# Клиентское программирование на JavaScript

## Зачем нужен JavaScript

JavaScript — объектно-ориентированный скриптовый язык программирования. Является диалектом языка ECMAScript.   
  
JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам.   
  
Основные архитектурные черты: динамическая типизация, слабая типизация, автоматическое управление памятью, прототипное программирование, функции как объекты первого класса.   
  
На JavaScript оказали влияние многие языки, при разработке была цель сделать язык похожим на Java, но при этом лёгким для использования непрограммистами[3]. Языком JavaScript не владеет какая-либо компания или организация, что отличает его от ряда языков программирования, используемых в веб-разработке.   
  
Название «JavaScript» является зарегистрированным товарным знаком компании Sun Microsystems, Inc.

## Подключение внешнего файла JavaScript к странице

<script type="text/javascript" language="javascript" src="/js/script.js"></script>

## Введение в JavaScript

Примеры программ Hello world, Sum.

Конкатенация, массивы.

## Получение элементов страницы по id

JavaScript позволяет работать с элементы HTML страницы и изменять свойства элементов.

Для тогда, чтобы однозначно определить элемент на странице, необходимо присвоить элементу уникальный идентификатор (id).

Пример,

<div id=”myContainer”>

<input id=”firstName” type=”text” name=”first\_name” value=”” />

…

</div>

JavaScript код для получения объектов будет выглядеть следующим образом:

var myContainer = document.getElementById( ‘myContainer’ );

var firstName = document.getElemebtById( ‘firstName’ );

alert( firstName.value );

## Свойства объектов

### Accessing HTML attributes using the DOM

Among the many talents of the DOM is its ability to retrieve and modify attributes inside HTML elements. Using just a small set of methods, you can intuitively get the width of a DIV, change the src of an image, or even remove the background image of the document!

The DOM supplies the following 3 methods to tackle HTML attributes:

-getAttribute()  
-setAttribute()  
-removeAttribute()

These methods can be used on any element that supports attributes.

**-getAttribute()**

GetAttribute() retrieves the corresponding value of an attribute. If the attribute does not exist, an empty string is returned. For example:

<img id="myimage" src="test.gif">

<script type="text/javascript">

//returns "test.gif"

var getvalue=document.getElementById("myimage").getAttribute("src")

</script>

**-setAttribute()**

As the name implies, setAttribute() dynamically modifies the value of an element's attribute. The method takes two parameters- the name of the attribute to set, and its new value. Using the same <IMG> example above:

<script type="text/javascript">

//image src is now "another.gif"

document.getElementById("myimage").setAttribute("src","another.gif")

</script>

**-removeAttribute()**

A whole new concept, removeAttribute() allows you to remove entire HTML attributes from an element! This is particularly handy when an attribute contains multiple values which you wish to all erase:

<div id="adiv" style="width:200px;height:30px;background-color:yellow">Some Div</div>

<script type="text/javascript">

//adiv now contains no style at all

document.getElementById("adiv").removeAttribute("style")

</script>

You're out of style adiv!

<a onmouseover="document.getElementById['foo'].style.visibility='visible'" href="foo">Foo</a>

<div id="foo" style="visibility: hidden;">Bar</div>

style.display = none | block | inline-block | inline

style.position = relative | absolute | fixed

style.marginTop = ….

## Добавление элементов в контейнтер

\* parent.appendChild(element);

\* parent.removeChild(element);

### addElement JavaScript Function

function addElement() {

var ni = document.getElementById('myDiv');

var numi = document.getElementById('theValue');

var num = (document.getElementById('theValue').value -1)+ 2;

numi.value = num;

var newdiv = document.createElement('div');

var divIdName = 'my'+num+'Div';

newdiv.setAttribute('id',divIdName);

newdiv.innerHTML = 'Element Number '+num+' has been added! <a href=\'#\' onclick=\'removeElement('+divIdName+')\'>Remove the div "'+divIdName+'"</a>';

ni.appendChild(newdiv);

}

### removeElement JavaScript Function

function removeElement(divNum) {

var d = document.getElementById('myDiv');

var olddiv = document.getElementById(divNum);

d.removeChild(olddiv);

}

## События

By using JavaScript, we have the ability to create dynamic web pages. Events are actions that can be detected by JavaScript.

Every element on a web page has certain events which can trigger a JavaScript. For example, we can use the onClick event of a button element to indicate that a function will run when a user clicks on the button. We define the events in the HTML tags.

Examples of events:

* A mouse click
* A web page or an image loading
* Mousing over a hot spot on the web page
* Selecting an input field in an HTML form
* Submitting an HTML form
* A keystroke

**Note:** Events are normally used in combination with functions, and the function will not be executed before the event occurs!

For a complete reference of the events recognized by JavaScript, go to our complete [JavaScript reference](http://www.w3schools.com/jsref/default.asp).

## onLoad and onUnload

The onLoad and onUnload events are triggered when the user enters or leaves the page.

The onLoad event is often used to check the visitor's browser type and browser version, and load the proper version of the web page based on the information.

Both the onLoad and onUnload events are also often used to deal with cookies that should be set when a user enters or leaves a page. For example, you could have a popup asking for the user's name upon his first arrival to your page. The name is then stored in a cookie. Next time the visitor arrives at your page, you could have another popup saying something like: "Welcome John Doe!".

## onFocus, onBlur and onChange

The onFocus, onBlur and onChange events are often used in combination with validation of form fields.

Below is an example of how to use the onChange event. The checkEmail() function will be called whenever the user changes the content of the field:

|  |
| --- |
| <input type="text" size="30" id="email" onchange="checkEmail()"> |

## onSubmit

The onSubmit event is used to validate ALL form fields before submitting it.

Below is an example of how to use the onSubmit event. The checkForm() function will be called when the user clicks the submit button in the form. If the field values are not accepted, the submit should be cancelled. The function checkForm() returns either true or false. If it returns true the form will be submitted, otherwise the submit will be cancelled:

|  |
| --- |
| <form method="post" action="xxx.htm" onsubmit="return checkForm()"> |

## onMouseOver

The onmouseover event can be used to trigger a function when the user mouse over an HTML eleme.

## Бонус. Сессии PHP

<?php  
// page1.php

session\_name( ‘my site session’ );  
session\_start();

echo 'Welcome to page #1';  
  
$\_SESSION['favcolor'] = 'green';  
$\_SESSION['animal']   = 'cat';  
$\_SESSION['time']     = time();  
  
// Works if session cookie was accepted  
echo '<br /><a href="page2.php">page 2</a>';  
  
// Or maybe pass along the session id, if needed  
echo '<br /><a href="page2.php?' . SID . '">page 2</a>';  
?>

|  |  |  |
| --- | --- | --- |
| **Event handler** | **Applies to:** | **Triggered when:** |
| **onAbort** | Image | The loading of the image is cancelled. |
| **onBlur** | Button, Checkbox, FileUpload, Layer, Password, Radio, Reset, Select, Submit, Text, TextArea, Window | The object in question loses focus (e.g. by clicking outside it or pressing the TAB key). |
| **onChange** | FileUpload, Select, Text, TextArea | The data in the form element is changed by the user. |
| **onClick** | Button, Document, Checkbox, Link, Radio, Reset, Submit | The object is clicked on. |
| **onDblClick** | Document, Link | The object is double-clicked on. |
| **onDragDrop** | Window | An icon is dragged and dropped into the browser. |
| **onError** | Image, Window | A JavaScript error occurs. |
| **onFocus** | Button, Checkbox, FileUpload, Layer, Password, Radio, Reset, Select, Submit, Text, TextArea, Window | The object in question gains focus (e.g. by clicking on it or pressing the TAB key). |
| **onKeyDown** | Document, Image, Link, TextArea | The user presses a key. |
| **onKeyPress** | Document, Image, Link, TextArea | The user presses or holds down a key. |
| **onKeyUp** | Document, Image, Link, TextArea | The user releases a key. |
| **onLoad** | Image, Window | The whole page has finished loading. |
| **onMouseDown** | Button, Document, Link | The user presses a mouse button. |
| **onMouseMove** | None | The user moves the mouse. |
| **onMouseOut** | Image, Link | The user moves the mouse away from the object. |
| **onMouseOver** | Image, Link | The user moves the mouse over the object. |
| **onMouseUp** | Button, Document, Link | The user releases a mouse button. |
| **onMove** | Window | The user moves the browser window or frame. |
| **onReset** | Form | The user clicks the form's Reset button. |
| **onResize** | Window | The user resizes the browser window or frame. |
| **onSelect** | Text, Textarea | The user selects text within the field. |
| **onSubmit** | Form | The user clicks the form's Submit button. |
| **onUnload** | Window | The user leaves the page. |