>

/\* стиль для элемента, который является элементом по умолчанию в группе элементов \*/

input:default {

background-color: red;

}

</style>

</head>

<body>

<form>

<input type="reset">

<input type="submit">

<input type="submit" >

<input type="submit">

</form>

</body>

### Disabled

<style>

/\* disabled - псевдокласс для получения не активных элементов форм \*/

textarea:disabled {

background: url(border.png);

border: 1px solid #666;

padding: 5px;

}

</style>

</head>

<body>

<form>

<textarea cols="30" rows="5">

Это заблокированное поле, в него нельзя ничего писать</textarea>

<textarea disabled cols="30" rows="5">

Это заблокированное поле, в него нельзя ничего писать</textarea>

</form>

</body>

### Enabled

<script>

window.onload = function () {

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

check.onclick = function () {

var sbmt = document.getElementById('form1');

if (check.checked) {

// переводим элемент формы в активное состояние.

sbmt.submit.disabled = false;

}

else {

sbmt.submit.disabled = true;

}

}

}

</script>

<style>

/\* :enabled - псевдоклаcс для получения доступных элементов форм (по умолчанию элементы форм enabled) \*/

input:enabled {

background: green;

}

</style>

</head>

<body>

<form id="form1">

<p>

<input type="checkbox" name="agree" id="check">

Я согласен со всеми условиями

</p>

<p>

<!--Изначально элемент формы не активен-->

<input type="submit" name="submit" value="Отправить" disabled>

</p>

</form>

</body>

### Empty

<style>

/\* :empty - псевдокласс, который представляет пустые элементы - те, которые не содержат дочерних элементов, текста или пробелов \*/

span:empty {

background: red;

padding: 3px;

margin: 7px;

}

span:empty::before {

content: "Пустой элемент";

font-weight: bold;

}

</style>

</head>

<body>

<span></span>

<p>

Lorem ipsum dolor<span></span> sit amet, consectetur adipiscing elit fusce vel sapien elit in malesuada semper mi, id sollicitudin urna fermentum ut fusce varius nisl ac ipsum gravida vel pretium tellus.

<span></span>

</p>

</body>

### Of Type

<style>

/\* :first-of-type - получение первого элемента p в списке дочерних элементов \*/

p:first-of-type {

background: red;

}

/\* :last-of-type - получение последнего элемента p в списке дочерних элементов \*/

p:last-of-type {

background: green;

}

</style>

</head>

<body>

<h1>Заголовок</h1>

<p>Первый параграф.</p> <!--first-of-type-->

<p>Второй параграф.</p>

<p>Третий параграф.</p>

<p>Четвертый параграф.</p> <!--last-of-type-->

<div>

<p>Первый параграф.</p> <!--first-of-type-->

<p>Второй параграф.</p>

<p>Третий параграф.</p>

<p>Четвертый параграф.</p> <!--last-of-type-->

</div>

</body>

### Interminate

<style>

/\* :indeterminate - элементы в неопределенном состояние. Состояние задается через JavaScript код. \*/

:indeterminate + span {

background-color: limegreen;

}

</style>

<script>

window.onload = function () {

var inputs = document.querySelectorAll("input[type=checkbox]");

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

inputs[i].indeterminate = true;

}

}

</script>

</head>

<body>

<form>

<p>

<input type="checkbox">

<span>Option 1</span>

</p>

<p>

<input type="checkbox">

<span>Option 2</span>

</p>

<p>

<input type="checkbox">

<span>Option 2</span>

</p>

</form>

### Form validation

<style>

/\* :invalid - псевдокласс для получения элементов форм в которых допущены ошибки \*/

input:invalid {

background: #fdd;

font-weight: bold;

}

/\* :valid - пслевдокалсс для получения элеметов форм, которые корректно заполнены \*/

input:valid {

background: #dfd;

}

</style>

</head>

<body>

<form>

<p>Адрес сайта</p>

<p>

<!--В правильном формате http://edu.cbsystematics.com/-->

<input type="url" required />

</p>

<p>Адрес электронной почты</p>

<p>

<!--В правильном формате edu@cbsystematics.com-->

<input type="email" required />

</p>

<p>

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

</p>

</form>

</body>

### Child

<style>

/\* первый параграф в определенном контейнере \*/

p:first-of-type {

background-color: green;

}

/\* первый дочерний элемент любого div\*/

div :first-child {

background-color: blue;

}

/\* последний дочерний элемент любого div\*/

div :last-child {

background-color: red;

}

</style>

</head>

<body>

<h1>Заголовок</h1>

<p>Первый параграф.</p> <!--p:first-of-type-->

<p>Второй параграф.</p>

<p>Третий параграф.</p>

<p>Четвертый параграф.</p>

<div>

<span>Первая строка</span> <!--div :first-child-->

<br />

<span>Вторая строка</span>

<br />

<span>Третья строка</span>

<br />

<span>Четвертая строка</span> <!--div :last-child-->

</div>

</body>

### Not

<style>

/\* все параграфы зеленого цвета \*/

p {

color: green;

}

/\* :not() псевдокласс для элементов которые не содержат указанный селектор \*/

/\* все элементы, которые не являются параграфами - красного цвета\*/

:not(p) {

color: red;

}

</style>

</head>

<body>

<h1>Заголовк</h1>

<p>Параграф</p>

<p>Еще один параграф</p>

<div>Текст в dive</div>

<a href="#" target="\_blank">Ссылка</a>

</body>

### Not

<style>

/\* правило для всех input, которые не используют атрибут type="submit" \*/

input:not([type=submit]) {

border: 1px solid #999;

border-radius: 3px;

color: #aaa;

padding: 4px 2px;

margin: 2px;

width: 40%;

}

</style>

</head>

<body>

<form>

Поле

<input name="name1" value="value" /><br />

Поле

<input name="name2" value="value" /><br />

Поле

<input name="name3" value="value" /><br />

Поле

<input name="name4" value="value" /><br />

Поле

<input name="name5" value="value" /><br />

<input type="submit" name="name6" value="Отправить" />

</form>

</body>

### Nth-of-child

<style>

p {

color: white;

border-radius: 4px;

margin: 4px;

padding: 4px;

}

/\* Псевдокласс :nth-child используется для добавления стиля к элементам на основе нумерации в дереве элементов. \*/

/\* все нечетные параграфы \*/

p:nth-child(odd) {

background-color: green;

}

/\* все четные параграфы \*/

p:nth-child(even) {

background-color: red;

}

</style>

</head>

<body>

<h1>Заголовок</h1>

<p>0 Paragraph</p>

<p>1 Paragraph</p>

<p>2 Paragraph</p>

<p>3 Paragraph</p>

<p>4 Paragraph</p>

<p>5 Paragraph</p>

</body>

### Nth-of-child

<style>

p {

border-radius: 4px;

margin: 4px;

padding: 4px;

}

/\* получить шестой элемент \*/

p:nth-child(6) {

background-color: red;

}

/\*

n - счетчик, который постоянно увеличивается на 1 начиная со значения 0.

3 \* 0 + 1 = 1

3 \* 1 + 1 = 4

3 \* 2 + 1 = 7

3 \* 3 + 1 = 10

правило используется для элементов с порядковым номером 1, 4, 7, 10

\*/

p:nth-child(3n+1) {

background-color: yellow;

}

/\* 2, 5, 8\*/

p:nth-child(3n+2) {

background-color: orange;

}

</style>

</head>

<body>

<p>

Paragraph 1

</p>

<p>

Paragraph 2

</p>

<p>

Paragraph 3

</p>

Nth-of-type

<style>

/\* Псевдокласс :nth-of-type используется для добавления стиля к элементам указанного типа на основе нумерации в дереве элементов. \*/

/\* второй параграф в одном родительском контейнера\*/

#id1 p:nth-of-type(2) {

background: red;

}

/\* второй элемент в одном контейнере который является параграфом \*/

#id2 p:nth-child(2) {

background: green;

}

</style>

</head>

<body>

<div id="id1">

<h1>Header 1</h1>

<p>Paragraph 1</p>

<p>Paragraph 2</p> <!--p:nth-of-type(2)-->

<p>Paragraph 3</p>

</div>

<div id="id2">

<h1>Header 2</h1>

<p>Paragraph 1</p> <!--p:nth-child(2)-->

<p>Paragraph 2</p>

</div>

</body>

### Only\_of\_type

<style>

/\* Псевдокласс :only-of-type применяется к дочернему элементу указанного типа, только если он единственный у родителя \*/

p:only-of-type {

border: 5px double red;

}

</style>

</head>

<body>

<div>

<p>Paragraph 1</p>

<p>Paragraph 2</p> <!--удалите данный параграф-->

</div>

<div>

<p>Paragraph 1</p>

</div>

<div>

<span>Span 1</span>

<p>Paragraph 1</p>

</div>

</body>

### Only child

<style>

/\* правило для параграфов, которые являются единственным дочерним элементом \*/

p:only-child {

background: red;

}

</style>

</head>

<body>

<div>

<p>Первый параграф.</p>

<p>Второй параграф.</p>

</div>

<div>----------------------------</div>

<div>

<p>Первый параграф.</p>

</div>

### Optional

<style>

/\* стиль для элемента который не использует атрибут required \*/

input[type=text]:optional {

border: 2px dashed grey;

}

/\* стиль для элемента с атрибутом required\*/

input:required {

border: 2px dashed red;

}

</style>

</head>

<body>

<form>

<p>Пожалуйста, укажите ваше имя и возраст.</p>

<p>

Имя:

<input type="text" name="name">

</p>

<p>

Возраст:

<input type="number" min="18" required name="old" title="Заполните Ваш возраст">

</p>

<p>

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

</p>

</form>

</body>

### Required

<style>

p > label{

width:100px;

float:left;

margin:3px 0px 0px 0px

}

/\* label для элемента с атрибутом required должен заканчиваться символом '\*' \*/

input:required + label:after {

content: " \*";

}

</style>

</head>

<body>

<form>

<p>

<!--

Для того что бы создать селектор для label которые находиться перед элементом required в разметке label

размещаем за input для использования селектора E + F но с помощью стилей перемещаем label перед input

-->

<input type="text" required="required" />

<label>Login</label>

</p>

<p>

<input type="text" required="required" />

<label>Password</label>

</p>

<p>

<input type="text" required="required" />

<label>Email</label>

</p>

<p>

<input type="text" />

<label>Address</label>

</p>

<p>

<input type="text" />

<label>Country</label>

</p>

</form>

### Read only

<style>

/\* элементы форм не доступные для редактирования (c атрибутом readonly="readonly")\*/

input:read-only {

opacity: 0.7;

border: 1px solid red;

}

/\* элементы форм, которые поддерживают редактирование \*/

input:read-write {

border: 2px solid green;

}

</style>

</head>

<body>

<form>

<p>

<input type="text" size="40" value="Введенный текст не может изменяться" readonly="readonly">

</p>

<p>

<input type="text" size="40" value="Этот текст может изменяться">

</p>

<p>

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

</p>

</form>

### Root

<head>

<title></title>

<style>

/\*

Псевдокласс :root определяет корневой элемент документа

В HTML этот селектор всегда соответствует элементу <html>

\*/

:root {

background: green;

}

:root :nth-child(1) {

text-decoration: underline;

color:white;

}

</style>

</head>

<body>

<h1>Header</h1>

</body>

### Value selector

<title>attr^=value</title>

<style>

/\* Значение атрибута class начинается с 'test' \*/

div[class^="test"] {

background: green;

}

</style>

</head>

<body>

<div class="first\_test">Первый div.</div>

<div class="second">Второй div.</div>

<div class="test">Третий div.</div>

<div class="test4">Четвертый div.</div>

<p class="test">Параграф.</p>

</body>

### Appearance

<style>

input[type="checkbox"] {

background: url(switch.png); /\* переключатель выключен \*/

height: 26px;

width: 32px;

cursor: pointer;

/\* отключаем стандартное отображения элемента input checkbox\*/

-webkit-appearance: none;

-moz-appearance: none;

appearance: none;

}

input[type="checkbox"]:checked {

background-position: 0 -26px; /\* переключатель включен \*/

}

/\* указываем что div должен отображаться как кнопка \*/

div {

-moz-appearance: button;

-webkit-appearance: button;

appearance: button;

width: 75px;

padding: 10px;

}

</style>

</head>

<body>

<p>Щелкните по переключателю</p>

<p>

<input type="checkbox" name="c1" />

</p>

<p>

<input type="checkbox" name="c2" checked="checked" />

</p>

<p>

<input type="checkbox" name="c3" />

</p>

<div>This is DIV</div>

<a href="http://htmlbook.ru/blog/stilnye-chekboksy-ne-dlya-vsekh">HtmlBook</a>

</body>

## Gradient

### Linear gradient

<style>

div {

float: left;

margin: 5px;

width: 300px;

height: 150px;

padding: 10px;

border: 1px #000 solid;

text-decoration: underline;

}

#id1 {

/\*

Градиент распространяется сверху вниз от белого к зеленому.

Используется два градиент стопа

\*/

background: -webkit-linear-gradient(top, white, green);

background: -moz-linear-gradient(top, white, green);

background: -ms-linear-gradient(top, white, green);

background: linear-gradient(top, white, green);

}

#id2 {

/\*

Градиент распространяется слева направо от белого к зеленому.

\*/

background: -webkit-linear-gradient(left, white, green);

background: -moz-linear-gradient(left, white, green);

background: -ms-linear-gradient(left, white, green);

background: linear-gradient(left, white, green);

}

#id3 {

/\*

Направление градиента задается указанием угла наклона.

Значение 0deg и 360deg соответствует вертикальному градиенту.

При изменение значения угла вращение градиента происходит по часовой стрелке

\*/

background: -webkit-linear-gradient(0deg, white, green);

background: -moz-linear-gradient(0deg, white, green);

background: -ms-linear-gradient(0deg, white, green);

background: linear-gradient(0deg, white, green);

}

#id4 {

background: -webkit-linear-gradient(270deg, white, green);

background: -o-linear-gradient(270deg, white, green);

background: -moz-linear-gradient(270deg, white, green);

background: -ms-linear-gradient(270deg, white, green);

background: linear-gradient(270deg, white, green);

}

</style>

</head>

<body>

<p>1. Примеры задания направлений градиентов с помощью ключевых слов: </p>

<div id="id1">linear-gradient(top, white, green)</div>

<div id="id2">linear-gradient(left, white, green)</div>

<br style="clear: both;" />

<p>2. Примеры задания направлений градиентов с помощью градусов: </p>

<div id="id3">linear-gradient(0deg, white, green)</div>

<div id="id4">linear-gradient(270deg, white, green)</div>

</body>

### Linear gradient more colors

<style type="text/css">

div {

margin: 10px;

height: 200px;

width: 600px;

border: 1px #000 solid;

float: left;

}

p {

background-color: rgba(0, 0, 0, 0.7);

color: white;

padding: 10px;

font-weight: bold;

font-family: 'Segoe UI';

}

b {

color: red;

}

#id1 {

/\*

При создание градиента с несколькими сточками перехода цвета используется пара значений - цвет позиция

background: linear-gradient(left, - направление

white 0%, - в начале градиента белый цвет

green 50%, - средина градиента - зеленый

black 100%); - конец - черный

\*/

background: -webkit-linear-gradient(left, white 0%, green 50%, black 100%);

background: -moz-linear-gradient(left, white 0%, green 50%, black 100%);

background: -ms-linear-gradient(left, white 0%, green 50%, black 100%);

background: linear-gradient(left, white 0%, green 50%, black 100%);

}

#id2 {

background: -webkit-linear-gradient(left, #8F04A8 0%, #7CE700 60%, #FFE100 100%);

background: -moz-linear-gradient(left, #8F04A8 0%, #7CE700 60%, #FFE100 100%);

background: -ms-linear-gradient(left, #8F04A8 0%, #7CE700 60%, #FFE100 100%);

background: linear-gradient(left, #8F04A8 0%, #7CE700 60%, #FFE100 100%);

}

</style>

</head>

<body>

<div id="id1">

<p>linear-gradient(left, white <b>0%</b>, green <b>50%</b>, black <b>100%</b>)</p>

</div>

<div id="id2">

<p>linear-gradient(left, #8F04A8 <b>0%</b>, #7CE700 <b>60%</b>, #FFE100 <b>100%</b>)</p>

</div>

</body>

### Radial gradient

<style>

div {

margin: 10px;

height: 300px;

width: 350px;

border: 1px #000 solid;

float: left;

}

p {

background-color: rgba(0,0,0,0.2);

color: white;

padding: 10px;

font-weight: bold;

border-bottom: 1px white solid;

font-family: 'Segoe UI';

}

b {

color: red;

}

#id1 {

/\*

radial-gradient - создание радиального градиента

\*/

background: -webkit-radial-gradient(white 20%, black 40%);

background: -o-radial-gradient(white 20%, black 40%);

background: -moz-radial-gradient(white 20%, black 40%);

background: -ms-radial-gradient(white 20%, black 40%);

background: radial-gradient(white 20%, black 40%);

}

#id2 {

/\*

возможные типы градиентов - circle, ellipse

\*/

background: -webkit-radial-gradient(ellipse, #8F04A8 0%, #5D016D 40%, black 60%);

background: -o-radial-gradient(ellipse, #8F04A8 0%, #5D016D 40%, black 60%);

background: -moz-radial-gradient(ellipse, #8F04A8 0%, #5D016D 40%, black 60%);

background: -ms-radial-gradient(ellipse, #8F04A8 0%, #5D016D 40%, black 60%);

background: radial-gradient(ellipse, #8F04A8 0%, #5D016D 40%, black 60%);

}

#id3 {

/\*

первые два параметра - координаты точки с которой распространяется градиент

\*/

background: -webkit-radial-gradient(20% 10%, circle, #8F04A8 0%, #5D016D 40%, black 60%);

background: -o-radial-gradient(20% 10%, circle, #8F04A8 0%, #5D016D 40%, black 60%);

background: -moz-radial-gradient(20% 10%, circle, #8F04A8 0%, #5D016D 40%, black 60%);

background: -ms-radial-gradient(20% 10%, circle, #8F04A8 0%, #5D016D 40%, black 60%);

background: radial-gradient(20% 10%, circle, #8F04A8 0%, #5D016D 40%, black 60%);

}

</style>

</head>

<body>

<div id="id1">

<p><b>radial-gradient</b>(white 20%, black 40%)</p>

</div>

<div id="id2">

<p>radial-gradient(<b>circle</b>, #8F04A8 0%, #5D016D 40%,black 60%)</p>

</div>

<div id="id3">

<p>radial-gradient(<b>20% 10%</b>, circle, #8F04A8 0%, #5D016D 40%, black 60%)</p>

</div>

</body>

### Repeating linear gradient

<style type="text/css">

div {

margin: 10px;

height: 200px;

width: 700px;

border: 1px #000 solid;

float: left;

}

p {

background-color: rgba(0,0,0,1);

color: white;

padding: 10px;

font-weight: bold;

font-family: 'Segoe UI';

}

b {

color: red;

}

#id1 {

background: -webkit-repeating-linear-gradient(top left, white, white 5px, black 5px, black 10px);

background: -moz-repeating-linear-gradient(top left, white, white 5px, black 5px, black 10px);

background: -ms-repeating-linear-gradient(top left, white, white 5px, black 5px, black 10px);

background: repeating-linear-gradient(top left, white, white 5px, black 5px, black 10px);

}

#id2 {

background: -webkit-repeating-radial-gradient(circle, #8F04A8 0%, #5D016D 40%, black 60%);

background: -moz-repeating-radial-gradient(circle, #8F04A8 0%, #5D016D 40%, black 60%);

background: -ms-repeating-radial-gradient(circle, #8F04A8 0%, #5D016D 40%, black 60%);

background: repeating-radial-gradient(circle, #8F04A8 0px, #5D016D 40px, black 60px);

}

</style>

</head>

<body>

<div id="id1">

<p><b>repeating-linear-gradient</b>(top left, white, white 5px, black 5px, black 10px)</p>

</div>

<div id="id2">

<p><b>repeating-radial-gradient</b>(circle, #8F04A8 0%, #5D016D 40%, black 60%)</p>

</div>

</body>