



# جاوا اسکریپت

دیوید فلاناگان

ترجمه‌ی دکتر قاسم کیانی مقدم

**Edited by Foxit PDF Editor  
Copyright (c) by Foxit Corporation, 2003 - 2010  
For Evaluation Only.**

# جاوا اسکریپت

دیوید فلاناگان

ترجمه‌ی دکتر قاسم کیانی مقدم

## فهرست

۵.....	زبان جاوا اسکریپت.....	۱
۲۷.....	جاوا اسکریپت سمت مشتری .....	۲
۵۳.....	مرجع جاوا اسکریپت .....	۳
۱۴۴ .....	واژه‌نامه.....	۴

# ۱ زبان جاوا اسکریپت

جاوا اسکریپت یک زبان اسکریپتنویسی سبک وزن مبتنی بر اشیا است که می‌توان آن را در صفحات HTML جا داد. در این کتاب ابتدا جاوا اسکریپت هسته را بررسی می‌کنیم، و سپس مطالبی درباره جاوا اسکریپت سمت مشتری که در مرورگر استفاده می‌شود، بیان می‌کنیم. قسمت انتهایی کتاب مرجع سریعی برای توابع جاوا اسکریپت هسته و سمت مشتری است.

## ۱/۱ دستور

دستور زبان جاوا اسکریپت مبتنی بر زبان جاوا است. خود زبان جاوا هم بر مبنای دستور زبان C و C++ بنا شده است. بنا بر این، دستور زبان جاوا اسکریپت برای برنامه‌نویسانی که قبلاً به زبان‌های جاوا، C، و یا C++ برنامه‌نویسی کرده‌اند، بسیار آشنا به نظر خواهد رسید.

## ۱/۱/۱ حروف کوچک و بزرگ

در جاوا اسکریپت حروف کوچک و بزرگ با هم متفاوت‌اند. تمام کلیدواژه‌ها با حروف کوچک نوشته می‌شوند. نام تمام متغیرها، توابع، و شناسه‌های دیگر باید از نظر حروف کوچک و بزرگ همواره یکسان نوشته شود.

## ۱/۱/۲ فضای سفید

جاوا اسکریپت از فضای سفید بین نمادها چشمپوشی می‌کند. می‌توانید از فضا، جهش، و سر سطر برای آرایش و خواناسازی متن برنامه‌ی خود استفاده کنید.

## ۱/۱/۳ ویرگول نقطه

دستورات جاوا اسکریپت با ویرگول نقطه تمام می‌شود. اما وقتی که بعد از یک دستورالعمل به سطر بعد می‌رویم، می‌توانیم ویرگول نقطه‌ی انتهایی را حذف کنیم. دقت کنید که این مسئله باعث می‌شود که از نظر اینکه در متن جاوا اسکریپت

کی می توانیم به سطر بعد برویم، محدودیت ایجاد می شود. یعنی یک قطعه از دستور العمل را اگر به تنهایی بتواند دستور العمل کاملی به شمار رود، نمی توان در یک سطر قرار داد، و ادامه‌ی آن را در سطر بعد نوشت.

## ۱/۱/۴ توضیحات

جاوا اسکریپت هر دو نوع توضیح C و C++ را پشتیبانی می کند. متن بین دو علامت /\* و \*/, هر چند سطر باشد، توضیح به شمار می رود، و جاوا اسکریپت از آن صرف نظر می کند. در ضمن، هر متنی بین علامت // و پایان سطر باشد، توضیح به شمار می رود، و جاوا اسکریپت از آن صرف نظر می کند. چند مثال:

```
// This is a single-line, C++-style comment.  
/*  
 * This is a multi-line, C-style comment.  
 * Here is the second line.  
 */  
/* Another comment. */ // This too.
```

## ۱/۱/۵ شناسه‌ها

در جاوا اسکریپت، شناسه‌ها شامل نام متغیرها، توابع، و برچسب‌ها هستند. شناسه‌ها متشکل از حروف، ارقام، و نویسه‌های \_ و \$ هستند. با این حال، اولین نویسه‌ی یک شناسه باید یک رقم باشد. موارد زیر به عنوان شناسه قابل قبول‌اند:

```
i  
my_variable_name  
v13  
$str
```

## ۱/۱/۶ کلیدواژه‌ها

کلیدواژه‌های زیر بخشی از زبان جاوا اسکریپت هستند، و برای تفسیرگر جاوا اسکریپت معنای خاصی دارند. بنا بر این، از آنها نمی توان به عنوان شناسه استفاده کرد:

break	do	if
switch	typeof	case
else	in	this
var	catch	false
instanceof	throw	void
continue	finally	new
true	while	default
for	null	try
with	delete	function
return		

در ضمن، واژه‌های زیر نیز در جاوا اسکریپت برای استفاده‌ی آینده ذخیره شده‌اند. از این کلمات نیز نمی‌توانید به عنوان شناسه استفاده کنید:

abstract	enum	int
short	boolean	export
interface	static	byte
extends	long	super
char	final	native
synchronized	class	float
package	throws	const
goto	private	transient
debugger	implements	protected
volatile	double	import
public		

به علاوه، باید از ایجاد متغیرهایی که نام آنها با خصلت‌ها و روش‌های سراسری یکسان است، اجتناب کنید: مثلاً به صفحات مرجع مربوط به `Global`، `Object`، و `Window` مراجعه کنید. در داخل تابع، از شناسه arguments به عنوان نام آوند یا متغیر محلی استفاده نکنید.

## ۱/۲ متغیرها

متغیرها با استفاده از دستورالعمل `var` اعلام یا آغازش می‌شوند:

```
var i = 1+2+3;  
var x = 3, message = 'hello world';
```

اعلام متغیرها در متن سطح بالای جاوا اسکریپت را می‌توان حذف کرد، ولی

اعلام متغیرهای محلی در بدن‌های یک تابع لازم است.

متغیرهای جاوا اسکریپت بدون نوع‌اند: مقدار یک متغیر از هر نوع داده‌ای می‌تواند باشد.

متغیرهای سراسری به صورت خصلت‌های یک شیء سراسری پیاده‌سازی شده‌اند. متغیرهای محلی در درون تابع‌ها به صورت خصلت‌های شیء آوند تابع پیاده‌سازی می‌شوند. متغیرهای سراسری در سر تا سر برنامه‌ی جاوا اسکریپت قابل دستیابی‌اند. متغیرهایی که در درون یک تابع اعلام شده‌اند، تنها در درون تابع قابل دستیابی‌اند. بر خلاف C، C++، و جاوا، جاوا اسکریپت قلمرو سطح بلوک ندارد: متغیرهایی که در درون قلابهای تابع تعریف می‌شوند، منحصر به آن قطعه نیستند، و از خارج آن هم قابل دستیابی‌اند.

## ۱/۳ نوعهای داده‌ای

جاوا اسکریپت سه نوع داده‌ای بدوی را پشتیبانی می‌کند: اعداد، مقادیر بولی، و رشته‌ها؛ و نیز سه نوع داده‌ای مرکب دارد: اشیا و آرایه‌ها، به علاوه، انواع اختصاصی اشیا نیز دارد که نماینده‌ی توابع، عبارت‌های مرتب، و تاریخ هستند.

### ۱/۳/۱ اعداد

اعداد در جاوا اسکریپت به صورت قالب ۶۴ بیتی شناور نمایش داده می‌شوند. جاوا اسکریپت بین اعداد صحیح و اعداد ممیز شناور افتراقی قابل نمی‌شود. مقادیر عددی در برنامه‌ی جاوا اسکریپت به صورت معمول ظاهر می‌شوند: رشته‌ای از ارقام، با ممیز اعشاری اختیاری و نمای اختیاری. مثال:

```
1  
3.14  
0001  
6.02e23
```

اعداد صحیح ممکن است با نماد شانزدهگانی نیز نمایش داده شوند. اعداد شانزدهگانی با  $0x$  شروع می‌شوند:

```
0xFF // The number 255 in hexadecimal
```

وقتی یک عملیات عددی سرریز می‌شود، مقدار ویژه‌ای بر می‌گرداند که معرف مثبت یا منفی بی‌نهایت است. وقتی زیرریز می‌شود، صفر بر می‌گرداند. هنگامی که عملیاتی مانند جذر گرفتن از یک عدد منفی یک مقدار خطای یا بی‌معنی بر می‌گرداند، مقدار ویژه‌ی NaN را بر می‌گرداند که معرف مقداری است که یک عدد نیست. برای آزمودن از نظر این مقدار، از تابع isNaN استفاده کنید.

شیء Number ثابت‌های عددی مفیدی را تعریف می‌کند. شیء Math تابع ریاضی مختلفی را مانند () Math.random()، Math.pow()، Math.sin()، Math.cos() تعریف می‌نماید.

### ۱/۳/۲ مقادیر بولی

نوع بولی دو مقدار ممکن دارد، که با کلیدواژه‌های true و false نشان داده می‌شوند. این مقادیر معرف درست یا غلط، روشن یا خاموش، بله یا نه، و یا هر چیز دیگری که بتوان آن را با یک بیت اطلاعات نشان داد، هستند.

### ۱/۳/۳ رشته‌ها

یک رشته‌ی جاوا اسکریپت عبارت از رشته‌ای از حروف، ارقام، و سایر

نویسه‌ها از مجموعه‌ی نویسه‌های یونیکد<sup>۱</sup> ۱۶ بیتی است.  
مقادیر رشته‌ای در جاوا اسکریپت با علامت نقل قول منفرد یا دوتایی مشخص  
می‌شوند. هر نوع نقل قول را می‌توان در داخل نوع دیگر قرار داد:

```
'testing'  
"3.14"  
'name="myform'"  
"Wouldn't you prefer O'Reilly's book?"
```

هرگاه در داخل یک مقدار رشته‌ای نویسه‌ی \ واقع شود، معنای نویسه‌ی  
بعدی را تغییر می‌دهد. این ترکیبات ویژه در جدول زیر نشان داده شده‌اند:

معنا	ترکیب
فضای وارون	\b
صفحه‌ی جدید	\f
سطر جدید	\n
سر سطر	\r
جهش	\t
آپستروف یا نقل قول منفرد که پایان دهنده‌ی رشته نیست	\'
نقل قول دوتایی که پایان دهنده‌ی رشته نیست	\"
نویسه‌ی کچ خط وارون	\।
نویسه‌ای که رمز آن در رمزگذاری لاتین ۱ با عدد شانزدهگانی dd مشخص شده است	\xdd
نویسه‌ای که رمز آن در رمزگذاری پیشکد با عدد شانزدهگانی dddd مشخص شده است	\uddddd

کلاس String روش‌های زیادی تعریف می‌کند که می‌توانید برای کار روی رشته‌ها از آنها استفاده کنید. در ضمن، این کلاس خصلت length را تعریف می‌کند که تعداد نویسه‌های موجود در رشته را مشخص می‌نماید.  
عملگر جمع (+) رشته‌ها را به هم وصل می‌کند. عملگر تساوی (==) دو رشته را بررسی می‌کند ببیند آیا دقیقاً حاوی رشته‌های نویسه‌ای یکسانی هستند یا نه. (این مقایسه بر روی مقدار رشته‌ها صورت می‌پذیرد و نه بر اساس نشانی آنها، بر خلاف آنچه برنامه‌نویسان C، C++، و یا جاوا ممکن است انتظار داشته باشند). عملگر عدم تساوی (!) عکس این کار را انجام می‌دهد. عملگرهای رابطه‌ای (<، >, =، <=، >=)، رشته‌ها را با استفاده از ترتیب الفبایی مقایسه می‌کنند.

رشته‌های جاوا اسکریپت تغییرنایزیرند، یعنی به هیچ ترتیبی نمی‌توان محتويات یک رشته را تغییر داد. روش‌هایی که روی رشته‌ها عمل می‌کنند، معمولاً نسخه‌ی تغییر یافته‌ای از رشته را بر می‌گردانند.

<sup>1</sup> Unicode.

## ۱/۳/۴ اشیا

یک شیء یک نوع داده‌ای مرکب است که دارای تعدادی خصلت است. هر خصلت یک نام و یک مقدار دارد. برای دستیابی به خصلت‌های یک شیء از عملگر استفاده می‌شود. مثلاً می‌توانید مقادیر خصلت‌های یک شیء را به صورت زیر بخوانید و بنویسید:

```
o.x = 1;  
o.y = 2;  
o.total = o.x + o.y;
```

برخلاف C، C++، و جاوا، در جاوا اسکریپت خصلت‌های شیء از قبل تعریف نشده‌اند؛ به هر شیئی می‌توان هر خصلتی را منتب کرد. اشیای جاوا اسکریپت آرایه‌های ارتباطی هستند: آنها مقادیر داده‌ای دلخواه را با نام‌های دلخواه همراه می‌کنند. بدین سبب، خصلت‌های اشیا را با نماد آرایه می‌توان بیان کرد:

```
o["x"] = 1;  
o["y"] = 2;
```

اشیا با عملگر new ساخته می‌شوند. می‌توانید به صورت زیر شیئی بدون خصلت بسازید:

```
var o = new Object();
```

اما به طور معمول از سازنده‌های از پیش تعریف شده برای ساخت اشیایی که عضو کلاسی از اشیا هستند و از قبل تعدادی خصلت و روش مفید برای آنها تعریف شده است، استفاده می‌کنید. مثلاً با جمله‌ی زیر، می‌توانید یک شیء Date ایجاد کنید که معرف زمان حاضر است:

```
var now = new Date();
```

همچنین، می‌توانید خودتان کلاس‌های اشیای و سازنده‌های ویژه بسازید. انجام این کار را بعداً نشان خواهیم داد.

در جاوا اسکریپت ۱/۲ و بعد از آن، می‌توانید اشیا را با تعریف مستقیم در متن برنامه ایجاد کنید. تعریف مستقیم شامل لیستی از مقادیر نام و مقدار است که به صورت name:value داخل قلاب واقع شده است. به عنوان مثال:

```
var o = {x:1, y:2, total:3};
```

به قسمت Object و Date در بخش مرجع مراجعه کنید.

## ۱/۳/۵ آرایه‌ها

آرایه نوعی شیء است که، به جای مقادیر نامگذاری شده، مقادیر شماره‌گذاری شده دارد. برای دسترسی به مقادیر شماره‌گذاری شده‌ی یک آرایه از

عملگر [ ] استفاده می‌شود:

```
a[0] = 1;  
a[1] = a[0] + a[0];
```

اولین عنصر یک آرایه‌ی جاوا اسکریپت، عنصر ۰ است. هر آرایه‌ی یک خصلت length دارد که تعداد عناصر موجود در آرایه را نشان می‌دهد. آخرین عنصر یک آرایه‌ی جاوا اسکریپت، عنصر length - ۱ است. عناصر آرایه می‌توانند حاوی هر نوع مقدار دیگری، از جمله اشیا و آرایه‌های دیگر، باشند، و لازم نیست که مقادیر تمام عناصر یک آرایه از یک نوع باشند.

برای ساختن یک آرایه از سازنده‌ی Array استفاده می‌کنید:

```
var a = new Array();           // Empty array  
var b = new Array(10);         // 10 elements  
var c = new Array(1, 2, 3);    // Elements 1, 2, 3
```

از جاوا اسکریپت ۱/۲ به بعد، می‌توانید آرایه‌ها را مستقیماً در متن برنامه ایجاد کنید. در این حالت، مقادیر را در درون کروشه با ویرگول از هم جدا می‌کنید. برای مثال:

```
var a = [1, 2, 3];  
var b = [1, true, [1, 2], {x:1, y:2}, "Hello"];
```

برای اطلاع از شماری از روش‌های مفید کار با آرایه‌ها به مبحث آرایه در قسمت مرجع مراجعه کنید.

## ۱۳/۶ توابع و روش‌ها

تابع قطعه‌ای از متن برنامه‌ی جاوا اسکریپت است که یک بار تعریف می‌شود، و می‌توان آن را به دفعات در برنامه فراخوانی کرد. تعریف تابع به صورت زیر است:

```
function sum(x, y)  
  return x + y;  
}
```

تابع با استفاده از عملگر () و با دادن لیستی از مقادیر آوند فراخوانی می‌شوند:

```
var total = sum(1, 2); // Total is now 3
```

در جاوا اسکریپت ۱/۱، می‌توانید با استفاده از سازنده‌ی Function() تابع ایجاد کنید:

```
var sum = new Function("x", "y", "return x+y;");
```

در جاوا اسکریپت ۱/۲ به بعد، تابع با دستور مستقیم تعریف می‌شوند، و لذا سازنده‌ی Function() منسوخ شده است:

```
var sum = function(x, y) { return x+y; }
```

هنگامی که تابعی به خصلت یک شیء اختصاص داده می‌شود، به آن یک روش آن شیء می‌گویند. در درون بدن‌های یک روش، کلیدواژه‌ی this به شیئی

اشاره می کند که تابع، خصلت آن است.  
در داخل بدنی یک تابع، آرایه `[ ]` حاوی مجموعه‌ی کامل آوندهایی است که به تابع داده شده است. به مبحث توابع و آوندها در قسمت مرجع مراجعه کنید.

## undefined و null ۱/۳/۷

در جاوا اسکریپت دو مقدار وجود دارند که از هیچ‌کدام از انواع فوق‌الذکر نیستند. کلیدواژه‌ی `null` در جاوا اسکریپت مقدار ویژه‌ای است که نشان دهنده‌ی «علم مقدار» است. اگر متغیری حاوی `null` باشد، می‌فهمید که هیچ مقدار مجازی از نوع دیگر ندارد. دیگر مقدار خاص در جاوا اسکریپت، کلیدواژه‌ی `undefined` است. این مقدار در متغیرهای آغازش نشده وجود دارد، و زمانی نیز که خصلتی را که وجود ندارد، می‌پرسید، بر گردانده می‌شود. در جاوا اسکریپت `1/5 undefined`، متغیر سراسری از پیش تعريف شده‌ای به نام `undefined` وجود دارد که حاوی مقدار ویژه‌ی تعريف نشده است. `undefined` و `null` مقاصد مشابهی دارند، و عملگر `==` آنها را مساوی می‌انگارد. اگر بخواهید بین آنها تمیز قابل شوید، باید از عملگر `===` استفاده کنید.

## ۱/۴ عبارت‌ها و عملگرها

برای تشکیل عبارت‌ها در جاوا اسکریپت، مقادیر را (که ممکن است ثابت‌ها، متغیرها، خصلت‌های شیء، عناصر آرایه، و یا فراخوانی تابع باشند) با استفاده از عملگر `=` هم ترکیب می‌کنیم. برای تقسیم کردن یک عبارت به چند عبارت فرعی و تغییر دادن ترتیب پیش‌فرض ارزیابی عبارت می‌توان از پرانتز استفاده کرد. چند مثال:

```
1+2  
total/n  
sum(o.x, a[3])++
```

جاوا اسکریپت مجموعه‌ی کاملی از عملگرها را تعريف کرده است، که اکثر آنها برای برنامه‌نویسان C، C++، و جاوا آشنا به نظر خواهد رسید. لیست عملگرها در جدول زیر آمده است، و بعد مختصرًا در باره‌ی عملگرهای استاندارد و غیراستاندارد بحث خواهیم کرد. در ستونی که جهت عملگر را نشان می‌دهد، L نشان دهنده‌ی چپ به راست، و R نشان دهنده‌ی راست به چپ است.

نقدم	جهت	عملگر	عمل مریوطه
۱۵	L	.	دستیابی به خصلت‌های شیء
	L	[ ]	دستیابی به عناصر آرایه
	L	( )	فاخوانی یک تابع
	R	new	ایجاد شیء جدید
۱۴	R	++	افزایش قلیل یا بعدی (عملگر یک عملوندی)
	R	--	کاهش قلیل یا بعدی (عملگر یک عملوندی)
	R	-	نهای یک عملوندی (منفی)
	R	+	به علاوه یک عملوندی (عدم عملگر)
	R	~	متتم بیتی (یک عملوندی)
	R	!	متتم منطقی (یک عملوندی)
	R	delete	حذف تعریف یک خصلت (یک عملوندی)
	R	(جاوا اسکریپت ۱/۲)	
	R	typeof	بر گرداندن نوع داده (یک عملوندی) (جاوا اسکریپت ۱/۱)
	R	void	بر گرداندن مقدار تعریف نشده (یک عملوندی)
	R	(جاوا اسکریپت ۱/۱)	
۱۳	L	%، /، *	ضرب، تقسیم، باقیمانده
۱۲	L	-، +	جمع، تفریق
	L	+	ادغام رشته‌ها
۱۱	L	<<	انتقال چپ عدد صحیح
	L	>>	انتقال راست، بسط علامت
	L	>>>	انتقال راست، بسط صفر
۱۰	L	<=، <	کوچکتر، کوچکتر یا مساوی
	L	>=، >	بزرگتر، بزرگتر یا مساوی
	L	instanceof	وارسی نوع شیء (جاوا اسکریپت ۱/۵)
	L	in	وارسی وجود خصلت (جاوا اسکریپت ۱/۵)
۹	L	==	آزمون تساوی
	L	!=	آزمون عدم تساوی
	L	====	آزمون یکسانی (جاوا اسکریپت ۱/۳)
	L	! ==	آزمون عدم یکسانی (جاوا اسکریپت ۱/۳)
۸	L	&	AND بیتی صحیح
۷	L	^	XOR بیتی صحیح
۶	L		OR بیتی صحیح
۵	L	&&	AND منطقی
۴	L		OR منطقی
۳	R	? :	عملگر شرطی (۳ عملوند)
۲	R	=	اختصاص
	R	, =، +=، *=، -=	اختصاص با عمل
۱	L	,	ارزیابی چندگانه

عملگرهای جاوا اسکریپت که با زبان‌های C, C++, و جاوا تفاوت دارند، شامل موارد زیر هستند:

**• == و ===**

عملگر تساوی جاوا اسکریپت، یعنی `==`، تساوی را به صورت آزاد تعريف می کند، و امکان تبدیل نوع را نیز می دهد. مثلاً، عدد 3 و رشته‌ی "3" را برابر می گیرد، `false` و 0 را برابر می گیرد، `null` و `undefined` را نیز مساوی فرض می کند. عملگر یکسانی، یعنی `===` دقیق‌تر است: فقط زمانی `true` بر می گرداند که دو عملوند یکسان باشند: یعنی نوع یکسانی داشته و مقدارشان برابر باشد. به همین ترتیب، عملگر عدم یکسانی جاوا اسکریپت، `!=`، اکیدتر از عملگر `==` است.

**• عملگرهای رشته‌ای**

در جاوا اسکریپت، عملگر `+ علاوه بر جمع زدن آوندهای عددی، آوندهای رشته‌ای را هم ادغام می کند. عملگرهای == و === رشته‌ها را بر اساس مقدار مقایسه می کنند بینند آیا دقیقاً حاوی نویسه‌های یکسانی هستند، یا نه. عملگرهای <, >, => رشته‌ها را بر اساس ترتیب الفبایی با هم مقایسه می کنند.`

**• typeof**

نوع عملوند را به صورت رشته بر می گرداند. مقادیر `"number"`, `"function"`, `"object"`, `"boolean"`, `"string"` و یا `"undefined"` بر می گرداند. اگر عملوند `null` باشد، مقدار `"object"` بر می گردد.

**• instanceof**

در صورتی مقدار `true` بر می گرداند که شیء طرف چپ با تابع سازنده‌ی طرف راست (مانند `Date` یا `RegExp`) ساخته شده باشد.

**• in**

در صورتی مقدار `true` بر می گردد که شیء طرف راست واجد (یا وارث) خصلتی که نام آن در طرف چپ آمده است، باشد.

**• delete**

یک خصلت شیء را پاک می کند. دقت کنید که این معادل آن نیست که به خصلت مقدار `null` بدهیم. اگر خصلت را نتوان حذف کرد، مقدار `false` بر می گردد، و گرنه مقدار `true` بر می گردد.

**void •**

از عملوند صرف نظر کرده و مقدار undefined برمی‌گرداند.

## ۱/۵ دستورالعمل‌های جاوا اسکریپت

یک برنامه‌ی جاوا اسکریپت، سلسله‌ای از دستورالعمل‌های جاوا اسکریپت است. اکثر دستورالعمل‌های جاوا اسکریپت مانند زبان‌های C، C++ و جاوا نوشته می‌شوند.

### ۱/۵/۱ دستورالعمل‌های عبارتی

هر عبارت جاوا اسکریپت می‌تواند به تنهایی به عنوان یک دستورالعمل تلقی شود. اختصاص مقدار، فراخوانی روش، افزایش، و کاهش دستورالعمل‌های عبارتی هستند. برای نمونه:

```
s = "hello world";  
x = Math.sqrt(4);  
x++;
```

### ۱/۵/۲ دستورالعمل‌های مرکب

هنگامی که سلسله‌ای از دستورالعمل‌ها در درون قلاب قرار داده شود، به عنوان یک دستورالعمل مرکب تلقی می‌شود. مثلاً بدنی یک حلقه‌ی while از یک دستورالعمل تشکیل می‌شود. اگر بخواهید که چندین دستورالعمل چند بار انجام شوند، از دستورالعمل مرکب استفاده کنید. این تکنیک به طور شایع برای if، for، و سایر دستورالعمل‌هایی که بعداً خواهیم گفت، مورد استفاده قرار می‌گیرد.

### ۱/۵/۳ دستورالعمل‌های خالی

دستورالعمل خالی صرفاً متشکل از یک ویرگول نقطه است. گاه می‌توان برای نوشتن حلقه‌ای با بدنی خالی از دستورالعمل خالی استفاده کرد.

### ۱/۵/۴ دستورالعمل‌های برچسب دار

از جاوا اسکریپت ۱/۲ به بعد، به هر دستورالعمل می‌توان نامی به عنوان برچسب اختصاص داد. با این کار، می‌توان در دستورالعمل‌های break و continue از این برچسب‌ها استفاده کرد.

```
label : statement
```

## ۱/۵/۵ مرجع الفبایی دستورالعمل‌ها

در بندهای زیر، تمام دستورالعمل‌های جاوا اسکریپت به ترتیب الفبایی ذکر می‌شوند.

### **break •**

دستورالعمل `break` اجرای درونی ترین حلقه و یا در مورد جاوا اسکریپت ۱/۲ به بعد، حلقه‌ی نام برده شده را متوقف می‌کند:

```
break ;  
break label ;
```

### **case •**

این یک دستورالعمل واقعی نیست، بلکه کلیدواژه‌ای برای برچسب زدن دستورالعمل‌ها در درون یک دستورالعمل `switch` در جاوا اسکریپت ۱/۲ یا بعد از آن است:

```
case constant-expression :  
    statements  
    [ break ; ]
```

به خاطر ماهیت دستورالعمل `switch`، گروهی از دستورالعمل‌ها که با برچسب `case` مشخص شده‌اند، معمولاً باید به دستورالعمل `break` ختم بشوند.

### **continue •**

دستورالعمل `continue` درونی ترین حلقه، و یا در جاوا اسکریپت ۱/۲ یا بعد از آن، حلقه‌ی نام برده شده، را دوباره آغاز می‌کند:

```
continue ;  
continue label ;
```

### **default •**

این هم مانند `case` یک دستورالعمل واقعی نیست، بلکه برچسبی است که در درون دستورالعمل `switch` در جاوا اسکریپت ۱/۲ یا بعد از آن ظاهر می‌شود:

```
default:  
    statements  
    [ break ; ]
```

### **do/while •**

حلقه‌ی `do/while` تا وقتی که مقدار یک عبارت `true` باشد، دستورالعملی را به طور مکرر اجرا می‌کند. این دستورالعمل مانند حلقه‌ی

است، جز اینکه شرط حلقه در پایان حلقه قرار می‌گیرد (و آزموده می‌شود). این بدان معنا است که بدنی حلقه دستکم یک بار اجرا می‌شود:

```
do
    statement
while (expression);
```

این دستورالعمل در جاوا اسکریپت<sup>۱</sup>/۲ اضافه شد. در نت‌اسکیپ<sup>۳</sup>، دستورالعمل continue در حلقه‌ی do/while درست کار نمی‌کند.

#### **for •**

دستورالعمل for یک حلقه‌ی سهل‌الاستفاده است که در آن عبارت‌های آغازش و افزایش با عبارت شرط حلقه ترکیب می‌شوند:

```
for (initialize ; test ; update)
    statement
```

حلقه‌ی for تا وقتی که عبارت test مقدار true داشته باشد، statement را اجرا می‌کند. این حلقه عبارت initialize را یک بار قبل از شروع حلقه اجرا می‌کند، و در پایان هر دور تکرار، عبارت update را ارزیابی می‌نماید.

#### **for/in •**

این عبارت روی خصلت‌های شیء تعیین شده حلقه می‌زند:

```
for (variable in object)
    statement
```

حلقه‌ی for/in برای هر خصلت یک شیء، دستورالعمل را یک بار انجام می‌دهد. قبل از هر بار تکرار، مقدار خصلت را به متغیر تعیین شده اختصاص می‌دهد. بعضی از خواص شیء‌های از پیش تعریف شده‌ی جاوا اسکریپت به وسیله‌ی این حلقه احصا نمی‌شود، ولی خصلت‌های تعیین شده توسط کاربر همیشه بر شماری می‌شوند.

#### **function •**

این دستورالعمل تابعی را در برنامه‌ی جاوا اسکریپت تعریف می‌کند:

```
function funcname (args) {
    statements
}
```

دستورالعمل فوق تابعی به نام funcname تعریف می‌کند که بدنی آن متشکل از دستورالعمل‌های ارائه شده است، و آوندهای آن با args

<sup>1</sup> Netscape.

مشخص شده است. `args` لیستی مشکل از صفر یا چند نام آوند است که با ویرگول از هم جدا شده‌اند. در بدنه‌ی تابع می‌توان از این آوندها برای اشاره به پارامترهایی که به تابع داده می‌شود، استفاده کرد.

#### **if/else •**

دستورالعمل `if` در صورتی که عبارت داده شده `true` باشد، دستورالعمل مربوطه را اجرا می‌کند:

```
if (expression)
    statement
```

اگر بند `else` هم وجود داشته باشد، دستورالعمل در صورت بودن عبارت، دستورالعمل دیگری را اجرا می‌کند:

```
if (expression)
    statement
else
    statement2
```

هر بند `else` را می‌توان دوباره با یک دستورالعمل `if/else` ادغام کرد، تا دستورالعمل `else if` به دست آید:

```
if (expression)
    statement
else if (expression2)
    statement2
else
    statement3
```

#### **return •**

این دستورالعمل باعث می‌شود که تابعی که فعلاً در حال اجرا است، متوقف شود و به فراخواننده‌ی خود باز گردد. اگر به دنبال آن عبارتی قرار گرفته باشد، مقدار آن عبارت به عنوان مقدار بازگشته تابع برگردانده می‌شود:

```
return ;
return expression ;
```

#### **switch •**

دستورالعمل `switch` یک انشعب چندشاخه‌ای است. این دستورالعمل ابتدا عبارتی را ارزیابی می‌کند، و بعد به دستورالعملی که با برچسب `case` مشخص شده و منطبق بر مقدار به دست آمده است، می‌رود. اگر مقدار هیچکدام از دستورالعمل‌های مشخص شده با `case` مطابقت با مقدار مورد نظر نداشته باشد، دستورالعمل `switch` در صورت وجود به دستورالعملی که با برچسب `default` مشخص شده است، می‌رود:

```
switch (expression) {
    case constant-expression: statements
    [ case constant-expression: statements ]
    [ ... ]
    default: statements
}
```

هر مجموعه از دستورالعمل های داخل دستورالعمل `switch` معمولاً با یک دستورالعمل `return` یا `break` تمام می شود، تا برنامه از یک مورد وارد مورد دیگر نشود.

#### **throw •**

دستورالعمل `throw` نشان دهنده خطأ است، و یا یک استثنای برابر می انگیزد. این امر موجب می شود که کنترل برنامه بلافصله به نزدیک ترین `try/catch/finally` استثنای منتقل شود (به دستورالعمل `throw` بر اساس ECMA ۳ تعریف شده و مراجعه کنید). دستورالعمل `throw` در جاوا اسکریپت ۱/۵ پیاده سازی شده است. دستور آن به صورت زیر است:

```
throw expression ;
```

این عبارت `expression` ممکن است به مقداری از هر نوع ارزیابی شود. (به مبحث خطأ در قسمت مرجع مراجعه کنید).

#### **try/catch/finally •**

دستورالعمل `try/catch/finally`، ساز و کار پردازش استثنایها در جاوا اسکریپت را تشکیل می دهد. این دستورالعمل در ECMA ۳ تعریف شده و در جاوا اسکریپت ۱/۵ پیاده سازی شده است. دستور آن به صورت زیر است:

```
try {
    statements
}
catch (argument) {
    statements
}
finally {
    statements
}
```

بند `try` در این دستورالعمل، قطعه ای از متن را مشخص می کند که استثنایها و خطاهای آن باید مورد پردازش قرار گیرند. اگر خطایی در برنامه صورت پذیرد، و یا استثنایی از درون قطعه `try` بر انگیخته شود، کنترل به دستورالعمل های پردازش استثنای `catch` منتقل می شود.

این بند شامل یک آوند منفرد و یا یک متغیر محلی است؛ مقداری که توسط استثنا ارائه می‌شود، به این متغیر محلی اختصاص داده می‌شود، تا اینکه بعداً دستورالعمل‌های بند `catch` بتوانند از آن استفاده کنند. بند `finally` حاوی دستورالعمل‌هایی است که صرف نظر از اینکه استثنا بر انگیخته شود یا نه، بعد از بندهای `try` یا `catch` اجرا می‌شوند. بندهای `finally` یا `catch` انتخابی‌اند، ولی هر دوی آنها را نمی‌توان حذف کرد.

#### **var •**

دستورالعمل `var` یک یا چند متغیر را اعلام کرده و به طور اختیاری آغازش می‌کند. اعلام متغیر در متن سطح بالا اختیاری است، ولی در داخل بدنی توابع باید متغیرهای محلی را اعلام کرد.

```
var name [ = value ] [ , name2 [ = value2 ] [ . . . ] ;
```

#### **while •**

دستورالعمل `while` یک حلقه‌ی اساسی است. این حلقه تا زمانی که عبارت داده شده `true` باشد، دستورالعمل درون خود را اجرا می‌کند:

```
while (expression)
    statement;
```

#### **with •**

این دستورالعمل، شئی را به زنجیره‌ی قلمرو اضافه می‌کند، به طوری که یک دستورالعمل در زمینه‌ی آن شیء اجرا می‌شود:

```
with (object)
    statement;
```

دستورالعمل `with` اثرات جانبی پیچیده‌ای دارد که به آسانی به ذهن تبادر نمی‌کند؛ قویاً توصیه می‌کنیم که از آن کمتر استفاده شود.

## ۱/۶ جاوا اسکریپت شیء‌گرا

اشیای جاوا اسکریپت آرایه‌های ارتباطی هستند که مقادیری را با نام‌های خصلت‌ها ارتباط می‌دهند. جاوا اسکریپت ساز و کار اشتقاچی ساده‌ای دارد و می‌توانید برای استفاده در برنامه‌های خود کلاس‌های جدیدی از اشیا تعریف کنید. برای تعریف یک کلاس جدید، ابتدا باید یک تابع سازنده بنویسید. سازنده مثل هر تابع دیگری است، جز اینکه با عملگر `new` فراخوانده می‌شود، و برای آغازش شیء جدید و

اشاره به آن از کلیدواژه this استفاده می‌کند. مثلاً در اینجا سازنده‌ای است می‌بینید که اشیایی از کلاس جدیدی به نام Point می‌سازد:

```
function Point(x, y) { // Constructor for Point
    this.x = x; // Initialize X coordinate
    this.y = y; // Initialize Y coordinate
}
```

هر تابع جاوا اسکریپت که به عنوان سازنده مورد استفاده قرار می‌گیرد، خصلتی به نام prototype دارد. این خصلت به شیء خاصی اشاره دارد که به عنوان سرمشق برای ساخت اشیا توسط سازنده مورد استفاده قرار می‌گیرد. هر گونه خصلتی روی این شیء سرمشق تعریف کنید، برای تمام اشیایی که با سازنده ساخته می‌شوند، به ارث می‌رسد. از شیء سرمشق معمولاً برای قرار دادن روش‌ها در دسترس تمام نمونه‌های یک کلاس استفاده می‌شود. تعریف کردن روش toString می‌دهد که نمونه‌های کلاس قابلیت تبدیل به رشته را داشته باشند. برای مثال:

```
// Define function literals and assign them
// to properties of the prototype object.
Point.prototype.distanceTo = function(that) {
    var dx = this.x - that.x;
    var dy = this.y - that.y;
    return Math.sqrt(dx*dx + dy*dy);
}
Point.prototype.toString = function () {
    return '(' + this.x + ', ' + this.y + ')';
}
```

اگر می‌خواهید روش‌ها یا خصلت‌های ایستا (کلاسی) تعریف کنید، می‌توانید آنها را مستقیماً به تابع سازنده اخصاص دهید و نه به شیء سرمشق. برای مثال:

```
// Define a commonly used Point constant
Point.ORIGIN = new Point(0, 0);
```

در قطعات متنی که مشاهده کردید، یک کلاس Point ساده تعریف شد که می‌توانیم از آن در متنی مانند متن زیر استفاده کنیم:

```
// Call constructor to create a new Point object
var p = new Point(3, 4);
// Invoke a method of the object, using a static
// property as the argument.
var d = p.distanceTo(Point.ORIGIN);
// Adding the object to a string implicitly
// invokes toString().
var msg = "Distance to " + p + " is " + d;
```

## ۱/۷ عبارت‌های مرتب

جاوا اسکریپت برای تطابق الگو قابلیت استفاده از عبارت‌های مرتب دارد و

دستور آن همانند زبان برنامه‌نویسی پرل<sup>۱</sup> است. جاوا اسکرپت ۱/۲ قابلیت پشتیانی از عبارت‌های مرتب پرل ۴ دارد، و در جاوا اسکرپت ۱/۵ خصوصیات بیشتری که مربوط به عبارت‌های مرتب در پرل ۵ است، به آن اضافه شده است. یک عبارت مرتب در جاوا اسکرپت به صورت سلسله‌ای از نویسه‌ها در داخل نویسه‌های کج خط (/) مشخص می‌شود، که به طور اختیاری ممکن است به دنبال آن نویسه‌های شاخص m (جستجوی سراسری)، n (جستجوی غیرحساس به حروف کوچک و بزرگ)، و g (حال چندسطری؛ این از اضافات جاوا اسکرپت ۱/۵ است)، قرار گرفته باشد. علاوه بر این دستور تعریف لفظی، می‌توان با استفاده از سازنده‌ی () RegExp، اشیاء RegExp ساخت. این سازنده، نویسه‌های الگو و شاخص را به صورت آوندهای رشته‌ای، بدون نویسه‌ی کج خط، می‌گیرد.

توضیح کامل دستور عبارت‌های مرتب خارج از حوصله‌ی این کتاب است، ولی جدول‌های موجود در قسمت‌های زیر، خلاصه‌ای از دستور را بیان می‌کنند.

## ۱/۷/۱ نویسه‌های لفظی

حروف، اعداد، و اکثر نویسه‌های دیگر در یک عبارت مرتب لفظی هستند: اینها صرفاً با خودشان تطبیق داده می‌شوند. با این حال، به طوری که در قسمت‌های زیر خواهیم دید، تعدادی از نویسه‌های نقطه‌گذاری و سلسله‌های گریز (که با \ شروع می‌شوند) هستند که معانی خاصی دارند. ساده‌ترین این سلسله‌های گریز روش‌های دیگری برای نمایش نویسه‌های لفظی فراهم می‌کنند:

نویسه	معنا
\n	با مقادیر لفظی نویسه‌های سطر جدید، سر سطر، و جهش منطبق می‌شوند
\t	با مقدار لفظی نویسه‌ی نقطه‌گذاری منطبق می‌شوند، و مانع از آن می‌شوند که
\ / ، \* ، \+ ، \? ، \x ، \n ، \u	معنای ویژه یا گریز داشته باشد
\nn	نویسه با رمز شائزدهگانی
\xxxx	نویسه‌ی یونیکد با رمز شائزدهگانی
\uxxxx	

## ۱/۷/۲ کلاس‌های نویسه

در دستور عبارت‌های مرتب، از کروشه برای نشان دادن مجموعه‌های نویسه‌ها و یا کلاس‌های موجود در یک الگو استفاده می‌شود. علاوه بر این، سلسله‌های گریز نیز پاره‌ای از کلاس‌های نویسه‌ای شایع را تعریف می‌کنند، که در جدول زیر نشان داده شده‌اند:

<sup>1</sup> Perl.

نویسه	معنا
[ . . . ]	با هر نویسه‌ی منفرد در داخل کروشه منطبق می‌شود
[ ^ . . . ]	با هر نویسه‌ی منفرد غیر موجود در داخل کروشه منطبق می‌شود
.	با هر نویسه‌ی غیر از سطر جدید منطبق می‌شود
\W , \w	با هر نویسه‌ی کلمه/غیر کلمه منطبق می‌شود
\S , \s	با هر نویسه‌ی فضای سفید/غیر فضای سفید منطبق می‌شود
\D , \d	با هر رقم/غیر رقم منطبق می‌شود

### ۱/۷/۳ تکرار

جدول زیر دستور عبارت‌های مرتب را برای تعیین اینکه یک انطباق چند بار می‌تواند تکرار شود، نشان می‌دهد.

نویسه	معنا
?	جمله‌ی اختیاری؛ صفر یا یک بار منطبق می‌شود
+	یک یا چند بار با جمله‌ی قبلی منطبق می‌شود
*	صفر یا چند بار با جمله‌ی قبلی منطبق می‌شود
{n}	دقیقاً n بار با جمله‌ی قبلی منطبق می‌شود
{n, }	n بار یا بیشتر با جمله‌ی قبلی منطبق می‌شود
{n, m}	لاقل n بار ولی حداکثر m بار با جمله‌ی قبلی منطبق می‌شود

در جاوا اسکریپت ۱/۵، به دنبال هر کدام از نویسه‌های تکرار می‌توان یک علامت سؤال قرار داد تا حرصشان کمتر شود، یعنی با کمترین تکرار ممکن منطبق شوند تا حدی که الگو بتواند درست در بیاید.

### ۱/۷/۴ گروهبندی و جایگزینی

در عبارت‌های مرتب برای گروهبندی عبارت‌های فرعی، درست مانند عبارت‌های ریاضی، از پرانتز استفاده می‌شود. مثلاً، پرانتز برای اینکه یک نویسه‌ی تکرار بر تمام یک زیرعبارت اعمال شود، مفید است. در ضمن می‌توان از آنها به همراه نویسه‌ی | استفاده کرد که برای نشان دادن قابل جایگزین بودن به کار می‌رود. گروه‌های قرار گرفته در داخل پرانتز رفتار خاصی دارند: هنگامی که تطابق الگو یافت می‌شود، متنی که با هر گروه منطبق است، ذخیره می‌شود، و می‌توان بر اساس شماره‌ی گروه به آن اشاره کرد. دستور این کار در جدول زیر نشان داده شده است:

نوبه	معنا
a   b	با یا b منطبق می شود
(sub)	زیرعبارت sub به یک جمله واحد گروه‌بندی می شود، و متن منطبق با آن به خاطر سپرده می شود
(?:sub)	زیرعبارت sub به یک جمله واحد گروه‌بندی می شود، ولی متن منطبق با آن شماره گذاری یا به خاطر سپرده نمی شود (جاوا اسکریپت ۱.۵)
\n	دقیقاً با همان نویسه‌هایی که با گروه شماره‌ی n منطبق شده بودند، منطبق می شود
\$n	در رشته‌های جایگزینی، متغیر زیرعبارت شماره‌ی n منطبق شده بود، جایگزین می کند

## ۱/۷/۵ تعیین موقعیت انطباق

یک لنگر در عبارت مرتب با موقعیتی در درون رشته (مثالاً اول یا آخر آن) منطبق می شود، بدون اینکه با هیچ‌کدام از نویسه‌های آن منطبق گردد. می توان از آن برای محدود کردن یا لنگر انداختن انطباق به موقعیت خاصی از رشته استفاده کرد.

نوبه	معنا
\$, ^	انطباق در اول یا آخر رشته و یا در مورد حالت چند سطري، در اول یا آخر سطر
\B, \b	انطباق در کران کلمه یا در غیر کران کلمه
(?:p)	بررسی پیش‌بینانه: انطباق نویسه‌های بعدی با الگوی داده شده بدون گنجاندن آنها در انطباق (جاوا اسکریپت ۱.۵)
(p)?	بررسی پیش‌بینانه منعی: عدم انطباق نویسه‌های بعدی با الگوی داده شده (جاوا اسکریپت ۱.۵)

## ۱/۸ روایت‌های جاوا اسکریپت

نت‌اسکیپ چند روایت از جاوا اسکریپت را تعریف کرده است، میکروسافت هم روایت‌های کمایش متناظری به نام "JScript" عرضه کرده است، و نهاد استاندارد سازی ECMA سه روایت از جاوا اسکریپت استاندارد تحت عنوان "ECMAScript" عرضه کرده است. در بندهای زیر این روایت‌های مختلف را بررسی کرده و ارتباط آنها را با یکدیگر بررسی می کنیم. هر درایه در قسمت مرجع حاوی اطلاعاتی است که نشان می دهد آن ویژگی در کدامیک از روایت‌ها ارائه شده است.

### ۱/۰ جاوا اسکریپت

روایت اولیه‌ی زبان جاوا اسکریپت. این روایت پر از اشکال بود و امروزه تقریباً منسوخ شده است. روایت مذکور در نت‌اسکیپ ۲ پیاده‌سازی شد.

### ۱/۱ جاوا اسکریپت

شیء آرایه‌ی واقعی را ارائه کرد؛ خطاهای مهم اصلاح شده بود. در

نت اسکیپ ۳ پیاده سازی شد.

• **جاوا اسکریپت ۱/۲**

دستور العمل switch، عبارت های مرتب، و شماری از ویژگی های دیگر به آن اضافه شد. تقریباً با ECMA ۱ سازگار است، ولی برخی ناهمانگی ها در آن به چشم می خورد. در نت اسکیپ ۴ پیاده سازی شد.

• **جاوا اسکریپت ۱/۳**

ناهمانگی های جاوا اسکریپت ۱/۲ اصلاح شده است. با ECMA ۱ سازگار است. در نت اسکیپ ۴/۵ پیاده سازی شد.

• **جاوا اسکریپت ۱/۴**

فقط در محصولات سرور نت اسکیپ پیاده سازی شد.

• **جاوا اسکریپت ۱/۵**

پردازش استثنای آن اضافه شد. با ECMA ۳ سازگار است. در Mozilla<sup>۱</sup> و نت اسکیپ ۶ پیاده سازی شد.

• **جی اسکریپت ۱/۰**

تقریباً معادل جاوا اسکریپت ۱/۰ است. در نسخه های اولیه ایترنت اکسپلورر<sup>۲</sup> ۳ پیاده سازی شد.

• **جی اسکریپت ۲/۰**

تقریباً معادل جاوا اسکریپت ۱/۱ است. در نسخه های بعدی ایترنت اکسپلورر ۳ پیاده سازی شد.

• **جی اسکریپت ۳/۰**

تقریباً معادل جاوا اسکریپت ۱/۳ است. با ECMA ۱ سازگار است. در ایترنت اکسپلورر ۴ پیاده سازی شد.

• **جی اسکریپت ۴/۰**

در هیچ مرورگر شبکه ای پیاده سازی نشد.

• **جی اسکریپت ۵/۰**

قابلیت پردازش استثنای دارد؛ به طور نسبی با ECMA ۳ سازگار است. در

<sup>1</sup> Mozilla.

<sup>2</sup> Internet Explorer.

اینترنت اکسپلورر ۵ پیاده‌سازی شد.

### • جی اسکریپت ۵/۵ •

تفقیباً معادل جاوا اسکریپت ۱/۵ است. کاملاً با ECMA ۳ سازگار است.  
در اینترنت اکسپلورر ۵/۵ و ۶ پیاده‌سازی شده است.

### ۱ ECMA •

اولین ویرایش استاندارد زبان جاوا اسکریپت. ویژگی‌های اساسی  
جاوا اسکریپت ۱/۱ را استانداردسازی کرده و چند ویژگی جدید نیز  
اضافه کرد. دستورالعمل switch و پشتیبانی از عبارت‌های مرتب را  
استانداردسازی نکرد. پیاده‌سازی‌های سازگار با آن، جاوا اسکریپت ۱/۳ و  
جی اسکریپت ۳/۰ هستند.

### ۲ ECMA •

نسخه‌ی نگهدارنده‌ی استاندارد که فقط به توضیح ابهامات بسنده کرده و  
هیچ ویژگی جدیدی معرفی نکرد.

### ۳ ECMA •

دستورالعمل switch، پشتیبانی از عبارت‌های مرتب، و پردازش استشانها  
را استانداردسازی کرد. پیاده‌سازی‌های سازگار با آن، جاوا اسکریپت ۱/۵  
و جی اسکریپت ۵/۵ هستند.

## ۲ جاوا اسکریپت سمت مشتری

منظور از جاوا اسکریپت سمت مشتری، متن جاوا اسکریپتی است که در درون صفحه‌ی HTML جای داده می‌شود و به وسیله‌ی مرورگر شبکه اجرا می‌شود. علاوه بر اشیای هسته که در قسمت قبل گفته شد، متن جاوا اسکریپت سمت مشتری به چند شیء دیگر نیز دسترسی دارد که معرف مرورگر، سند نشان داده شده در مرورگر، و محتوای سند هستند. برنامه‌های جاوا اسکریپت سمت مشتری معمولاً مبتنی بر رویداد هستند، که معنای آن این است که در پاسخ به تعاملات کاربر با مرورگر و سند، رویدادپردازهای جاوا اسکریپت اجرا می‌شوند. اسکریپت‌های جاوا اسکریپت سمت مشتری آنقدر قدرتمند است که می‌تواند رخدنهای امنیتی عملهای را در مرورگر شبکه ایجاد کند. بدین علت، مرورگرهای شبکه معمولاً اعمال اسکریپت‌های سمت مشتری را محدود می‌کنند. در این قسمت ابتدا چگونگی قرار دادن متن جاوا اسکریپت در پرونده‌های HTML را بیان می‌کیم، آنگاه به ارائه‌ی اشیای جاوا اسکریپت سمت مشتری، رویدادها و رویدادپردازهای جاوا اسکریپت، و محدودیت‌های امنیتی جاوا اسکریپت می‌پردازم.

### ۲/۱ جاوا اسکریپت در HTML

متن جاوا اسکریپت را می‌توان به صورت اسکریپت، رویدادپرداز، و نشانی در پرونده‌های HTML قرار داد، که ذیلاً آینه‌ها را شرح می‌دهیم.

#### ۲/۱/۱ برگه‌ی <script>

اکثراً متن جاوا اسکریپت در پرونده‌های HTML در درون برگه‌های و <script> و </script> واقع می‌شود. برای نمونه:

```
<script>
document.write("The time is: " + new Date());
</script>
```

در جاوا اسکریپت ۱/۱ و بعد از آن، می‌توانید از صفت src برگه‌ی <script> برای مشخص کردن نشانی یک اسکریپت خارجی استفاده کنید. اسکریپت موجود در نشانی مذکور خوانده شده واجرا می‌شود. معمولاً پرونده‌های متن

جاوا اسکریپت با پسوند `.js` مشخص می‌شوند. دقت کنید که در صورت استفاده از این صفت هم برگهی `</script>` مورد نیاز است:

```
<script src="library.js"></script>
```

در HTML می‌توان اسکریپت‌ها را به زبانی غیر از جاوا اسکریپت نیز نوشت، و مثلاً برخی از مرورگرها، مانند اینترنت اکسپلورر، از زبان‌های دیگری مانند وی‌بی‌اسکریپت<sup>۱</sup> نیز پشتیوانی می‌کنند. می‌توان برای تعیین زبان یک اسکریپت از صفت `language` استفاده کرد. مقدار پیش‌فرض این صفت در تمام مرورگرها "JavaScript" است، لذا معمولاً نیازی به نوشتن آن نیست. علاوه بر این، می‌توان از مقادیری مانند "JavaScript1.3" و "JavaScript1.5" برای تعیین روایت جاوا اسکریپت استفاده کرد. مرورگرهایی که از روایت مشخص شده پشتیوانی نمی‌کنند، از اسکریپت به سادگی چشمپوشی خواهند کرد.

در حقیقت، در HTML ۴ خصلت `language` برای برگهی `<script>` وجود ندارد. در عوض، روش رسمی برای تعیین زبان به کار رفته در اسکریپت، استفاده از خصلت `type` است. برای جاوا اسکریپت، مقدار این خصلت را نوع `text/javascript` MIME

```
<script src="functions.js"
       language="JavaScript1.5"
       type="text/javascript"></script>
```

## ۲/۱/۲ رویدادپردازها

متن جاوا اسکریپت را می‌توان به عنوان مقدار یک صفت رویدادپرداز برگهی HTML نیز به کار برد. نام صفت‌های رویدادپرداز همیشه با "on" شروع می‌شود. وقتی رویداد مربوطه بروز می‌کند، متن مربوط به رویدادپرداز اجرا می‌شود. مثلاً، در متن HTML زیر، یک دکمه ایجاد می‌شود، و برای خصلت `onclick` رویدادپردازی تعریف می‌شود که زمانی که کاربر روی دکمه کلیک می‌کند، پیغام را در پنجره‌ی پیغام نمایش می‌دهد:

```
<input type="button" value="Press Me"
       onclick="alert('Hello World!');">
```

لیست سایر خصلت‌های رویدادپرداز موجود بعداً ارائه خواهد شد.

## ۲/۱/۳ نشانی‌های جاوا اسکریپت

متن جاوا اسکریپت را به صورت نشانی نیز می‌توان ذکر کرد، که برای این منظور، از پروتکل کاذب `javascript` استفاده می‌شود. محتوای اینگونه نشانی‌ها

---

<sup>۱</sup> VBScript.

با ارزیابی متن جاوا اسکریپت و تبدیل مقدار حاصله به رشته تعیین می‌شود. اگر می‌خواهید از یک نشانی جاوا اسکریپت استفاده کنید که متن جاوا اسکریپت را اجرا کند، ولی هیچ مقداری بر نگرداند تا جایگزین سند فعلی شود، از عملگر void استفاده کنید:

```
<form action="javascript:void validate()">
```

## ۲/۲ شیء پنجره

شیء Window معرف یک پنجره‌ی مرورگر شبکه است. در جاوا اسکریپت سمت مشتری، شیء پنجره شیء سراسری تعریف کننده خصلت‌ها و روش‌های سطح بالا است. بنا بر این، خصلت‌ها و روش‌های Window، خصلت‌های سراسری و توابع سراسری هستند، و می‌توانید بدون پیشوند کردن نام شیء، به آنها اشاره کنید. یکی از خصلت‌های شیء پنجره خصلت window است که دوباره به خود شیء پنجره اشاره می‌کند:

```
window           // The global Window object
window.document // The document property of the window
document        // Or omit the object prefix
```

برای لیست کامل خصلت‌ها و روش‌های شیء پنجره به مبحث مربوطه در قسمت مرجع مراجعه کنید. در قسمت‌های زیر، موارد مهم‌تر این خصلت‌ها و روش‌ها را بررسی می‌کنیم، و فنون اصلی استفاده از شیء پنجره در برنامه‌نویسی سمت مشتری را نشان می‌دهیم. دقت کنید که مهم‌ترین خصلت شیء پنجره، document است، که به شیء Document اشاره دارد که توصیف کننده سندی است که در پنجره‌ی مرورگر نمایش داده می‌شود. شیء سند بعداً در مبحث جداگانه‌ای بررسی خواهد شد.

## ۲/۲/۱ پنجره‌های گفتگوی ساده

برای نمایش پنجره‌های گفتگوی ساده به مشتری سه راه وجود دارد.  
 () به شما امکان می‌دهد که پیغامی را برای مشتری نمایش دهید،  
 alert() به شما امکان می‌دهد که سؤال بلی/خیر ساده‌ای پرسید، و  
 confirm() به شما امکان می‌دهد که از مشتری بخواهید که یک رشته‌ی یک‌سطری را وارد کنند. مثال:

```
alert("Welcome to my home page!");
if (confirm("Do you want to play?")) {
    var n = prompt("Enter your name");
}
```

## ۲/۲/۲ سطرا و ضعیت

اکثر مرورگرهای شبکه دارای یک سطر وضعیت در پایین پنجره هستند که برای نمایش مقصد پیوندها و اطلاعات دیگر به کار می‌رود. با استفاده از خصلت status می‌توانید کاری کنید که متنی در این سطر وضعیت نمایش داده شود. متنی که به این خصلت بدهید، در ناحیه وضعیت ظاهر می‌شود، تا زمانی که خود شما یا مرورگر مقدار جدیدی را جایگزین مقدار قبلی کنید. همچنین، می‌توانید با استفاده از خصلت defaultStatus متن پیش‌فرضی را مشخص کنید که مرورگر زمانی که اطلاعات دیگری برای نمایش دادن در سطر وضعیت ندارد، آن را نمایش بدهد. مثلاً، در متن HTML زیر، پیوندی را می‌بینید که متن جاوا اسکریپت موجود در رویدادپرداز آن سبب می‌شود که به جای نشانی پیوند، متن دیگری در سطر وضعیت نمایش داده شود:

```
<a href="help.html"
    onmouseover="window.status='Help'; return true;">Help</a>
```

## ۲/۲/۳ زمان‌سنجهای

در جاوا اسکریپت سمت مشتری برای اجرا متن برنامه در زمان بروز یک رویداد از رویدادپرداز استفاده می‌شود. در صورتی که بخواهید متنی پس از گذشتن زمان مشخصی بر حسب میلی ثانیه، اجرا شود، می‌توانید از زمان‌سنجه استفاده کنید. برای اینکه پس از سپری شدن یک مدت زمان معین، یک رشته‌ی جاوا اسکریپت اجرا شود، از روش () setTimeout استفاده کنید، و رشته‌ی مربوطه و زمان مورد نظر را به عنوان پارامتر به آن بدهید. اگر بخواهید که این متن به صورت مکرر اجرا شود، از روش () setInterval استفاده کنید، و رشته‌ی مربوطه و فاصله‌ی زمانی مورد نظر را به عنوان پارامتر به آن بدهید. هر دو تابع مقداری بر می‌گردانند که می‌توانید آن را به ترتیب به روش () clearInterval یا () clearTimeout بدهید تا اجرای بعدی متن لغو شود. برای مثال:

```
var count = 0;
// Update status line every second
var timer = setInterval("status=++count", 1000);
// But stop updating after 5 seconds;
setTimeout("clearInterval(timer)", 5000);
```

## ۲/۲/۴ اطلاعات سیستم

خصلت‌های navigator و screen از شیء پنجره به اشیای مرورگر و صفحه‌ی نمایش اشاره می‌کنند، که خود خصلت‌هایی را تعریف می‌کنند که حاوی

اطلاعات سیستم، از قبیل نام و روایت مرورگر شبکه، سیستم عاملی که روی آن اجرا می‌شود، و وضوح صفحه‌ی نمایش کاربر هستند. به مباحث مربوط به مرورگر و صفحه‌ی نمایش در قسمت مراجعه کنید. شیء مرورگر عموماً زمانی استفاده می‌شود که می‌خواهیم متنی را اختصاصاً برای یک مرورگر شبکه‌ی خاص و یا برای روایت خاصی از مرورگر شبکه بنویسیم:

```
if (navigator.appName == "Netscape" &&
    parseInt(navigator.appVersion) == 4) {
    // Code for Netscape 4 goes here.
}
```

## ۲/۲/۵ ناویری مرورگر

خصلت location از شیء پنجره به محتوای نوار نشانی مرورگر (جایی که نشانی را در آن تایپ می‌کنید)، اشاره می‌کند. با خواندن مقدار این خصلت می‌توان به نشانی ای که در حال حاضر نمایش داده می‌شود، دست یافت. مهم‌تر از همه، با اختصاص مقدار جدید به خصلت location، می‌توان کاری کرد که مرورگر نشانی داده شده را بخواند و آن را نمایش بدهد:

```
// In old browsers, load a different page
if (parseInt(navigator.appVersion) <= 4)
    location = "staticpage.html";
```

دقت کنید که هر اسکریپت یا رویدادپردازی که به خصلت location مربوط به پنجره‌ی خود، نشانی جدیدی بدهد، باعث خواهد شد که با خوانده شدن آن سند، خودش هم از بین برود و اجرایش متوقف شود! گرچه مقدار خصلت location را مانند یک رشته می‌توان پرسجو یا تعیین کرد، ولی این خصلت در واقع به شیئی از نوع Location اشاره می‌کند. این شیء خصلت‌هایی دارد که به شما امکان می‌دهد که قسمت‌های مختلف نشانی حاضر را بخوانید یا بنویسید:

```
// Get the substring of the URL following ?
var query = location.search.substring(1);
// Scroll to a named portion of the document
location.hash = "#top";
```

به علاوه، روش reload() باعث می‌شود که مرورگر دوباره نشانی حاضر را بار کند.

خصلت history از شیء پنجره به شیء سابقه در پنجره‌ی مرورگر اشاره می‌کند. این شیء روش‌هایی دارد که به شما امکان می‌دهد که در بین نشانی‌های مرور شده‌ی آن جلو یا عقب بروید، درست همانطور که کاربر با دکمه‌های Back و Forward این کار را انجام می‌دهد:

```
history.back();      // Go back once
```

```
history.forward(); // Go forward
history.go(-3); // Go back three times
```

## ۲/۲/۶ کنترل پنجره

شیء پنجره روش‌هایی برای حرکت دادن، تغییر اندازه، و در نور دیدن پنجره دارد، و نیز روش‌هایی برای منتقل کردن کانون صفحه کلید به یک پنجره و یا خارج کردن از آن پنجره دارد. برای مثال:

```
// Automatically scroll 10 pixels a second
setInterval("scrollBy(0, 1)", 100);
```

برای اطلاعات بیشتر در خصوص روش‌های moveBy()، moveTo()، scrollBy()، scrollTo()، resizeBy()، resizeTo()، blur()، focus()، و () به مبحث شیء پنجره در قسمت مرجع، مراجعه کنید. مهم‌تر از این روش‌ها، که روی پنجره‌های موجود عمل می‌کنند، دو روش دیگر هستند: روش open() که پنجره‌ی مرورگر جدیدی ایجاد می‌کند، و روش close() که یک پنجره‌ی ایجاد شده با اسکریپت را می‌بندد. روش open() سه آوند می‌گیرد. آوند اول نشانی نمایش داده شده در پنجره است. آوند دوم نام اختیاری پنجره است. اگر یک پنجره‌ی قبلی با آن نام وجود داشته باشد، همان پنجره دوباره مورد استفاده قرار می‌گیرد، و پنجره‌ی جدیدی ایجاد نمی‌شود. آوند سوم رشته‌ای اختیاری است که اندازه پنجره‌ی جدید و مشخصات یا ویژگی‌هایی را که باید نمایش بدهد، تعیین می‌کند. برای مثال:

```
// Open a new window
w = open("new.html", "newwin", // URL and name
          "width=400, height=300, " + // size
          "location, menubar, " + // chrome
          "resizable, scrollbars, status, toolbar");
// And close that new window
w.close();
```

دقت کنید که اکثر مرورگرها فقط به اسکریپت‌ها اجازه می‌دهند که پنجره‌هایی را که خودشان باز کرده‌اند، بینند. همچنین، به خاطر رشد اخیر پنجره‌های جهشی آگهی آزار دهنده، بعضی از مرورگرها اصلاً به اسکریپت‌ها اجازه باز کردن هیچ پنجره‌ای نمی‌دهند.

## ۲/۲/۷ پنجره‌ها و کادرهای متعدد

به طوری که قبلًا گفتیم، با استفاده از روش open() متعلق به شیء پنجره می‌توان پنجره‌های مرورگر جدیدی ساخت که هر کدام معرف یک شیء پنجره‌ی جدید هستند. پنجره‌ای که یک اسکریپت در آن اجرا می‌شود، شیء سراسری برای آن

اسکریپت است، و می‌توانید از تمام خصلت‌ها و روش‌های آن شیء به طوری که گویی به صورت سراسری تعریف شده‌اند، استفاده کنید. لیکن زمانی که اسکریپتی که در یک پنجره اجرا می‌شود، نیازمند تعامل با پنجره‌ی دیگری است، لازم است که شیء Window به صراحت ذکر گردد:

```
// Create a new window and manipulate it
var w = open("newdoc.html");
w.alert("Hello new window");
w.setInterval("scrollBy(0, 1)", 50);
```

در HTML یک پنجره می‌تواند کادرهای متعدد داشته باشد. بسیاری از طراحان شبکه از کادرها دوری می‌کنند، ولی با این حال، استفاده از آنها نسبتاً شایع است. جاوا اسکریپت با هر کادر به عنوان یک شیء پنجره‌ی جداگانه رفتار می‌کند، و اسکریپت‌های موجود در کادرهای مختلف مستقل از هم اجرا می‌شوند. خصلت frames از شیء پنجره آرایه‌ای از اشیای پنجره است، که معرف زیرکادرهای یک

پنجره هستند:

```
// Scripts in framesets refer to frames like this:
frames[0].location = "frame1.html";
frames[1].location = "frame2.html";
// With deeply nested frames, you can use:
frames[1].frames[2].location = "frame2.3.html";
// Code in a frame refers to the top-level window:
top.status = "Hello from the frame";
```

خصلت parent از شیء پنجره به کادر یا پنجره‌ای که پنجره در آن واقع شده است، اشاره دارد. خصلت top به پنجره‌ی سطح بالای مرورگر که ریشه‌ی سلسه‌های کادرها است، اشاره می‌کند. (اگر شیء پنجره خود یک پنجره‌ی سطح بالا نه یک کادر باشد، خصلت‌های top و parent و صرفًا به خود شیء پنجره اشاره می‌کنند).

هر پنجره و کادر مرورگر، قرینه‌ی متفاوتی برای اجرای جاوا اسکریپت دارد، و در هر قرینه، شیء پنجره شیء سراسری است. این بدان معنا است که هر متغیر اعلام شده یا خصلت تعریف شده تبدیل به خصلت‌های شیء پنجره‌ی مربوطه می‌شوند. به این ترتیب، اسکریپتی در یک پنجره یا کادر می‌تواند از متغیرها و توابع تعریف شده در پنجره یا کادر دیگر استفاده کند. مثلاً، شایع است که توابع را در قسمت <head> پنجره‌ی سطح بالا تعریف می‌کنند، و بعد اسکریپت‌ها و رویدادبردارهای موجود در کادرهای درونی با استفاده از خصلت top به آن توابع دست پیدا می‌کنند:

```
// Code in a frame calls code in the top-level window.
top.stop_scrolling();
```

## ۲/۳ شیء سند

هر شیء پنجره یک خصلت document دارد که به شیئی از نوع سند اشاره می‌کند. شیء سند را به جهاتی می‌توان مهم‌تر از خود شیء پنجره دانست: در حالی که پنجره معرف پنجره‌ی مرورگر است، سند معرف سندی HTML است که در آن پنجره نمایش داده می‌شود. شیء سند خصلت‌های مختلفی دارد که به اشیایی اشاره می‌کنند که امکان دستیابی و تغییر محتویات سند را فراهم می‌نمایند. برای دستیابی به محتویات سند از روشی به نام مدل شیء سند، یا DOM، استفاده می‌شود، و هم‌اینک چند نوع DOM وجود دارند:

### • DOM قدیمی

نوع اولیه‌ی مدل شیء سند به موازات روایت‌های ابتدایی زبان جاوا اسکریپت توسعه یافت. این نوع مورد حمایت همه‌ی مرورگرها است، ولی تنها امکان دستیابی به برخی قسمت‌های کلیدی سند را، از قبیل فرم‌ها، عناصر فرم، و تصاویر، فراهم می‌کند.

### • W3C DOM

این مدل شیء سند امکان دستیابی و تغییر تمام محتوای سند را فراهم می‌کند، و به وسیله‌ی کنسرسیوم شبکه‌ی جهانی<sup>۱</sup> (W3C) استاندارد سازی شده است. این نوع لاقل به طور نسبی به وسیله‌ی نت‌اسکیپ ۶ و بالاتر، اینترنت اکسلپورر ۵ و بالاتر، و سایر مرورگرهای مدرن پشتیبانی می‌شود. W3C DOM به طور دقیق با IE4 DOM سازگار نیست، ولی بسیاری از ویژگی‌های قدیمی DOM اولیه را استاندارد سازی کرده است. این کتاب ویژگی‌های هسته‌ای استاندارد را پوشش می‌دهد، و زیرمجموعه‌ی ساده شده‌ای از DOM را که برای برنامه‌نویسان جاوا اسکریپت که با سندهای HTML کار می‌کنند، ارائه می‌نماید. برای توضیحات کامل به کتاب‌های جامع‌تر در باره‌ی جاوا اسکریپت مراجعه کنید.

### • IE4 DOM

ویرایش ۴ اینترنت اکسلپورر میکروسافت، DOM قدیمی را با ویژگی‌های جدیدی برای دستیابی و امکان تغییر تمام محتوای یک سند گسترش داد. این ویژگی‌ها هرگز استاندارد سازی نشدند، ولی بعضی از آنها در

---

<sup>۱</sup> World Wide Web Consortium.

مرورگرهای غیر میکروسافت نیز پشتیبانی می‌شوند.

در قسمت‌های زیر هر کدام از این DOM‌ها را با تفصیل بیشتر بررسی می‌کنیم، و توضیح خواهیم داد که چگونه می‌توانید از آنها برای دستیابی و تغییر دادن محتوای سند استفاده کنید.

## ۲/۴ DOM قدیمی

اولیه جاوا اسکریپت سمت مشتری از طریق خصلت‌های شیء سند امکان دستیابی به محتویات سند را فراهم می‌کند. خصلت‌های فقط-خواندنی متعددی، از قبیل title، URL، lastModified، و اطلاعاتی را در باره‌ی سند به عنوان یک کل در اختیار ما قرار می‌دهند. برای اطلاعات بیشتر در باره‌ی این خصلت‌های شیء سند به مباحث مربوطه در قسمت مرجع مراجعه نمایید. سایر خصلت‌ها، آرایه‌هایی هستند که به انواع خاص محتوای سند اشاره می‌کنند:

### forms[] •

آرایه‌ای از اشیای فرم که معرف فرم‌های موجود در سند است.

### images[] •

آرایه‌ای از اشیای تصویر که معرف تصویرهای موجود در سند است.

### applets[] •

آرایه‌ای از اشیا که معرف برنامک‌های جاوای موجود در سند است. در واقع از جاوا اسکریپت می‌توان برای اسکریپتنویسی جاوا و کترل این برنامک‌ها استفاده کرد، ولی چگونگی انجام این کار فراتر از حد این کتاب است.

### links[] •

آرایه‌ای از اشیای پیوند که معرف پیوندهای موجود در سند است.

### anchors[] •

آرایه‌ای از اشیای لنگر که معرف لنگرهای موجود در سند است (لنگر به مکان‌های نامگذاری شده درون سند می‌گویند که با استفاده از صفت name از برگه‌ی HTML <a> ایجاد می‌شود).

این آرایه‌ها حاوی اشیای مربوطه به ترتیب بروز در سند هستند. بنا بر این، اولین فرم در یک سند document.forms[0] است، و سومین تصویر

[2] document.images[2].name به آنها با استفاده از صفت name است: سند، دادن نام به آنها برای اشاره فرم‌ها، تصاویر، و برنامک‌های هنگامی که به این طریق به یک فرم، تصویر، یا برنامک نامی داده می‌شود، می‌توان با استفاده از آن نام آن را در آرایه پیدا کنید، و یا اینکه مستقیماً آن را به صورت خصلتی از خود سند مورد استفاده قرار دهید:

```
document.forms["address"] // A named form
document.address // The same thing
```

شیء فرم خصوصاً جالب است. این شیء یک آرایه‌ی elements[] دارد که حاوی اشیایی است که عناصر فرم را به ترتیب ظهور در فرم نشان می‌دهند. برای جزئیات بیشتر در مورد این عناصر فرم به مباحث Select، Input، و Textarea در قسمت مرجع مراجعه فرمایید.

آرایه‌ی forms[] یک فرم تا حد زیادی مانند آرایه‌ی elements[] یک سند عمل می‌کند؛ این آرایه حاوی عناصر فرم به ترتیب ظهور در فرم است، ولی این هم امکان آن را به ما می‌دهد که عناصر را به نام مورد اشاره قرار دهیم. چکیده‌ی زیر را در نظر بگیرید:

```
<form name='address'><input name='street'></form>
برای اشاره به عنصر ورود متن در این فرم از چند راه می‌توانیم استفاده کنیم:
document.forms[0].elements[0]
document.address.elements['street']
document.address.street
```

DOM قدیمی هیچ راهی برای اشاره به محتوای سند غیر از فرم‌ها، عناصر فرم، تصاویر، برنامک‌ها، پیوندها، و لنگرها فراهم نمی‌کند. مثلاً، هیچ آرایه‌ای وجود ندارد که حاوی تمام برگه‌های <h1> باشد، و یا به هیچ طریقی نمی‌توان متن معمولی سند را گرفت. این نقصی است که به طوری که بعداً خواهیم دید، در DOM‌های W3C اینترنت اکسپلورر ۴ به آن پرداخته شده است. با این حال، به طوری که در قسمت‌های زیر خواهیم دید، DOM قدیمی با وجود محدودیت‌هایی که دارد، امکان آن را فراهم می‌کند که اسکریپت‌ها محتوای سند را به طور پویا تغییر دهند.

## ۲/۴/۱ سندهای ایجاد شده به طور پویا

علاوه بر خصلت‌های پیشگفته، شیء سند چندین روش مهم برای تولید پویای محتوای سند نیز دارد. با استفاده از روش write() می‌توانیم متنی را در سند در محل برگه‌ی <script> حاوی فراخوانی روش بنویسیم. برای مثال:

```
document.write("<p>Today is: " + new Date());
document.write("<p>Document updated: " +
  document.lastModified);
```

دقت کنید که متنی که به این ترتیب نوشته می‌شود، می‌تواند حاوی هر کدام از برگه‌های HTML نیز باشد؛ مرورگر بعد از اجرا کردن اسکریپت، متن نوشته شده توسط آن را تجزیه کرده و نمایش می‌دهد.

از روش `write()` در برگه‌ی `<script>` تنها در زمانی که صفحه هنوز در حال خوانده شدن است، می‌توان استفاده کرد. اگر سعی کنید که از آن در داخل رویدادپردازی استفاده کنید که بعد از بار شدن کامل سند فعل می‌شود، باعث پاک شدن سند و خود رویدادپرداز می‌شود. با این حال، می‌توانید از یک رویدادپرداز در یک پنجره یا کادر برای فعل کردن فراخوانی `document.write()` در یک پنجره‌ی دیگر استفاده کنید. اما برای این کار، باید محتویات کامل یک سند جدید را بنویسید، و یادتان باشد که وقتی کارتان تمام شد، روش `document.close()` را فراخوانی کنید:

```
var clock = open("", "", "width=400, height=30");
var d = clock.document; // Save typing below
setInterval("d.write(new Date());d.close();",
1000);
```

## ۲/۴/۲ فرم‌های پویا

به طوری که دیده‌ایم، آرایه‌ی `elements` از یک شیء فرم حاوی اشیایی متناظر با عناصر موجود در فرم است. بسیاری از این اشیا خصلت‌هایی دارند که می‌توانید از آن برای خواندن یا نوشتن مقدار نمایش داده شده در عنصر فرم استفاده کنید. این راه دیگری برای تغییر پویای محتویات سند است. مثلاً، متن زیر خصلت `value` در یک شیء `Text` را به گونه‌ای تعیین می‌کند که زمان محلی فعلی را نمایش دهد:

```
<form><input size=10></form> // An HTML form
<script> /* Display a clock in the form */
// The Text element we're working with.
var e = document.forms[0].elements[0];
// Code to display the time in that element
var s=e.value=(new Date()).toLocaleTimeString();
setInterval(s, 1000); // Run it every second
</script>
```

## ۲/۴/۳ اعتبارسنجی فرم

برگه‌ی `<form>` یک رویدادپرداز `onsubmit` دارد که زمانی که کاربر سعی می‌کند فرمی را تحویل دهد، فعال می‌شود. می‌توانید از این رویدادپرداز برای اعتبارسنجی استفاده کنید: یعنی مثلاً بررسی اینکه تمام فیلد‌های لازم در فرم پر شده‌اند. اگر رویدادپرداز `onsubmit` مقدار `false` برگرداند، فرم تحویل داده

نمی شود. برای مثال:

```
<form name="address" onsubmit="checkAddress()">
<!-- form elements go here -->
</form>
<script>
// A simple form validation function
function checkAddress() {
    var f = document.address; // The form to check
    // Loop through all elements
    for(var i = 0; i < f.elements.length; i++) {
        // Ignore all but text input elements
        if (f.elements[i].type != "text") continue;
        // Get the user's entry
        var text = f.elements[i].value;
        // If it is not filled in, alert the user
        if (text == null || text.length == 0) {
            alert("Please fill in all form fields.");
            return false;
        }
    }
}
</script>
```

## ۲/۴/۴ رواندازی تصاویر

DOM قدیمی به شما امکان ایجاد یکی از جلوه‌های ویژه‌ی شایع را می‌دهد، و آن توانایی جایگزینی پویای یک تصویر با تصویر دیگر است. از این فن عموماً برای رواندازی تصاویر استفاده می‌شود، یعنی اینکه وقتی نشانگر موشواره روی تصویر قرار می‌گیرد، تصویر عوض می‌شود. آرایه‌ی [ ] از شیء سند حاوی اشیایی images دارد که معرف تصاویر موجود در سند هستند. هر تصویر یک خصلت src دارد که نشانی تصویری را که باید نمایش داده شود، مشخص می‌کند. برای تغییر تصویر نمایش داده شده، کافی است که به این خصلت یک نشانی جدید بدهید: document.images[0].src = "newbanner.gif";

برای اینکه از این تکنیک جهت جلوه‌ی رواندازی تصاویر استفاده کنیم، باید از آن به همراه رویدادپردازهای onmouseover و onmouseout استفاده کنیم. این رویدادپردازها به ترتیب زمانی که موشواره وارد تصویر می‌شود و از آن خارج می‌شود، فعال می‌شوند. در مثال زیر، متن جاوا اسکریپت کوچکی را به همراه رویدادپردازها می‌بینید که جلوه‌ی رواندازی تصویر را ایجاد می‌کند:

```

```

هنگامی که قرار است که تصویری به صورت پویا نمایش داده شود، بهتر است که قبل آن را به حافظه‌ی نهان مرورگر بار کنیم تا موقع ظاهر شدن تأخیری

پیش نیاید. برای این منظور، می‌توانید از یک تصویر پویای خارج از صفحه استفاده کنید:

```
var i = new Image(); // Create Image object
i.src="b2.gif"; // Load, but don't display image
```

## ۲/۴/۵ کار با کوکی‌ها

خصلت cookie شیء سند خصلت ویژه‌ای است که با آن می‌توانید کوکی‌های مربوط به سند را بخوانید و بنویسید. برای همراه کردن یک کوکی گذرا با سند، کافی است که به خصلت کوکی، رشته‌ای به صورت زیر اختصاص دهید:

`name=value`

با این کار، یک کوکی با نام و مقدار مشخص شده برای این سند ساخته می‌شود. اگر بخواهید کوکی‌ای ایجاد کنید که حتی بعد از خارج شدن کاربر از مرورگر نیز ذخیره شود، باید با استفاده از رشته‌ای به صورت زیر، تاریخ انقضای آن را مشخص نمایید:

`name=value; expires=date`

تاریخ انقضا باید به همان صورتی باشد که به وسیله‌ی روش `Date.toGMTString()` بر کردانده می‌شود. اگر بخواهید که کوکی برای سندهای دیگر پایگاه اینترنتی شما نیز قابل دستیابی باشد، باید مسیر آن را نیز مشخص نمایید:

`name=value; expires=date; path=prefix`

یک سند واحد ممکن است با بیش از یک کوکی همراه باشد. برای خواندن کوکی‌های سند، کافی است که مقدار خصلت cookie را بخوانید. این رشته حاوی رشته‌های به صورت `name=value` است که با یک ویرگول نقطه و یک فضا از هم جدا شده‌اند. هنگام خواندن کوکی‌ها، هیچگاه به قسمت‌های `expires=` یا `path=` برخورد نمی‌کنید؛ در این رشته صرفاً نام و مقدار کوکی‌ها ذکر شده است. در اینجا تابعی را می‌بینید که مقدار یک کوکی با نام مشخص را در خصلت cookie پیدا می‌کند. فرض بر این است که در مقادیر کوکی‌ها هرگز نویسه‌ی ویرگول نقطه یافت نمی‌شود.

```
function getCookie(name) {
    // Split cookies into an array
    var cookies = document.cookie.split(' ; ');
    for(var i = 0; i < cookies.length; i++) {
        var c = cookies[i]; // One cookie
        var pos = c.indexOf('='); // Find = sign
        var n = c.substring(0, pos); // Get name
        if (n == name) // If it matches
            return c.substring(pos+1); // Return value
    }
    return null; // Can't find the named cookie
```

}

## ۲/۵ W3C DOM

این نوع DOM، ویژگی‌های DOM قدیمی را استاندارد می‌کند، لیکن ویژگی‌های مهم جدیدی را نیز اضافه می‌نماید. این نوع، علاوه بر آرایه‌های [ ]، forms [ ]، images [ ]، و سایر خصلت‌های شیء سند، روش‌هایی را نیز تعریف می‌کند که به اسکریپت‌ها اجازه می‌دهد که هر کدام از عناصر سند، و نه فقط عناصر خاصی مانند فرم‌ها و تصاویر، را بخوانند و یا تغییر دهند.

### ۲/۵/۱ یافتن عناصر بر اساس شناسه

هنگامی که سندی ایجاد می‌کنید که می‌خواهید عناصر خاصی از آن را با استفاده از اسکریپت تغییر دهید، می‌توانید به هر عنصر خاص یک خصلت id (شناسه) با مقدار منحصر به فرد بدهید. بعد می‌توانید برای دست یافتن به عنصر بر اساس شناسه‌ی آن، از روش getElementById () متعلق به شیء سند استفاده کنید:

```
<h1 id="title">Title</h1>
<script>
var t = document.getElementById("title");
</script>
```

### ۲/۵/۲ یافتن عناصر بر اساس نام برگه

راه دیگر برای دستیابی به عناصر سند، استفاده از نام برگه است. روش getElementsByTagName () متعلق به شیء سند، آرایه‌ای از تمام عناصر سند با آن نام را بر می‌گرداند. هر کدام از عناصر سند نیز همان روش را دارا است، و بنا بر این، می‌توانید مثلاً عناصری با برگه‌ی مورد نظر را که همگی زیرمجموعه‌ی عنصر دلخواهی هستند، را پیدا کنید:

```
// Get an array of all <ul> tags
var lists = document.getElementsByTagName("ul");
// Find the 3rd <li> tag in the second <ul>
var item = lists[1].getElementsByTagName("li")[2];
```

### ۲/۵/۳ عبور از درخت سند

W3C DOM هر سند را به صورت درختی نمایش می‌دهد. گره‌های این درخت نشان دهنده‌ی برگه‌های HTML، رشته‌های متن، و توضیحات HTML موجود در سند هستند. هر گره به وسیله‌ی یک شیء جاوا اسکریپت نمایش داده می‌شود که

مانند متن نمونه‌ی زیر، امکان حرکت کردن در درخت را برای شما فراهم می‌آورد:

```
// Look up a node in the document
var n = document.getElementById("mynode");
var p = n.parentNode; // The containing tag
var c0 = n.firstChild; // First child of n
var c1 = c0.nextSibling; // 2nd child of n
var c2 = n.childNodes[2]; // 3rd child of n
var last = n.lastChild; // last child of n
```

برای جزئیات بیشتر به قسمت مرجع مراجعه فرمایید.

خود شیء سند نیز نوعی گره است، و همان خصلت‌ها را دارا است. خصلت documentElement از شیء سند، به عنصر منفرد برگه‌ی <html> در سطح بالای هر سند HTML اشاره دارد، و خصلت body به برگه‌ی <body> اشاره می‌کند.

## ۲/۵/۴ انواع گره‌ها

هر گره موجود در درخت سند، یک خصلت nodeType دارد که نوع آن گره را تعیین می‌کند. انواع مختلف گره‌ها، زیرکلاس‌های متفاوت شیء گره هستند. بعضی از انواع گره‌ها که برای برنامه‌نویسانی که با HTML کار می‌کنند، اهمیت دارند، ذیلاً ذکر شده‌اند (انواع گره دیگری هم برای سندهای XML وجود دارند):

نوع گره	معنا
۱	عنصر: یک برگه‌ی HTML
۲	متن: متن موجود در سند
۸	توضیح: یک توضیح HTML
۹	سند: سند HTML

در گرهی که از نوع عنصر است، خصلت nodeName نام برگه‌ی مربوطه را نشان می‌دهد. در گره‌هایی که از نوع متن یا توضیح هستند، خصلت nodeValue متن سند یا متن توضیح را نشان می‌دهد. برای جزئیات بیشتر به مباحث مربوط به عنصر، متن، توضیح، و سند در قسمت مرجع مراجعه فرمایید. همچنین، برای اطلاع بیشتر درباره‌ی خصلت‌ها و روش‌های مشترک آنها به مبحث گره در قسمت مرجع مراجعه فرمایید.

## ۲/۵/۵ صفت‌های HTML

به طوری که در بالا دیدیم، برگه‌های HTML در یک درخت سند با اشیایی از نوع عنصر نمایش داده می‌شوند. در سندهای HTML، هر شیء عنصر خصلت‌هایی دارد که دقیقاً متناظر با صفت‌های برگه‌ی HTML هستند. برای مثال، برای خواندن یا

نوشتن مقدار صفت `caption` از یک برگه‌ی `<table>`، می‌توانید از خصلت `caption` شیء عنصر متناظر با آن برگه استفاده کنید. برای جزئیات بیشتر به مبحث مربوط به عنصر در قسمت مرجع مراجعه فرمایید.

## ۲/۵/۶ کار با عناصر سند

یک راه ساده برای تغییر دادن سنهای HTML با W3C DOM تعیین مقدار خصلت‌های متناظر با صفات HTML است. به طوری که در DOM قدیمی دیدیم، به عنوان مثال می‌توان مقدار خصلت `src` عنصری را که متناظر با یک برگه‌ی `img` است، تغییر داد. یک راه بسیار مؤثر برای تغییر عناصر سند، از طریق خصلت `style` است، که شیوه‌های CSS را کنترل می‌کند. بعداً این موضوع مهم را با جزئیات بیشتری بررسی خواهیم کرد.

## ۲/۵/۷ تغییر متن سند

برای تغییر محتوای متنی سند، کافی است که مقدار خصلت `nodeValue` را در یک گره متنی تغییر دهید:

```
// Find the first <h1> tag in the document
var h1 = document.getElementsByTagName("h1")[0];
// Set new text of its first child
h1.firstChild.nodeValue = "New heading";
```

علاوه بر تغییر خصلت `nodeValue`، شیء متن اجازه تغییر دادن خصلت `data` را نیز می‌دهد، و همچنین، روش‌هایی برای درج کردن، حذف کردن، افزودن، و یا جایگزین کردن متن دارد.

دقت کنید که مشکل متن مذکور این است که فرض می‌کند که محتوای برگه‌ی `<h1>` متن ساده است. حال اگر مثلاً محتوای برگه به صورت زیر باشد، این متن موفق نخواهد بود، زیرا در اینجا متن عنوان نوی برگه‌ی `<h1>` است، نه فرزند مستقیم آن:

```
<h1><i>Original Heading</i></h1>
```

یک راه برای حل این مسئله، استفاده از خصلت `innerHTML` گره عنوان است. این خصلت بخشی از DOM IE4 است، نه W3C DOM، ولی از آنجا که بسیار مفید است، در بسیاری از مرورگرهای مدرن پشتیبانی می‌شود. وقتی دوباره به بررسی IE4 DOM پردازیم، مجدداً با این خصلت رو به رو خواهیم شد. راه دیگر برای حل این مسئله، جایگزین کردن گره عنوان با یک برگه‌ی جدید `<h1>` و گره متن با متن دلخواه است، که این راه را در قسمت بعد می‌بینیم.

## ۲/۵/۸ تغییر ساختار سند

علاوه بر تغییر متن سند و صفات عناصر سند، امکان W3C DOM ساختار درخت خود سند را نیز می‌دهد. برای این کار از روش‌های گره که امکان درج، پیوست، حذف، و جایگزین کردن فرزندان یک گره را می‌دهند، و روش‌های سند که امکان ایجاد گره‌های عنصر و متن جدید را فراهم می‌آورند، انجام می‌شود. متن زیر نشان دهنده‌ی این مطلب است:

```
// Find a <ol> element by name:  
var list = document.getElementById("mylist");  
// Create a new <li> element  
var item = document.createElement("li");  
// Append it to the list  
list.appendChild(item);  
// Create a Text node  
var text = document.createTextNode("new item");  
// Append it to the new <li> node  
item.appendChild(text);  
// Remove the new item from the list  
list.removeChild(item);  
// Place the new item at the start of the list  
list.insertBefore(item, list.firstChild);
```

باز به عنوان یک مثال دیگر، متن زیر یکتابع جاوا اسکریپت را نشان می‌دهد که از W3C DOM برای پرنگ کردن یک گره دلخواه سند استفاده می‌کند؛ این تابع برای گره مذکور والد جدیدی از نوع **<b>** ایجاد می‌کند:

```
function embolden(node) { // Embolden node n  
    var b = document.createElement("b");  
    var p = n.parentNode; // Get parent of n  
    p.replaceChild(b, n); // Replace n with <b>  
    b.appendChild(n); // Insert n into <b> tag  
}
```

## ۲/۶ IE4 DOM اینترنت اکسپلورر ۴

IE4 DOM در روایت ۴ مرورگر اینترنت اکسپلورر میکروسافت عرضه شد.

این نوع یک DOM قوی ولی استاندارد نشده است که قابلیت‌های آن مشابه DOM W3C است. روایت‌های ۵ و بالاتر اینترنت اکسپلورر اکثر ویژگی‌های پایه‌ی DOM W3C را پشتیبانی می‌کنند، لیکن به علت آنکه هنوز هم از IE4 DOM به طور شایع استفاده می‌شود، بحث زیر را در مورد آن ارائه می‌کنیم. در قسمت زیر، IE4 DOM از نظر تفاوت‌هایی که با W3C DOM دارد، مورد بحث قرار می‌گیرد، لذا لازم است که قبلًا با W3C DOM آشنایی داشته باشید.

## ۲/۶/۱ دستیابی به عناصر سند

از روش getElementById() IE4 DOM پشتیانی نمی‌کند. در عوض، برای یافتن عناصر دلخواه سند باید از آرایهی [all] متعلق به شیء سند استفاده کنند:

```
var list = document.all["mylist"];
list = document.all.mylist; // this also works
```

گاری غیرعادی می‌کند و آن اینکه روی آرایهی [all] یک روش () تعريف می‌کند، که این هم برای عناصر سند و هم برای خود شیء سند وجود دارد.

در اینجا نحوه‌ی یافتن برگه‌های <li> را در داخل اولین برگه‌ی <ul> می‌بینید:

```
var lists = document.all.tags("UL");
var items = lists[0].all.tags("LI");
```

دقت کنید که برای روش () all.tags() باشد نام برگه‌ی HTML را با حروف بزرگ وارد کنید.

## ۲/۶/۲ عبور از درخت سند

عبور از درخت سند در IE4 DOM به همان صورت W3C DOM است. تفاوت فقط در نام خصلت‌های مربوطه است: اینترنت اکسپلورر ۴ به جای children[], خصلت [parentNode] دارد، و به جای parentElement خصلت firstChild، nextSibling، و خصلت‌های مربوطه در W3C DOM ندارد. یک تفاوت عمده بین IE4 DOM و W3C DOM در این است که IE4 DOM حاوی برگه‌های HTML است: توضیحات چشمپوشی می‌شوند، و متن سند نیز بخشی از خود درخت نیست. در عوض، متن موجود در داخل هر عنصری از طریق خصلت‌های innerText و innerHTML شیء عنصر قابل دستیابی است. (در قسمت بعد، اطلاعات بیشتری درباره‌ی innerHTML ارائه خواهد شد).

## ۲/۶/۳ تغییر دادن محتوا و ساختار سند

گره‌های یک درخت سند اینترنت اکسپلورر ۴، اشیای عنصر مشابه گره عنصر در W3C DOM هستند. این اشیا هم، مانند گره‌های عنصر در DOM، خصلت‌هایی منتظر با صفات برگه‌های HTML دارند، و می‌توانید مقدار این خصلت‌ها را به دلخواه بخوانید یا بنویسید. برای تغییر محتوای متنی یک عنصر سند، متن مورد نظر را به خصلت innerText آن اختصاص دهید. با این کار هر برگه یا

متنی در داخل آن عنصر پاک می‌شود، و متن معین شده جایگزین آن می‌گردد. هیچ روش صریحی برای ایجاد، درج، حذف، و یا جایگزینی IE4 DOM گرهای یک درخت سند ندارد. اما از خصلت بسیار مهم innerHTML پشتیبانی HTML می‌کند، که به شما امکان می‌دهد که محتوای هر عنصر سند را با یک رشته‌ی HTML دلخواه جایگزین کنید. انجام این کار باعث فراخوانی تجزیه‌گر HTML می‌شود، و لذا این روش نسبت به تغییر مستقیم خود گره‌ها کارایی کمتری دارد. با این حال، این خصلت بسیار سودمند است، به حدی که موزیلا، نت‌اسکیپ ۶ و بالاتر، و سایر مرورگرهای مدرن این خصلت را — با آنکه استاندارد نیست — پیاده‌سازی کرده‌اند.

IE4 DOM دارای خصلت مشابه دیگری به نام outerHTML نیز هست که عنصر و محتوای آن را جایگزین می‌کند، و نیز روش‌های insertAdjacentText() و insertAdjacentHTML() دارد. از اینها زیاد استفاده نمی‌شود، و به علاوه، پشتیبانی از آنها در خارج از اینترنت اکسپلورر برخلاف innerHTML زیاد نیست؛ می‌توانید در باره‌ی آنها در مبحث مربوط به عنصر در قسمت مرجع اطلاعات بیشتری کسب کنید.

## ۲/۶/۴ سازگاری DOM

اگر بخواهید اسکریپتی بنویسید که در صورت موجود بودن از W3C DOM استفاده کند، و در غیر این صورت از IE4 DOM در صورت فراهم بودن بهره بگیرد، می‌توانید از روش خاصی استفاده کنید که در آن با بررسی موجود بودن یک روش یا خصلت مشخص می‌شود که پشتیبانی از DOM مورد نظر وجود دارد یا نه. برای مثال:

```
if (document.getElementById) {
    // If the W3C method exists, use it
}
else if (document.all) {
    // If the all[] array exists, use it
}
else {
    // Otherwise use the legacy DOM
}
```

## ۲/۷ CSS: اسکریپت‌نویسی شیوه‌های DHTML

با HTML پویا حاصل تر کیب کردن HTML، CSS، و DOM است: از اسکریپت برای تغییر پویای شیوه استفاده می‌شود، که ممکن است شامل مواردی مانند موقعیت و مرئی بودن عناصر سند باشد. در DOM‌های

W3C و IE4، هر عنصر سند یک خصلت style دارد که متاظر با صفت در HTML است و شیوه‌ی مستقیم عنصر را تعیین می‌کند. اما، خصلت style به جای اینکه به رشته‌ی ساده‌ای اشاره داشته باشد، معروف یک شیء شیوه است که خصلت‌هایی متاظر با صفات CSS شیوه دارد.

برای مثال، اگر یک عنصر e یک خصلت style داشته باشد که مقدار صفت CSS رنگ (color) را تعیین کرده باشد، برای خواندن مقدار آن می‌توانید از عبارت e.style.color استفاده کنید. هنگامی که نام یک صفت CSS دارای فاصله است، نام خصلت متاظر آن در جاوا اسکریپت فاقد فاصله است، و از حروف بزرگ مختلط با حروف کوچک استفاده می‌کند. بنا بر این، برای تعیین مقدار صفت background-color CSS در یک عنصر e، از خصلت background-color استفاده کنید. در جاوا اسکریپت e.style.backgroundColor دارد: صفت float CSS در جاوا اسکریپت یک واژه‌ی ذخیره شده است، و لذا نام خصلت متاظر آن در جاوا اسکریپت cssFloat است.

استاندارد CSS خصلت‌های زیادی تعریف می‌کند که می‌توانید از آن برای تنظیم منظره‌ی بصری سندهای خود استفاده کنید. لیستی از این خصلت‌ها در جدولی در مبحث شیوه در قسمت مرجع آمده است. خصلت‌های موقعیت و ظهور بالاخص برای اسکریپتنویسی پویا کاربرد دارند. اگر به خصلت position مقدار مطلق یک عنصر سند (بر حسب پیکسل، درصد، یا واحدهای دیگر) استفاده کنید. به همین ترتیب، خصلت‌های width و height به عنصر را تعیین می‌کنند. یک عنصر را می‌توان در آغاز با دادن مقدار hidden به خصلت visibility نامرئی کرد، و سپس در زمان مناسب با دادن مقدار visible آن را مرئی کرد.

دقت کنید که مقدار تمام خصلت‌های شیوه همواره از نوع رشته است، حتی برای خصلت‌هایی مانند left و width که معرف عدد هستند. هنگام تعیین این خصلت‌های طولی و بعدی، باید حتماً اعداد را به رشته تبدیل کنید، و واحد آن را هم به این رشته اضافه کنید (معمولًاً از رشته‌ی px که معرف پیکسل است، استفاده می‌شود). در جدول زیر، این خصلت‌های مربوط به موقعیت و نمایش به طور خلاصه معرفی شده‌اند:

خصیلت	توصیف/مقادیر
position	چگونگی قرار گیری موقعیت عنصر. <code>fixed</code> , <code>relative</code> , <code>absolute</code> , <code>static</code> (پیش فرض).
left, top	مختصات X و Y لبه‌های بالا و چپ عنصر.
width	پهنای عنصر
height	بلندای عنصر.
zIndex	ترتیب پشت‌هاسازی. مقادیر این خصلت عدد صحیح‌اند. عناصری که مقدار این خصلت آنها بالاتر است، روی عناصری که مقدار پایین‌ترین دارند، ترسیم می‌شوند.
display	چگونگی نمایش عنصر. مقادیر شایع عبارت‌اند از <code>block</code> , <code>inline</code> , <code>none</code> برای عناصری که اصلاً ترسیم نمی‌شوند.
visibility	مرئی (visible) یا نامرئی (hidden) بودن عنصر. برای عناصر مخفی هم فضا اختصاص داده می‌شود، مگر آنکه موقعیت خاصی به آنها داده شده باشد.
overflow	وقتی محتوای عنصر از اندازه‌ی آن تجاوز کند، چه اتفاقی بیفتد. مقادیر: <code>visible</code> (محتوا سریز می‌شود); <code>hidden</code> (محتوای مازاد مخفی است); <code>scroll</code> (نوار پیمایی همیشگی نشان داده شود); و <code>auto</code> (نوار پیما فقط در صورت لزوم).
clip	چه بخشی از محتوای عنصر نمایش داده شود. دستور: <code>rect(left, top, right, bottom)</code>

متن زیر یک پویانمایی ساده‌ی DHTML را نشان می‌دهد. هر بار فراخوانی می‌شود، تابع `nextFrame()` یک عنصر را ۱۰ پیکسل به طرف راست می‌برد و از `setTimeout()` استفاده می‌کند تا به جاوا اسکریپت بگوید که آن را ۵۰ میلی ثانیه بعد دوباره فراخوانی کند. پس از ۲۰ بار فراخوانی، تابع با استفاده از خصلت `visibility` عنصر را مخفی می‌کند و دست از فراخوانی مجدد خود می‌کشد.

```
<h1 id='title'>DHTML Animation<h1>
<script>
// Look up the element to animate
var e = document.getElementById("title");
// Make it positionable.
e.style.position = "absolute";
var frame = 0; // Initialize frame counter.
// This function moves the element one frame
// at a time, then hides it when done.
function nextFrame() {
    if (frame++ < 20) { // Only do 20 frames
        e.style.left = (10 * frame) + "px";
        // Call ourselves again in 50ms.
        setTimeout("nextFrame()", 50);
    }
    else e.style.visibility="hidden"; // Hide it.
}
nextFrame(); // Start animating now!
</script>
```

## ۲/۸ رویدادها و رویدادپردازی

در آغاز این قسمت دیدیم که یک راه برای قرار دادن جاوا اسکریپت سمت مستتری در درون سندهای HTML، استفاده از صفت‌های رویدادپرداز برگه‌های HTML است. در جدول زیر، لیست صفت‌های رویدادپرداز استاندارد HTML برگه‌هایی که این صفت‌ها برای آنها قابل استفاده هستند، ذکر شده است. در ستون اول جدول، نام صفت رویدادپرداز آمده است: این نامها همیشه با on شروع می‌شوند. در ستون دوم جدول، برگه‌های HTML که این صفت‌ها را می‌توان برای آنها به کار برد، ذکر شده است. در ضمن، در صورت لزوم ذکر شده است که چه رویدادهایی متن رویدادپرداز را فراخوانی می‌کنند.

رویدادپرداز	برگه‌های پشتیبانی کننده / زمان فراخوانی
onabort	<img> قطع بار شدن تصویر
onblur	<body> و عنصرهای فرم؛ محرف شدن کانون توجه صفحه کلید از پنجره یا عنصر
onchange	عنصرهای فرم؛ تغییر مقدار نمایش داده شده
onclick	تمام عنصرهای فرم؛ فشار داده شدن و رها شدن دکمه‌ی موشواره؛ برای لغو، باید مقدار
ondblclick	برگردان false
onerror	تمام عنصرهای دوکلیک موشواره
onfocus	<img> ناموفق بودن بار شدن تصویر
onkeydown	<body> و عنصرهای فرم؛ منتقل شدن کانون توجه صفحه کلید به پنجره یا عنصر
onkeypress	<body> و عنصرهای فرم؛ فشار داده شدن کلید؛ برای لغو، باید مقدار
onkeyup	برگردان false
onload	<body> و عنصرهای فرم؛ رها شدن کلید
onmousedown	سنده، تصویر، یا شیء
onmousemove	تمام عنصرهای فرم؛ فشار داده شدن دکمه‌ی موشواره
onmouseout	تمام عنصرهای حرکت نشانگر موشواره
onmouseover	تمام عنصرهای خارج شدن موشواره از روی عنصر
onreset	تمام عنصرهای وارد شدن موشواره به روی عنصر؛ برای جلوگیری از نمایش نشانی پیوند در سطر وضعیت پایین مروگر، باید مقدار true برگردان
onmouseup	تمام عنصرهای رها شدن دکمه‌ی موشواره
onresize	<form> تغییر اندازه‌ی پنجره
onsubmit	<form> تغاضای تحويل فرم؛ برای جلوگیری از تحويل، باید مقدار
onunload	برگردان false

دقت کنید که وقتی مرورگر بعضی از رویدادپردازها، از قبیل onclick، onsubmit، onmouseover و onmouseout را فراخوانی می‌کند، به مقدار برگشتی رویدادپرداز (در صورت وجود) نگاه می‌کند تا بیند آیا باید عمل پیشفرض مربوط به رویداد را انجام دهد یا نه. به طور معمول، اگر یک رویدادپرداز مقدار false برگرداند، عمل پیشفرض (مانند رفتن به یک پیوند یا تحویل دادن یک فرم) انجام نمی‌شود. استثنای این مطلب، رویدادپرداز onmouseover است: وقتی موشواره روی یک پیوند حرکت می‌کند، مرورگر نشانی پیوند را در سطر وضعیت خود نمایش می‌دهد مگر آن که رویدادپرداز مقدار true برگرداند.

## ۲/۸/۱ رویدادپردازها به عنوان تابع‌های جاوا اسکریپت

دیدیم که در انواع مختلف مدل شیء سند (DOM)، برگه‌های HTML به عنوان اشیای جاوا اسکریپت نمایش داده می‌شوند، و صفات برگه به عنوان خصلت‌های شیء منظور می‌گردند. اگر سند شما فقط حاوی یک برگه‌ی <form> با یک صفت رویدادپرداز onsubmit باشد، رویدادپرداز مذکور به صورت زیر قابل دسترس است:

```
document.forms[0].onsubmit
```

گرچه صفات رویدادپرداز HTML به عنوان رشته‌های متن جاوا اسکریپت نوشته می‌شوند، ولی، در واقع، مقدار خصلت جاوا اسکریپت متناظر با آنها از نوع رشته‌ی متن نیست، بلکه تابع است. لذا برای ایجاد یک رویدادپرداز جدید کافی است که یک تابع را به عنوان مقدار به خصلت مربوطه اختصاص دهید:

```
function validate() { // Form validation function
    // check validity here
    return valid;        // return true or false
}
// Now check user input before submitting it
document.forms[0].onsubmit = validate;
```

## ۲/۸/۲ رویدادپردازی پیشرفته

در قسمت‌های قبل، مدل اساسی رویدادپردازی در جاوا اسکریپت سمت مشتری مورد بحث قرار گرفت. رویدادپردازی ویژگی‌های پیشرفته‌ای نیز دارد، ولی متأسفانه سه مدل رویداد وجود دارد که با هم ناسازگارند: مدل استاندارد W3C، مدل اینترنت اکسپلورر (میکروسافت استاندارد W3C را پذیرفته است)، و مدل نت‌اسکیپ<sup>۴</sup>. این مدل‌های رویداد خیلی پیچیده‌اند، لذا در قسمت زیر ما تنها خلاصه‌ای از ویژگی‌های پیشرفته‌ای این مدل‌ها را بیان می‌کنیم. برای جزئیات بیشتر به کتاب‌های مفصل‌تر مراجعه کنید.

## • جزئیات رویداد

در مدل‌های پیشرفته‌ی رویدادپردازی، جزئیات رویداد، از قبیل نوع رویداد، وضعیت دکمه‌ها و مختصات موشواره، حالت کلیدهای تغییرگر، وغیره، به عنوان خصلت‌های شیء رویداد در اختیار ما قرار می‌گیرد. در مدل‌های رویداد W3C نتسکیپ، این شیء رویداد به عنوان آوند به رویدادپرداز داده می‌شود. در مدل اینترنت اکسپلورر، شیء رویداد به عنوان آوند داده نمی‌شود، بلکه در خصلت event پنجره‌ای که رویداد در آن روی می‌دهد، ذخیره می‌شود. متأسفانه در این سه مدل پیشرفته از نام‌های متفاوتی برای خصلت‌های رویداد استفاده می‌شود، و بنا بر این، سازگاری بین مرورگرهای مختلف مشکل است. برای اطلاعات بیشتر درباره‌ی شیء رویداد در هر یک از این سه مدل به مبحث رویداد در قسمت مرجع مراجعه فرمایید.

## • انتشار رویداد

در مدل پایه‌ی رویدادها، رویدادپردازها فقط برای عنصری از سند که رویداد روی آن بروز کرده است، فراخوانده می‌شوند. اما در مدل‌های پیشرفته، رویدادها می‌توانند در درخت عناصر سند به طرف بالا یا پایین عنصر مربوطه انتشار پیدا کنند، و به وسیله‌ی بیش از یک رویدادپرداز، مورد پردازش قرار گیرند. در مدل‌های نتسکیپ و W3C، رویدادها از شیء سند شروع می‌شوند و در درخت عناصر سند پایین می‌آیند تا به عنصری برسند که رویداد برای آن روی داده است. اگر هر کدام از عناصر بالاتر رویدادپردازهای ثبت شده‌ای داشته باشد، این رویدادپردازها رویداد را می‌گیرند و اولین قدم را در پردازش آن بر می‌دارند. در مدل‌های اینترنت اکسپلورر و W3C، برخی از انواع رویدادها، مانند کلیک موشواره، بعد از اینکه در عنصر منع مورد پردازش قرار گرفتند، حباب وار به قسمت‌های بالاتر سند سرایت می‌کنند. به این ترتیب، برای پردازش تمام کلیک‌های موشواره‌ای که روی عناصر درون یک عنصر `<div>` به وقوع می‌پیوندند، می‌توانید یک رویدادپرداز برای آن عنصر به ثبت برسانید. تمام انواع رویدادپرداز ارسالی، حبابی، و طبیعی این توانایی را دارند که مانع از انتشار بیشتر رویداد شوند، ولی روش انجام این کار در هر مدل متفاوت است.

### • ثبت رویدادپرداز

در مدل رویداد W3C، رویدادپردازها صرفاً به عنوان مقدار به خصلت‌های اشیای سند اختصاص داده نمی‌شوند. بلکه در این مدل، هر شیء سند یک روش `() addEventListener` دارد که با فراخوانی آن می‌توانید یک تایپ رویدادپرداز را برای رویداد مورد نظر خود به ثبت برسانید. به این طریق، در برنامه‌های پیشرفته می‌توان برای یک رویداد، چند رویدادپرداز به ثبت رسانید.

## ۲/۹ محدودیت‌های امنیتی جاوا اسکریپت

به دلایل امنیتی، معمولاً در پیاده‌سازی‌های جاوا اسکریپت سمت مشتری محدودیت‌هایی بر اعمالی که اسکریپت‌ها می‌توانند انجام دهند، اعمال می‌شود. واضح ترین محدودیت‌ها حذف قابلیت‌های خطرناک است: مثلاً، اسکریپت‌های سمت مشتری به هیچ طریقی نمی‌توانند پرونده‌ای را از روی دیسک محلی یک کاربر پاک کنند. محدودیت‌های دیگری نیز برای جلوگیری از افساسازی اطلاعات خصوصی و یا جلوگیری از آزار کاربران توسط اسکریپت‌ها وضع شده‌اند. البته لیست استانداردی از محدودیت‌های امنیتی وجود ندارد، ولی موارد زیر محدودیت‌هایی هستند که در پیاده‌سازی‌های معمول مرورگرها یافت می‌شوند. تلاش نکنید اسکریپت‌هایی بنویسید که این کارها را انجام بدند؛ اگر هم این اسکریپت‌ها در مرورگر خودتان کار کنند، ممکن است در مرورگرهای دیگر با مشکل مواجه شوند.

### • سیاست مبدأ یکسان

اسکریپت‌ها فقط خصلت‌های پنجره‌ها و سند‌هایی را که از سرور شبکه‌ی یکسانی بار شده باشند، می‌توانند بخوانند. این محدودیت غلاظ و شدادی برای اسکریپت‌نویسی بین‌پنجره‌ای است، و مانع از آن می‌شوند که اسکریپت‌ها به مطالب سند‌های نامربوط دیگری که کاربر می‌خواند، دسترسی پیدا کنند. علاوه بر این، این محدودیت مانع از به ثبت رساندن رویدادپردازها و رویدادهای کلاهبردارانه روی سند‌های نامربوط می‌شود.

### • فرآگذاری پرونده‌ها

اسکریپت‌ها قادر نیستند مقدار خصلت `value` را در عنصر فرم `FileUpload` تعیین کنند.

### • فرستادن نامه‌ی الکترونیک و خبر

اسکریپت‌ها نمی‌توانند بدون تأیید کاربر فرمی را برای نشانی‌های news: یا mailto: بفرستند.

### • بستن پنجره‌ها

یک اسکریپت فقط می‌تواند پنجره‌های مرورگری را که خود ایجاد کرده است، بیندد، مگر اینکه تأیید کاربر را بگیرد.

### • سرک کشیدن به حافظه‌ی نهانی

اسکریپت نمی‌تواند نشانی‌های about:cache:about: را از قبیل بخواند.

### • پنجره‌های پنهان و تزیینات پنجره

اسکریپت نمی‌تواند پنجره‌های کوچک یا خارج از صفحه‌ی نمایش و یا پنجره‌های بدون نوار عنوان ایجاد کند.

دقت کنید که لیست محدودیت‌های امنیتی ثابت نیست. با افزایش استفاده از جاوا اسکریپت، آگهی دهنده‌گان و اشخاص موذی کم کم شروع به استفاده‌های موذیانه از آن کرده‌اند. در نتیجه، مرورگرهای جدید مانند موزیلا ۱/۰، امکانات قابل تنظیم توسط کاربری برای محدودیت‌های اسکریپتی دارند که مانع از باز کردن پنجره‌های جدید (مانند پنجره‌های آگهی) توسط اسکریپت و یا حرکت دادن یا تغییر اندازه‌ی پنجره‌های موجود می‌شوند.

## ۳ مرجع جاوا اسکریپت

قسمت باقیمانده‌ی این کتاب حاوی مرجع سریعی برای رابط برنامه‌نویسی کاربردی جاوا اسکریپت هسته و سمت مشتری است. در اینجا، توابع جاوا اسکریپت هسته به طور کامل و DOM قدیمی (سطح  $\circ$ ) مورد بررسی قرار می‌گیرد، و DOM سطح ۲ W3C نیز به طور ساده شده ارائه می‌شود. قسمت‌هایی از رابط برنامه‌نویسی مذکور که به برنامه‌نویسان جاوا اسکریپت در سندهای HTML مربوط نمی‌شود، حذف شده است. در آغاز هر مبحث، مشخص شده است که آیا جزئی از جاوا اسکریپت هسته است یا جاوا اسکریپت سمت مشتری، و اینکه ویژگی مورد نظر در چه روایتی از جاوا اسکریپت، چه مروگرهایی، و چه نسخه‌ای از DOM ارائه شده است.

از آنجا که جاوا اسکریپت زبانی با نوع‌های ضعیف است، لذا کلاس‌ها و اشیای موجود در رابط برنامه‌نویسی جاوا اسکریپت لیست رسمی و مشخصی ندارد. لیست درایه‌هایی که بحث خواهند شد، ذیلاً درج شده است. می‌توانید مبحث مورد نظر خود را در این لیست به سهولت پیدا کنید.

موقعیت خاصی در سند همراه با یک نام	Anchor
یک برنامک جاوا	Applet
آندهای یک تابع	Arguments
ایجاد و عملیات آرایه	Array
صفت یک عنصر سند	Attr
یک شیء لغافه برای مقادیر بولی	Boolean
یک توضیح HTML	Comment
خطای DOM را نشان می‌دهد	DOMEception
سند ایجاد می‌کند، و ویزگی‌های DOM را بررسی می‌کند	DOMImplementation
انجام عملیات تاریخ و زمان	Date
یک سند HTML	Document
عملیات بر روی تعدادی از گره‌های با هم	DocumentFragment
یک برگه‌ی HTML در یک سند	Element
أنواع استثنای از پیش تعریف شده	Error
جزئیات رویداد	Event
یک فرم ورودی HTML	Form
یک تابع جاوا اسکریپت	Function
خلصت‌ها و توابع سراسری	Global
سابقه‌ی مرور	History
یک تصویر HTML	Image
یک عنصر ورودی در فرم	Input
یک لایه‌ی مستقل در سند	Layer
<area> یا <a> پیوند از نوع	Link
نشانی فعلی مرورگر	Location
توابع و ثابت‌های ریاضی	Math
اطلاعات در باره‌ی مرورگر	Navigator
گرگه‌ی در درخت سند	Node
پیش‌تیانی از اعداد	Number
کلاس پایه‌ی همه‌ی اشیا در جاوا اسکریپت	Object
یک گرینه‌ی قبل انتخاب	Option
عبارت مرتب برای انطباق الگو	RegExp
اطلاعات در باره‌ی صفحه‌ی نمایش	Screen
یک لیست نمایش گرینه‌ها	Select
کار با رشته‌ها	String
خواص مستقیم CSS یک عنصر HTML	Style
قطعه‌ای از متن در یک سند	Text
ورودی متن چندسطري	Textarea
پنجره یا کادر مرورگر	Window

## Anchor ۳/۱

(جاوا اسکریپت سمت مشتری ۱/۲)

یک محل نامگذاری شده در سند.

مشتق شده از: Element

### دستور ۳/۱/۱

```
document.anchors[index]  
document.anchors[name]
```

### شرح ۳/۱/۲

شیء Anchor نشان دهنده یک برگه‌ی  با صفت name است، که برای ایجاد یک محل نامگذاری شده در سند به کار می‌رود.

### خصلت‌ها ۳/۱/۳

name •

مقدار صفت name از برگه‌ی .

### ارجاع ۳/۱/۴

```
Location.hash, Link, Document.anchors[]
```

## Applet ۳/۲

(جاوا اسکریپت سمت مشتری ۱/۱)

یک برنامک جاوا.

### دستور ۳/۲/۱

```
document.applets[i]  
document.applets[appName]  
document.appletName
```

### خصلت‌ها و روش‌ها ۳/۲/۲

خصلت‌ها و روش‌های یک شیء برنامک، همان فیلدها و روش‌های عمومی برنامک جاوابی هستند که این شیء معرف آن است. متن جاوا اسکریپت می‌تواند

فیلدهای جاوا را بخواند یا بنویسد، و روش‌های جاوای برنامک را فراخوانی کند.

## Arguments ۳/۳

(جاوا اسکریپت هسته ۱/۱؛ جی اسکریپت ۲/۰؛ ECMA ۱) آوندهای یک تابع.

### ۳/۳/۱ دستور

`arguments[n]`  
`arguments.length`

### ۳/۳/۲ شرح

شیء آوندها فقط در درون بدنی یک تابع تعریف شده است، و در درون بدنی هر تابع، متغیر `arguments` به شیء آوندها، آرایه‌ای است که عناصر آن مقادیری هستند که به عنوان آوند به دارد. شیء آوندها، آرایه‌ای است که عناصر آن مقادیری هستند که به عنوان آوند به تابع داده شده‌اند. عنصر ۰ آوند اول است، عنصر ۱ آوند دوم، و الخ. تمام مقادیری که به عنوان آوند به تابع داده شده‌اند، تبدیل به عنصرهای شیء آوندها می‌شوند، ولو آنکه در تعریف تابع به این آوندها نامی داده نشده باشد.

### ۳/۳/۳ خصلت‌ها

#### callee •

شارهای به تابعی که هم‌اکنون در حال اجرا است. برای تراجع در توابع بی‌نام مفید است. جاوا اسکریپت ۱/۲؛ جی اسکریپت ۵/۵؛ ECMA ۱؛ فقط در درون بدنی یک تابع تعریف شده است.

#### length •

تعداد آوندهای داده شده به تابع. جاوا اسکریپت ۱/۱؛ جی اسکریپت ۲، ECMA ۱؛ فقط در درون بدنی یک تابع تعریف شده است.

### ۳/۳/۴ ارجاع

`.Function`

**Array ۳/۴**

(جاوا اسکریپت هسته ۱؛ جی اسکریپت ۰؛ ECMA ۲) ایجاد آرایه و کار کردن با آن.

**۳/۴/۱ سازنده**

```
new Array()           // empty
new Array(n)          // n undefined elements
new Array(e0, e1, ...) // specified elements
```

**۳/۴/۲ دستور مستقیم**

در جاوا اسکریپت ۱/۲، جی اسکریپت ۰، و ECMA ۳ می توانید با قرار دادن لیستی از عبارت‌ها در درون کروشه که با ویرگول از هم جدا شده‌اند، یک آرایه را ایجاد و آغازش کنید. مقادیر این عبارت‌ها، عناصر آرایه خواهند بود. برای مثال:

```
var a = [1, true, 'abc'];
var b = [a[0], a[0]*2, f(x)];
```

**۳/۴/۳ خصلت‌ها****length •**

یک عدد صحیح خواندنی/نوشتني که تعداد عناصر آرایه را نشان می‌دهد، و یا، در مواردی که عناصر آرایه مجاور هم نیستند، عددی را نشان می‌دهد که یک واحد از آخرین عنصر موجود در آرایه بالاتر است. تغییر دادن مقدار این خصلت، آرایه را از وسط قطع کرده و یا اینکه آن را بسط می‌دهد.

**۳/۴/۴ روش‌ها****concat(value, ...)**

آرایه‌ی جدیدی بر می‌گرداند که از ادغام هر یک از آوندهای داده شده با این آرایه تشکیل می‌شود. اگر هر کدام از آوندهای concat() خود یک آرایه باشد، عناصر آن به این آرایه ملحق می‌شوند، نه خود آن. جاوا اسکریپت ۱/۲؛ جی اسکریپت ۰؛ ECMA ۳.

**join(separator) •**

رشته‌ای را بر می‌گرداند که از تبدیل هر یک از عناصر آرایه به رشته و

سپس به هم پیوستن همه‌ی آنها با استفاده از separator به عنوان جدا کننده در بین آنها، حاصل می‌شود.

#### **pop()** •

آخرین عنصر آرایه را برابر می‌دارد و بر می‌گرداند، و طول آرایه را یک واحد کم می‌کند. جاوا اسکریپت ۱/۲؛ جی اسکریپت ۵/۵؛ ECMA ۳.

#### **push(value, ...)** •

مقدار یا مقادیر داده شده را به انتهای آرایه ملحق می‌کند، و طول جدید آرایه را برابر می‌گرداند. جاوا اسکریپت ۱/۲؛ جی اسکریپت ۵/۵؛ ECMA ۳.

#### **reverse()** •

ترتیب عناصر آرایه را معکوس می‌کند. چیزی بر نمی‌گرداند.

#### **shift()** •

اولین عنصر آرایه را حذف می‌کند، و بر می‌گرداند، و هر کدام از عناصر باقیمانده را یک شماره جلوتر می‌آورد، و طول آرایه را یک واحد کاهش می‌دهد. جاوا اسکریپت ۱/۲؛ جی اسکریپت ۵/۵؛ ECMA ۳.

#### **slice(start, end)** •

آرایه‌ی جدیدی بر می‌گرداند که حاوی عناصر آرایه از عنصر شماره‌ی start تا عنصر شماره‌ی end (به استثنای خود آن عنصر) است. جاوا اسکریپت ۱/۲؛ جی اسکریپت ۵/۵؛ ECMA ۳.

#### **sort(orderfunc)** •

عناصر آرایه را مرتب می‌کند، و اشاره‌ای به آن آرایه را برابر می‌گرداند. دقت کنید که خود آرایه مرتب می‌شود، نه اینکه نسخه‌ی جدیدی از آن ساخته شود. آوند اختیاری orderfunc می‌تواند تابعی را برای تعیین چگونگی مرتب‌سازی ارائه نماید.تابع مذکور باید دو آوند بگیرد، و در صورتی که آوند اول کمتر از دومی باشد، مقداری کمتر از ۰ بر گرداند، اگر هر دو برابر باشند، ۰ بر گرداند، و اگر آوند اول بزرگ‌تر از دومی باشد، مقداری بالاتر از ۰ بر گرداند.

#### **splice(start, deleteCount, value, ...)** •

تعداد مشخص شده از عناصر آرایه را با شروع از شماره‌ی تعیین شده حذف می‌کند، و بعد آوندهای باقیمانده را در آن محل در آرایه درج

می کند. آرایه ای بر می گرداند که حاوی عناصر حذف شده است.  
جاوا اسکریپت ۱/۲؛ جی اسکریپت ۵/۵؛ ECMA ۳.

#### **toLocaleString()** •

نمایش رشته ای محلی شده ای آرایه را بر می گرداند. جاوا اسکریپت ۱/۵؛  
جی اسکریپت ۵/۵؛ ECMA ۱.

#### **toString()** •

نمایش رشته ای آرایه را بر می گرداند.

#### **unshift(value, ...)** •

آوند یا آوندهای داده شده را در آغاز آرایه درج می کند، و عناصر قبلی  
را جابجا می کند، تا برای اینجا جا باز شود. طول جدید آرایه را  
بر می گرداند. جاوا اسکریپت ۱/۲؛ جی اسکریپت ۵/۵؛ ECMA ۳.

## **Attr ۳/۵**

(DOM سطح ۱)

صفت یک عنصر HTML

مشتق شده از: Node

#### **خصلت ها ۳/۵**

##### **name** •

نام صفت. فقط خواندنی.

##### **ownerElement** •

شیء عنصری که حاوی این صفت است. فقط خواندنی. DOM سطح ۲.

##### **specified** •

در صورتی که مقدار صفت به صراحة در متن سند و یا توسط یک  
اسکریپت تعیین شده باشد، مقدار آن true است، و در غیر این صورت،  
مقدار آن false است. فقط خواندنی.

##### **value** •

مقدار صفت به صورت یک رشته. خواندنی / نوشتنی.

## ارجاع ۳/۵/۲

```
Document.createAttribute()  
Element.getAttributeNode()  
.Element.setAttributeNode()
```

## Boolean ۳/۶

(جاوا اسکریپت هسته ۱/۱؛ جی اسکریپت ۲/۰؛ ECMA ۱) شیء لفافه برای مقادیر بولی.

### سازنده ۳/۶/۱

```
new Boolean(value)  
Boolean(value)
```

وقتی () به عنوان یک تابع، بدون عملگر `new` فراخوانده شود، مقدار داده شده را به یک مقدار بولی (نه یک شیء `Boolean`) تبدیل می‌کند، و آن را بر می‌گرداند. تمام مقادیر تبدیل به مقدار `true` می‌شوند، به جزء `NaN`، `null`، `undefined`، و رشته‌ی تهی، ".". وقتی سازنده‌ی `(new Boolean()` با عملگر `new` فراخوانده شود، همان کار را انجام می‌دهد، و حاصل را در یک شیء `Boolean` می‌پیچد.

### روش‌ها ۳/۶/۲

#### toString() •

بسته به مقدار شیء بولی، مقدار "true" یا "false" بر می‌گردد.

#### valueOf() •

مقدار بولی بدوى را که درون شیء بولی پیچیده شده است، بر می‌گردد.

## Comment ۳/۷

(۲ سطح DOM)  
یک توضیح `.HTML`  
مشتق شده از: `Node`

## ۳/۷/۱ خصلت‌ها

گره‌های توضیح دقیقاً دارای همان خصلت‌های گره‌های متن هستند.

## ۳/۷/۲ روش‌ها

گره‌های توضیح تمام روش‌های گره‌های متن را، به جز `splitText()` پشتیانی می‌کنند.

## ۳/۷/۳ ارجاع

`.Text`

## ۳/۸ DOMException

خطای DOM سطح ۱ را نشان می‌دهد.

## ۳/۸/۱ خصلت‌ها

`code` •

رمز خطا که جزئیاتی را در باره‌ی آنچه سبب بر انگیخته شدن استثنا شده است، نشان می‌دهد. بعضی از مقادیر ممکن‌های این خصلت (و معنای آنها) توسط ثابت‌های ذکر شده در زیر تعریف شده‌اند.

## ۳/۸/۲ ثابت‌ها

ثبت‌های زیر مقادیر `code` را که ممکن است در کار با سندهای HTML با آنها مواجه شوید، نشان می‌دهند. دقت کنید که این ثابت‌ها خصلت‌های ایستای DOMException هستند، نه خصلت‌های اشیای نمونه‌ی حاصل از آن.

`DOMException.INDEX_SIZE_ERR = 1` •

خطای خروج از کران در مورد نمایه‌ی آرایه یا رشته.

`DOMException.HIERARCHY_REQUEST_ERR = 3` •

تلash برای گذاشتن یک گره در یک محل غیرقانونی در درخت انشعابات سند.

**DOMEexception.WRONG\_DOCUMENT\_ERR = 4 •**

تلash برای استفاده از گره در سندی غیر از سندی که آن را ساخته است.

**DOMEexception.INVALID\_CHARACTER\_ERR = 5 •**

نویسه‌ای غیرقانونی (مثلاً در نام یک عنصر) یافت شده است.

**DOMEexception.NOT\_FOUND\_ERR = 8 •**

یک گره در جایی که باید می‌بود، یافت نشد.

**DOMEexception.NOT\_SUPPORTED\_ERR = 9 •**

یک روش یا خصلت در پیاده‌سازی کنونی DOM پشتیبانی نمی‌شود.

**DOMEexception.INUSE\_ATTRIBUTE\_ERR = 10 •**

تلash برای ربط دادن یک Attr به یک Element در شرایطی که قبلاً به گره Element دیگری ربط داده شده است.

**DOMEexception.SYNTAX\_ERR = 12 •**

خطای دستوری، مثلاً در تعیین خصلت CSS

**DOMImplementation ۳/۹****(۱) سطح DOM**

سند می‌سازد و خصوصیات DOM را وارسی می‌کند.

**۳/۹/۱ دستور**

`document.implementation`

**۳/۹/۲ روش‌ها****`createHTMLDocument(title)` •**

یک شیء جدید از نوع سند HTML ایجاد می‌کند، و در آن عناصر `<html>`، `<head>`، `<body>`، `<title>` و `title` قرار می‌دهد، و آن را بر می‌گرداند. `title` متنی است که در عنصر `<title>` درج می‌شود.

.۲ سطح DOM

**`hasFeature(feature, version)` •**

اگر پیاده‌سازی موجود ویراست مشخص شده از ویژگی داده شده را

پشتیانی بکند، مقدار true بر می‌گردداند، و در غیر این صورت، مقدار false بر می‌گردداند. اگر هیچ ویراستی داده نشده باشد، در صورتی مقدار true بر می‌گردداند که این پیاده‌سازی ویژگی مشخص شده را به طور کامل با تمام ویراست‌ها پشتیانی بکند. *version* و *feature* هر دو از نوع رشته هستند؛ مثلاً "core", "1.0" یا ".html", "2.0"

## Date ۳/۱۰

(جاوا اسکریپت هسته ۱/۰؛ جی‌اسکریپت ۱/۰؛ ECMA ۱) کار با تاریخ و زمان.

### ۳/۱۰/۱ سازنده

```
new Date();           // current time
new Date(milliseconds) // from timestamp
new Date(datestring); // parse string
new Date(year, month, day, hours, minutes, seconds, ms)
```

سازنده‌ی () بدون آوند، یک شیء Date ایجاد می‌کند که روی تاریخ و زمان فعلی تنظیم شده است. اگر یک آوند عددی به آن داده شود، آن را به عنوان نمایش درونی تاریخ بر حسب میلی ثانیه، به گونه‌ای که توسط روش getTime() بر گردانده می‌شود، فرض می‌کند. اگر یک آوند رشته‌ای داده شود، آن را به عنوان نمایش رشته‌ای یک تاریخ تغییر می‌کند. در غیر این صورت، به سازنده دو تا هفت آوند عددی داده می‌شود که فیلدهای تاریخ و زمان محلی را مشخص می‌کنند. تمام فیلدها، به استثنای دو تای اول (سال و ماه) اختیاری هستند. روش ایستای () Date.UTC، جایگزین مناسبی است که به جای زمان محلی از زمان جهانی استفاده می‌کند.

هنگامی که () بدون عملگر new فرا خوانده شود، به آوندهای داده شده توجهی نمی‌کند، و نمایش رشته‌ای تاریخ و زمان کنونی را بر می‌گردداند.

### ۳/۱۰/۲ روش‌ها

شیء Date هیچ‌گونه خصلتی ندارد، بلکه هر گونه دستیابی به مقادیر تاریخ و زمان از طریق روش‌های آن صورت می‌گیرد. اکثر روش‌ها دو شکل دارند: یکی نوعی که با زمان محلی کار می‌کند، و دیگری نوعی که واجد حروف "UTC" در نام خود است، و با استفاده از زمان جهانی (GMT) یا UTC کار می‌کند. این

روش‌های جفتی را در زیر ذکر می‌کنیم. دقت کنید که مقادیر برگشتی و آوندهای اختیاری اکثر روش‌های `set()` که ذیلاً ذکر می‌شود، قبل از استانداردسازی ECMA پشتیبانی نمی‌شوند. برای اطلاع از دامنه‌های قانونی هر کدام از فیلدهای تاریخ به روش گوناگون `get()` مراجعه کنید.

#### **get [UTC] Date()**

روز ماه را به وقت محلی یا جهانی بر می‌گرداند. مقدار برگشتی بین ۱ و ۳۱ است.

#### **get [UTC] Day()**

روز هفته را به وقت محلی یا جهانی بر می‌گرداند. مقدار برگشتی بین ۰ (یکشنبه) و ۶ (شنبه) است.

#### **get [UTC] FullYear()**

سال را با تمام ۴ رقم آن به وقت محلی یا جهانی بر می‌گرداند.  
جاوا اسکریپت `۱/۲`؛ جی اسکریپت `۳/۰`؛ ECMA `۱`.

#### **get [UTC] Hours()**

فیلد ساعت را به وقت محلی یا جهانی بر می‌گرداند. مقدار برگشتی بین ۰ (نیمه شب) و ۲۳ (۱۱ شب) است.

#### **get [UTC] Milliseconds()**

فیلد میلی‌ثانیه را به وقت محلی یا جهانی بر می‌گرداند.  
جاوا اسکریپت `۱/۲`؛ جی اسکریپت `۳/۰`؛ ECMA `۱`.

#### **get [UTC] Minutes()**

فیلد دقیقه را به وقت محلی یا جهانی بر می‌گرداند. مقدار برگشتی بین ۰ و ۵۹ است.

#### **get [UTC] Month()**

فیلد ماه را به وقت محلی یا جهانی بر می‌گرداند. مقدار برگشتی بین ۰ (ژانویه) و ۱۱ (دسامبر) است.

#### **get [UTC] Seconds()**

فیلد ثانیه را به وقت محلی یا جهانی بر می‌گرداند. مقدار برگشتی بین ۰ و ۵۹ است.

**getTime()** •

نمایش درونی تاریخ را بر حسب میلی ثانیه بر می گرداند؛ در واقع، تعداد میلی ثانیه های سپری شده از نیمه شب ۱ ژانویه ۱۹۷۰ (به وقت جهانی) و تاریخ و زمان موجود در شیء Date را بر می گرداند. دقت کنید که این مقدار ربطی به منطقه زمانی (وقت محلی) ندارد.

**getTimezoneOffset()** •

اختلاف بین نمایش محلی و جهانی این تاریخ و زمان را بر حسب دقیقه بر می گرداند. دقت کنید مقدار بر گردانده شده بستگی به فعال بودن سیستم تغییر ساعت در تابستان نیز دارد.

**getYear()** •

مقدار فیلد سال منهای ۱۹۰۰ را بر می گرداند. این روش به نفع روش `getFullYear()` منسوخ شده است.

**set[UTC]Date(day\_of\_month)** •

مقدار فیلد روز ماه را به وقت محلی یا جهانی تعیین می کند. نمایش میلی ثانیه ای تاریخ جدید را بر می گرداند.

**set[UTC]FullYear(year, month, day)** •

مقدار سال (و به طور اختیاری ماه و روز) را به وقت محلی یا جهانی تعیین می کند. نمایش میلی ثانیه ای تاریخ جدید را بر می گرداند. جاوا اسکریپت ۱؛ جی اسکریپت ۳؛ ECMA ۱/۲.

**set[UTC]Hours(hours, mins, secs, ms)** •

مقدار ساعت (و به طور اختیاری فیلدهای دقیقه، ثانیه، و میلی ثانیه) را به وقت محلی یا جهانی تعیین می کند. نمایش میلی ثانیه ای تاریخ جدید را بر می گرداند.

**set[UTC]Milliseconds(millis)** •

مقدار فیلد میلی ثانیه ای تاریخ را به وقت محلی یا جهانی تعیین می کند. نمایش میلی ثانیه ای تاریخ جدید را بر می گرداند. جاوا اسکریپت ۱؛ جی اسکریپت ۳؛ ECMA ۱/۲.

**set[UTC]Minutes(minutes, seconds, millis)** •

مقدار فیلد دقیقه (و به طور اختیاری فیلدهای ثانیه و میلی ثانیه) را به وقت

محلى یا جهانی تعیین می‌کند. نمایش میلی‌ثانیه‌ی تاریخ جدید را بر می‌گرداند.

#### **set[UTC]Month(month, day)** •

مقدار فیلد ماه (و به طور اختیاری فیلد روز ماه) را به وقت محلی یا جهانی تعیین می‌کند. نمایش میلی‌ثانیه‌ی تاریخ جدید را برابر می‌گرداند.

#### **set[UTC]Seconds(seconds, millis)** •

مقدار فیلد ثانیه (و به طور اختیاری فیلد میلی‌ثانیه) را به وقت محلی یا جهانی تعیین می‌کند. نمایش میلی‌ثانیه‌ی تاریخ جدید را برابر می‌گرداند.

#### **setTime(milliseconds)** •

مقدار نمایش درونی میلی‌ثانیه‌ی تاریخ را تعیین می‌کند. مقدار آوند *milliseconds* را برابر می‌گرداند.

#### **setYear(year)** •

مقدار فیلد سال ۲ رقمی را تعیین می‌کند. این روش به نفع روش `set[UTC]FullYear()` منسوخ شده است.

#### **toDateString()** •

قسمت تاریخ را با استفاده از وقت محلی به صورت رشته‌ای برابر می‌گرداند. جاوا اسکریپت<sup>۱/۵</sup>؛ جی‌اسکریپت<sup>۵/۵</sup>؛ ECMA<sup>.۳</sup>.

#### **toGMTString()** •

قسمت تاریخ را با استفاده از وقت گرینویچ به صورت رشته‌ای برابر می‌گرداند. این روش به نفع روش `toUTCString()` منسوخ شده است.

#### **toLocaleDateString()** •

قسمت تاریخ را با استفاده از وقت محلی و با استفاده از قراردادهای محلی فرمت تاریخ به صورت رشته‌ای برابر می‌گرداند. جاوا اسکریپت<sup>۱/۵</sup>؛ جی‌اسکریپت<sup>۵/۵</sup>؛ ECMA<sup>.۳</sup>.

#### **toLocaleString()** •

تاریخ را با استفاده از وقت محلی و با استفاده از قراردادهای محلی فرمت تاریخ به صورت رشته‌ای برابر می‌گرداند.

### **toLocaleTimeString()** •

قسمت زمان را با استفاده از وقت محلی و با استفاده از فواردادهای محلی فرمات تاریخ به صورت رشته‌ای بر می‌گرداند. جاوا اسکریپت ۱؛<sup>۳</sup> ECMA جی اسکریپت ۵/۵؛<sup>۴</sup> ۵/۵

### **toString()** •

نمایش رشته‌ای تاریخ را با استفاده از وقت محلی بر می‌گرداند.

### **toTimeString()** •

قسمت زمان را با استفاده از وقت محلی به صورت رشته‌ای بر می‌گرداند. جاوا اسکریپت ۱؛<sup>۵</sup> جی اسکریپت ۵/۵؛<sup>۶</sup> ECMA

### **toUTCString()** •

شیء تاریخ را با استفاده از زمان جهانی به یک رشته تبدیل می‌کند و آن را بر می‌گرداند. جاوا اسکریپت ۱؛<sup>۷</sup> جی اسکریپت ۰؛<sup>۸</sup> ECMA ۱

### **valueOf()** •

نمایش میلی‌ثانیه‌ی تاریخ را، درست مانند روش `getTime()` بر می‌گرداند. جی اسکریپت ۱؛<sup>۹</sup> ECMA ۱

## **۳/۱۰/۳ تابع‌های ایستا**

علاوه بر روش‌های نمونه که قبلاً گفتیم، شیء Date دو روش ایستا نیز تعریف کرده است. این روش‌ها از طریق خود سازنده‌ی Date() فراخوانی می‌شوند، و نه از طریق اشیای Date.

### **Date.parse(date)** •

یک نمایش رشته‌ای تاریخ و زمان را تجزیه می‌کند، و نمایش درونی میلی‌ثانیه را باز می‌گرداند.

### **Date.UTC(yr, mon, day, hr, min, sec, ms)** •

نمایش درونی میلی‌ثانیه را برای تاریخ و زمان داده شده باز می‌گرداند.

## **Document ۳/۱۱**

جاوا اسکریپت سمت مشتری DOM سطح ۱؛<sup>۱۰</sup> HTML

یک سند

## مشتق شده از: Node (در DOM سطح ۱)

### ۳/۱۱/۱ دستور

`window.document`  
`document`

### ۳/۱۱/۲ شرح

شیء Document نشان دهنده‌ی یک سند HTML است، و یکی از مهم‌ترین اشیا در جاوا اسکریپت سمت مشتری است. این شیء در جاوا اسکریپت ۱/۰ ارائه شد، و در جاوا اسکریپت ۱/۱، تعدادی روش و خصلت به آن اضافه شد. نت‌اسکیپ و اینترنت اکسپلورر هر کدام روش‌ها و خصلت‌های غیراستانداردی به شیء سند اضافه می‌کنند، W3C DOM نیز در استاندارد خود خصلت‌ها و روش‌های دیگری ارائه کرده است.

### ۳/۱۱/۳ خصلت‌های مشترک

تمام پیاده‌سازی‌های شیء سند خصلت‌های زیر را پشتیبانی می‌کنند. متعاقباً، لیست‌های جداگانه‌ای نیز برای خصلت‌های تعریف شده در DOM، W3C DOM، نت‌اسکیپ، و اینترنت اکسپلورر ارائه خواهیم کرد.

#### alinkColor •

رشته‌ای که رنگ پیوندهای فعال شده را مشخص می‌کند. منسوخ شده است.

#### anchors[] •

آرایه‌ی اشیای لنگر، بدین صورت که برای هر لنگری که در سند وجود دارد، یک شیء در این آرایه موجود است. جاوا اسکریپت ۱/۲.

#### applets[] •

آرایه‌ی اشیای برنامک، بدین صورت که برای هر برنامکی که در سند وجود دارد، یک شیء در این آرایه موجود است. جاوا اسکریپت ۱/۲.

#### bgColor •

رشته‌ای که رنگ زمینه‌ی سند را مشخص می‌کند. منسوخ شده است.

#### cookie •

خصلتی با مقدار رشته‌ای که رفتار خاصی دارد، و امکان خواندن/نوشتن

کوکی های مربوط به سند را فراهم می کند.

#### **domain •**

رشته ای که دامنه اینترنتی مبدأ سند را مشخص می کند. برای مقاصد امنیتی مورد استفاده قرار می گیرد. جاوا اسکریپت ۱/۱.

#### **embeds [] •**

آرایه ای از اشیا که معرف داده های خوابانده شده در سند با استفاده از برگه هی `<embed>` هستند. مترادف `[ ]` `plugins` است. بعضی از جازن ها و کنترل های ActiveX را می توان با استفاده از جاوا اسکریپت کنترل کرد. رابط برنامه نویسی این کار بستگی به نوع کنترل مورد نظر دارد. جاوا اسکریپت ۱/۲.

#### **fgColor •**

رشته ای که رنگ پیش فرض متن سند را مشخص می کند. منسون شده است.

#### **forms [] •**

آرایه ای از اشیای فرم، شامل یک شیء برای هر فرم موجود در سند.

#### **images [] •**

آرایه ای از اشیای تصویر، شامل یک شیء برای هر تصویری که با استفاده از برگه هی `<img>` در سند قرار داده شده است. جاوا اسکریپت ۱/۱.

#### **lastModified •**

رشته ای فقط -خواندنی، که تاریخ آخرین تغییر انجام شده روی سند را (بر اساس گزارش سرور شبکه) نشان می دهد. جاوا اسکریپت ۱/۰.

#### **linkColor •**

رشته ای که رنگ پیوندهای بازدید نشده را نشان می دهد. منسون شده است.

#### **links [] •**

آرایه ای از اشیای پیوند، شامل یک شیء برای هر پیوند ابرمتن موجود در سند.

**location •**

نشانی سند. به نفع خصلت URL منسوخ شده است.

**plugins[] •**

متراծ آرایه‌ی [ ] embeds. جاوا اسکریپت ۱/۱.

**referrer •**

رشته‌ای فقط-خواندنی، که نشانی سندی را که به سند جاری پیوند داده بود، در صورت وجود، نشان می‌دهد.

**title •**

محتوای متنی برگه‌ی <title>. تا قبل از DOM سطح ۱، فقط-خواندنی بود.

**URL •**

رشته‌ای فقط-خواندنی، که نشانی سند را نشان می‌دهد. جاوا اسکریپت ۱/۱.

**vlinkColor •**

رشته‌ای که رنگ پیوندهای بازدید شده را نشان می‌دهد. منسوخ شده است.

## ۳/۱۱/۴ خصلت‌های W3C DOM

در مرورگرهای سازگار با DOM، شیء سند خصلت‌های گره را به ارت می‌برد، و خصلت‌های اضافی زیر نیز برای آن تعریف شده است.

**body •**

اشاره‌ای به شیء عنصری که نشان دهنده‌ی برگه‌ی <body> این سند است.

**defaultView •**

پنجره‌ای که سند در آن نشان داده می‌شود. DOM سطح ۲.

**documentElement •**

اشاره‌ای فقط-خواندنی به برگه‌ی <html> سند.

### implementation •

شیء DOMImplementation که نشان دهنده‌ی پیاده‌سازی سازنده‌ی سند است. فقط-خواندنی.

## ۳/۱۱/۵ خصلت‌های اینترنت اکسپلورر ۴

خصلت‌های غیراستاندارد (و انتقال‌ناپذیر) زیر در اینترنت اکسپلورر ۴ و بعد از آن تعریف شده‌اند.

### activeElement •

خصلتی فقط-خواندنی که به عنصر ورودی‌ای که هم‌اکنون فعال است (یعنی کانون ورودی را در دست دارد) اشاره می‌کند.

### all[] •

آرایه‌ای از تمام اشیای عنصر موجود در سند. در این آرایه می‌توان بر اساس شماره به ترتیب به عناصر دستیابی پیدا کرد، و یا اینکه می‌توان به جای شماره از `id` یا `name` استفاده کرد.

### charset •

نویسگان سند.

### children[] •

آرایه‌ای متشكل از عنصرهای HTML که مستقیماً از سند منشعب شده‌اند. دقت کنید که این با `all` متفاوت است، زیرا `all` حاوی تمام عناصر موجود در سند صرف نظر از موقعیت آنها در درخت سند است.

### defaultCharset •

نویسگان پیش‌فرض سند.

### expando •

این خصلت، اگر مقدار `false` به آن داده شود، مانع از گسترش یافتن اشیای سمت مشتری می‌شود. به عبارت دیگر، اگر برنامه‌ای سعی داشته باشد که در شیء به خصلتی که وجود ندارد، مقدار بدهد، خطای زمان اجرا ایجاد می‌شود. این کار می‌تواند به یافتن اشکالات برنامه‌نویسی که ناشی از اشتباه تایپی در نوشتن نام خصلت است، کمک کند. این خصلت خصوصاً می‌تواند برای برنامه‌نویسانی که جدیداً به زبان جاوا اسکریپت

روی آورده‌اند، و از قبل عادت به زبان‌هایی دارند که حروف کوچک و بزرگ در آنها اهمیت ندارد، سودمند واقع شود. گرچه این خصلت فقط در اینترنت اکسپلورر کار می‌کند، ولی مقداردهی به آن در نتاسکیپ هم بی‌خطر است (گرچه اثربخش ندارد).

**parentWindow •**

پنجره‌ای که حاوی این سند است.

**readyState •**

وضعیت بار شدن سند را مشخص می‌کند. مقدار آن یکی از چهار رشته‌ی زیر است:

**uninitialized •**

سند شروع به بار شدن نکرده است.

**loading •**

سند در حال بار شدن است.

**interactive •**

سند به قدر کافی بار شده که کاربر بتواند با آن تعامل کند.

**complete •**

سند کاملاً بار شده است.

## ۴/۱۱/۳ خصلت‌های نتاسکیپ

خصلت‌های غیراستاندارد (و انتقال‌ناپذیر) زیر در نتاسکیپ ۴ تعریف شده‌اند.

**height •**

بلندای سند، بر حسب پیکسل.

**layers [] •**

آرایه‌ای از اشیای لایه که معرف لایه‌های موجود در سند است. این خصلت فقط در نتاسکیپ ۴ موجود است؛ از نتاسکیپ ۶ به بعد حذف شده است.

**width •**

پهنه‌ای سند، بر حسب پیکسل.

## ۳/۱۱/۷ روش‌های مشترک

تمام پیاده‌سازی‌های شیء سند روش‌های زیر را پشتیبانی می‌کنند. متعاقباً، لیست‌های جداگانه‌ای نیز برای روش‌های تعریف شده در W3C DOM، نت‌اسکیپ، و اینترنت اکسپلورر ارائه خواهیم کرد.

### **clear()** •

تمام محتویات سند را پاک می‌کند، و هیچ چیز بر نمی‌گردد. جاوا اسکریپت ۱/۰ در جاوا اسکریپت ۱/۱ منسوخ شده است.

### **close()** •

جریان سند را که با استفاده از روش `(open)` باز شده است، می‌بندد، و چیزی بر نمی‌گردد.

### **open()** •

محتویات فعلی سند را پاک می‌کند، و جریانی باز می‌کند که محتویات سند را می‌توان در آن نوشت. چیزی بر نمی‌گردد. جاوا اسکریپت ۱/۰.

### **write(value, ...)** •

رشته یا رشته‌های داده شده را در سندی که هم‌اکنون در حال تجزیه شدن است، درج می‌کند، و یا به آخر سندی که با `(open)` باز شده است، اضافه می‌کند.

### **writeln(value, ...)** •

همانند `(write)` است، جز اینکه به آخر خروجی یک نویسه‌ی سطر جدید اضافه می‌کند. چیزی بر نمی‌گردد. جاوا اسکریپت ۱/۰.

## ۳/۱۱/۸ روش‌های W3C DOM

در مرورگرهای سازگار با DOM، شیء سند روش‌های گره را به ارث می‌برد، و روش‌های اضافی زیر نیز برای آن تعریف شده است.

### **createAttribute(name)** •

یک گره جدید Attr با نام داده شده ایجاد می‌کند و بر می‌گردد.

### **createComment(text)** •

یک گره جدید Comment با متن داده شده ایجاد می‌کند و

بر می‌گرداند.

**createDocumentFragment()** •

یک گره جدید DocumentFragment خالی ایجاد می‌کند و بر می‌گرداند.

**createElement(tagName)** •

یک گره جدید Element با نام برگه‌ی داده شده ایجاد می‌کند و بر می‌گرداند.

**createTextNode(text)** •

یک گره جدید Text با متن داده شده ایجاد می‌کند و بر می‌گرداند.

**getElementById(id)** •

عنصری را در این سند که مقدار صفت id آن معادل با مقدار داده شده است، بر می‌گرداند، و اگر چنین عنصری موجود نباشد، null بر می‌گرداند.

**getElementsByName(name)** •

آرایه‌ای از گره‌های تمام عناصر موجود در سند را که مقدار صفت name آنها معادل مقدار داده شده است، بر می‌گرداند. اگر هیچ عنصری با این ویژگی موجود نباشد، آرایه‌ای با طول صفر بر می‌گرداند.

**getElementsByTagName(tagname)** •

آرایه‌ای از گره‌های تمام عناصر موجود در سند را که نام برگه‌ی آنها معادل نام داده شده است، بر می‌گرداند. ترتیب این عناصرها در این آرایه، همان ترتیب آنها در متن سند است.

**importNode(importedNode, deep)** •

یک نسخه از گرهی در سند دیگر که قابل درج در سند حاضر است، ایجاد می‌کند، و آن را بر می‌گرداند. اگر به آوند deep مقدار داده شود، به صورت تراجعی گره‌های منشعب شده را هم کپی می‌کند. .۲ سطح DOM

## ۳/۱۱/۹ روش‌های نت‌اسکیپ

`getSelection()` •

متن انتخاب شده‌ی فعلی در سند را بدون برگه‌های HTML بر می‌گرداند.

## ۳/۱۱/۱۰ روش‌های اینترنت اکسپلورر ۴

`elementFromPoint(x, y)` •

عنصری را که در نقطه‌ی داده شده واقع است، بر می‌گرداند.

## ۳/۱۱/۱۱ رویدادپردازها

در مرورگرهای سازگار با DOM و اینترنت اکسپلورر ۴، شیء سند همان رویدادپردازهای عمومی شیء عنصر را پشتیبانی می‌کند. گرچه پردازندۀ‌های `onunload` و `onload` از نظر منطقی به شیء سند تعلق دارند، ولی به عنوان خصلت‌های شیء پنجره پیاده‌سازی شده‌اند.

## ۳/۱۱/۱۲ ارجاع

`Link`, `Layer`, `Image`, `Form`, `Element`, `Applet`, `Anchor`  
`Window`

## DocumentFragment ۳/۱۲

(DOM سطح ۱)

گردهایی که با هم مورد دستکاری قرار می‌گیرند.

مشتق شده از: `Node`

## ۳/۱۲/۱ شرح

شیء `DocumentFragment`, روش‌ها و خصلت‌های `Node` را به ارث می‌برد، و هیچ خصلت یا روش جدیدی تعریف نمی‌کند. اما یک رفتار خیلی مهم دارد: وقتی که یک `DocumentFragment` در داخل سند درج می‌شود، خود شیء درج نمی‌شود، بلکه گرهای منشعب از آن در سند درج می‌گردد. این باعث می‌شود که `DocumentFragment` به عنوان جایگاهی برای نگهداری گردهایی که می‌خواهید همه را با هم در داخل سند درج کنید، بسیار سودمند باشد.

## ۳/۱۲/۲ ارجاع

`.Document.createDocumentFragment()`

## Element ۳/۱۳

DOM سطح ۱، اینترنت اکسپلورر ۴  
یک برگه‌ی HTML در یک سند  
مشتق شده از: Node (در DOM سطح ۱)

### ۳/۱۳/۱ شرح

شیء عنصر نشان دهنده‌ی یک عنصر یا برگه‌ی HTML است. اینترنت اکسپلورر ۴ و بعداً مرورگرهای سازگار با DOM، مانند اینترنت اکسپلورر ۵ به بالا و نت اسکیپ ۶ به بالا امکان دسترسی به تمام عناصر سند را فراهم می‌کنند. به علاوه، برای هر عنصر خصلت‌ها و روش‌های زیر را تعریف می‌کنند. متأسفانه، خصلت‌ها و روش‌های اینترنت اکسپلورر ۴ با خصلت‌ها و روش‌های W3C و استاندارد DOM متفاوت است. از این جهت، آنها را در گروههای مجزا در زیر بیان می‌کنیم.

### ۳/۱۳/۲ خصلت‌های W3C DOM

در مرورگرهای شبکه که از W3C DOM پشتیبانی می‌کنند، تمام عناصر در سند HTML خصلت‌هایی متناظر با صفت‌های HTML خود دارند، که صفت‌های عمومی، مانند `dir`, `id`, `lang`، و `title` نیز از این جمله‌اند. در مواردی که صفت HTML از چند کلمه تشکیل می‌شود، در خصلت متناظر با آن از حروف کوچک و بزرگ برای مشخص کردن کلمات استفاده می‌شود. در سایر موارد، خصلت‌ها در جاوا اسکریپت با حروف کوچک است (مانند `id` و `href` در مقابل `tagIndex` و `accessKey`). در دو مورد، نام صفت HTML کلمه‌ای ذخیره شده در زبان جاوا اسکریپت است، و لذا برای این صفات از نام‌های ویژه‌ای استفاده شده است. جاوا اسکریپت از خصلت `className` برای اشاره به صفت `class` مربوط به تمام برگه‌های HTML استفاده می‌کند. همچنین، برای صفت `for` مربوط به برگه‌های `<script>` و `<label>` از خصلت `htmlFor` استفاده می‌نماید. تمام عناصر، علاوه بر صفات HTML خود، دارای خصلت‌های زیر نیز هستند. در ضمن، به یاد داشته باشید که در مرورگرهای سازگار با DOM، تمام عنصرهای HTML خصلت‌های شیء `Node` را به ارث می‌برند.

#### **className •**

مقدار رشته‌ای صفت `class` عنصر، که یک یا چند کلاس CSS را ذکر می‌کند. دقت کنید که نام این خصلت `className` است، نه `class`، زیرا واژه‌ی اخیر از واژگان ذخیره شده‌ی جاوا اسکریپت است.

#### **style •**

یک شیء `Style` که معرف صفت `style` عنصر HTML است.

#### **tagName •**

نام برگه‌ی عنصر که یک خصلت فقط-خواندنی است؛ صرف نظر از اینکه نام برگه در سند با حروف بزرگ یا کوچک نوشته شده باشد، نام برگه با حروف بزرگ برگردانده می‌شود. در سندهای XHTML، این مقدار با حروف کوچک است.

### ۳/۱۳/۳ خصلت‌های DOM اینترنت اکسپلورر

اینترنت اکسپلورر ۴ و ویرایش‌های بعدی یک DOM تجاری تعریف کرده‌اند. در DOM اینترنت اکسپلورر ۴، همانند W3C DOM، هر عنصر HTML خواص جاوا اسکریپتی متناظر با صفت‌های HTML خود دارد. علاوه بر این، DOM اینترنت اکسپلورر ۴ خصلت‌های زیر را نیز برای هر عنصر تعریف کرده است:

#### **all[] •**

آرایه‌ای از تمام اشیای عنصر که از این عنصر منشعب شده‌اند. این آرایه را می‌توان شماره‌گذاری کرد، و به ترتیب حضور عناصر در متن سند به آنها دسترسی پیدا کرد. راه دیگر استفاده از `id` یا `name` برای یافتن عنصر در این آرایه است. همچنین، رک. `Document.all[]`.

#### **children[] •**

آرایه‌ای از اشیای عنصر که مستقیماً از این عنصر منشعب شده‌اند. دقت کنید که DOM اینترنت اکسپلورر ۴ هیچ شیئی متناظر با گره‌های متن یا توضیح ندارد، بنا بر این، فرزندان یک عنصر خود عنصرهای دیگری هستند.

#### **className •**

رشته‌ای خواندنی/نوشتني که مقدار صفت `class` یک عنصر را تعیین می‌کند.

**document •**

اشاره‌ای به شیء سندی که در بر دارنده‌ی این عنصر است.

**innerHTML •**

متن HTML موجود در درون عنصر، به استثنای برگه‌ی ورودی و خروجی خود عنصر. هر مقداری به این خصلت داده شود، جایگزین محتوای عنصر می‌شود. از آنجا که این خصلت غیر استاندارد بسیار قوی و پرکاربرد است، مروارگرهای دیگر هم مانند نت‌اسکیپ ۶ و موزیلا آن را پیاده‌سازی کرده‌اند.

**innerText •**

متن ساده‌ی موجود در درون عنصر، به استثنای برگه‌ی ورودی و خروجی خود عنصر. هر مقداری به این خصلت داده شود، آن را بدون تجزیه جایگزین محتوای عنصر می‌کند.

**offsetHeight •**

بلندای عنصر و کل محتوای آن بر حسب پیکسل.

**offsetLeft •**

مختصه‌ی X عنصر نسبت به عنصر گنجایه‌ی `.offsetParent`.

**offsetParent •**

عنصر گنجایه‌ای را تعریف می‌کند که مختصات `offsetLeft` و `offsetTop` بر حسب آن اندازه‌گیری می‌شود. برای اکثر عنصرها، `offsetParent` شیء سندی است که آنها را در خود جای داده است. اما اگر عنصری والدی داشته باشد که به صورت پویا محل آن تعیین شده باشد، همان والد به عنوان `offsetParent` تلقی می‌شود. خانه‌های جدول نسبت به ردیفی که در آن واقع شده‌اند، جای داده می‌شوند.

**offsetTop •**

مختصه‌ی Y عنصر، نسبت به عنصر گنجایه‌ی `.offsetParent`.

**offsetWidth •**

پهنه‌ای عنصر و کل محتویات آن بر حسب پیکسل.

**outerHTML •**

متن HTML عنصر، مشتمل بر برگه‌ی ورودی و خروجی خود عنصر. هر مقداری به این خصلت داده شود، به طور کامل جایگزین عنصر و محتویات آن می‌شود.

**outerText •**

متن ساده‌ی یک عنصر، مشتمل بر برگه‌ی ورودی و خروجی خود عنصر. هر مقداری به این خصلت داده شود، به صورت متن تجزیه نشده جایگزین عنصر و محتویات آن می‌شود.

**parentElement •**

عنصری که والد مستقیم این عنصر است. این خصلت فقط-خواندنی است.

**sourceIndex •**

نمایه‌ی عنصر در آرایه‌ی [ ] Document.all سندی که آن را در خود جای داده است.

**style •**

یک شیء شیوه که نشان دهنده‌ی صفت شیوه‌ی CSS مستقیم این عنصر است. تعیین خصلت‌های این شیء، شیوه‌ی نمایش این عنصر را تغییر می‌دهد.

**tagName •**

رشته‌ای فقط-خواندنی که نام برگه‌ی HTML این عنصر را نشان می‌دهد.

## ۳/۱۳/۴ W3C DOM روش‌های

در مرورگرهایی که از W3C DOM پشتیبانی می‌کنند، تمام عنصرها علاوه بر اینکه روش‌های Node را به ارث می‌برند، روش‌های زیر را نیز دارا هستند. بسیاری از این روش‌های برای خواندن یا نوشتمن مقادیر صفت‌ها به کار می‌روند، و به ندرت مورد استفاده قرار می‌گیرند، زیرا اشیای عنصر دارای خصلت‌هایی متناظر با هر کدام از صفات HTML خود هستند.

**getAttribute(name) •**

مقدار صفت مشخص شده را به صورت یک رشته بر می‌گرداند.

**getAttributeNode(name)** •

مقدار صفت مشخص شده را به صورت یک گره Attr برمی‌گرداند.

**getElementsByName(name)** •

آرایه‌ای از تمام عناصر مشق شده از این عنصر را که دارای نام برگهی داده شده هستند، به ترتیبی ظهور در متن سند برمی‌گرداند.

**hasAttribute(name)** •

اگر عنصر صفتی با نام داده شده داشته باشد، مقدار true و در غیر این صورت، مقدار false برمی‌گرداند. DOM سطح ۲.

**removeAttribute(name)** •

صفت مشخص شده را از این عنصر حذف می‌کند، و چیزی برمی‌گرداند.

**removeAttributeNode(oldAttr)** •

گره Attr مشخص شده را از لیست صفت‌های این عنصر حذف می‌کند.  
گره Attr حذف شده را برمی‌گرداند.

**setAttribute(name, value)** •

به صفتی که نام آن داده شده است، مقدار داده شده را اختصاص می‌دهد، و چیزی برمی‌گرداند.

**setAttributeNode(newAttr)** •

گره Attr مشخص شده را به لیست صفت‌های این عنصر اضافه می‌کند.  
اگر صفتی با همان نام از قبل موجود باشد، مقدار جدید جایگزین آن می‌شود. گره Attr جایگزین شده با newAttr را برمی‌گرداند، و یا در صورتی هیچگونه جایگزینی انجام نشده باشد، مقدار null برمی‌گرداند.

**۳/۱۳/۵ روش‌های DOM اینترنت اکسپلورر**

اینترنت اکسپلورر ۴ و ویرایش‌های بعدی، روش‌های غیراستاندارد زیر را برای همهی عنصرهای سند پشتیبانی می‌کنند.

**contains(target)** •

در صورتی که این عنصر حاوی عنصر target باشد، مقدار true و

در غیر این صورت، مقدار false بر می‌گرداند.

**getAttribute(name)** •

مقدار صفت مشخص شده از این عنصر را برابر می‌گرداند، و اگر چنین صفتی وجود نداشته باشد، مقدار null بر می‌گرداند.

**insertAdjacentHTML(where, text)** •

متن HTML را در نزدیکی این عنصر در موقعیتی که به وسیله‌ی where مشخص می‌شود، در داخل سند درج می‌کند. where باید یکی از رشته‌های "AfterBegin", "BeforeBegin", "AfterEnd" و یا "BeforeEnd" باشد. چیزی بر نمی‌گرداند.

**insertAdjacentText(where, text)** •

متن ساده‌ی text را در نزدیکی این عنصر در موقعیتی که به وسیله‌ی where مشخص می‌شود، در داخل سند درج می‌کند. چیزی بر نمی‌گرداند.

**removeAttribute(name)** •

صفت مشخص شده و مقدار آن را از عنصر حذف می‌کند. در صورت موقیت مقدار true، و در غیر این صورت، مقدار false بر می‌گرداند.

**scrollIntoView(top)** •

سند را در صفحه جا به جا می‌کند تا اینکه این عنصر در بالا یا پایین پنجره قرار گیرد. اگر به top مقدار true داده شود و یا اینکه حذف شود، عنصر در بالای پنجره ظاهر می‌شود. اگر به top مقدار false داده شود، عنصر در پایین پنجره ظاهر می‌شود.

**setAttribute(name, value)** •

به صفت مشخص شده، مقدار رشته‌ای داده شده را اختصاص می‌دهد، و هیچ چیز بر نمی‌گرداند.

## رویدادپردازها ۳/۱۳/۶

عنصرهای یک سند HTML رویدادپردازهای زیر را برای پاسخ به رویدادهای خام موشواره و صفحه کلید تعریف می‌کنند. انواع خاص عنصر (مانند اشیای FORM و

(Input) ممکن است رویدادپردازهای تخصصی تری (مثلًاً onsubmit و onchange) تعریف کنند، که به رویدادهای ورودی خام تفسیرهای خاصی را نسبت می‌دهند.

#### **onclick •**

هنگامی که کاربر روی عنصر کلیک می‌کند، فراخوانده می‌کند.

#### **ondblclick •**

هنگامی که کاربر روی عنصر دوکلیک می‌کند، فراخوانده می‌کند.

#### **onhelp •**

هنگامی که کاربر تقاضای کمک می‌کند، فراخوانده می‌کند. فقط اینترنت اکسپلورر.

#### **onkeydown •**

هنگامی که کاربر کلیدی را فشار می‌دهد، فراخوانده می‌کند.

#### **onkeypress •**

هنگامی که کاربر کلیدی را فشار داده و رها می‌کند، فراخوانده می‌کند.

#### **onkeyup •**

هنگامی که کاربر کلیدی را رها می‌کند، فراخوانده می‌کند.

#### **onmousedown •**

هنگامی که کاربر یک دکمهٔ موشواره را فشار می‌دهد، فراخوانده می‌کند.

#### **onmousemove •**

هنگامی که موشواره حرکت می‌کند، فراخوانده می‌شود.

#### **onmouseout •**

هنگامی که کاربر موشواره را از روی عنصر خارج می‌کند، فراخوانده می‌کند.

#### **onmouseover •**

هنگامی که کاربر موشواره را روی عنصر حرکت می‌دهد، فراخوانده می‌کند.

### onmouseup •

هنگامی که کاربر یک دکمه‌ی موشواره را رها می‌کند، فراخوانده می‌کند.

## ۳/۱۳/۷ ارجاع

.Textarea، Select، Node، Input، Form

### Error ۳/۱۴

(جاوا اسکریپت هسته ۱/۵؛ جی اسکریپت ۵/۵؛ ECMA ۳)

انواع استثنای از پیش تعریف شده.

مشتق شده از Object

## ۳/۱۴/۱ سازنده

```
new Error(message)
new EvalError(message)
new RangeError(message)
new ReferenceError(message)
new SyntaxError(message)
new TypeError(message)
new URIError(message)
```

این سازنده‌ها نمونه‌ای از کلاس Error و یا یکی از زیرکلاس‌های آن می‌سازند. آوند message اختیاری است.

### ۳/۱۴/۲ خصلت‌ها

و تمام زیرکلاس‌های آن دو خصلت دارند:

#### message •

پیغام خطایی که جزئیات استثنا را نشان می‌دهد. این خصلت حاوی رشته‌ی داده شده به سازنده و یا یک رشته‌ی پیش‌فرض وابسته به پیاده‌سازی است.

#### name •

رشته‌ای که نوع استثنا را نشان می‌دهد. این خصلت همواره نام سازنده‌ای است که برای ساختن شیء استثنا مورد استفاده قرار گرفته است.

## ۳/۱۴/۳ روش‌ها

### **toString()** •

نمایش رشته‌ای این شیء خطرا برابر می‌گرداند.

## Event ۳/۱۵

DOM سطح ۲، اینترنت اکسپلورر ۴، نت‌اسکیپ ۴)

جزئیات رویداد

## ۳/۱۵/۱ شرح

شیء رویداد هم جزئیات رویداد را مشخص می‌کند، و هم چگونگی انتشار آن را تعیین می‌نماید. DOM سطح ۲ یک شیء رویداد استاندارد تعریف کرده است، ولی اینترنت اکسپلورر ۴، ۵ و ۶ به جای آن از شیء مخصوصی استفاده می‌کند. نت‌اسکیپ ۴ نیز شیء مخصوصی دارد که با هر دو متفاوت است. DOM سطح ۲ رویدادهای صفحه کلید را استانداردسازی نکرده است، بنا بر این، شیء رویداد نت‌اسکیپ ۴ ممکن است هنوز هم برای برنامه‌نویسانی که به رویدادهای کلید در نت‌اسکیپ ۶ و بعد را آن علاقه‌مندند، سودمند باشد. خصلت‌های شیء رویداد DOM، اینترنت اکسپلورر، و نت‌اسکیپ ۴ ذیلا در قسمت‌های جداگانه ارائه می‌شود. در مدل‌های رویداد DOM و نت‌اسکیپ، شیء رویداد به عنوان آوند به رویدادپرداز داده می‌شود. اما در مدل رویداد اینترنت اکسپلورر، شیء رویدادی که جدیدترین رویداد را توصیف می‌کند، در خصلت event شیء پنجره ذخیره می‌شود.

## ۳/۱۵/۲ ثابت‌های DOM

این ثابت‌ها مقادیر مجاز برای خصلت eventPhase هستند؛ اینها فاز کنونی انتشار رویداد را برای رویداد مربوطه نشان می‌دهند.

### **Event.CAPTURING\_PHASE = 1** •

رویداد در مرحله ارسال است.

### **Event.AT\_TARGET = 2** •

رویداد در حال پردازش شدن توسط گره هدف است.

#### **Event.BUBBLING\_PHASE = 3 •**

رویداد در مرحله‌ی انتشار جبابی است.

### **DOM ۳/۱۵/۳ خصلت‌های**

تمام خصلت‌های این شیء فقط خواندنی‌اند.

#### **altKey •**

اگر در زمان بروز رویداد، کلید Alt فشار داده شده باشد، مقدار آن true است. برای رویدادهای موشواره تعریف شده است.

#### **bubbles •**

اگر رویداد از نوعی باشد که انتشار جبابی پیدا می‌کند، مقدار آن true است. برای تمام رویدادها تعریف شده است.

#### **button •**

مشخص می‌کند که در رویدادهای فشار داده شدن یا رها شدن و یا کلیک کردن، وضعیت کدام دکمه‌ی موشواره تغییر کرده است. معرف دکمه‌ی چپ، ۱ دکمه‌ی وسط، و ۲ دکمه‌ی راست است. دقت کنید که این خصلت فقط زمانی تعریف شده است که دکمه‌ای تغییر وضعیت بدهد: بنا بر این، مثلاً زمانی که در حین بروز یک رویداد حرکت موشواره دکمه‌ای پایین نگهداشته شده است، این خصلت استفاده نمی‌شود. به علاوه، این خصلت به صورت چندیتی نیست، یعنی اگر بیش از یک دکمه‌ی موشواره پایین نگهداشته شده باشد، آنها را گزارش نمی‌دهد. نت‌اسکیپ ۶/۰ به جای ۰، ۱، و ۲ از ۱، ۲، و ۳ استفاده می‌کند. این مسئله در نت‌اسکیپ ۶/۱ اصلاح شده است.

#### **cancelable •**

در صورتی که بتوان با فراخوانی preventDefault() عمل پیش‌فرض مربوط به رویداد را لغو کرد، مقدار این خصلت true و در غیر این صورت، false است.

#### **clientX, clientY •**

این خصلت‌ها مختصات X و Y نشانگر موشواره را نسبت به ناحیه‌ی مشتری پنجره‌ی مرورگر نشان می‌دهند. دقت کنید که این خصلت‌ها در نوردهیدن سند را در نظر نمی‌گیرند.

**ctrlKey •**

اگر در زمان بروز رویداد، کلید Ctrl فشار داده شده باشد، مقدار آن true است. برای رویدادهای موشواره تعریف شده است.

**currentTarget •**

گرہ سند کہ ہم اکنون در حال رسیدگی به این رویداد است. در حین گرفتن و انتشار حبابی رویداد، این گرہ متفاوت از گرہ target است. برای تمام رویدادها تعریف شده است.

**detail •**

تعداد کلیک: ۱ برای تک کلیک، ۲ برای دو کلیک، ۳ برای سه کلیک، و الخ. برای رویدادهای کلیک، فشار داده شدن، و رہا شدن موشواره تعریف شده است.

**eventPhase •**

مرحله‌ی کنونی انتشار رویداد. ثابت‌های تعریف شده در بالا سه مقدار مجاز این خصلت را نشان می‌دهند. برای تمام رویدادها تعریف شده است.

**metaKey •**

اگر در زمان بروز رویداد، کلید Meta فشار داده شده باشد، مقدار آن true است. برای رویدادهای موشواره تعریف شده است.

**relatedTarget •**

برای رویدادهای mouseover، این گرہی است که موشواره قبل از وارد شدن به گرہ فعلی از آن خارج شده است. برای رویدادهای mouseout، این گرہی است که موشواره بعد از خارج شدن از گرہ فعلی به آن وارد شده است. برای سایر انواع رویداد، تعریف نشده است.

**screenX, screenY •**

این خصلت‌ها مختصات X و Y نشانگر موشواره را نسبت به گوشی بالا و چپ صفحه‌ی کاربر تعیین می‌کنند. برای رویدادهای موشواره تعریف شده است.

**shiftKey •**

اگر در زمان بروز رویداد، کلید Shift فشار داده شده باشد، مقدار آن true است. برای رویدادهای موشواره تعریف شده است.

**target •**

گره هدف این رویداد؛ گرهی که رویداد را ایجاد کرده است. دقت کنید که این می‌تواند هر گرهی، حتی گره متن باشد، و منحصر به گرههای عنصر نیست. برای تمام رویدادها تعریف شده است.

**timeStamp •**

یک شیء تاریخ که تاریخ و ساعت بروز رویداد را نشان می‌دهد. برای تمام رویدادها تعریف شده است، ولی پیاده‌سازی‌ها مجبور نیستند تاریخ معتبری را گزارش کنند.

**type •**

نوع رویداد. این همان نام رویدادپرداز با برداشتن حروف `on` از آغاز آن است. مثلاً، رویداد `click`، `load`، و یا `mousedown`. برای تمام رویدادها تعریف شده است.

**view •**

شیء پنجره‌ای که رویداد در آن پدید آمده است.

## ۳/۱۵/۴ روش‌های DOM

**preventDefault() •**

به مرورگر شبکه می‌گوید که عمل پیش‌فرض مربوط به این رویداد را (اگر چنین عملی وجود داشته باشد)، انجام ندهد. اگر رویداد از نوع قابل لغو نباشد، این روش اثری ندارد. چیزی بر نمی‌گردد.

**stopPropagation() •**

از انتشار بیشتر رویداد در مراحل ارسال، هدف، و انتشار حبابی جلوگیری می‌کند. چیزی بر نمی‌گردد.

## ۳/۱۵/۵ خصلت‌های اینترنت اکسپلورر ۴

**altKey •**

مقداری بولی که مشخص می‌کند در زمان بروز رویداد، کلید `Alt` فشار داده شده است یا نه.

**button •**

در رویدادهای موشواره، خصلت button مشخص می‌کند چه کلید یا کلیدهای موشواره فشا رداده شده‌اند. این عدد صحیح فقط-خواندنی یک بیت‌ماسک است: بیت ۱ در صورتی روشن است که دکمه‌ی چپ موشواره فشار داده شده باشد؛ بیت ۲ در صورتی روشن است که دکمه‌ی راست موشواره فشار داده شده باشد؛ بیت ۴ در صورتی روشن است که دکمه‌ی میانی موشواره (در موشواره‌های سدکمه‌ای) فشار داده شده باشد.

**cancelBubble •**

اگر رویداد پرداز بخواهد از انتشار رویداد به اشیای بالاتر جلوگیری کند، باید به این خصلت مقدار true بدهد.

**clientX, clientY •**

مختصات X و Y محل بروز رویداد، نسبت به صفحه‌ی مرورگر شیکه.

**ctrlKey •**

مقداری بولی که مشخص می‌کند در زمان بروز رویداد، کلید Ctrl فشار داده شده است یا نه.

**fromElement •**

برای رویدادهای mouseout و mouseover به شیئی fromElement اشاره دارد که نشانگر موشواره از آن خارج می‌شود.

**keyCode •**

برای رویدادهای صفحه‌کلید، keyCode رمز نویسه‌ی یونیکد ایجاد شده به وسیله‌ی کلید مورد نظر را مشخص می‌کند.

**offsetX, offsetY •**

مختصات X و Y محل بروز رویداد، نسبت به سیستم مختصات عنصر مبدأ .srcElement رویداد (رک).

**returnValue •**

اگر به این خصلت مقداری داده شود، مقدار آن بر مقداری که واقعاً به وسیله‌ی رویداد پرداز برگردانده می‌شود، تقدم دارد. برای لغو عمل پیش‌فرض عنصر مبدائی که رویداد روی آن رخ داده است، به این خصلت مقدار false بدهید.

**screenX, screenY •**

مختصات X و Y محل بروز رویداد، نسبت به صفحه‌ی نمایش.

**shiftKey •**

مقداری بولی که مشخص می‌کند در زمان بروز رویداد، کلید Shift فشار داده شده است یا نه.

**srcElement •**

شیء پنجره، سند، و یا عنصری که رویداد را ایجاد کرده است.

**toElement •**

برای رویدادهای mouseover و mouseout به شیئی toElement اشاره دارد که نشانگر موشواره به آن داخل می‌شود.

**type •**

یک خصلت رشته‌ای که نوع رویداد را مشخص می‌کند. مقدار آن نام رویدادپرداز منهای پیشوند on است. بنا بر این، وقتی رویدادپرداز () فراخوانده می‌شود، مقدار خصلت type شیء رویداد click است.

**x, y •**

مختصات X و Y محل بروز رویداد. این خصلت، مختصات را نسبت به درونی ترین عنصر در بر گیرنده که با استفاده از CSS به صورت پویا تعیین محل شده است، بیان می‌کند.

## ۳/۱۵/۶ خصلت‌های نت‌اسکیپ

**height •**

فقط در رویدادهای resize تعیین می‌شود. بلندای جدید پنجره یا کادر را پس از تغییر اندازه مشخص می‌کند.

**layerX, layerY •**

مختصات X و Y محل بروز رویداد را نسبت به لایه‌ی در بر گیرنده مشخص می‌کند.

**modifiers •**

مشخص می کند که کدام کلیدهای تغییرگر در هنگام بروز رویداد، فشار داده شده‌اند. این مقدار عددی، یک بیت‌ماسک است که می‌تواند مشکل‌Event.ALTA\_MASK از کدام از ثابت‌های Event.META\_MASK، Event.CONTROL\_MASK، Event.SHIFT\_MASK باشد. به علت یک خطأ، این خصلت در نت‌اسکریپت ۶/۱ تعریف شده است.

**pageX, pageY •**

مختصات X و Y محل بروز رویداد، نسبت به صفحه‌ی مرورگر. دقت کنید که این مختصات نسبت به صفحه‌ی سطح بالا هستند، نه نسبت به لایه‌ی در بر گیرنده.

**screenX, screenY •**

مختصات X و Y محل بروز رویداد، نسبت به صفحه‌ی نمایش.

**target •**

شیء پنجره، سند، لایه، و یا عنصری که رویداد در آن بروز کرده است.

**type •**

یک خصلت رشته‌ای که نوع رویداد را مشخص می‌کند. مقدار آن نام رویدادپرداز منهای پیشوند on است. بنا بر این، وقتی رویدادپرداز onclick() فراخوانده می‌شود، مقدار خصلت type شیء رویداد click است.

**which •**

برای رویدادهای صفحه‌کلید یا موشواره which مشخص می‌کند که کدام کلید یا دکمه‌ی موشواره فشار داده یا رها شده است. برای رویدادهای صفحه‌کلید، این خصلت حاوی رمز نویسه‌ای کلید فشار داده شده است. برای رویدادهای موشواره، مقدار آن ۱، ۲، و یا ۳ است، که به ترتیب، دکمه‌ی چپ، میانی، و راست را نشان می‌دهد.

**width •**

فقط در رویدادهای resize تعیین می‌شود. پهنانی جدید پنجره یا کادر را پس از تغییر اندازه مشخص می‌کند.

### x, y •

مختصات X و Y محل بروز رویداد. این خصلت‌ها مترادف با layerX و layerY هستند، و موقعیت را نسبت به لایه‌ی در بر گیرنده (در صورت وجود) نشان می‌دهند.

## Form ۳/۱۶

(جاوا اسکریپت سمت مشتری ۱۰)

یک فرم ورودی HTML

مشتق شده از: Element

### ۳/۱۶/۱ دستور

```
document.forms[form_number]  
document.forms[form_name]  
document.form_name
```

### ۳/۱۶/۲ خصلت‌ها

شیء فرم، خصلت‌هایی را برای هر کدام از صفت‌های عنصر <form> در HTML، از قبیل action، encoding، method و target تعريف می‌کند. به علاوه، خصلت‌های زیر را هم تعريف می‌کند.

### elements [] •

آرایه‌ای فقط-خواندنی از شیء‌های ورودی که نشان دهنده‌ی عنصرهای موجود در فرم هستند. آرایه را می‌توان با شماره نمایه‌گذاری کرد، و یا برای عنصرهایی که مقدار صفت name آنها تعیین شده است، از نام استفاده کرد.

### length •

تعداد عنصرهای موجود در فرم. معادل elements.length است.

### ۳/۱۶/۳ روش‌ها

### reset () •

مقدار هر کدام از عناصر ورودی فرم را به مقدار پیش‌فرض بر می‌گرداند.

چیزی بر نمی‌گرداند. جاوا اسکریپت ۱/۱.

#### submit () •

فرم را تحویل می‌دهد، ولی رویدادپرداز onsubmit را فعال نمی‌کند.  
چیزی بر نمی‌گرداند.

### ۳/۱۶/۴ رویدادپردازها

#### onreset •

درست پیش از بر گرداندن مقدار عناصر فرم فراخوانی می‌شود. برای جلوگیری از بر گردانده شدن مقدار عنصرها، باید مقدار false بر گرداند.

#### onsubmit •

درست پیش از تحویل داده شدن فرم فراخوانی می‌شود. این رویدادپرداز، امکان بررسی صحت موارد وارد شده توسط کاربر را فراهم می‌کند. برای جلوگیری از تحویل داده شدن فرم، باید مقدار false بر گرداند.

### ۳/۱۶/۵ ارجاع

.Textarea ،Select ،Input Element

### Function ۳/۱۷

(جاوا اسکریپت هسته ۱/۰؛ جی اسکریپت ۱/۰؛ ECMA ۱) اسکریپت تابع جاوا اسکریپت.

### ۳/۱۷/۱ سازنده

```
new Function(argument_names..., body)
```

این سازنده در جاوا اسکریپت ۱/۱ وارد شد. ولی در جاوا اسکریپت ۱/۲ منسوخ شده، و به جای آن از دستور مستقیم تعریف تابع استفاده می‌شود.

### ۳/۱۷/۲ خصلت‌ها

#### length •

تعداد آوندهای نام برده شده در هنگام تعریف تابع را نشان می‌دهد. برای

مشخص کردن تعداد واقعی آوندهای داده شده، می‌توانید از Arguments.length استفاده کنید. جاوا اسکریپت ۱، ECMA ۲، جی اسکریپت ۱،

#### prototype •

شیئی که برای یک تابع سازنده، تمام خصلت‌ها و روش‌هایی را تعریف می‌کند که بین اشیای ساخته شده توسط سازنده مشترک است. جاوا اسکریپت ۱؛ جی اسکریپت ۲؛ ECMA ۱،

### روش‌ها ۳/۱۷/۳

#### apply(thisobj, args) •

تابع را به عنوان روشی از thisobj فراخوانی می‌کند، و عنصرهای آرایه‌ی args را به عنوان آوندهای تابع به آن می‌دهد. مقدار بر گردانده شده توسط تابع را بر می‌گرداند. جاوا اسکریپت ۲؛ جی اسکریپت ۵؛ ECMA ۳،

#### call(thisobj, args...) •

تابع را به عنوان روشی از jz فراخوانی می‌کند، و آوندهای باقیمانده را به عنوان آوندهای تابع به آن می‌دهد. مقدار بر گردانده شده توسط تابع را بر می‌گرداند. جاوا اسکریپت ۵؛ جی اسکریپت ۵؛ ECMA ۳.

#### toString() •

نمایش رشته‌ای تابع را بر می‌گرداند. در برخی از پیاده‌سازی‌ها، نمایش رشته‌ای به صورت متن برنامه‌ی تابع است. جاوا اسکریپت ۰، جی اسکریپت ۲، ECMA ۱،

### ارجاع ۳/۱۷/۴

Arguments

#### Global ۳/۱۸

(۱) ECMA ۰؛ جی اسکریپت ۱؛ جاوا اسکریپت هسته ۱،

## خصلت‌ها و تابع‌های سراسری.

### ۳/۱۸/۱ دستور

this

### ۳/۱۸/۲ شرح

شیء Global، خصلت‌ها و روش‌های سراسری را در خود نگه می‌دارد. این خصلت‌ها و روش‌ها را لازم نیست از طریق شیء خاصی فراخوانی کنیم. هر گونه متغیر یا تابعی در متن سطح بالا تعریف کنید، تبدیل به خصلت‌های شیء سراسری می‌شود. شیء سراسری نامی ندارد، ولی می‌توانید در متن سطح بالا (یعنی در خارج از روش‌ها) با استفاده از کلیدواژه‌ی this به آن اشاره کنید. در جاوا اسکریپت سمت مشتری، شیء پسجهو به عنوان شیء سراسری عمل می‌کند. این شیء، خصلت‌ها و روش‌های متعدد دیگری نیز دارد، و می‌توان به صورت window به آن اشاره کرد.

### ۳/۱۸/۳ خصلت‌های سراسری

#### Infinity •

مقداری عددی که مثبت بینهایت را نشان می‌دهد. جاوا اسکریپت هسته .۱؛ جی اسکریپت .۳؛ ECMA .۱/۳

#### NaN •

مقدار غیر عددی. جاوا اسکریپت هسته .۱/۳؛ جی اسکریپت .۳؛ ECMA .۱

#### undefined •

مقدار تعریف نشده. جاوا اسکریپت هسته .۱/۵؛ جی اسکریپت .۵/۵؛ ECMA .۳

### ۳/۱۸/۴ تابع‌های سراسری

#### decodeURI(uri) •

یک نسخه‌ی رمزگشایی شده از uri را بر می‌گرداند، که در آن به جای سلسله‌های گریز شانزدهگانی، نویسه‌ی مربوطه قرار داده شده است. جاوا اسکریپت هسته .۱/۵؛ جی اسکریپت .۵/۵؛ ECMA .۳

#### **decodeURIComponent(s)** •

یک نسخه‌ی رمزگشایی شده از s را برمی‌گرداند، که در آن به جای سلسله‌های گریز شاتزدهگانی، نویسه‌ی مربوطه قرار داده شده است. جاوا اسکریپت هسته ۱/۵؛ جی اسکریپت ۵/۵؛ ECMA ۳.

#### **encodeURI(uri)** •

یک نسخه‌ی رمزگذاری شده از uri را برمی‌گرداند، که در آن به جای برخی نویسه‌ها، سلسله‌های گریز شاتزدهگانی قرار داده شده است. نویسه‌هایی مانند #، ?، و @ که برای جدا کردن اجزای URI به کار می‌روند، تبدیل به سلسله‌ی گریز نمی‌شوند. جاوا اسکریپت هسته ۱/۵؛ جی اسکریپت ۵/۵؛ ECMA ۳.

#### **encodeURIComponent(s)** •

یک نسخه‌ی رمزگذاری شده از s را برمی‌گرداند، که در آن به جای برخی نویسه‌ها، سلسله‌های گریز شاتزدهگانی قرار داده شده است. نویسه‌های نقطه‌گذاری را هم که برای جدا کردن اجزای URI به کار می‌روند، تبدیل به سلسله‌ی گریز می‌کند. جاوا اسکریپت هسته ۱/۵؛ جی اسکریپت ۵/۵؛ ECMA ۳.

#### **escape(s)** •

یک نسخه‌ی رمزگذاری شده از s را برمی‌گرداند، که در آن به جای برخی نویسه‌ها، سلسله‌های گریز شاتزدهگانی قرار داده شده است. جاوا اسکریپت هسته ۱/۰؛ جی اسکریپت ۱/۰؛ ECMA ۱؛ در ۳ ECMA منسوخ شده و به جای آن باید از encodeURI و encodeURIComponent استفاده کرد.

#### **eval(code)** •

رشته‌ای از متن جاوا اسکریپت را ارزیابی می‌کند، و حاصل را برمی‌گرداند.

#### **isFinite(n)** •

در صورتی که n عدد متناهی بوده و یا قابل تبدیل به چنین عددی باشد، true برمی‌گرداند. اگر مقدار n (یا حاصل تبدیل آن) NaN (غیرعدد) و یا مثبت یا منفی بی‌نهایت باشد، false برمی‌گرداند. جاوا اسکریپت ۱/۲؛ جی اسکریپت ۳/۰؛ ۱ ECMA.

**isNaN(x) •**

اگر مقدار x (یا حاصل تبدیل آن) NaN (غیر عدد) باشد، true بر می‌گرداند. در صورتی که x یک مقدار عددی بوده و یا قابل تبدیل به آن باشد، false بر می‌گرداند. جاوا اسکریپت ۱/۰ جی اسکریپت ۱/۰ .۱ ECMA

**parseFloat(s) •**

رشته‌ی s (و یا قطعه‌ی پیشوند s را) به یک عدد تبدیل می‌کند، و آن عدد را برابر می‌گرداند. اگر s با عدد معتبری شروع نشده باشد، مقدار NaN (و یا ۰ در جاوا اسکریپت ۱/۰) بر می‌گرداند. جاوا اسکریپت ۱/۰ جی اسکریپت ۱/۱ .۱ ECMA

**parseInt(s, radix) •**

رشته‌ی s (و یا قطعه‌ی پیشوند s را) به یک عدد صحیح تیدیل می‌کند، و آن عدد را برابر می‌گرداند. اگر s با عدد معتبری شروع نشده باشد، مقدار NaN (و یا ۰ در جاوا اسکریپت ۱/۰) بر می‌گرداند. آوند اختیاری radix، مبنای عدد (بین ۲ و ۳۶) را مشخص می‌کند. اگر این آوند داده نشده باشد، مبنای پیش‌فرض ۱۰، و یا در صورتی که رشته با پیشوند شانزده‌گانی یعنی "۰x" یا "۰X" شروع شده باشد، ۱۶ است. جاوا اسکریپت ۱/۰؛ جی اسکریپت ۱/۱ .۱ ECMA

**unescape(s) •**

رشته‌ای را که با استفاده از escape() رمزگذاری شده است، رمزگشایی می‌کند. نسخه‌ی رمزگشایی شده‌ی s را برابر می‌گرداند. جاوا اسکریپت ۱/۰؛ جی اسکریپت ۱/۰ .۱ در ۳ منسخه شده، و باید به جای آن از decodeURIComponent() استفاده کنید.

**ارجاع ۳/۱۸/۵**

Window

**History ۳/۱۹**

(جاوا اسکریپت سمت مشتری ۱/۰)

عقب یا جلو رفتن در سابقه‌ی مرور.

## ۳/۱۹/۱ دستور

`window.history`  
`history`

## ۳/۱۹/۲ روش‌ها

### `back()` •

در سابقه‌ی مرور، به نشانی مرور شده‌ی قبلی بر می‌گردد. چیزی بر نمی‌گرداند.

### `forward()` •

در سابقه‌ی مرور، جلو می‌رود. چیزی بر نمی‌گرداند.

### `go(n)` •

به نشانی `n` نسبت به نشانی فعلی می‌رود. اگر این روش با مقدار `-1` فراخوانده شود، همانند فراخواندن `(())` `back()` است. چیزی بر نمی‌گرداند.

## Image ۳/۲۰

جاوا اسکریپت سمت مشتری (۱/۱)

یک تصویر HTML

مشتق شده از: Element

## ۳/۲۰/۱ دستور

`document.images[i]`  
`document.images[image-name]`  
`document.image-name`

## ۳/۲۰/۲ سازنده

`new Image(width, height);`

این سازنده تصویری خارج از صفحه ایجاد می‌کند که نمی‌توان آن را نمایش داد. آوندهای `width` و `height` اختیاری هستند. اگر در شیء حاصله، مقدار صفت `src` تعیین شود، سبب می‌شود که مرورگر تصویر را به حافظه‌ی نهانی خود پیش‌خوانی کند.

## ۳/۲۰/۳ خصلت‌ها

شیء تصویر برای هر یک از صفت‌های عنصر `<img>`, از قبیل `src`, `width`, `height`, `border`, `vspace`, `hspace`, `height`, `width`, `border` خصلتی تعریف می‌کند. علاوه بر این، خصلت‌های خاص زیر را هم دارا است:

### **complete** •

اگر تصویر هنوز در حال بار شدن باشد، مقدار آن `false` است. اگر خواندن تصویر تمام شده و یا در اثنای آن خطایی بروز کرده است، مقدار آن `true` است. این خصلت فقط-خواندنی است.

### **src** •

رشته‌ای خواندنی/نوشتني که نشانی تصویر را نشان می‌دهد. این خصلت صرفاً معنکس کننده‌ی صفت `src` برگه‌ی `<img>` است، ولی علت توجه خاص به آن در اینجا این است که بسیاری از جلوه‌های مهم HTML پویا (DHTML) با تعیین مقدار این خصلت به صورت پویا انجام می‌شود، که سبب می‌شود تصویری با تصویر دیگر جایگزین شود.

## ۳/۲۰/۴ رویدادپردازها

شیء تصویر رویدادپرداز عنصر را به ارث می‌برد، و در ضمن، موارد زیر را هم تعریف می‌کند:

### **onabort** •

اگر کاربر فروگذاری تصویر را قطع کند، فراخوانده می‌شود.

### **onerror** •

اگر هنگام فروگذاری تصویر خطایی بروز کند، فراخوانده می‌شود.

### **onload** •

زمانی که بار شدن تصویر با موفقیت به اتمام رسید، فراخوانده می‌شود.

## Input ۳/۲۱

(جاوا اسکریپت سمت مشتری ۱/۰)

یک عنصر ورودی فرم

مشتق شده از: Element

## ۳/۲۱/۱ دستور

```
form.elements[i]
form.elements[name]
form.name
```

### ۳/۲۱/۲ خصلت‌ها

شیء ورودی خصلت‌هایی متناظر با هر یک از صفت‌های برگهی `checked`، `disabled`، `readOnly`، `tabIndex`، `size` و `maxLength` از قبیل `<input>` دارد. علاوه بر این، خصلت‌های زیر را نیز تعریف می‌کند:

#### **checked** •

یک مقدار بولی خواندنی/نوشتنی که مشخص می‌کند یک عنصر ورودی از نوع چک‌باکس یا دکمه‌ی رادیویی، چک شده (`true`) است یا نه (`false`).

#### **defaultChecked** •

یک مقدار بولی خواندنی/نوشتنی که مشخص می‌کند یک عنصر ورودی از نوع چک‌باکس یا دکمه‌ی رادیویی، وقتی در آغاز تشکیل می‌شود و یا به حالت اولیه بر گردانده می‌شود، چک شده است یا نه.

#### **defaultValue** •

مقداری رشته‌ای که متن نشان داده شده در ورودی «متن» یا «گذرواژه» در آغاز تشکیل و یا پس از بر گردانده شدن به مقدار اولیه را مشخص می‌کند. به دلایل امنیتی، این خصلت بر عناصر ورودی از نوع «برونده» اثری ندارد.

#### **form** •

اشاره‌ای فقط-خواندنی به شیء فرمی که عنصر در آن قرار گرفته است. این خصلت برای تمام انواع عنصرهای ورودی تعریف شده است.

#### **name** •

نام این عنصر ورودی، که به وسیله‌ی صفت `name` در HTML تعیین می‌شود. این خصلت برای تمام انواع عنصرهای ورودی تعریف شده است.

#### **type** •

رشته‌ای که نوع عنصر فرم را مشخص می‌کند. این خصلت منعکس

کنده‌ی مقدار صفت type در HTML است. مقادیر قانونی آن در جدول زیر لیست شده است؛ مقدار پیش‌فرض "text" است. اشای Textarea و Submit هم یک خصلت دارند که مقادیر ممکن آن "textarea"، "select-multiple"، "select-one" است. جاوا اسکریپت ۱/۱.

نوع	شرح
"button"	دکمه
"checkbox"	چک‌باکس
"file"	عنصر فراگذاری پرونده
"hidden"	عنصر پنهان
"image"	دکمه‌ی گرافیکی تحویل فرم
"password"	فیلد ورود متن مخفی شده
"radio"	دکمه‌های رادیویی مانعه‌الجمع
"reset"	دکمه‌ی برگردان
"text"	ورودی متن یک‌سطری
"submit"	دکمه‌ی تحویل فرم

#### value •

مقدار رشته‌ای که هنگام تحویل فرم فرستاده می‌شود. برای عناصر ورودی از نوع متن، گذر واژه، و پرونده، این متن قابل ویرایشی نشان داده شده در درون عنصر است. می‌توانید برای تغییر مقدار نمایش داده شده، مقدار این خصلت را تغییر دهید. برای عناصرهای ورودی از نوع دکمه، تحویل، و برگردان، مقدار این خصلت رشته‌ای است که به عنوان برجسب روی دکمه نمایش داده می‌شود. در انواع دیگر، رشته‌ی value نمایش داده نمی‌شود. دقت کنید که به دلایل امنیتی، خصلت value در عناصرهای نوع پرونده معمولاً فقط خواندنی است.

#### روش‌ها ۳/۲۱/۳

##### blur() •

کانون صفحه کلید را تحویل می‌دهد و چیزی بر نمی‌گرداند. برای تمام عناصرها غیر از نوع پنهان تعیین شده است.

##### click() •

مانند کلیک شدن موشواره روی عنصر عمل می‌کند، و چیزی

بر نمی گرداند. برای انواع عنصرهای دکمه‌ای — دکمه، چک‌باکس، رادیو، برگردان، و تحويل — تعریف شده است.

**focus()** •

کانون صفحه کلید را می‌گیرد و چیزی بر نمی گرداند. برای تمام عنصرها غیر از نوع پنهان تعیین شده است.

**select()** •

متن موجود در عنصر را انتخاب می‌کند و چیزی بر نمی گرداند. برای عنصرهای نوع متن، گذرواژه، و پرونده کار می‌کند. برای شیء Textarea هم تعریف شده است.

**رویدادپردازها ۳/۲۱/۴**

**onblur** •

وقتی که عنصر کانون صفحه کلید را تحويل می‌دهد، فراخوانی می‌شود. برای تمام عنصرها غیر از نوع پنهان تعیین شده است.

**onchange** •

این رویدادپرداز، برای عنصرهای ورود متن، شامل متن، گذرواژه، و پرونده، زمانی فراخوانده می‌شود که کاربر متن درون عنصر را تغییر می‌دهد و کانون صفحه کلید را از عنصر بیرون می‌برد، بدان معنا که کار ویرایش تمام شده است. به عبارت دیگر، برای هر کلیدی که زده می‌شود، این فراخوانی صورت نمی‌پذیرد.

**onclick** •

این رویدادپرداز، برای عنصرهای دکمه‌ای — دکمه، چک‌باکس، رادیو، برگردان، و تحويل — هنگام کلیک شدن موشاوه روی عنصر فراخوانده می‌شود. برای انواع تحويل و برگردان، اگر اینتابع false برابر گرداند، کار تحويل یا برگردان انجام نمی‌شود.

**onfocus** •

هنگامی که عنصر کانون صفحه کلید را دریافت می‌کند، فراخوانی می‌شود. برای تمام عنصرها غیر از نوع پنهان تعیین شده است.

## ۳/۲۱/۵ ارجاع

.Textarea ,Select ,Option ,Form

## Layer ۳/۲۲

(فقط نت اسکیپ ۴ سمت مشتری)  
یک لایه‌ی مستقل سند.

### ۳/۲۲/۱ دستور

```
document.layers[i]  
document.layers[layer-name]  
document.layer-name
```

### ۳/۲۲/۲ سازنده

```
new Layer(width, parent_layer)
```

### ۳/۲۲/۳ شرح

شیء لایه فقط در نت اسکیپ ۴ پشتیانی می‌شود، و در نت اسکیپ ۶ حذف شده است. این شیء کاملاً غیراستاندارد است، ولی علت ذکرشن در اینجا این است که تنها راه تعیین پویایی جای اشیا در نت اسکیپ ۴ است. هر عنصر HTML که به صفت position CSS آن مقدار absolute داده شده باشد، در جاوا اسکریپت با یک لایه نمایش داده می‌شود. در ضمن، می‌توانید با برگه‌ی غیراستاندارد <layer> یا با سازنده‌ی () لایه ایجاد کنید.

### ۳/۲۲/۴ خصلت‌ها

#### above •

لایه‌ی روی این لایه، اگر وجود داشته باشد. فقط -خواندنی.

#### background •

تصویر زمینه‌ی لایه.

#### below •

لایه‌ی زیر این لایه، اگر وجود داشته باشد. فقط -خواندنی.

**bgColor •**

رنگ زمینه‌ی لایه.

**clip.bottom •**

مختصه‌ی Y لبه‌ی پایینی ناحیه‌ی برش لایه، نسبت به top.

**clip.height •**

بلندای ناحیه‌ی برش لایه. با دادن مقدار جدید به آن، مقدار clip.bottom هم تغییر می‌کند.

**clip.left •**

مختصه‌ی X لبه‌ی چپ ناحیه‌ی برش لایه، نسبت به left.

**clip.right •**

مختصه‌ی X لبه‌ی راست ناحیه‌ی برش لایه، نسبت به right.

**clip.top •**

مختصه‌ی Y لبه‌ی بالایی ناحیه‌ی برش لایه، نسبت به top.

**clip.width •**

پهنه‌ی ناحیه‌ی برش لایه. با دادن مقدار جدید به آن، مقدار clip.right هم تغییر می‌کند.

**document •**

اشاره‌ای فقط-خواندنی به شیء سند که در درون لایه واقع است.

**hidden •**

پنهان یا مرئی بودن لایه را مشخص می‌کند. برای پنهان کردن لایه به آن مقدار true بدهید، و برای مرئی کردن آن، مقدار false به آن بدهید.

**layers[] •**

آرایه‌ای حاوی تمام شیء‌های لایه که از این لایه منشعب شده‌اند. این همانند آرایه‌ی لایه‌های document.layers[] است.

**left •**

مختصه‌ی X این لایه، نسبت به لایه یا سند حاوی آن. تغییر مقدار این خصلت سبب انتقال لایه به چپ یا راست می‌شود. left مترادف با x است.

**name •**

صفت name برگه‌ی HTML مربوط به این لایه.

**pageX, pageY •**

مختصه‌ی X و Y این لایه نسبت به سند سطح بالا. دقت کنید که این مختصات نسبت به صفحه‌ی سطح بالا است، نه نسبت به هیچکدام از لایه‌های حاوی این لایه.

**parentLayer •**

شاره‌ای فقط-خواندنی به لایه یا پنجره‌ی حاوی (والد) این لایه.

**siblingAbove, siblingBelow •**

این خصلت‌ها لایه‌ی رو یا زیر این لایه را که با آن برادر است (یعنی والد یکسانی دارد) نشان می‌دهد. اگر چنین لایه‌هایی وجود نداشته باشد، این خصلت null بر می‌گرداند.

**src •**

رشته‌ای خواندنی/نوشتني که نشانی محتویات لایه را مشخص می‌کند. دادن نشانی جدید به این خصلت سبب می‌شود که مروارگر محتویات آن نشانی را بخواند و آن را در لایه نمایش دهد.

**top •**

مختصه‌ی Y این لایه، نسبت به لایه یا سند در برگیرنده‌ی آن. تعیین این خصلت سبب بالا یا پایین رفتن لایه می‌شود. top متراff با y است.

**visibility •**

رشته‌ای خواندنی/نوشتني که مرئی بودن لایه را مشخص می‌کند. سه مقدار قانونی دارد: "inherit"، "show" و "hide".

**window •**

شیء پنجره‌ای که این لایه، صرف نظر از عمق، در آن قرار گرفته است.

**x, y •**

مختصات X و Y لایه. x متراff با left، و y متراff با خصلت top است.

**zIndex •**

موقعیت لایه در ترتیب `z` یا ترتیب پشته‌ای لایه‌ها. وقتی دو لایه روی هم می‌افتد، لایه‌ای که `zIndex` بالاتری داشته باشد، در بالا قرار می‌گیرد و لایه‌ی دیگر را محو می‌کند. اگر دو لایه‌ی همسطح دارای `zIndex` مساوی باشند، لایه‌ای که در آرایه‌ی `[layers]` بعدتر قرار گرفته است، بعداً ظاهر می‌شود و لذا روی لایه‌ای که قبلاً ظاهر شده است، می‌افتد.

**۳/۲۲/۵ روش‌ها****`load(src, width)` •**

یک نشانی را به لایه بار می‌کند، مقدار پهنای لایه را مطابق مقدار داده شده تنظیم می‌کند، و چیزی بر نمی‌گرداند.

**`moveAbove(other_layer)` •**

این لایه را به بالای لایه‌ی دیگر می‌برد، و چیزی بر نمی‌گرداند.

**`moveBelow(other_layer)` •**

این لایه را به زیر لایه‌ی دیگر می‌برد، و چیزی بر نمی‌گرداند.

**`moveBy(dx, dy)` •**

لایه را نسبت به موقعیت فعلی حرکت می‌دهد، و چیزی بر نمی‌گرداند.

**`moveTo(x, y)` •**

لایه را به موقعیت `(y, x)` نسبت به لایه‌ی والد یا پنجره می‌برد، و چیزی بر نمی‌گرداند.

**`moveToAbsolute(x, y)` •**

لایه را به موقعیت `(y, x)` نسبت به صفحه می‌برد، و چیزی بر نمی‌گرداند.

**`resizeBy(dw, dh)` •**

اندازه‌ی لایه را به میزان داده شده تغییر می‌دهد، و چیزی بر نمی‌گرداند.

**`resizeTo(width, height)` •**

اندازه‌ی لایه را به اندازه‌ی داده شده تغییر می‌دهد، و چیزی بر نمی‌گرداند.

## Link ۳/۲۳

(جاوا اسکریپت سمت مشتری ۱/۰)

یک پیوند <a> یا <area>

مشتق شده از: Element

### ۳/۲۳/۱ دستور

```
document.links[i]
```

### ۳/۲۳/۲ خصلت‌ها

بسیاری از خصلت‌های پیوند، در واقع، معرف قسمتی از نشانی آن هستند. برای هر کدام از خصلت‌های زیر، مقداری که داده می‌شود، بر اساس نشانی (فرضی) زیر است:

```
http://www.oreilly.com:1234/catalog/search.html?  
q=JavaScript&m=10#results
```

#### hash •

یک خصلت رشته‌ای خواندنی/نوشتني که قسمت لنگر نشانی را، به همراه نویسه‌ی # ابتدای آن، مشخص می‌کند. مثال: "#result".

#### host •

یک خصلت رشته‌ای خواندنی/نوشتني که قسمت نام میزبان و شماره‌ی درگاه نشانی را مشخص می‌کند. مثال: "www.oreilly.com:1234".

#### hostname •

یک خصلت رشته‌ای خواندنی/نوشتني که قسمت نام میزبان نشانی را مشخص می‌کند. مثال: "www.oreilly.com".

#### href •

یک خصلت رشته‌ای خواندنی/نوشتني که متن کامل نشانی را نشان می‌دهد.

#### pathname •

یک خصلت رشته‌ای خواندنی/نوشتني که قسمت نام مسیر نشانی را مشخص می‌کند. مثال: "/catalog/search.html".

**port •**

یک خصلت رشته‌ای خواندنی/نوشتی که قسمت شماره‌ی درگاه نشانی را مشخص می‌کند. مثال: "1234".

**protocol •**

یک خصلت رشته‌ای خواندنی/نوشتی که قسمت پروتکل نشانی را مشخص می‌کند. مثال: "http:".

**search •**

یک خصلت رشته‌ای خواندنی/نوشتی که قسمت پرسجوی نشانی را، به همراه نویسه‌ی ? ابتدای آن، مشخص می‌کند. مثال: ".?q=JavaScript&m=10"

**target •**

یک خصلت رشته‌ای خواندنی/نوشتی که نام یک شیء پنجره (یعنی کادر یا پنجره‌ی سطح بالای مرورگر) را که سند پیوندی باید در آن نمایش داده شود، مشخص می‌کند. این خصلت معادل صفت استاندارد target در HTML است. در ضمن، می‌توان از نام‌های خاص "`_blank`"، "`_parent`"، "`_self`" و "`_top`" استفاده کرد.

**رویدادپردازها ۳/۲۳/۳**

**onclick •**

وقتی کاربر روی پیوند کلیک می‌کند، فراخوانی می‌شود. در جاوا اسکریپت ۱/۱، این رویدادپرداز می‌تواند با برگرداندن مقدار `false`، مانع از تعقیب پیوند شود.

**onmouseout •**

وقتی کاربر موشواره را از روی عنصر بیرون می‌برد، فراخوانی می‌شود. در جاوا اسکریپت ۱/۱

**onmouseover •**

وقتی کاربر موشواره را روی پیوند می‌برد، فراخوانی می‌شود. در اینجا می‌توان مقداری برای خصلت `status` پنجره‌ی فعلی تعیین کرد. اگر مقدار `true` برگردانید، مرورگر می‌فهمد که نباید نشانی را در سطر

وضعیت نمایش دهد.

## ۳/۲۳/۴ ارجاع

`Location`, `Anchor`

## Location ۳/۲۴

(جاوا اسکریپت سمت مشتری)  
مکان کنونی مرورگر.

## ۳/۲۴/۱ دستور

`location`  
`window.location`

## ۳/۲۴/۲ خصلت‌ها

شیء مکان دارای همان خصلت‌های مربوط به نشانی موجود در شیء پیوند است، به استثنای `target`. برای شرح `hostname`, `host`, `hash`, `search`, `protocol`, `port`, `pathname` و `search` به شیء پیوند مراجعه کنید. تعیین هر کدام از این صفات سبب می‌شود که مرورگر سند را از نشانی جدید بار کند. به عنوان میانبر، می‌توانید با اختصاص دادن یک رشته‌ی نشانی به خصلت `location` پنجره، سند جدیدی بار کنید.

## ۳/۲۴/۳ روش‌ها

### `reload(force)` •

سند جاری را از حافظه‌ی نهان و یا سرور دوباره می‌خواند. آوند `force` اختیاری است. اگر مقدار آن `true` باشد، باعث بازخوانی کامل می‌شود، ولو آنکه سند تغییر نکرده باشد. چیزی بر نمی‌گرداند. جاوا اسکریپت ۱/۱

### `replace(url)` •

سند جاری را با سند دیگری جایگزین می‌کند، بدون اینکه درایه‌ی جدیدی در سابقه‌ی مرورگر ایجاد کند. چیزی بر نمی‌گرداند. جاوا اسکریپت ۱/۱

## ۳/۲۴/۴ ارجاع

.Window.location Link

## ۳/۲۵ Math

(جاوا اسکریپت هسته ۱/۰، جی اسکریپت ۱/۰) ECMA تابع‌ها و ثابت‌های ریاضی.

### ۳/۲۵/۱ دستور

Math.constant  
Math.function()

### ۳/۲۵/۲ شرح

شیء Math محلی برای تابع‌ها و ثابت‌های ریاضی است. این کلاس، بر خلاف Date و String، طبقه‌ای از اشیا را تعریف نمی‌کند. هیچگونه سازنده‌ی وجود ندارد، و توابعی مانند Math.sin() فقط تابع‌اند، نه روش‌هایی که روی شیء عمل کنند.

### ۳/۲۵/۳ ثابت‌ها

#### Math.E •

ثابت  $e$ ، مبنای لگاریتم طبیعی.

#### Math.LN10 •

لگاریتم طبیعی  $10$ .

#### Math.LN2 •

لگاریتم طبیعی  $2$ .

#### Math.LOG10E •

لگاریتم مبنای  $10$   $e$ .

#### Math.LOG2E •

لگاریتم مبنای  $2$   $e$ .

**Math.PI •**

ثابت  $\pi$

**Math.SQRT1\_2 •**

۱ تقسیم بر ریشه‌ی دوم ۲.

**Math.SQRT2 •**

ریشه‌ی دوم ۲.

**تابع‌ها ۳/۲۵/۴****Math.abs(x) •**

قدر مطلق  $x$  را برابر می‌گرداند.

**Math.acos(x) •**

آرک‌کسینوس  $x$  را برابر می‌گرداند؛ مقدار برگشتی بین  $0^\circ$  و  $\pi$  رادیان است.

**Math.asin(x) •**

آرک‌سینوس  $x$  را برابر می‌گرداند؛ مقدار برگشتی بین  $-\frac{\pi}{2}$  و  $\frac{\pi}{2}$  رادیان است.

**Math.atan(x) •**

آرک‌تاتزانت  $x$  را برابر می‌گرداند؛ مقدار برگشتی بین  $-\frac{\pi}{2}$  و  $\frac{\pi}{2}$  رادیان است.

**Math.atan2(y, x) •**

مقداری بین  $-\pi$  و  $\pi$  رادیان بر می‌گرداند که معرف زاویه‌ی بین قسمت مثبت محور X و نقطه‌ی  $(y, x)$  است. به ترتیب آوندهای تابع توجه کنید.

**Math.ceil(x) •**

نزدیک‌ترین عدد صحیحی را که بزرگ‌تر یا برابر با  $x$  است، بر می‌گرداند.

**Math.cos(x) •**

کسینوس مقدار داده شده‌ی  $x$  را برابر می‌گرداند.

**Math.exp(x)** •

مقدار ثابت  $e$  به توان  $x$  را برابر می‌گرداند.

**Math.floor(x)** •

نزدیک‌ترین عدد صحیحی را که کوچک‌تر یا برابر با  $x$  است، برابر می‌گرداند.

**Math.log(x)** •

لگاریتم طبیعی  $x$  را برابر می‌گرداند.

**Math.max(args...)** •

بزرگ‌ترین آوند را از میان آوندهای داده شده برابر می‌گرداند. اگر آوندی داده نشده باشد، `Infinity` برابر می‌گرداند. اگر یکی از آوندها `Nan` باشد، و یا مقداری غیر عددی باشد که نتوان آن را به عدد تبدیل کرد، `Nan` برابر می‌گرداند. تا قبل از ECMA ۳، این تابع دقیقاً به ۲ آوند نیاز داشت.

**Math.min(args...)** •

کوچک‌ترین آوند را از میان آوندهای داده شده برابر می‌گرداند. اگر آوندی داده نشده باشد، `Infinity` برابر می‌گرداند. اگر یکی از آوندها `Nan` باشد، و یا مقداری غیر عددی باشد که نتوان آن را به عدد تبدیل کرد، `Nan` برابر می‌گرداند. تا قبل از ECMA ۳، این تابع دقیقاً به ۲ آوند نیاز داشت.

**Math.pow(x, y)** •

مقدار  $x$  به توان  $y$  را برابر می‌گرداند.

**Math.random()** •

یک عدد شبه‌تصادی را بین  $0^{\circ}$  و  $1^{\circ}$  برابر می‌گرداند. جاوا اسکریپت  $1/1$ ؛ جی اسکریپت  $1/1$ ؛ ECMA  $.1$ .

**Math.round(x)** •

نزدیک‌ترین عدد صحیح به  $x$  را برابر می‌گرداند.

**Math.sin(x)** •

سینوس  $x$  را برابر می‌گرداند.

**Math.sqrt(x) •**

جذر  $x$  را برمی‌گرداند. اگر  $x$  کمتر از صفر باشد، NaN برمی‌گرداند.

**Math.tan(x) •**

تانژانت  $x$  را برمی‌گرداند.

**ارجاع ۳/۲۵/۵**

Number

**Navigator ۳/۲۶**

(جاوا اسکریپت سمت مشتری ۱/۰)

اطلاعات درباره‌ی مرورگر.

**۳/۲۶/۱ دستور**

navigator

**۳/۲۶/۲ خصلت‌ها****appCodeName •**

رشته‌ای فقط-خواندنی که لقب مرورگر را بیان می‌کند. این لقب در تمام مرورگرهای نتسکیپ "Mozilla" است. برای سازگاری، در مرورگرهای اینترنت اکسلورر هم این لقب "Mozilla" است.

**appName •**

رشته‌ای فقط-خواندنی که نام مرورگر را بیان می‌کند. برای نتسکیپ، مقدار این خصلت "Netscape" است. برای اینترنت اکسلورر، مقدار این خصلت "Microsoft Internet Explorer" است.

**appVersion •**

رشته‌ای فقط-خواندنی که اطلاعات مربوط به روایت و سکوی مرورگر را بیان می‌کند. قسمت اول این رشته، شماره‌ی روایت است. اگر این رشته را به تابع () parseInt() بدهید، فقط مقدار روایت اصلی را برمی‌گرداند، و اگر آن را به تابع () parseFloat() بدهید، مقدار روایت اصلی و فرعی را به صورت یک عدد میز شناور برمی‌گرداند. بقیه‌ی مقدار

رشته‌ای این خصلت، جزئیات دیگری را در باره‌ی روایت مرورگر، از جمله سیستم عاملی که روی آن اجرا می‌شود، را نشان می‌دهد. اما متأسفانه قالب این اطلاعات بین مرورگرهای مختلف بسیار متغیر است.

**cookieEnabled •**

یک مقدار بولی فقط-خواندنی که نشان می‌دهد کوکی‌ها در مرورگر فعال هستند (true) یا نه (false). اینترنت اکسپلورر ۴، نتاسکیپ ۶.

**language •**

رشته‌ای فقط-خواندنی که زبان پیشفرض روایت مرورگر را نشان می‌دهد. مقدار این خصلت، رمز استاندارد دوحرفی زبان مورد نظر است، مثلاً "en" برای انگلیسی و "fr" برای فرانسوی [و یا "fa" برای فارسی]. در ضمن، ممکن است رشته‌ای پنج حرفی باشد که علاوه بر زبان، ناحیه‌ی جغرافیایی را هم نشان می‌دهد، مثلاً "fr\_CA" که معرف فرانسوی کانادا است [و یا "fa\_IR" برای فارسی ایران]. نتاسکیپ ۴؛ دقت کنید که اینترنت اکسپلورر ۴ دو خصلت متفاوت در رابطه با زبان ارائه می‌کند.

**platform •**

رشته‌ای فقط-خواندنی که سیستم عامل و/یا سکوی سخت‌افزاری مورد نیاز مرورگر را نشان می‌دهد. گرچه مجموعه‌ی مقادیر استانداردی برای این خصلت وجود ندارد، ولی برخی از مقادیر معمول آن "Win32"، "MacPPC"، و "Linux i586" هستند. جی‌اسکریپت ۱/۲

**systemLanguage •**

رشته‌ای فقط-خواندنی که زبان پیشفرض سیستم عامل را با همان رمزهایی که در مورد خصلت ویژه‌ی نتاسکیپ language گفته شد، بیان می‌کند. اینترنت اکسپلورر ۴.

**userAgent •**

رشته‌ای فقط-خواندنی که مقدار مورد استفاده برای سرفصل عامل کاربر در تقاضاهای HTTP توسط مرورگر را تعیین می‌کند. عموماً مقدار آن متشکل از مقدار navigator.appCodeName به علاوه‌ی یک کج خط و بعد مقدار navigator.appVersion است.

**userLanguage •**

رشته‌ای فقط-خواندنی که زبان مرجح کاربر را با همان رمزهایی که در مورد خصلت ویژه‌ی نت اسکیپ language گفته شد، بیان می‌کند. اینترنت اکسپلورر ۴.

**روش‌ها ۳/۲۶/۳****javaEnabled() •**

اگر جاوا در مرورگر فعلی پشتیبانی شده و فعال باشد، مقدار true، و در غیر این صورت، مقدار false بر می‌گرداند. جاوا اسکریپت ۱/۱.

**ارجاع ۳/۲۶/۴**

.Screen

**Node ۳/۲۷**

(۱) سطح DOM  
گرهی در درخت سند.

**۳/۲۷/۱ زیرکلاس‌ها**

DocumentFragment, Document, Comment, Attr  
.Text, Element

**۳/۲۷/۲ ثابت‌ها**

تمام گره‌ها در یک سند HTML نمونه‌هایی از زیرکلاس‌های گره هستند که در بالا بر شمرده شده‌اند. هر شیء گره یک خصلت nodeType دارد که مشخص می‌کند نمونه‌ای از کدامیک از زیرکلاس‌های فوق است. ثابت‌های زیر، مقادیر قانونی nodeType هستند. دقت کنید که اینها خصلت‌های ایستای Node هستند، نه خصلت‌های اشیایی که از نوع گره ساخته می‌شوند. اینها در اینترنت اکسپلورر ۴، ۵، و ۶ تعریف نشده‌اند؛ در مرورگرهای مذکور باید از مقادیر عددی مربوطه استفاده کنید:

Node.ELEMENT_NODE	= 1; // Element
Node.ATTRIBUTE_NODE	= 2; // Attr
Node.TEXT_NODE	= 3; // Text

```

Node.COMMENT_NODE      = 8; // Comment
Node.DOCUMENT_NODE     = 9; // Document
Node.DOCUMENT_FRAGMENT_NODE = 11; // DocumentFragment

```

## ۳/۲۷/۳ خصلت‌ها

### **attributes[]** •

اگر گرهی یک عنصر باشد، خصلت attributes آرایه‌ای فقط-خواندنی از اشیای Attr است که معرف صفت‌های عنصر هستند. این آرایه را می‌توان به شماره یا نام صفت نمایه‌دهی کرد. چون برای هر صفت HTML یک خصلت متاظر در عنصر وجود دارد، لذا معمولاً از آرایه‌ی attributes[] استفاده نمی‌شود.

### **childNodes[]** •

آرایه‌ای فقط-خواندنی از گره‌ها است که حاوی گره‌های فرزند این گره است. اگر گره فرزندی نداشته باشد، این خصلت آرایه‌ای با طول صفر است.

### **firstChild** •

این خصلت فقط-خواندنی به اولین گره فرزند این گره اشاره می‌کند، و اگر گره فرزندی نداشته باشد، مقدار آن null است.

### **lastChild** •

این خصلت فقط-خواندنی به آخرین گره فرزند این گره اشاره می‌کند، و اگر گره فرزندی نداشته باشد، مقدار آن null است.

### **nextSibling** •

گره خواهری که در آرایه‌ی childNodes[] مربوط به parentNode بلافاصله بعد از این گره قرار می‌گیرد، و یا null در زمانی که چنین گرهی وجود نداشته باشد. فقط-خواندنی.

### **nodeName** •

نام گره. برای گره‌های عنصر، این خصلت نام عنصر را مشخص می‌کند، که می‌توان با خصلت tagName نیز آن را به دست آورد. برای گره‌های Attr، این خصلت نام صفت را مشخص می‌کند. برای انواع دیگر گره، این مقدار یک رشته‌ی ثابت است که نوع گره را مشخص می‌کند. فقط-خواندنی.

**nodeType •**

نوع گره. مقادیر قانونی این خصلت ثابت‌هایی هستند که در بالا گفته شد.

**nodeValue •**

مقدار رشته‌ای یک گره. برای گره‌های متن و توضیح، این خصلت حاوی محتوای متنی گره است. برای گره‌های Attr، حاوی مقدار صفت است. این خصلت خواندنی‌نوشتی است.

**ownerDocument •**

شیء سندی که این گره بخشی از آن است. برای گره‌های سند، این خصلت null است. فقط-خواندنی.

**parentNode •**

گره والد یا گنجایه‌ی این گره، و یا null در صورتی که والدی نداشته باشد. وقت کنید که گره‌های سند و Attr هرگز گره والد ندارند. در گره‌هایی که از سند بر داشته شده و یا جدیداً ایجاد شده‌اند و هنوز به درخت سند افزوده نشده‌اند، مقدار این خصلت null است. فقط-خواندنی.

**previousSibling •**

گره خواهری که در آرایه childNodes [] مربوط به parentNode بلا فاصله قبل از این گره قرار می‌گیرد، و یا null در زمانی که چنین گرهی وجود نداشته باشد.

**روش‌ها ۳/۲۷/۴****addEventListener(type, listener, useCapture)**

یک رویدادشون را برای این گره به ثبت می‌رساند. type رشته‌ای است که نوع رویداد را با برداشتن حروف "on" از آغاز آن بیان می‌کند (مثلاً click یا submit). listener تابع رویدادپرداز است. وقتی این تابع فراخوانده می‌شود، شیئی از نوع Event به عنوان آوند به آن داده می‌شود. اگر مقدار useCapture صحیح (true) باشد، این رویدادپرداز از نوع گیرنده است. اگر false باشد و یا وجود نداشته باشد، یک رویدادپرداز عادی است. چیزی بر نمی‌گرداند. DOM سطح ۲؛

در اینترنت اکسپلورر ۴، ۵، و یا ۶ پشتیبانی نمی‌شود.

#### **appendChild(newChild)** •

گره newChild را با افزودن به انتهای آرایه‌یchildNodes[ ] این گره، به درخت سند اضافه می‌کند. اگر گره از قبل در درخت سند باشد، اول بر داشته می‌شود، و بعد در موقعیت جدید اضافه می‌شود. آوند newChild را بر می‌گرداند.

#### **cloneNode(deep)** •

نسخه‌ی دیگری از این گره بر می‌گرداند. اگر مقدار deep صحیح باشد، فرزندان گره نیز به صورت تراجی نسخه‌برداری می‌شوند.

#### **hasAttributes()** •

اگر این گره یک عنصر دارای برخی صفات باشد، صحیح بر می‌گرداند. ۲ سطح DOM

#### **hasChildNodes()** •

اگر گره فرزندی داشته باشد، صحیح بر می‌گرداند.

#### **insertBefore(newChild, refChild)** •

گره newChild را درست قبل از گره refChild به درخت سند اضافه می‌کند. refChild باید فرزندی از این گره باشد. اگر گره newChild از قبل در درخت سند باشد، اول بر داشته می‌شود. newChild را بر می‌گرداند.

#### **isSupported(feature, version)** •

اگر ویژگی مورد نظر با روایت مشخص شده پشتیبانی شده باشد، مقدار صحیح بر می‌گرداند. همچنان‌ین، رک. ۲ سطح DOM DOMImplementation.hasFeature()

#### **normalize()** •

تمام فرزندان این گره را که از نوع گره متی هستند، بهنجار می‌کند، یعنی گره‌های متی خالی را حذف می‌کند، و گره‌های متی مجاور را با هم ادغام می‌کند. چیزی بر نمی‌گرداند.

**removeChild(oldChild)** •

گره oldChild را از درخت سند بر می‌دارد. باید oldChild فرزندی از این گره باشد. oldChild را بر می‌گرداند.

**removeEventListener(type, listener, useCapture)**

رویدادشونی مشخص شده را بر می‌گرداند. چیزی بر نمی‌گرداند. DOM سطح ۲ در اینترنت اکسپلورر ۴، ۵ و یا ۶ پشتیبانی نمی‌شود.

**replaceChild(newChild, oldChild)** •

به جای گره oldChild (که باید از فرزندان این گره باشد)، گره newChild را می‌گذارد. اگر newChild از قبل در درخت سند باشد، اول از جای قبلی بر داشته می‌شود. oldChild را بر می‌گرداند.

**۳/۲۷/۵ ارجاع**

DocumentFragment, Document, Comment, Attr  
.Text, Element

**Number ۳/۲۸**

(۱) ECMA ۲/۰؛ جی اسکریپت ۱؛ جاوا اسکریپت ۲/۰؛ پشتیبانی برای اعداد.

**۳/۲۸/۱ سازنده**

```
new Number(value)
Number(value)
```

سازنده‌ی () به همراه عملگر new آوند خود را به مقداری عددی تبدیل می‌کند، و شیء عدد جدیدی را که به دور آن مقدار پیچیده شده است، بر می‌گرداند. بدون new () یک تابع تبدیلی است که آوند خود را به عدد تبدیل می‌کند، و آن مقدار را بر می‌گرداند.

**۳/۲۸/۲ ثابت‌ها**

این ثابت‌ها خصلت‌های خود Number هستند، نه شیء‌های عدد ساخته شده از آن.

**Number.MAX\_VALUE •**

بزرگ‌ترین عدد قابل نمایش. تقریباً  $1.79E+308$

**Number.MIN\_VALUE •**

کوچک‌ترین عدد قابل نمایش. تقریباً  $.5E-324$

**Number.NaN •**

مقدار غیر عددی. همان `NaN` سراسری.

**Number.NEGATIVE\_INFINITY •**

مقدار منهای بی‌نهایت.

**Number.POSITIVE\_INFINITY •**

مقدار بی‌نهایت. همان `Infinity` سراسری.

## ۳/۲۸/۳ روش‌ها

**toExponential(digits) •**

نمایش رشته‌ای عدد را به صورت نمایی با یک رقم صحیح و `digits` رقم اعشاری بر می‌گرداند. برای رسیدن به طول مورد نظر، قسمت کسری گرد شده و یا به آن صفر اضافه می‌شود. مقدار `digits` باید بین  $0$  و  $۲۰$  باشد، و اگر حذف شود، هر تعداد رقم لازم باشد، استفاده می‌شود. جاوا اسکریپت  $1/5$ ; جی‌اسکریپت  $5/5$ ; ECMA  $.3$ .

**toFixed(digits) •**

نمایش رشته‌ای عدد را به صورت غیر نمایی دقیقاً با تعداد `digits` رقم اعشاری بر می‌گرداند. مقدار `digits` باید بین  $0$  و  $۲۰$  باشد. برای رسیدن به طول مورد نظر، قسمت کسری گرد شده و یا به آن صفر اضافه می‌شود. جاوا اسکریپت  $1/5$ ; جی‌اسکریپت  $5/5$ ; ECMA  $.3$ .

**toLocaleString() •**

نمایش رشته‌ای خاصی از عدد را که بر اساس قراردادهای محلی آراسته شده است، بر می‌گرداند. مثلاً این آرایش شامل نوع نویسه‌ی مورد استفاده برای میز اعشاری و جدا کننده‌ی هزارگان است. جاوا اسکریپت  $1/5$ ; جی‌اسکریپت  $5/5$ ; ECMA  $.3$ .

**toPrecision(precision)** •

نمایش رشته‌ای عدد را با تعداد رقم معنی‌دار معادل precision برمی‌گرداند. precision باید بین ۱ و ۲۱ باشد. رشته‌ی برمی‌گردانده شده در صورت امکان از نماد ممیز ثابت و یا از نماد نمایی استفاده می‌کند. برای رسیدن به طول مورد نظر، قسمت کسری گرد شده و یا به آن صفر اضافه می‌شود. جاوا اسکریپت<sup>۱</sup>؛ جی اسکریپت<sup>۲</sup>؛ ECMA<sup>۳</sup>.

**toString(radix)** •

عدد را در مبنای داده شده به رشته تبدیل کرده و این رشته را برمی‌گرداند. radix باید بین ۲ و ۳۶ باشد. اگر حذف شود، از مبنای ۱۰ استفاده می‌شود.

**ارجاع ۳/۲۸/۴**

Math

**Object ۳/۲۹**

(جاوا اسکریپت هسته<sup>۱</sup>؛ جی اسکریپت<sup>۲</sup>؛ ECMA<sup>۳</sup>)  
کلاس والد همه‌ی اشیای جاوا اسکریپت.

**۳/۲۹/۱ سازنده**`new Object();`

این سازنده شیئی خالی ایجاد می‌کند که می‌توانید خصلت‌های دلخواه به آن اضافه کنید.

**۳/۲۹/۲ خصلت‌ها**

تمام اشیای جاوا اسکریپت، صرف نظر از اینکه چگونه ایجاد شده باشند، دارای خصلت‌های زیر هستند.

**constructor •**

شارهای به تابع جاوا اسکریپتی که سازنده‌ی شیء بوده است.  
جاوا اسکریپت<sup>۱</sup>؛ جی اسکریپت<sup>۲</sup>؛ ECMA<sup>۳</sup>.

## ۳/۲۹/۳ روش‌ها

تمام اشیای جاوا اسکریپت، صرف نظر از اینکه چگونه ایجاد شده باشد، دارای روش‌های زیر هستند.

### **hasOwnProperty(propname)** •

اگر شیء خصلتی غیرموروژی با نام مشخص شده داشته باشد، true بر می‌گرداند. اما اگر شیء فاقد آن خصلت باشد، و یا آن را از شیء سرمشق خود به ارث برده باشد، false بر می‌گرداند.  
جاوا اسکریپت ۱/۵؛ جی اسکریپت ۵/۵؛<sup>۳</sup> ECMA

### **isPrototypeOf(o)** •

اگر این شیء سرمشق o باشد، true بر می‌گرداند. اما اگر o یک شیء نباشد، و یا این شیء سرمشق آن نباشد، false بر می‌گرداند.  
جاوا اسکریپت ۱/۵؛ جی اسکریپت ۵/۵؛<sup>۳</sup> ECMA

### **propertyIsEnumerable(propname)** •

اگر شیء یک خصلت برشمردنی با نام مشخص شده داشته باشد، درست بر می‌گرداند، والا غلط بر می‌گرداند. خصلت‌های برشمردنی خصلت‌هایی هستند که در حلقه‌های for/in بر شمرده می‌شوند.  
جاوا اسکریپت ۱/۵؛ جی اسکریپت ۵/۵؛<sup>۳</sup> ECMA

### **toLocaleString()** •

نمایش رشته‌ای محلی شده‌ی این شیء را بر می‌گرداند. در پیاده‌سازی پیش‌فرض، این رشته صرفاً toString() را فرا می‌خواند، اما زیرکلاس‌ها می‌توانند برای محلی‌سازی آن را تغییر دهند.  
جاوا اسکریپت ۱/۵؛ جی اسکریپت ۵/۵؛<sup>۳</sup> ECMA

### **toString()** •

نمایش رشته‌ای شیء را بر می‌گرداند. پیاده‌سازی پیش‌فرضی که در کلاس Object ارائه شده، خیلی عمومی است، و چندان اطلاعات مفیدی به ما نمی‌دهد. لیکن زیرکلاس‌ها معمولاً این روش را جایگزین می‌کنند، و آن را به گونه‌ای تعریف می‌کنند که اطلاعات سودمندتری ارائه دهد.  
جاوا اسکریپت ۱/۰؛ جی اسکریپت ۲/۰؛<sup>۱</sup> ECMA

**valueOf()** •

مقدار بدوی شیء را در صورت وجود بر می‌گرداند. در مورد اشیایی که از نوع Object هستند، این روش صرفاً خود شیء را بر می‌گرداند. اما زیرکلاس‌هایی همچون عدد و بولی، مقدار بدوی مربوط به خود را بر می‌گرداند. جاوا اسکریپت ۱/۱؛ جی‌اسکریپت ۲/۰، ECMA ۱.

**ارجاع ۳/۲۹/۴**

.String Number Function Boolean Array

**Option ۳/۳۰**

(جاوا اسکریپت سمت مشتری ۱/۰)

یک گزینه‌ی قابل انتخاب.

Element مشتق شده از:

**دستور ۳/۳۰/۱**

`select.options[i]`

**سازنده ۳/۳۰/۲**

در جاوا اسکریپت ۱/۱ و بعد از آن، می‌توان اشیای گزینه را به صورت پویا با

استفاده از سازنده‌ی Option() ایجاد کرد:

```
new Option(text, value, defaultSelected, selected)
```

**خصلت‌ها ۳/۳۰/۳****defaultSelected** •

یک مقدار بولی خواندنی/نوشتگی که مشخص می‌کند در آغاز این گزینه از Select انتخاب شده است یا نه.

**index** •

عدد صحیح فقط-خواندنی که شماره‌ی این گزینه را در آرایه‌ی options[] شیء انتخاب حاوی آن نشان می‌دهد.

**selected** •

یک مقدار بولی خواندنی/نوشتگی که مشخص می‌کند در حال حاضر این

گزینه انتخاب شده است یا نه. برای بررسی انتخاب شدن یک گزینه می‌توانید از این خصلت استفاده کنید. به علاوه، با دادن مقدار جدید به آن می‌توانید گزینه را انتخاب کنید یا بر عکس. دقت کنید که وقتی به این صورت انتخاب گزینه را تغییر می‌دهید، رویداد پرداز `Select.onchange()` فراخوانده نمی‌شود.

#### **text •**

رشته‌ای خواندنی/نوشتني که به عنوان گزینه برای کاربر نشان داده می‌شود.

#### **value •**

رشته‌ای خواندنی/نوشتني که در صورت انتخاب شدن این گزینه در هنگام تحويل داده شدن فرم به سرور شبکه داده می‌شود.

### **۳/۳۰/۴ ارجاع**

`.Select`

### **RegExp ۳/۳۱**

(۳) ECMA ۳/۰؛ جی اسکریپت ۱/۲؛ جاوا اسکریپت هسته؛ عبارت‌های مرتب برای انطباق الگو.

### **۳/۳۱/۱ دستور مستقیم**

`/pattern/attributes`

### **۳/۳۱/۲ سازنده**

```
new RegExp(pattern, attributes)
```

الگوهای عبارت‌های مرتب با استفاده از دستور پیچیده‌ای بیان می‌شوند که قبلاً در این کتاب در باره‌ی آنها صحبت کردیم.

### **۳/۳۱/۳ خصلت‌های اشیای نمونه**

#### **global •**

یک مقدار بولی فقط-خواندنی که مشخص می‌کند که این شیء دارای صفت `o` است، و لذا انطباق را به صورت سراسری انجام می‌دهد.

**ignoreCase •**

یک مقدار بولی فقط-خواندنی که مشخص می‌کند که این شیء دارای صفت ن است، و لذا انطباق را بدون توجه به حروف کوچک و بزرگ انجام می‌دهد.

**lastIndex •**

برای اشیای RegExp سراسری، این خصلت خواندنی/نوشتني موقعیت نویسه‌ی بلافصله بعد از آخرین انطباق را نشان می‌دهد؛ این اولین نویسه‌ای است که برای انطباق بعدی مورد بررسی قرار می‌گیرد.

**multiline •**

یک مقدار بولی فقط-خواندنی که مشخص می‌کند که این شیء دارای صفت m است، و لذا انطباق را به صورت چندسطری انجام می‌دهد.

**source •**

رشته‌ای فقط-خواندنی که متن مبدأ عبارت مرتب pattern را به استثنای کچ خطها و صفت‌ها در خود نگه می‌دارد.

**روش‌ها ۳/۳۱/۴****exec(string) •**

رشته را با این عبارت مرتب انطباق می‌دهد، و آرایه‌ای از نتایج انطباق را بر می‌گرداند، و یا اگر هیچ انطباقی یافت نشد، null بر می‌گرداند. عنصر ۰ آرایه عبارت انطباق یافته است. عناصر بعدی آرایه حاوی زیررشته‌هایی هستند که با زیرعبارت‌های موجود در عبارت مرتب منطبق شده‌اند. آرایه‌ی برگردانده شده یک خصلت index نیز دارد که موقعیت شروع انطباق را بر می‌گرداند.

**test(string) •**

اگر string حاوی متن قابل انطباق با این عبارت مرتب باشد، صحیح بر می‌گرداند، و گرنه غلط.

**ارجاع ۳/۳۱/۵**

`String.replace()`, `String.match()`

.String.search()

## Screen ۳/۳۲

(جاوا اسکریپت سمت مشتری ۱/۲)  
اطلاعات در باره‌ی صفحه‌ی نمایش.

۳/۳۲/۱ دستور

screen

۳/۳۲/۲ خصلت‌ها

**availHeight** •

بلندای موجود صفحه بر حسب پیکسل.

**availWidth** •

پهنانی موجود صفحه بر حسب پیکسل.

**colorDepth** •

ژرفای جعبه‌رنگ مرورگر، و یا تعداد بیت‌بر‌پیکسل صفحه‌ی نمایش.

**height** •

بلندای کلی صفحه بر حسب پیکسل.

**width** •

پهنانی کلی صفحه بر حسب پیکسل.

۳/۳۲/۳ ارجاع

Navigator

## Select ۳/۳۳

(جی اسکریپت سمت مشتری ۱/۰)  
یک لیست انتخاب گرافیکی  
مشتق شده از: Element

## ۳/۳۳/۱ دستور

```
form.elements[i]
form.elements[element_name]
form.element_name
```

### ۳/۳۳/۲ خصلت‌ها

شیء انتخاب یک خصلت برای هر کدام از صفت‌های برگه‌ی <select> در HTML از قبیل name، multiple، disabled و size، دارد. علاوه بر این، خصلت‌های زیر را هم تعریف کرده است:

- **form**

شیء فرمی که حاوی این شیء انتخاب است. فقط-خواندنی.

- **length**

یک عدد صحیح فقط-خواندنی که تعداد عنصرهای موجود آرایه‌ی options[] را مشخص می‌کند. مقدار این خصلت برابر با options.length است.

- **options[]**

آرایه‌ای از اشیای گزینه که هر کدام معرف یکی از گزینه‌های موجود در شیء انتخاب است. می‌توانید با دادن مقدار کمتری به options.length، این آرایه را کوتاه‌تر کنید (حتی با دادن مقدار صفر می‌توانید تمام گزینه‌های آن را بردارید). با دادن مقدار null به یکی از عناصر آرایه، می‌توانید گزینه‌ی مربوط به آن را بردارید — این کار بقیه گزینه‌های بعد از آن را جا به جا می‌کند، و آرایه را یک عنصر کوتاه‌تر می‌کند. می‌توانید با استفاده از سازنده‌ی Option() یک گزینه‌ی جدید ایجاد کنید و آن را به options.length اختصاص دهید، و به این ترتیب یک گزینه به انتهای گزینه‌ها اضافه کنید.

- **selectedIndex**

یک عدد صحیح خواندنی/نوشتی که شماره‌ی گزینه‌ی انتخاب شده را در شیء انتخاب نشان می‌دهد. اگر هیچ گزینه‌ای انتخاب نشده باشد، مقدار این خصلت 1- است. اگر بیش از یک گزینه انتخاب شده باشند، فقط شماره‌ی اولین گزینه‌ی انتخاب شده را نشان selectedIndex

می دهد. اگر به این خصلت مقداری داده شود، تمام گزینه های دیگر از حالت انتخاب خارج می شوند. اگر مقدار ۱- شود، تمام گزینه ها از حالت انتخاب خارج می شوند.

**type •**

یک خصلت رشته ای فقط - خواندنی که نوع عنصر را مشخص می کند. اگر شیء انتخاب فقط اجازه انتخاب یک گزینه را بدهد (یعنی در تعریف آن صفت `multiple` HTML ذکر نشده باشد)، مقدار این خصلت "select-one" است. در غیر این صورت، مقدار آن - `multiple`.  
`Input.type` اس - ت. همچنان - یعنی، رک.

جاوا اسکریپت ۱/۱

**۳/۳۳/۳ روش ها**

**add(new, old) •**

شیء گزینه `new` را درست قبل از گزینه `old` در آرایه `options` [] درج می کند. اگر `old` مقدار `null` داشته باشد، گزینه `new` به انتهای آرایه اضافه می شود. چیزی بر نمی گرداند. DOM سطح ۱.

**blur() •**

کانون صفحه کلید را تحویل می دهد و چیزی بر نمی گرداند.

**focus() •**

کانون صفحه کلید را در اختیار می گیرد و چیزی بر نمی گرداند.

**remove(n) •**

عنصر `n` آرایه `options` [] را برابر می دارد. چیزی بر نمی گرداند. DOM سطح ۱.

**۳/۳۳/۴ رویداد پردازها**

**onblur •**

وقتی کانون ورودی از دست می رود، فرا خوانده می شود.

**onchange •**

وقتی کاربر یکی از اقلام را انتخاب می کند یا از حالت انتخاب شده خارج

می کند، فرا خوانده می شود.

**onfocus •**

وقتی کانون ورودی به دست می آید، فرا خوانده می شود.

**۳/۳۳/۵ ارجاع**

.Option، Input، Form

**String ۳/۳۴**

(جاوا اسکریپت هسته ۱/۰؛ جی اسکریپت ۱/۰؛ ECMA ۱)

کار با رشته ها.

مشتق شده از Element

**۳/۳۴/۱ سازنده**

String(s)  
new String(s)

بدون عملگر new تابع String آوند خود را به یک رشته تبدیل می کند. با عملگر new String ایجاد سازنده ای است که مقدار داده شده را در یک شیء رشته قرار می دهد.

**۳/۳۴/۲ خصلت ها**

**length •**

تعداد نویسه ها در رشته. فقط خواندنی.

**۳/۳۴/۳ روش ها**

**charAt(n) •**

نویسه هی موجود در موقعیت n در رشته را برابر می گرداند.

**charCodeAt(n) •**

رمز یونیکد نویسه هی موجود در موقعیت n در رشته را برابر می گرداند.  
جاوا اسکریپت ۱/۲؛ جی اسکریپت ۵/۵؛ ECMA ۱

رشته‌ی جدیدی بر می‌گرداند که از تبدیل کردن هر یک از آوندها به رشته و ادغام رشته‌های حاصله به دست می‌آید. جاوا اسکریپت ۱/۲؛ جی اسکریپت ۳/۰؛ ECMA ۳/۳.

**concat(value, ...)** •

موقعیت اولین ظهور substring در این رشته را از نمایه‌ی start به بعد بر می‌گرداند، و اگر یافت نشد، مقدار ۱- بر می‌گرداند. اگر حذف شود، مقدار آن صفر منظور می‌شود.

**lastIndexOf(substring, start)** •

موقعیت آخرین ظهور substring در این رشته را تا نمایه‌ی start بر می‌گرداند، و اگر یافت نشد، مقدار ۱- بر می‌گرداند. اگر حذف شود، مقدار آن برابر طول رشته منظور می‌شود.

**match(regexp)** •

این رشته را با عبارت مرتب داده شده مقایسه می‌کند، و آرایه‌ای مشکل از نتایج انطباق بر می‌گرداند، و اگر هیچگونه انطباقی یافت نشده باشد، null بر می‌گرداند. اگر regexp یک عبارت مرتب سراسری نباشد، در آن صورت، آرایه‌ی بر گردانده شده همان حاصل روش "g" داشته باشد، عناصر آرایه‌ی بر گردانده شده حاوی متن انطباق‌های یافت شده است. جاوا اسکریپت ۱/۲؛ جی اسکریپت ۳/۰؛ ECMA ۳/۳.

**replace(regexp, replacement)** •

رشته‌ی جدیدی بر می‌گرداند که در آن متن منطبق شده با regexp با replacement جایگزین شده است. regexp ممکن است یک عبارت مرتب و یا یک رشته‌ی ساده باشد. replacement ممکن است یک رشته باشد، که حاوی سلسله‌های گزین اختیاری عبارت مرتب (از قبیل \$1) باشد که به وسیله‌ی قسمت‌هایی از متن منطبق شده جایگزین می‌شوند. در ضمن، می‌تواند تابعی باشد که رشته‌ی جایگزین را بر اساس جزئیات انطباق داده شده در آوندها محاسبه می‌کند. جاوا اسکریپت ۱/۲؛ جی اسکریپت ۳/۰؛ ECMA ۳/۳.

• **search(regexp)**

موقعیت شروع اولین زیرشته‌ی این رشته را که با `regexp` منطبق می‌شود، بر می‌گرداند، و یا اگر انطباقی یافت نشود، ۱- بر می‌گرداند.  
جاوا اسکریپت ۱/۲؛ جی اسکریپت ۳/۰؛ ECMA ۳.

• **slice(start, end)**

رشته‌ی جدیدی را بر می‌گرداند که حاوی تمام نویسه‌های این رشته از شماره‌ی `start` (شامل خود آن) تا شماره‌ی `end` (اما بدون آن) است. اگر `end` حذف شود، قطعه‌ی تا پایان رشته ادامه می‌یابد. آوندهای منفی به معنای شمارش از آخر رشته هستند. جاوا اسکریپت ۱؛ جی اسکریپت ۳/۰؛ ECMA ۳.

• **split(delimiter, limit)**

آرایه‌ای از رشته‌ها بر می‌گرداند که از در هم شکستن رشته تشکیل می‌شود؛ محل‌های قطع رشته به وسیله‌ی `delimiter` مشخص می‌شود. ممکن است یک رشته یا یک عبارت مرتب باشد. اگر `delimiter` یک عبارت مرتب با زیرعبارتی در درون پرانتز باشد، متن جدا کننده‌ی منطبق شده با زیرعبارت در آرایه‌ی بر گردانده شده گنجانده می‌شود. همچنین، رک. `Array.join()` جاوا اسکریپت ۱؛ جی اسکریپت ۳/۰؛ ECMA ۱.

• **substring(from, to)**

رشته‌ی جدیدی مشکل از نویسه‌های شماره‌ی `from` تا `to` - ۱ این رشته بر می‌گرداند. اگر `to` حذف شود، زیررشته تا آخر رشته ادامه می‌یابد. آوندهای منفی مجاز نیستند.

• **substr(start, length)**

رشته‌ی جدیدی مشکل از تعداد `length` نویسه از شماره‌ی `from` این رشته بر می‌گرداند؛ اگر `length` حذف شود، زیررشته تا آخر رشته ادامه می‌یابد. جاوا اسکریپت ۱/۲؛ جی اسکریپت ۳/۰؛ این تابع غیراستاندارد است، و بهتر است به جای آن از `slice()` یا `substring()` استفاده کنید.

• **toLowerCase()**

نسخه‌ی دیگری از این رشته را بر می‌گرداند، که در آن، تمام حروف

بزرگ در صورت موجود بودن، به حرف کوچک معادل خود تبدیل شده‌اند.

#### **toUpperCase() •**

نسخه‌ی دیگری از این رشته را بر می‌گرداند، که در آن، تمام حروف کوچک در صورت موجود بودن، به حرف بزرگ معادل خود تبدیل شده‌اند.

### **۳/۳۴/۴ توابع ایستا**

#### **String.fromCharCode(c1, c2, ... ) •**

رشته‌ی جدیدی بر می‌گرداند که از نویسه‌هایی با رمزهای داده شده تشکیل شده است. جاوا اسکریپت ۱/۲؛ جی اسکریپت ۳/۰؛ ECMA ۱.

### **Style ۳/۳۵**

سطح ۲؛ اینترنت اکسپلورر ۴)

خصلت‌های CSS مستقیم یک عنصر.

### **۳/۳۵/۱ دستور**

`element.style`

### **۳/۳۵/۲ خصلت‌ها**

شیء شیوه خصلت‌های زیادی دارد: یعنی به ازای هر صفت CSS تعریف شده در تعیین نامه‌ی CSS2، یک خصلت معادل دارد. نام خصلت معادل نام صفت است، با مختصر تغییراتی که برای اجتناب از خطاهای دستوری در جاوا اسکریپت لازم است. صفت‌های چند کلمه‌ای که حاوی خط فاصله هستند، مانند `font-family`، در جاوا اسکریپت بدون خط فاصله نوشته می‌شوند، و بعد از کلمه‌ی اول، هر کدام از کلمات بعدی با حروف بزرگ شروع می‌شوند: `fontFamily`. به علاوه، از آنجا که نام صفت `float` در جاوا اسکریپت یک کلمه‌ی ذخیره شده است، لذا به جاوا اسکریپت آن از `cssFloat` استفاده می‌شود.

خصلت‌های بصری CSS در جدول زیر نشان داده شده‌اند. از آنجا که خصلت‌ها مستقیماً متناظر با صفت‌های CSS هستند، لذا برای هر کدام در اینجا شرح جداگانه‌ای ارائه نمی‌شود. برای اطلاع از معنا و مقادیر قانونی هر صفت، به یک مرجع

CSS مراجعه کنید. دقت کنید که مرورگرهای فعلی همه‌ی این خصلت‌ها را پیاده‌سازی نمی‌کنند.

تمام خصلت‌ها رشته‌ای‌اند، و در کار کردن با خصلت‌هایی که مقادیر عددی دارند، باید دقت کرد. برای گرفتن مقدار این خصلت‌ها باید از `parseFloat()` برای تبدیل کردن رشته به عدد استفاده کنید. برای مقداردهی به اینگونه خصلت‌ها، باید عدد را به رشته تبدیل کنید، که معمولاً این کار را با افزودن واحد آن انجام می‌دهید، مانند "px".

background	counterIncrement	orphans
backgroundAttachment	counterReset	outline
backgroundColor	cssFloat	outlineColor
backgroundImage	cursor	outlineStyle
backgroundPosition	direction	outlineWidth
backgroundRepeat	display	overflow
border	emptyCells	padding
borderBottom	font	paddingBottom
borderBottomColor	fontFamily	paddingLeft
borderBottomStyle	fontSize	paddingRight
borderBottomWidth	fontSizeAdjust	paddingTop
borderCollapse	fontStretch	page
borderColor	fontStyle	pageBreakAfter
borderLeft	fontVariant	pageBreakBefore
borderLeftColor	fontWeight	pageBreakInside
borderLeftStyle	height	position
borderLeftWidth	left	quotes
borderRight	letterSpacing	right
borderRightColor	lineHeight	size
borderRightStyle	listStyle	tableLayout
borderRightWidth	listStyleImage	textAlign
borderSpacing	listStylePosition	textDecoration
borderStyle	listStyleType	textIndent
borderTop	margin	textShadow
borderTopColor	marginBottom	textTransform
borderTopStyle	marginLeft	top
borderTopWidth	marginRight	unicodeBidi
borderWidth	marginTop	verticalAlign
bottom	markerOffset	visibility
captionSide	marks	whiteSpace
clear	maxHeight	widows
clip	maxWidth	width
color	minHeight	wordSpacing
content	minWidth	zIndex

قطعه‌ای از متن در یک سند.

مشتق شده از: Node

## ۳/۳۶/۱ **شرح**

شیء متن معرف قطعه‌ای از متن ساده بدون نشانه‌گذاری در یک درخت سند است. البته، این را باید با عنصر ورودی متن یک‌سطری در HTML، که با شیء Input نشان داده می‌شود، اشتباہ کنید.

## ۳/۳۶/۲ **خصلت‌ها**

**data** •

رشته‌ی متنی موجود در این گره.

**length** •

تعداد نویسه‌های موجود در این گره. فقط خواندنی.

## ۳/۳۶/۳ **روش‌ها**

**appendData(text)** •

متن داده شده را به آخر متن این گره اضافه می‌کند، و چیزی بر نمی‌گرداند.

**deleteData(offset, count)** •

متن این گره را از نویسه‌ی شماره‌ی offset نویسه به تعداد count حذف می‌کند. چیزی بر نمی‌گرداند.

**insertData(offset, text)** •

متن داده شده را در موقعیت نویسه‌ی offset به متن این گره اضافه می‌کند. چیزی بر نمی‌گرداند.

**replaceData(offset, count, text)** •

متن این گره را از نویسه‌ی شماره‌ی offset نویسه به تعداد count نویسه با متن داده شده جایگزین می‌کند. چیزی بر نمی‌گرداند.

**splitText(offset)** •

این گره متنی را در نویسه‌ی موقعیت offset به دو گره تبدیل می‌کند،

و گرده جدید را بعد از گرده اصلی به سند اضافه می کند، و آن را بر می گرداند.

**substringData(offset, count)** •

رشته ای مشکل از count نویسه، از نویسه شماره offset به بعد، بر می گرداند.

**ارجاع ۳/۳۶/۴**

`Node.normalize()`

**Textarea ۳/۳۷**

(جاوا اسکریپت سمت مشتری ۱/۰)  
ورودی متن چندسطری.  
مشتق شده از Element

**دستور ۳/۳۷/۱**

```
form.elements[i]  
form.elements[name]  
form.name
```

**شرح ۳/۳۷/۲**

شیء Textarea (ناحیه متنی) خیلی شبیه شیء Input (ورودی) است.

**خصیلت‌ها ۳/۳۷/۳**

شیء Textarea خصیلت‌هایی را برای هر یک از صفت‌های برگه‌ی HTML، از قبیل `disabled`, `defaultValue`, `cols`, `name`, `rows` و `readOnly` تعريف می‌کند. علاوه بر این، خصیلت‌های زیر را هم تعريف می‌کند:

**form** •

شیء فرمی که حاوی این شیء Textarea است. فقط خواندنی.

**type** •

یک خصیلت فقط خواندنی که نوع عنصر را مشخص می‌کند؛ در مورد

شیء `Textarea`، مقدار آن همواره "textarea" است.

**value •**

رشته‌ای خواندنی/نوشتني که متن موجود داخل این ناحیه‌ی متنی را مشخص می‌کند. مقدار اولیه‌ی آن، همان مقدار خصلت است. `defaultValue`

**۳/۳۷/۴ روش‌ها**

**blur() •**

کانون صفحه کلید را تحويل داده و چیزی بر نمی‌گرداند.

**focus() •**

کانون صفحه کلید را تحويل گرفته و چیزی بر نمی‌گرداند.

**select() •**

کل محتوای ناحیه‌ی متنی را انتخاب می‌کند. چیزی بر نمی‌گرداند.

**۳/۳۷/۵ رویدادپردازها**

**onblur •**

وقتی کانون صفحه کلید از دست می‌رود، فرا خوانده می‌شود.

**onchange •**

وقتی کاربر محتوای ناحیه‌ی متنی را تغییر داده، و کانون صفحه کلید را به جای دیگری منتقل می‌کند، فرا خوانده می‌شود. این رویدادپرداز فقط وقتی فرا خوانده می‌شود که کاربر کار ویرایش متن داخل ناحیه‌ی متنی را به پایان ببرد.

**onfocus •**

وقتی کانون صفحه کلید حاصل می‌شود، فرا خوانده می‌شود.

**۳/۳۷/۶ ارجاع**

.Input, Form, Element

## **Window ۳/۳۸**

(جاوا اسکریپت سمت مشتری ۱/۰)  
پنجره یا کادر مرورگر.

### **۳/۳۸/۱ دستور**

```
self  
window  
window.frames[i]
```

### **۳/۳۸/۲ خصلت‌ها**

شیء پنجره خصلت‌های زیر را تعریف می‌کند. خصلت‌های انتقال ناپذیر مختص مرورگر بعد از این لیست به صورت جداگانه فهرست شده‌اند. دقت کنید که شیء پنجره، در جاوا اسکریپت سمت مشتری، شیء سراسری محسوب می‌شود؛ بنا بر این، شیء پنجره واجد خصلت‌هایی که برای شیء سراسری ذکر شده‌اند، نیز است.

#### **closed •**

یک مقدار بولی فقط-خواندنی که مشخص می‌کند پنجره بسته شده است یا نه.

#### **defaultStatus •**

رشته‌ای خواندنی/نوشتی که پیغامی را که در صورت نبودن پیغام دیگر، در سطر وضعیت مرورگر نمایش داده می‌شود، تعیین می‌کند.

#### **document •**

شاره‌ای فقط-خواندنی به شیء سند موجود در این پنجره یا کادر. رک.  
.Document

#### **frames[] •**

آرایه‌ای از اشیای پنجره، که معرف کادرهای موجود در داخل پنجره است. دقت کنید که این کادرها هر کدام ممکن است مشتمل بر کادرهایی در درون خود باشند، که از طریق آرایه‌ی frames[] هر کدام از آنها قابل دستیابی است.

#### **history •**

شاره‌ای فقط-خواندنی به شیء سابقه‌ی مربوط به این پنجره یا کادر. رک.

#### **.History**

##### **length •**

تعداد کادرهای موجود در داخل این پنجره یا کادر را نشان می‌دهد.  
معادل frames.length است.

##### **location •**

شیء مکان مربوط به این پنجره یا کادر. رک. Location. این خصلت رفتاری ویژه دارد؛ اگر رشته‌ی یک نشانی را به آن بدهید، مرورگر آن نشانی را می‌خواند و نمایش می‌دهد.

##### **name •**

رشته‌ای که حاوی نام پنجره یا کادر است. نام از طریق روش `name` در برگه‌ی `Window.open()` و یا با صفت `name` در `<frame>` مشخص می‌شود. در جاوا اسکریپت ۱/۰ فقط خواندنی؛ در جاوا اسکریپت ۱/۱ خواندنی/نوشتی.

##### **navigator •**

اشاره‌ای فقط-خواندنی به شیء مرورگر، که اطلاعات مربوط به روایت و پیکربندی مرورگر شبکه را نشان می‌دهد. رک. Navigator.

##### **opener •**

اشاره‌ای خواندنی/نوشتی به شیء پنجره‌ای که این پنجره را باز کرده است. جاوا اسکریپت ۱/۱.

##### **parent •**

اشاره‌ای فقط-خواندنی به شیء پنجره‌ای که حاوی این پنجره یا کادر است. اگر این پنجره یک پنجه‌ی سطح بالا باشد، parent به خود آن اشاره می‌کند.

##### **screen •**

اشاره‌ای فقط-خواندنی به شیء صفحه‌ی نمایش که اطلاعاتی را در باره‌ی صفحه‌ی نمایشی که مرورگر در آن نمایش داده می‌شود، نشان می‌دهد. رک. Screen. جاوا اسکریپت ۱/۲.

**self •**

اشاره‌ای فقط-خواندنی به خود این پنجره. این مترادف با خصلت window است.

**status •**

رشته‌ای خواندنی/نوشتنی که می‌توان برای نمایش پیغامی گذرا در سطر وضعیت مرورگر از آن استفاده کرد.

**top •**

اشاره‌ای فقط-خواندنی به پنجره‌ی سطح بالایی که حاوی این پنجره است. اگر این پنجره یک پنجره‌ی سطح بالا باشد، top به خود آن اشاره می‌کند.

**window •**

خصلت window مترادف با خصلت self است؛ حاوی اشاره‌ای به این پنجره است.

## ۳/۳۸/۳ خصلت‌های نت‌اسکیپ ۴

**innerHeight, innerWidth •**

خصلت‌هایی خواندنی/نوشتنی که بلندا و پنهانی ناحیه‌ی نمایش سند در این پنجره را بر حسب پیکسل مشخص می‌کنند. این بعداً، مشتمل بر بلندای نوار منو و ابزار و امثال آن نیست.

**outerHeight, outerWidth •**

خصلت‌هایی خواندنی/نوشتنی که کل بلندا و پنهانی این پنجره را مشخص می‌کنند. این بعداً، مشتمل بر بلندای نوار منو و ابزار و امثال آن است.

**pageXOffset, pageYOffset •**

اعداد صحیح فقط-خواندنی که تعداد پیکسل‌های نورده‌یده شده‌ی سند به راست (pageXOffset) و پایین (pageYOffset) را نشان می‌دهند.

**screenX, screenY •**

اعداد صحیح فقط-خواندنی که مختصات X و Y گوشی بالا و چپ پنجره را نسبت به صفحه‌ی نمایش نشان می‌دهند. اگر این پنجره یک کادر باشد، این خصلت‌ها مختصات پنجره‌ی سطح بالایی را که این کادر در آن

واقع شده است، نشان می دهدند.

## ۳/۳۸/۴ خصلت های اینترنت اکسپلورر ۴

### **clientInformation •**

این خصلت مختص اینترنت اکسپلورر، متراff دخالت navigator است، و به شیء مرورگر اشاره دارد.

### **event •**

این خصلت یک شیء رویداد است که جزئیات جدیدترین رویدادی را که در این پنجره رخ داده است، نشان می دهد. در مدل رویداد اینترنت اکسپلورر، شیء رویداد به عنوان آوند به رویدادپرداز داده نمی شود، بلکه به این خصلت اختصاص داده می شود.

## ۳/۳۸/۵ روش ها

شیء پنجره روش های انتقال پذیر زیر را دارا است. در ضمن، از آنجا که شیء پنجره در جاوا اسکریپت سمت مشتری، شیء سراسری به شمار می رود، لذا روش های ذکر شده برای شیء سراسری را هم دارا است.

### **alert(message) •**

رشته‌ی message را در یک پنجره‌ی گفتگو نشان می دهد. چیزی بر نمی گرداند. جاوا اسکریپت ۱/۰.

### **blur() •**

کانون صفحه کلید را تحویل می دهد. چیزی بر نمی گرداند. جاوا اسکریپت ۱/۱.

### **clearInterval(intervalId) •**

اجرای تکراری مشخص شده با شناسه‌ی intervalId را لغو می کند. رک. () setInterval. چیزی بر نمی گرداند. جاوا اسکریپت ۱/۲.

### **clearTimeout(timeoutId) •**

زمان انتظار مشخص شده با شناسه‌ی timeoutId را لغو می کند. رک. () setTimeout. چیزی بر نمی گرداند. جاوا اسکریپت ۱/۰.

**close()** •

پنجره را می‌بندد و چیزی بر نمی‌گرداند. جاوا اسکریپت ۱/۰.

**confirm(question)** •

متن question را در یک پنجره‌ی گفتگو نشان می‌دهد، و منتظر یک جواب آری یا نه می‌شود. اگر کاربر دکمه‌ی OK را بزند، true بر می‌گرداند، و اگر دکمه‌ی Cancel را بزند، false بر می‌گرداند. جاوا اسکریپت ۱/۰.

**focus()** •

کانون صفحه کلید را تقاضا می‌کند؛ در ضمن، در اکثر سکوها این روش باعث می‌شود که پنجره‌ی مورد نظر به جلوی پنجره‌های دیگر بیاید. چیزی بر نمی‌گرداند. جاوا اسکریپت ۱/۱.

**getComputedStyle(elt)** •

یک شیء شیوه‌ی فقط-خواندنی بر می‌گرداند که حاوی تمام شیوه‌های CSS (و نه فقط شیوه‌های مستقیم) مربوط به عنصر elt است. صفات‌های موقعیت، مانند top، left و width در اینجا همیشه بر حسب پیکسل بر گردانده می‌شود. DOM سطح ۲.

**moveBy(dx, dy)** •

پنجره را از موقعیت فعلی به میزان مشخص شده جا به جا می‌کند. چیزی بر نمی‌گرداند. جاوا اسکریپت ۱/۲.

**moveTo(x, y)** •

پنجره را به موقعیت مشخص شده منتقل می‌کند. چیزی بر نمی‌گرداند. جاوا اسکریپت ۱/۲.

**open(url, name, features)** •

نشانی url را در پنجره‌ی با نام مشخص شده نشان می‌دهد. اگر آوند نام حذف شود و یا پنجره‌ای با آن نام وجود نداشته باشد، پنجره‌ی جدیدی ایجاد می‌شود. آوند اختیاری features رشتہ‌ای است که اندازه و ترتیبات پنجره‌ی جدید را به صورت لیستی جدا شده با ویرگول معین می‌کند. نام ویژگی‌هایی که در عموماً همه‌ی سکوها پشتیبانی می‌شوند، عبارت‌اند از: location، height=pixels و width=pixels.

مراجع جاوا اسکریپت ۱۴۱

toolbar، status و menubar اکسپلورر، موقعیت پنجره به صورت top=y و left=x مشخص می شود. در نت اسکریپت، به جای آن از screenY=y و screenX=x استفاده کنید. شیء پنجره موجود یا جدید را برع می گرداند. جاوا اسکریپت ۱/۰.

• **print()**

همانند کلیک کردن دکمه‌ی «چاپ» پنجره‌ی مرورگر عمل می‌کند، و چیزی بر نمی‌گرداند. نت اسکریپت ۴؛ اینترنت اکسپلورر ۵.

• **prompt(message, default)**

پیغام message را در یک پنجره‌ی گفتگو نشان می‌دهد، و منتظر می‌شود که کاربر متن پاسخ را وارد کند. مقدار اختیاری default را به عنوان پاسخ پیش‌فرض نشان می‌دهد. رشته‌ی وارد شده توسط کاربر را بر می‌گرداند، و یا اگر کاربر رشته‌ای وارد نکند، یک رشته‌ی خالی بر می‌گرداند، و در صورتی که کاربر دکمه‌ی «انصراف» را بزند، null بر می‌گرداند. جاوا اسکریپت ۱/۰.

• **resizeBy(dw, dh)**

پنجره را به میزان مشخص شده تغییر اندازه می‌دهد، و چیزی بر نمی‌گرداند. جاوا اسکریپت ۱/۲.

• **resizeTo(width, height)**

پنجره را به اندازه‌ی مشخص شده می‌رساند، و چیزی بر نمی‌گرداند. جاوا اسکریپت ۱/۲.

• **scroll(x, y)**

پنجره را به مختصات مشخص شده اسکرول می‌کند، و چیزی بر نمی‌گرداند. جاوا اسکریپت ۱/۱. در جاوا اسکریپت ۱/۲ به نفع scrollTo() منسون شده است.

• **scrollBy(dx, dy)**

پنجره را به میزان مشخص شده اسکرول می‌کند، و چیزی بر نمی‌گرداند. جاوا اسکریپت ۱/۲.

**scrollTo(x, y) •**

پنجره را به موقعیت مشخص شده اسکرول می کند، و چیزی بر نمی گرداند. جاوا اسکریپت ۱/۲.

**setInterval(code, interval, args...) •**

رشته‌ی متن جاوا اسکریپت `interval` را هر `code` میلی ثانیه ارزیابی می کند. در نت اسکریپت ۴ و اینترنت اکسپلورر ۵ `code` می تواند به جای یک رشته، اشاره‌ای به یکتابع باشد. در آن حالت، تابع مشخص شده هر `interval` میلی ثانیه یک بار اجرا می شود. در نت اسکریپت، هر گونه آوندی بعد از `interval` وجود داشته باشد، در موقع فرآخوانی به عنوان آوند به تابع مذکور داده می شود، ولی این ویژگی به وسیله‌ی اینترنت اکسپلورر پشتیبانی نمی شود. یک شناسه بر می گرداند که می توان برای لغو اجرای تکراری، آن را به عنوان آوند به روش `clearInterval()` داد. جاوا اسکریپت ۱/۲.

**setTimeout(code, delay) •**

رشته‌ی متن جاوا اسکریپت `code` را پس از سپری شدن `delay` میلی ثانیه ارزیابی می کند. در نت اسکریپت ۴ و اینترنت اکسپلورر ۵ `code` می تواند به جای یک رشته، اشاره‌ای به یکتابع باشد؛ به بحث مربوط به `setInterval()` مراجعه کنید. یک شناسه بر می گرداند که می توان برای لغو اجرای تعلیقی، آن را به عنوان آوند به روش `clearTimeout()` داد. وقت کنید که این روش بلا فاصله بر می گردد، و برای برگشتن منتظر سپری شدن زمان `delay` نمی شود. جاوا اسکریپت ۱/۰.

## ۳۶۸/۶ رویدادپردازها

رویدادپردازهای شیء پنجره به وسیله‌ی صفت‌های برگه‌ی `<body>` سند تعریف می شوند.

**onblur •**

زمانی که پنجره کانون صفحه کلید را از دست می دهد، فرآخوانده می شود.

**onerror •**

وقتی یک خطای جاوا اسکریپت بروز می‌کند، فرا خوانده می‌شود. این رویدادپرداز ویژه‌ای است که با سه آوند فرا خوانده می‌شود: پیغام خطای نشانی سند حاوی خطای شماره‌ی سطر خطای در صورت موجود بودن.

**onfocus •**

زمانی که پنجره کانون صفحه کلید را به دست می‌آورد، فرا خوانده می‌شود.

**onload •**

وقتی سند (یا مجموعه‌ی کادر) به طور کامل خوانده شد، فرا خوانده می‌شود.

**onresize •**

وقتی اندازه‌ی پنجره تغییر می‌کند، فرا خوانده می‌شود.

**onunload •**

وقتی مرورگر سند فعلی را ترک می‌کند، فرا خوانده می‌شود.

ارجاع ۳/۳۸/۷

.Document

## ۴ و اژه‌نامه

### ۴/۱ فارسی به انگلیسی

applet	برنامک	hypertext	ابرمن
clause	بند	execute	اجرا
Boolean	بولی	assignment	اخصاص
infinity	بی‌نهایت	format	آرایش
response	پاسخ	array	آرایه
bold	پرنیک	associated array	آرایه‌ی ارتباطی
query	پرسجو	exception	استشا
file	پرونده	script	اسکریپت
extension	پسوند	scripting	اسکریپت‌نویسی
stack	پشه	reference	اشاره
hidden	پنهان	pointer	اشاره‌گر
dynamic	پویا	bug	اشکال
implementation	پیاده‌سازی	validation	اعتبارسنجی
preload	پیش‌خوانی	declaration	اعلام
default	پیش‌فرض	initialization	آغازش
configuration	پیکربندی	increment	افراش
pixel	پیکسل	security	امنیت
link	پیوند	select	انتخاب
function	تابع	propagation	انتشار
parse	تجزیه	portable	انتقال‌یدر
parser	تجزیه‌گر	non-portable	انتقال‌ناپذیر
submit	تحویل	match	انطباق
compile	تدوین	argument	آوند
compiler	تدوین‌گر	load	بارشدن
recursion	ترابع	body	بدنه
recursive	تراتجی	primitive	بدوی
z-order	ترتیب Z	sibling	برادر/خواهر
decorations	تریبیتات	label	برچسب
image	تصویر	enumeration	برشماری
suspended	تعليقی	enumerable	برشمردنی
specification	تعیین‌نامه	reset	برگردان
modifier	تغییر‌گر	tag	برگه

event	رویداد	immutable	تغییرناپذیر
event handler	رویدادپرداز	interpret	تفسیر
event listener	رویدادشنو	interpreter	تفسیرگر
runtime	زمان اجرا	request	تقاضا
timer	زمان منج	comment	توضیح
substring	زیررشته	constant	ثابت
underflow	زیربریز	register	ثبت کردن
subclass	زیرکلاس	plug-in	جازن
history	سابقه	client-side	جاوا اسکریپت سمت مشتری
constructor	سازنده	JavaScript	جاوا اسکریپت
global	سراسری	core JavaScript	هسته
overflow	سرریز	thousands separator	جدا کننده هزارگان
carriage return	سرسطر	stream	جریان
header	سرفصل	palette	جعبه نگ
prototype	سرمشق	tab	جهش
top-level	سطح بالا	popup	جهشی
line feed	سطر جدید	checkbox	چک باکس
status line	سطر وضعیت	cache	حافظه نهان
platform	سکو	loop	حلقه
escape sequence	سلسله گریز	property	خصلت
document	سند	private	خصوصی
operating system	سیستم عامل	error	خطأ
hexadecimal	شانزدهگانی	embed	خواباندن
ID	شناسه	sibling	خواهر برادر
identifier	شناسه	scrolling	در نور دیدن
object	شیء	entry	درایه
object-oriented	شیء گرای	insert	درج کردن
style	شیوه	tree	درخت
attribute	صفت	command	دستور
form feed	صفحه‌ی جدید	syntax	دستور
screen	صفحه‌ی نمایش	instruction	دستور العمل
visibility	ظهور	button	دکمه
user agent	عامل کاربر	radio button	دکمه‌ی رادیویی
expression	عبارت	double-click	دوکلیک
regular expression	عبارت مرتب	interface	رابط
integer	عدد صحیح	relational	رابطه‌ای
non-identity	عدم یکسانی	string	رشتہ
operator	عملگر	string	رشتہ
operand	عملوند	digit	رقم
public	عمومی	encoding	رمزگذاری
element	عنصر	rollover	روایت
caption	عنوان	version	روایت
hyphen	فاصله	method	روش

**Edited by Foxit PDF Editor  
Copyright (c) by Foxit Corporation, 2003 - 2010  
For Evaluation Only.**

visible	مرئی	calling	فراخوانی
browser	مروورگر	upload	فراگذاری
navigator	مروورگر	child	فرزند
path	مسیر	download	فروگذاری
client	مشتری	space	فضا
location	مکان	white space	فضای سفید
decimal point	میز اعشاری	backspace	فضای وارون
floating-point	میز شناور	read-only	فقط خواندنی
deprecated	منسوخ	context	قیرینه
time zone	منطقه‌ی زمانی	curly brace	قلاب
inherited	میراثی	domain	قلمرو
mouse	موسواره	frame	کادر
position	موقعیت	focus	کانون
shortcut	میانبر	decrement	کاهش
clipping region	ناحیه‌ی برش	slash	کچخط
textarea	ناحیه‌ی متنی	backslash	کچخط وارون
hidden	نامرئی	bound	کران
navigation	ناوبری	keyword	کلیدواژه
copy	نسخه	click	کلیک
copy	نسخه‌برداری	capture	گرفتن (رویداد)
cursor	نشانگر	node	گره
markup	نمایه‌گذاری	option	گزینه
URL	نشانی	container	گچایه
quote	نقل قول	pattern	الگو
exponent	نما	layer	لایه
symbol	نماد	wrapping	لغافه
index	نمایه	literal	لفظی
indexing	نمایه‌دهی	nickname	لقب
exponential	نمایی	anchor	لنگر
instance	نمونه	source	مبدأ
address bar	نووار نشانی	radix	مبنا
scrollbar	نووار پیما	variable	متغیر
character set	نویسگان	variable	متغیر
charset	نویسگان	source code	متن برنامه
character	نویسه	plain text	متن ساده
parent	والد	finite	متناهی
input	ورودی	frame set	مجموعه‌ی کادر
resolution	وضوح	local	محلي
version	ویراست	localized	محلي شده
semicolon	ویرگول‌نقطه	coordinate	مختصه
identity	یکسانی	document object	مدل شیء سند
		model	

## ۴/۲ انگلیسی به فارسی

container	گنجایه	address bar	نوار نشانی
context	قریب	anchor	لنگر
coordinate	محضه	applet	برنامک
copy	نسخه	argument	آوند
copy	نسخه برداری	array	آرایه
core JavaScript	جاوا اسکریپت هسته	assignment	اختصاص
curly brace	قلاب	associated array	آرایه ای ارتباطی
cursor	نشانگر	attribute	صفت
decimal point	میز اعشاری	backslash	کجخط وارون
declaration	اعلام	backspace	فضای وارون
decorations	تریینات	body	بدنه
decrement	کاهش	bold	پررنگ
default	پیش فرض	Boolean	بولی
deprecated	منسوخ	bound	کران
digit	رقم	browser	مرورگر
document	سنده	bug	اشکال
document object	مدل شیء سنده	button	دکمه
model		cache	حافظه نهان
domain	قلمرو	calling	فراخوانی
double-click	دو کلیک	caption	عنوان
download	فروگذاری	capture	گرفتن (رویداد)
dynamic	پویا	carriage return	سرسطر
element	عنصر	character	نویسه
embed	خواباندن	character set	نویسگان
encoding	رمزگذاری	charset	نویسگان
entry	درایه	checkbox	چک باکس
enumerable	برشمردنی	child	فرزنده
enumeration	برشماری	clause	بند
error	خطا	click	کلیک
escape sequence	سلسله‌ی گریز	client	مشتری
event	رویداد	client-side	جاوا اسکریپت سمت
event handler	رویداد پرداز	JavaScript	مشتری
event listener	رویدادشنو	clipping region	ناحیه برش
exception	استثنای	command	دستور
execute	اجرا	comment	توضیح
exponent	نما	compile	تدوین
exponential	نمایی	compiler	تدوینگر
expression	عبارت	configuration	پیکربندی
extension	پسوند	constant	ثابت
file	پرونده	constructor	سازنده

local	محلى	finite	متناهي
localized	محلى شده	floating-point	مميز شناور
location	مكان	focus	قانون
loop	حلقه	form feed	صفحه جديد
markup	نشانه گذاري	format	آرایش
match	انطباق	frame	کادر
method	روش	frame set	مجموعه کادر
modifier	تغییر گر	function	تابع
mouse	موشواره	global	سراسري
navigation	ناوبري	header	سرفصل
navigator	مرور گر	hexadecimal	شانزدهگانی
nickname	لقب	hidden	پنهان
node	گره	hidden	نامرئي
non-identity	عدم يكسانی	history	سابقه
non-portable	انتقال ناپذير	hypertext	ابرمن
object	شيء	hyphen	فاصله
object-oriented	شيء گرا	ID	شناسه
operand	عملوند	identifier	شناسه
operating system	سيستم عامل	identity	يکسانی
operator	عملگر	image	تصوير
option	گزينه	immutable	غيرقابل تغيير
overflow	سرريز	implementation	پيادسازی
palette	چعبه رنگ	increment	افرايش
parent	والد	index	نمایه
parse	تجزیه	indexing	نمایه دهنی
parser	تجزیه گر	infinity	بینهايت
path	مسير	inherited	موروثی
pattern	الگو	initialization	آغازش
pixel	ピクセル	input	ورودی
plain text	متن ساده	insert	درج كردن
platform	سکو	instance	نمونه
plug-in	جازن	instruction	دستور العمل
pointer	اشاره گر	integer	عدد صحيح
popup	جهشی	interface	رابط
portable	انتقال پذير	interpret	تفسير
position	موقعیت	interpreter	تفسير گر
preload	پیش خوانی	keyword	کلید واژه
primitive	بدوي	label	برچسب
private	خصوصي	layer	لایه
propagation	انتشار	line feed	سطر جديد
property	خصلت	link	پيوند
prototype	سرمشق	literal	لفظي
public	عمومي	load	بار شدن

**Edited by Foxit PDF Editor**  
**Copyright (c) by Foxit Corporation, 2003 - 2010**  
**For Evaluation Only.**

واژه‌نامه ۱۴۹ ○

status line	سطر وضعیت	query	پرسچو
stream	جریان	quote	نقل قول
string	رشته	radio button	دکمه‌ی رادیویی
string	رشته	radix	مینا
style	شیوه	read-only	فقط-خواندنی
subclass	زیر کلاس	recursion	تراجع
submit	تحویل	recursive	تراجعي
substring	زیر رشته	reference	اشاره
suspended	تبلیغی	register	ثبت کردن
symbol	نماد	regular expression	عبارت مرتب
syntax	دستور	relational	رابطه‌ای
tab	جهش	request	تقاضا
tag	برگه	reset	برگردان
textarea	ناحیه متنی	resolution	وضوح
thousands separator	جدا کنندۀ هزارگان	response	پاسخ
time zone	منطقه‌ی زمانی	rollover	رواندازی
timer	زمان سنج	runtime	زمان اجرا
top-level	سطح بالا	screen	صفحه‌ی نمایش
tree	درخت	script	اسکریپت
underflow	زیر ریز	scripting	اسکریپت نویسی
upload	فرآگذاری	scrollbar	نوار پیما
URL	نشانی	scrolling	در نور دیدن
user agent	عامل کاربر	security	امنیت
validation	اعتبار سنجی	select	انتخاب
variable	متغیر	semicolon	ویرگول نقطه
variable	متغیر	shortcut	میانبر
version	روایت	sibling	برادر/خواهر
version	ویراست	sibling	خواهر برادر
visibility	ظهور	slash	کچخط
visible	مرئی	source	مبدأ
white space	فضای سفید	source code	متن برنامه
wrapping	لغافه	space	فضا
z-order	ترتیب z	specification	تعیین نامه
		stack	پشتہ