فصل اول: آشنایی با جاوااسکریپت

مقایسه برنامه‌نویسی و اسکریپت‌نویسی

برنامه نویسی: استفاده از یک زبان برنامه نویسی کامل برای تولید نرم‌افزار است. مانند C#، Visual Basic، Delphi و ...

اسکریپت نویسی: قطعه کدی است که درون کد دیگری نوشته می‌شود. مانند javascript، vbscript، php و ...

ویژگیها و قابلیتهای جاوا اسکریپت:

۱. اولین بار توسط شرکت netscape عرضه شد.
۲. به بزرگی و کوچکی حروف حساس است.
۳. استفاده از ; در انتهای هر دستور اختیاری است.
۴. یک زبان شئ گرا است.
۵. برای انجام محاسبات می‌توان از آن استفاده کرد.
۶. برای اعتبارسنجی فرمها می‌توان از آن استفاده کرد.
۷. به رویدادها (کلیک، دابل کلیک و ...) پاسخ می دهد.
۸. می توان از برنامه های ایجاد شده توسط برنامه نویسان دیگر استفاده کرد. (نمایش ساعت و تاریخ در صفحه و ...)
۹. یک زبان اسکریپتی سمت سرویس گیرنده محسوب می‌شود.

زبانهای سمت سرویس گیرنده: زبانهایی هستند که برای اجرا نیاز به سرور ندارند مانند ... ,javascript, vbscript, jscript

زبانهای سمت سرور: زبانهایی هستند که برای اجرا نیاز به سرور دارند مانند ... ,php, asp, jsp, cold futionِ

ابزارهای کدنویسی جاوا اسکریپت:

- محیط‌های متنی مانند نرم افزار notepad
- محیط‌های طراحی مانند نرم افزار Dreamweaver

روش‌های افزودن کدهای جاوا اسکریپت به صفحه وب (html):

روش داخلی

درج مستقیم کدها درون صفحه وب با استفاده از تگ <script> در بخش head یا body

```
<script type=”text/javascript>
دستورات زبان جاوااسکریپت
</script>
```

نکته: اگر تگ <script> در بخش head قرار بگیرد کدهای javascript قبل از بارگذاری صفحه اجرا می شوند. اما اگر در بخش body تعریف شوند در حین بارگذاری صفحه اجرا می شوند.

روش خارجی

درج کدها درون یک فایل خارجی با پسوند js و پیوند (link) آن به صفحه وب با استفاده از تگ <script> بصورت زیر:

```
<script  type="text/javascript"  src="نام و مسیر فایل حاوی کدهای جاوااسکریپت"> </script>
```

روش خطی

درج کدها در هر خط از تگ‌های html مانند زیر:

```
<a href=”javascript:window.close()”>خروج</a>
```

نکته۱: در صورتیکه اجرای اسکریپت‌ها در مرورگر غیرفعال یا محدود شده باشد می‌توان با تگ <noscript> این موضوع را به کاربر اطلاع داد. بصورت زیر:

```
<noscript>اجرای کدهای جاوااسکریپت روی مرورگر شما غیرفعال است.</noscript>
```

نکته۲: در صورتی که مرورگرِ کاربر قدیمی باشد و کدهای جاوااسکریپت را پشتیبانی نکند برای اینکه مشکلی در نمایش صفحه بوجود نیاید کدهای جاوااسکریپت را می‌توان درون علامت‌های <!-- و //--> قرار داد:

```
<!--
کدهای جاوااسکریپت
//-->
```

توضیحات در جاوااسکریپت

۱. برای درج توضیحات یک خطی در ابتدای خط از عملگر // استفاده می‌شود.

۲. توضیحات چند خطی را بین علامت‌های /* */ قرار می‌هند.

فصل دوم: آشنایی با مدل اشیاء سند DOM(Document Object Model)

DOM مدلی است که دسترسی به عناصر موجود در صفحه را با استفاده از اشیاء فراهم می‌آورد.

شیء گرایی در جاوا اسکریپت

در زبان جاوااسکریپت هر مفهومی که دارای ویژگی‌های زیر باشد شئ نامیده می شود:

۱. خصوصیات (properties) : مانند نام یک عنصر
۲. رفتار یا متد (method): عملیاتی که یک عنصر قادر به انجام آنها است.
۳. رویداد (event) : اتفاقاتی که برای یک عنصر رخ می‌دهد.

آشنایی با اشیاء جاوااسکریپت

شئ window

برای دستکاری پنجره کاربرد دارد. عملیاتی مانند بازکردن پنجره، بستن پنجره، نمایش پیغام در یک پنجره و ... توسط این شیء انجام می‌شود. مهمترین خصوصیات و متدهای این شیء در جدول زیر آمده است:

موضوع	نام	کاربرد
خصوصیت	window.status	دسترسی به نوار وضعیت پنجره
متد	window.open()	باز کردن یک پنجره جدید
	window.close()	بستن پنجره
	window.print()	چاپ صفحه جاری
	window.alert()	نمایش پیغام‌های مناسب در یک پنجره جدا
	window.prompt()	دریافت ورودی از کاربر در یک پنجره جدا
	window.confirm()	نمایش پیغام تایید با دو دکمه Ok و cancel
	window.moveTo(x,y)	انتقال پنجره به نقطه مشخص شده
	window.moveBy(dx,dy)	جابجایی پنجره به اندازه مشخص شده

مثال۱: دستور زیر باعث نمایش پیغام «به دنیای جاوااسکریپت خوش آمدید» در یک پنجره جدا می‌شود:

```
window.alert("به دنیای جاوااسکریپت خوش آمدید");
```

مثال۲: دستور زیر باعث می‌شود با کلیک روی پیوند ایجاد شده پنجره مرورگر بسته شود. (استفاده از کدهای جاوااسکریپت به روش خطی)

```
<a href="javascript:window.close()">خروج </a>
```

مثال۳: دستور زیر باعث باز شدن سایت google در یک پنجره جدا با عنوان «گوگل» و ابعاد ۴۰۰*۳۰۰ می‌شود.

```
window.open("http:www.google.com","گوگل","width=400,height=300");
```

مثال۴: پایین آوردن پنجره از بالای نمایشگر به اندازه ۱۰۰ نقطه و نمایش ساعت و تاریخ جاری سیستم در نوار وضعیت

```
Window.moveTo(0,100);
Window.status=Date();
```

شئ document

با استفاده از متدها و خصوصیات این شیء می‌توان به تک تک عناصر صفحه دسترسی پیدا کرد.

موضوع	نام	کاربرد
خصوصیت	document.title	خواندن عنوان صفحه و تغییر عنوان صفحه
	document.url	خواندن آدرس صفحه جاری
	document.referer	خواندن آدرس صفحه ای که صفحه جاری را فراخوانی کرده است.
	Document.linkcolor	تغییر رنگ پیوندهای موجود در صفحه
متد	document.write()	چاپ عبارات دلخواه روی صفحه
	document.getElemetById()	دسترسی به عناصر صفحه بر اساس شناسه ی عنصر(id)
	document.getElemetByName()	دسترسی به عناصر صفحه بر اساس نام عنصر(name)
	document.getElemetByTagName()	دسترسی به عناصر صفحه بر اساس نام تگ

مثال۱: کد زیر باعث می‌شود پاراگراف دوم که شناسه آن p2 است به رنگ قرمز درآید. (دسترسی به پاراگراف بر اساس شناسه)

```
<p  id="p1">کامپیوتر</p>
<p  id="p2">حسابداری</p>
<p  id="p3">تاسیسات</p>
<script type="text/javascript>
   document.getElementById("p2").style.color="red";
</script>
```

مثال۲: دستور زیر عنوان صفحه را به «جاوااسکریپت» تغییر می‌دهد.

```
document.title="جاوااسکریپت";
```

مثال۳: دستور زیر ابتدا عبارت «عنوان صفحه عبارت است از:» را روی صفحه نمایش می‌دهد سپس عنوان صفحه را خوانده و آنرا روی صفحه مرورگر چاپ می‌کند.

```
document.write("عنوان صفحه عبارت است از:");
document.write(document.title);
```

شئ history

امکان دسترسی به تاریخچه مرورگر را فراهم می‌آورد.

موضوع	نام	کاربرد
خصوصیت	length	تعداد نشانی (آدرس) موجود در لیست تاریخچه را نمایش می‌دهد.
متد	back()	نشانی قبلی موجود در تاریخچه مرورگر را بارگذاری می‌کند (معادل دکمه back در مرورگر)
	forward()	نشانی بعدی موجود در تاریخچه مرورگر را بارگذاری می‌کند (معادل دکمه forward در مرورگر)
	go()	نشانی خاصی را از لیست تاریخچه مرورگر بارگذاری می‌کند.

مثال۱: نمایش تعداد نشانی در لیست تاریخچه

```
document.write(history.length);
```

مثال۲: کلیک روی پیوند موجود در صفحه و برگشت به صفحه قبلی

```
<a href="javascript:history.back()">برگشت</a>
```

شئ location

کار با نشانی‌ها (آدرس‌های اینترنتی) توسط این شیء امکان‌پذیر است.

موضوع	نام	کاربرد
خصوصیت	href	نشانی کامل صفحه را برمی‌گرداند.
	host	نام دامنه وب سایت جاری را برمی‌گرداند.
متد	assign("url")	فراخوانی صفحه‌ای که آدرس آن در داخل پرانتز قید شده است.
	reload()	صفحه جاری را مجدداً بارگذاری می‌کند. (معادل دکمه refresh در مرورگر)
	replace(("url")	جایگزین کردن صفحه‌ای که نشانی آن داخل پرانتز درج شده با صفحه جاری

مثال: یک پیوند با عبارت refresh ایجاد کنید که با کلیک روی آن صفحه نوسازی شود.

```
<a href="javascript:location.reload()">refresh</a>
```

مثال۲: نام دامنه وب سایت جاری را نمایش دهید.

```
document.write(location.host);
```

شیء navigator

اطلاعات مربوط به مرورگر از طریق این شیء قابل استخراج است.

	نام	کاربرد
خصوصیت	appName	دسترسی به نام مرورگر
	appVersion	دسترسی به نسخه مرورگر

مثال: دستوری بنویسید که نام مرورگری که کاربر از آن استفاده می‌کند را نمایش دهد.

```
document.write(navigator.appName)
```

فصل سوم: کار با داده‌ها و متغیرها

متغیر

مخزنی است که از آن برای نگه‌داری داده ها و اطلاعات استفاده می شود. هر متغیر دارای یک نام است که برای دسترسی به داده‌های موجود در متغیر از آن استفاده می‌شود.

قوانین نامگذاری متغیر

۱. حروف کوچک و بزرگ در نامگذاری متغیرها متفاوت هستند.
۲. نام متغیر با یک حرف الفبا یا خط زیر (_) شروع می شود.
۳. کلمات کلیدی جاوااسکریپت در نامگذاری مجاز نیست.
۴. کلمات رزرو شده جاوااسکریپت در نامگذاری مجاز نیست.

نکته۱: کلمات کلیدی همان دستورات زبان جاوااسکریپت هستند مانند if، for، while و ...

نکته۲: کلمات رزرو شده کلماتی هستند که در جاوااسکریپت معنی و مفهوم خاصی دارند مانند int، float، byte و ...

سوال: با توجه به قوانین نامگذاری متغیرها کدام یک از نام‌های زیر معتبر است؟

الف- _sum معتبر است.
ب- var معتبر نیست. از کلمات کلیدی است.
ج- x1 معتبر است.
د- 1a معتبر نیست. نام متغیر نباید با عدد شروع شود.
ه- return معتبر نیست. از کلمات کلیدی است.
و- Num معتبر است.

نحوه‌ی تعریف متغیر

```
var   نام متغیر ;
```

مثال: تعریف متغیر با حالتهای مختلف

```
var   x;                تعریف متغیر بدون مقدار اولیه
var y=5;                تعریف متغیر با مقدار اولیه‌ی ۵
var m,n=”ali”;          تعریف دو متغیر یکی بدون مقدار اولیه و دیگری با مقدار اولیه
```

نکته۱: تعریف متغیرها با استفاده از var اختیاری است.

نکته۲: در جاوااسکریپت نیاز نیست نوع متغیرها تعیین شود و نوع متغیرها هنگام انتساب یک مقدار به آن توسط خود برنامه تعیین می شود.

عملگر انتساب (=)

برای مقدار دادن به متغیرها از این عملگر استفاده می شود.

مثال: با توجه به نکته‌ی فوق در قطعه کد زیر، متغیر x از نوع رشته‌ای و متغیر y از نوع عددی در نظر گرفته می‌شود.

```
var  x,y;
x="ali";
y=20;
```

تابع typeof()

با این دستور می‌توان نوع متغیر را مشاهده کرد.

مثال: دستور زیر باعث نمایش عبارت string روی صفحه می‌شود.

```
var x=”ali”;
document.write(typeof(x));
```

داده

آنچه در متغیر ذخیره می‌شود داده نام دارد.

انواع داده در جاوااسکریپت

الف- انواع داده اولیه

این نوع داده‌ها که لیست آنها در زیر آمده است نیاز به تعریف ندارند و هنگام انتساب روی متغیرها اعمال می‌شوند.

۱. عددی (number)
۲. رشته ای (string)

۳. منطقی (boolean)

ب- انواع داده ارجاعی

اگر بخواهیم با متغیرها مانند یک شیء برخورد کنیم یعنی اینکه از ویژگی‌ها و متدهای اشیاء استفاده کنیم آنها را بصورت ارجاعی تعریف می کنیم.

۱. شئ عددی: Number()
۲. شئ رشته‌ای: String()
۳. شئ منطقی: Boolean()

نحوه‌ی تعریف متغیر از نوع داده ارجاعی

```
var   متغیر =new  نوع داده ارجاعی;
```

مثال: متغیر s از نوع شیء ارجاعی تعریف شده است.

```
var  s=new String();
```

به این ترتیب می‌توان از خصوصیات و متدهای مربوط به شیء رشته‌ای روی متغیر s استفاده کنیم.

مثال: خصوصیت length طول رشته را نمایش می‌دهد. خروجی دستورات زیر عدد ۱۰ است.

```
s="javascript"
document.write(s.length);   // 10
```

نکته: هنگام تعریف دادهای ارجاعی می‌توان به آنها مقدار اولیه داد.

مثال:

```
Var s=new String("javascript");
```

تبدیل انواع داده

انواع داده‌ی فوق را می‌توان به یکدیگر تبدیل کرد. با استفاده از دستورات زیر:

الف- متدهای تبدیل که عبارتند از:

۱. toString() : تبدیل به رشته
۲. parseInt() : تبدیل به عدد صحیح
۳. parseFloat() : تبدیل به عدد اعشاری

ب- توابع تبدیل که عبارتند از:

۱. String() : تبدیل به رشته
۲. Number() : تبدیل به عدد
۳. Boolean() : تبدیل به مقدار منطقی

با استفاده از متدها و توابع فوق می‌توان:

- اعدادی که بصورت رشته هستند مثلاً "15" را به عدد واقعی یعنی 15 تبدیل کرد و برعکس.
- اعداد 0 و 1 را می توان به ترتیب به مقادیر منطقی false و true تبدیل کرد.
- مقادیر منطقی true و false را می‌توان به رشته‌های true و false تبدیل کرد.

مثال۱: توسط قطعه کد زیر ابتدا عدد 100 به رشته "100" تبدیل شده سپس نوع آن (string) نمایش داده می‌شود.

```
var n=100,s;
s=n.toString()
document.write(typeof(s));
```

مثال۲: توسط قطعه کد زیر عدد 0 به نوع منطقیِ معادل آن یعنی flase تبدیل شده و نمایش داده می‌شود.

```
b=Boolean(0);
document.write(b);
```

مثال۳: خروجی دستورات زیر را مشخص کنید

```
parseInt("15")          // → 15          تبدیل رشته به عدد
String(15);             //→ "15"         تبدیل عدد به رشته
parseFloat("12.75")     // → 12.75        تبدیل رشته به عدد اعشاری
parseInt("12.75")       //→ 12           تبدیل رشته به عدد صحیح
parseInt("ali")         //→ NaN          پیغام خطا: تبدیل قابل انجام نیست
parseInt("20 book")     // → 20          تبدیل رشته به عدد
```

نکته۱: عبارت NaN از Not a Number مشتق شده است و به معنی این است که پارامتر داده شده عدد نیست و عمل تبدیل قابل انجام نیست.
نکته۲: در متد toString() چنانچه در داخل پرانتز اعداد ۲ یا ۱۶ ذکر شود عدد حاصل به ترتیب در مبنای ۲ یا مبنای ۱۶ نمایش داده می‌شود.
مثال: در قطعه کد زیر عدد 5 در مبنای ۲ بصورت 101 نمایش داده می‌شود.

```
n=5;
s=n.toString(2);
document.write(s);          // 101
```

انواع عملگر در جاوا اسکریپت

الف- عملگرهای ریاضی

این عملگرها برای انجام محاسبات ریاضی کاربرد دارند.

عملگر ←	+	-	*	/	%	++	--
مفهوم ←	جمع	تفریق	ضرب	تقسیم	باقیمانده	افزایش یک واحد	کاهش یک واحد

نکته۱: در دستورات محاسباتی و انتساب، چنانچه عملگرهای ++ و - - قبل از یک متغیر قرار گیرند ابتدا یک واحد به متغیر اضافه یا کسر می‌شود سپس محاسبات لازم انجام می‌شود اما اگر عملگرهای ++ و - - بعد از یک متغیر قرار گیرند محاسبات لازم با مقدار فعلی عملگر انجام می‌شود و بعد یک واحد به متغیر اضافه یا کسر می‌شود.
مثال: با توجه به قطعه کد زیر ابتدا یک واحد به x اضافه می شود، سپس حاصل در y قرار می گیرد.

```
var x=11,y=15;
y=++x;                  // x=12,y=12
x=(y*x)/4;              //  x=( 12*12)/4=36
document.write(x);      // 36
document.write(y);      // 12
```

مثال: با توجه به قطعه کد زیر ابتدا مقدار فعلی x در y قرار می گیرد سپس یک واحد به آن اضافه می‌شود.

```
var x=11,y=0;
y=x++;                  // y=11,x=12
document.write(x);      // 12
document.write(y);      // 11
```

نکته۲: عملگر + علاوه بر محاسبه جمع اعداد و متغیرها برای اتصال رشته‌ها نیز استفاده می‌شود.
مثال: دو عدد از ورودی دریافت کرده سپس حاصل جمع آنها را نمایش دهید.

```
var n,m,sum
n=window.prompt("عدد اول را وارد کنید");
m=window.prompt("عدد دوم را وارد کنید");
sum=Number(n)+Number(m);
document.write(sum);
```

اعداد دریافتی توسط متد window.prompt بطور پیش‌فرض رشته در نظر گرفته می‌شود و به همین خاطر باید هنگام محاسبه جمع با استفاده از تابع تبدیل Number() تبدیل به عدد شوند در غیر اینصورت اعداد دریافتی بجای اینکه جمع شوند با هم الحاق می‌شوند.

ب- عملگرهای انتساب

برای انتساب یک مقدار به یک متغیر استفاده می‌شوند.

عملگر ←	=	+=	-=	*=	/=	%=
مفهوم ←	انتساب	انتساب جمع	انتساب تفریق	انتساب ضرب	انتساب تقسیم	انتساب باقیمانده

بجز عملگر = در سایر عملگرها به این صورت عمل می‌شود که به عنوان مثال x+=y یعنی x=x+y و x+=2 یعنی x=x+2.
مثال: کد زیر چه نتیجه‌ای در بر دارد.

```
var x=4;y=5;z=6;
z%=x;                   // z=z%x; → z=2
y+=z++;                 // y=y+z++; → y=7, z=3
document.write(y);      // 7
document.write(z);      // 3
```

در دستور z%=x باقیمانده z بر x محاسبه و در z قرار می‌گیرد.
در دستور y+=z++ مقدار فعلی z با y جمع شده و حاصل در y قرار می گیرد سپس یک واحد به z اضافه می‌شود.

ج- عملگرهای مقایسه‌ای

برای مقایسه اعداد و متغیرها بکار می‌روند و نتیجه را بصورت true یا false برمی‌گردانند.

عملگر ←	==	===	!=	>	>=	<	<=
مفهوم ←	بررسی از نظر مقدار	بررسی از نظر نوع و مقدار	نامساوی	بزرگتر	بزرگتر مساوی	کوچکتر	کوچکتر مساوی

سوال: حاصل "10"==10 چیست؟

پاسخ: چون هر دو مقدار یکسان هستند و در عملگر == نوع داده (رشته‌ای یا عددی بودن) بررسی نمی‌شود بنابراین حاصل true است.

سوال: حاصل "10"===10 چیست؟

پاسخ: در این مقایسه مقادیر یکسان هستند ولی یکی رشته‌ای و دیگری عددی است از آنجا که در عملگر === نوع داده بررسی می‌شود حاصل false است.

مثال: قطعه کد زیر چه نتیجه‌ای در بردارد؟

```
Var x=10;r
r=(x*2) >= 20;              // r=(10*2)>=20 →(20>=20)→r=true
document.write(r);          // true
document.write(typeof(r));  // boolean
document.write(x==10);      // true
document.write(x==="10");   // false
```

د- عملگرهای منطقی

برای ترکیب چند شرط استفاده می شوند و حاصل را بصورت true یا false بر می‌گردانند.

عملگر ←	&&	||	!
مفهوم ←	و	یا	نه (معکوس)

در جدول زیر نتیجه حاصل از ترکیب دو مقایسه با استفاده از این عملگرها آمده است:

مقایسه ۱	مقایسه ۲	&&	||
T	t	t	t
T	f	f	t
F	t	f	t
F	f	f	f

نکته۱: در جدول فوق t به معنی درست و f به معنی نادرست است.

نکته۲: نتیجه عملگر && زمانی درست است که هر دو مقایسه درست باشد در سایر حالات نتیجه نادرست است.

نکته۳: نتیجه عملگر || زمانی نادرست است که هر دو مقایسه نادرست باشد در سایر حالات نتیجه درست است.

مثال۱: خروجی کد زیر چیست؟

```
var x=10, y=12 , z=false;
document.write(x>y || (y*2)<10);   //  10>12 ||24<10 → false || false → false
document.write(!z && y>x);         // !false && 12>10 → true && true → true
```

مثال۲: خروجی کد زیر چیست؟

```
var x=15, y=11 , z=true;
document.write(!(z && y>x));       // !(true && 11>15) → !(true && false) →!(false) → true
```

در این مثال ابتدا داخل پرانتز محاسبه می‌شود سپس توسط عملگر ! نتیجه معکوس می شود.

نکته: در عباراتی مانند مثال ۱ که از عملگرهای ریاضی، مقایسه‌ای و منطقی استفاده شده است ابتدا عملگرهای ریاضی محاسبه می‌شوند سپس عملگرهای مقایسه‌ای و در نهایت عملگرهای منطقی.

فصل چهارم: کنترل روند اجرای برنامه

دستورات شرطی

الف- دستور if

در شرایطی که لازم است یک شرط بررسی شود و در صورت درستی شرط، قطعه کدی اجرا گردد از این دستور استفاده می‌شود. در صورت نادرست بودن شرط کنترل برنامه به دستور بعد از if منتقل می شود.

```
If (شرط)
{
دستوراتی که در صورت درستی شرط اجرا می‌شوند.
}
```

مثال۱: در قطعه کد زیر یک عدد از ورودی دریافت شده و چنانچه عدد کوچکتراز ۱۰ باشد پیغام مناسب نمایش داده می‌شود.

```
n=window.prompt("یک عدد وارد کنید");
```

```
if (n<10)
{
  document.write("عدد وارد شده کوچکتر از ۱۰ است");
}
```

نکته: در صورت درست بودن شرط اگر بخواهیم فقط یک دستور اجرا شود می‌توانیم آکولادها({}) را حذف کنیم.

مثال۱: در قطعه کد زیر یک عدد از ورودی دریافت شده و چنانچه عدد سه رقمی باشد پیغام مناسب نمایش داده می‌شود.

```
n=window.prompt("یک عدد وارد کنید");
if (n>=100 && n<=999)
  document.write("عدد وارد شده کوچکتر از ۱۰ است");
```

سوال: نتیجه قطعه کد زیر چیست؟

```
Var n=20;
If   (n<10)
{
   Document.write("عدد وارد شده کوچکتر از ۱۰ است");
}
```

پاسخ: چون شرط نادرست است هیچ عبارتی نمایش داده نمی‌شود

ب- دستور if...else

شکل کاملتر دستور if است و چنانچه سرط درست باشد مجموعه‌ای از دستورات اجرا می‌شوند و اگر شرط نادرست باشد مجموعه‌ی دیگری از دستورات اجرا خواهند شد سپس کنترل برنامه به دستور بعد از if منتقل می‌شود.

```
If  (شرط)
{
دستوراتی که بر اساس درستی شرط اجرا می‌شوند.
}
Else
{
دستوراتی که بر اساس نادرستی شرط اجرا می‌شوند.
}
```

مثال۱: در قطعه کد زیر نمره سه درس دانش آموز دریافت شده و بعد از محاسبه معدل وضعیت دانش آموز (قبول یا مردود) نمایش داده می شود.

```
n1=window.prompt("نمره درس ۱ وارد کنید");
n2=window.prompt("نمره درس ۲ وارد کنید");
n3=window.prompt("نمره درس ۳ وارد کنید");
avg=(n1+n2+n3)/3;
if (avg>=10)
  document.write("قبول");
else
  document.write("مردود");
```

در این مثال چون در صورت درستی یا نادرستی شرط فقط یک دستور اجرا می‌شود آکولاد درج نشده است.

مثال۲: در قطعه کد زیر یک عدد دریافت شده و سپس زوج یا فرد بودن آن نمایش داده می شود.

```
n=window.prompt("یک عدد وارد کنید");
if (n%2==0)
  document.write("زوج");
else
  document.write("فرد");
```

اگر باقیمانده عدد بر ۲ صفر شود عدد زوج در غیر اینصورت عدد فرد است.

ج- دستور switch

وقتی تعداد شرط‌هایی که می‌خواهیم متناظر با شرط عملیات خاصی انجام شود، زیاد است بهتر است بجای if از دستور switch استفاده شود.

```
switch (متغیر یا عبارت محاسباتی مورد مقایسه)
{
```

```
 case مقدار ۱ :
     دستورات مربوط به مقدار ۱
     break;
 case مقدار ۲ :
     دستورات مربوط به مقدار ۲
     break;
.
.
.
 default:
    دستورات
}
```

در دستور فوق در صورت درست بودن هر یک از مقادیر، دستورات متناظر با آن اجرا می‌شود و چنانچه هیچ یک از مقادیر برقرار نباشد دستورات بخش default اجرا خواهد شد.

مثال: در قطعه برنامه زیر عددی بین ۱ تا ۴ دریافت شده و معادل آن یکی از فصل‌های سال نمایش داده می‌شود

```
N=window.prompt("یک عدد وارد کنید");
switch  (n)
{
   case 1:
     document.write("بهار");
    break;
   case 2:
     document.write("تابستان");
    break;
   case 3:
     document.write("پاییز");
    break;
   case 4:
     document.write("زمستان");
    break;
default:
     document.write("عدد وارد شده معتبر نیست");
}
```

حلقه‌های تکرار

گاهی لازم است تا دستور یا دستوراتی از برنامه چندین مرتبه اجرا شوند(تکرار شوند)، در این صورت از حلقه تکرار استفاده می‌شود.

اجزاء حلقه تکرار

۱. متغیر شمارنده: یک متغیر کمکی برای شمارش تعداد دفعات تکرار و قبل از شروع حلقه مقداراولیه می‌گیرد.
۲. گام افزایش یا کاهش : مقدار افزایش یا کاهش متغیر شمارنده در هر بار اجرای حلقه
۳. بدنه حلقه : دستوراتی که باید تکرار شوند.
۴. شرط خاتمه : شرطی که با توجه به متغیرشمارنده، پایان حلقه را مشخص می کند.

انواع حلقه تکرار

حلقه تکرار معین : در صورتی که تعداد تکرار دستورات مشخص باشد از این حلقه استفاده می شود. مانند حلقه for

حلقه تکرار نامعین: در حالتی که تعداد تکرار دستورات مشخص نیست از این حلقه ها استفاده می‌شود. مانند حلقه while و حلقه do ... while

الف- حلقه for

```
for (متغیر شمارنده = مقدار اولیه ; شرط خاتمه حلقه ; گام افزایش یا کاهش )
 {
    بدنه حلقه
 }
```

مثال۱: دستور زیر عبارت javascript را ۱۰ بار روی صفحه چاپ می‌کند.

```
for (x=1;x<=10;++x)
```

```
{
   document.write("javascript");
   document.write("<br/>");
}
```

مثال۲: دستور زیر باعث نمایش اعداد زوج بین ۲ تا ۲۰ می شود.

```
for (x=2;x<=20;x+=2)
{
   document.write(x);
   document.write("<br/>");
}
```

مثال۴: دستور زیر باعث نمایش توان دوم اعداد ۱ تا ۱۰ می‌شود.

```
for (x=1;x<=10;++x)
{
   document.write(x);
   document.write("<sup>2</sup>=");
   document.write(x*x);
   document.write("<br/>");
}
```

نکته: در حلقه‌های تکرار چنانچه بخواهیم فقط یک دستور تکرار شود می‌توانیم آکولاد را ننویسیم.

مثال۳: دستور زیر باعث نمایش اعداد ۱۰ تا ۱ می شود.

```
for (x=10;x>=1;--x)
   document.writeln(x);
```

ب- حلقه while

```
مقدار اولیه = متغیر شمارنده
while (شرط)
{
بدنه حلقه
}
```

مثال: قطعه کدی بنویسید که عبارت «خوش آمدید» را ۵ بار روی صفحه نمایش دهد.

```
var n=1;
while (n<=5)
{
document.write("خوش آمدید");
document.write("<br/>");
}
```

ج- حلقه do...while

عملکرد این حلقه با حلقه while یکسان است با این تفاوت که چون شرط حلقه در انتهای حلقه بررسی می‌شود دستورات داخل حلقه بدون در نظر گرفتن شرط حلقه، حداقل یکبار اجرا می شوند.

```
مقدار اولیه = متغیر شمارنده
do
{
بدنه حلقه
}
 while (شرط)
```

مثال: قطعه کدی بنویسید که یک عدد بزرگتر از صفر دریافت کرده و عبارت javascript را به تعداد عدد تکرار کند چنانچه عدد صفر و کوچکتر از آن وارد شود پیغام مناسبی نمایش داده شود.

```
n=window.prompt("یک عدد وارد کنید");
if (n>0)
{
  do
    {
      document.write("javascript");
      document.write("<br/");
      --n;
```

```
   }
  while (n>0)
}
else
{
  window.alert("عدد وارد شده درست نیست");
}
```

سوال: در قطعه کد زیر چه اتفاقی می‌افتد؟

```
var i=0;
while (i<0)
{
  document.write(i);
  ++i;
}
do
{
  document.write(i);
  ++i;
}
while (i<0)
```

پاسخ: در هر دو حلقه شرط از همان ابتدا نادرست است بنابراین حلقه while اجرا نمی شود اما در حلقه do...while چون شرط حلقه در انتها بررسی می شود یکبار اجرا شده و مقدار صفر نمایش داده می شود.

حلقه تودرتو

در صورتی که در بدنه حلقه تکرار از یک حلقه تکرار دیگر استفاده شود به آن حلقه تودرتو می‌گویند. دستوراتی که در بدنه حلقه داخلی قرار بگیرند به اندازه حاصلضرب حلقه داخلی در حلقه خارجی اجرا می‌شوند.

سوال: با اجرای کد زیر کلمه javascript چندبار روی صفحه نوشته می‌شود؟

```
for(i=1;i<=4;++i)
{
    for(j=1;j<=3;++j)
     {
       document.write("javascript");
       document.write("<br/>");
     }
}
```

پاسخ: ۱۲ بار (4*3=12)

حلقه بی‌پایان

در صورتی که شرط حلقه همیشه درست باشد حلقه هیچ وقت پایان نمی‌پذیرد، به این حلقه، حلقه بی‌پایان (بی‌نهایت) می‌گویند.

مثال: حلقه زیر بی‌پایان است

```
While (true)
{
...
}
```

دستور break

برای خروج از حلقه تکرار کاربرد دارد. قبل از اینکه یک حلقه پایان بپذیرد می توانیم به کمک این دستور از حلقه خارج شویم همچنین در حلقه‌های بی‌پایان برای خروج از حلقه از این دستور استفاده می‌شود.

مثال: در دستورات زیر اعداد ۱ تا ۴ نمایش داده می‌شود و وقتی مقدار i برابر ۵ باشد دستور break اجرا شده و حلقه به پایان می‌رسد.

```
var i=1
while (true)
{
  document.writeln(i);
  ++i;
  if (i==5)
     break;
}
```

دستور continue

وجود این دستور در داخل حلقه باعث می‌شود دستورات بعد از continue نادیده گرفته شود و کنترل برنامه به ابتدای حلقه منتقل شود.

مثال: خروجی دستورات زیر چیست؟

```
for(i=1;i<=8;++i)
{
   if (i==4)
       continue;
   document.writeln(i);
}
```

هنگامیکه مقدار i برابر۴ می‌شود دستور continue اجرا شده و عدد ۴ چاپ نمی‌شود بنابراین خروجی عبارت است از: 1 2 3 5 6 7 8

مثالهای بیشتر

مثال۱- صفحه‌ای حاوی یک دکمه ایجاد کنید تا کاربر با کلیک روی آن اعداد فرد دو رقمی را مشاهده کند.

```
<script  type="text/javascript">
function  fard()
{
for(i=11;i<=99;i+=2)
  document.writeln(i);
}
</script>
<input   type="button"   value="اعداد فرد"  onclick="fard()"/>
```

مثال۲- خروجی کد زیر چیست؟

```
var i=5;
while (--i>1)
   document.writeln(i);
```

پاسخ:اعداد ۲ و ۳و ۴ بصورت 2 3 4 روی صفحه نمایش داده می‌شود.

مثال۳- کد زیر چه نتایجی را بر می‌گرداند؟

```
for(i=0;i<=25;i=i+2)
   document.write(i);
```

پاسخ: اعداد زوج 0 تا 25

مثال۴- کدی بنویسید که با استفاده از حلقه‌های تودرتو شکل زیر را ایجاد کند

```
*
* *
* * *
* * * *
* * * * *
for(i=1;i<=5;++i)
{
   for(j=1;j<=i;++j)
       document.write("*");
   document.write("<br/>");
}
```

در حلقه for داخلی فقط یک دستور تکرار می شود به همین خاطر از آکولاد استفاده نشده است.

مثال۵- کدی بنویسید که تعدادی عدد بزرگتر از صفر دریافت کرده و تعداد اعداد زوج را شمارش کرده نمایش دهد. اگر عدد صفر یا کوچکتر از آن وارد شود حلقه پایان پذیرد.

```
var z=0;
while (true)
{
   n=window.prompt("یک عدد وارد کنید");
   if (n<=0)
      break;
   if (n%2==0)
      ++z;
}
document.wriet(z);
```

متغیر z برای شمارش اعداد زوج در نظر گرفته شده است. هنگامیکه باقیمانده عدد دریافتی (n) بر ۲ صفر شود (یعنی عدد زوج است) به مقدار z یکی افزوده می شود و در پایان حلقه، مقدار آن نمایش داده می شود.

مثال۶- کدی بنویسید که تعدادی عدد بزرگتر از صفر دریافت کرده و میانگین آنها را نمایش دهد. ورود عدد صفر به معنی پایان حلقه است.

```
var c=0,sum=0,avg;
while (true)
{
   n=window.prompt("یک عدد وارد کنید");
   if (n<=0)
      break;
  sum=sum+n
  ++c;
}
avg=sum/c;
document.write(avg);
```

متغیر sum برای محاسبه مجموع اعداد دریافتی در نظر گرفته شده است و با هر بار اجرای حلقه مقدار آن با مقدار عدد دریافتی(n) جمع می شود.

متغیر c برای شمارش تعداد اعداد دریافتی در نظر گرفته شده و با هر بار اجرای حلقه یک واحد به آن افزوده می شود.

در نهایت بعد از پایان حلقه، مجموع اعداد بر تعداد اعداد تقسیم می‌شود و نتیجه بعنوان میانگین اعداد در متغیر avg قرار گرفته و نمایش می‌یابد.

فصل پنجم: کار با اشیاء جاوااسکریپت

شئ string

برای ذخیره‌سازی عبارات متنی و دستکاری رشته‌ها استفاده می‌شود.

تعریف رشته با حالت‌های مختلف

```
var s1=new String();              //تعریف شئ رشته‌ای بدون مقدار اولیه
var s2=new String("javascript"); // تعریف شئ رشته‌ای با مقدار اولیه
var s3="vbscript";                //تعریف متغیر رشته‌ای با مقدار اولیه
```

متغیر s2 بصورت زیر در حافظه قرار می‌گیرد.

j	a	v	a	s	c	r	i	p	t	←رشته
0	1	2	3	4	5	6	7	8	9	←اندیس(نمایه)

برای دسترسی به هر یک از کاراکترهای رشته از اندیس آن استفاده می‌شود.

مهمترین خصوصیات و متدهای رشته در جدول زیر آمده است.

موضوع	نام	کاربرد
خصوصیت	Length	طول رشته
متد	charAt(اندیس)	دسترسی به کاراکتر موجود در اندیس
	concat(رشته۱, رشته۲, ...)	اتصال رشته‌ها (معادل عملگر +)
	replace(رشته, رشته جایگزین)	جایگزین کردن رشته در کل عبارت
	search(رشته)	جستجوی در رشته و برگرداندن محل رشته (عدد ۱- عدم وجود رشته)
	split("محل تقسیم")	تقسیم رشته به چند قسمت
	substr(شروع, [طول])	جدا کردن بخشی از رشته که محل شروع و طول رشته باید مشخص شود. اگر طول رشته مشخص نشود تا آخر رشته برگردانده می‌شود.
	toLowerCase()	تبدیل رشته به حروف کوچک
	toUpperCase()	تبدیل رشته به حروف بزرگ
	big()	بزرگ کردن فونت
	small()	کوچک کردن فونت
	bold()	ضخیم کردن رشته
	italics()	مایل کردن رشته
	fontcolor("رنگ")	تغییر رنگ فونت

	link("نشانی")	تبدیل به پیوند
	blink()	چشمک زن (در مرورگرهای ie و chrome و safari عمل نمی‌کند)

مثال۱: خروجی قطعه کد زیر چیست؟

```
var s1=new String("java script");
var s2=new String("java");
document.write("<br/>"+s1.charAt(0));                  // j
document.write("<br/>"+s1.length);                     // 11
document.write("<br/>"+s1.replace("java","vb"));       // vb script
document.write("<br/>"+s1.split(" "));                 // java,script
document.write("<br/>"+s1.substr(5,6));                // script
document.write("<br/>"+s1.toUpperCase());              //  JAVA SCRIPT
```

مثال۲: کدی بنویسید که دو رشته از ورودی دریافت کرده و تشخیص دهد که آیا رشته دوم در رشته اول وجود دارد یا خیر.

```
var s1=new String();
var s2=new String();
s1=window.prompt("رشته اصلی را وارد کنید");
s2=window.prompt("رشته مورد جستجو را وارد کنید");
n= s1.search(s2)
if (n!=-1)
  window.alert("رشته مورد نظر پیدا شد");
else
  window.alert("رشته مورد نظر پیدا نشد ");
```

شیء Date

برای استخراج زمان و تاریخ کاربرد دارد.

تعریف متغیر از نوع شئ تاریخ با حالات مختلف

```
var d1=new Date();                  // زمان و تاریخ جاری سیستم در متغیر قرار می‌گیرد
var  d2=new Date(2012,8,13);        // تعریف متغیر بصورت شئ تاریخ با مقدار اولیه
```

مهمترین متدها و خصوصیات این شیء در جدول زیر آمده است.

موضوع	نام	کاربرد
تاریخ	getDate()	شماره روز در ماه عددی بین ۱ تا ۳۱
	getDay()	شماره روز در هفته عددی بین ۰ تا ۶
	getMonth()	شماره ماه در سال عددی بین ۰ تا ۱۱
	getFullYear	سال بصورت ۴ رقمی
ساعت	getHours()	ساعت (عددی بین ۰ تا ۲۳)
	getMinutes()	دقیقه (عددی بین ۰ تا ۵۹)
	getSeconds()	ثانیه (عددی بین ۰ تا ۵۹)
	getTimes()	زمان سپری شده از روز ۱۹۷۰/۱/۱ بر حسب میلی ثانیه
	setTimeout(تابع,تاخیر)	اجرای تابع بعد از گذشت زمان مورد نظر

مثال۱: کدی بنویسید که زمان جاری را نمایش دهد

```
var d=new Date();
t=getHours()+":"+getMinutes()+":"+getSeconds();
document.write(t);
```

دستور فوق فقط یکبار زمان را نمایش می‌دهد و برای اعمال تغییرات باید صفحه نوسازی شود. برای تغییر خودکار ساعت باید کد آنرا بصورت زیر تغییر داد.

```
function mytime()
{
        var d=new Date();
        h=d.getHours();
        m=d.getMinutes();
        s=d.getSeconds();
         t= h+":"+m+":"+s;
```

```
        document.getElementById("p1").innerHTML=t
        setTimeout("mytime()",500);
}
</script>
<body onload="mytime()">
<p id="p1"></p>
</body>
```

به این ترتیب بعد از گذشت هر ۵۰۰ میلی ثانیه مجدد ساعت نمایش داده می‌شود.

مثال۲: کدی بنویسید که نام روز در هفته را نمایش دهد.

```
var d=new Date();
day=d.getDay();
switch (day)
{
        case 0:
        document.write("یکشنبه");
        break;
        case 1:
        document.write("دوشنبه");
        break;
        case 2:
        document.write("سه شنبه");
        break;
        case 3:
        document.write("چهارشنبه");
        break;
        case 4:
        document.write("پنج شنبه");
        break;
        case 5:
        document.write("جمعه");
        break;
        case 6:
        document.write("شنبه");
        break;
}
```

مثال۳: کدی بنویسید که روزهای باقیمانده از سال ۲۰۱۲ را نمایش دهد.

```
var d=new Date();
var e=new Date(2012,11,31);        //پایان سال ۲۰۱۲ – شماره ماه عددی است از صفر تا ۱۱
day=(e-d)/86400000;
document.write(day);
```

انتهای سال ۲۰۱۲(2012,11,31) از تاریخ جاری سیستم کسر می‌شود که عددی است بر حسب میلی ثانیه. با تقسیم آن بر عدد ۸۶۴۰۰۰۰۰ (معادل یک روز بر حسب میلی ثانیه) تعداد روزهای باقیمانده از سال بدست می‌آید. (86400000=24*60*60*1000)

شیء Math

برای انجام محاسبات ریاضی استفاده می‌شوند.

شکل کاربرد:

خصوصیت یا متد.Math

مهمترین متدها و خصوصیات آن در جدول زیر آمده است.

موضوع	نام	کاربرد
خصوصیت	PI	عدد پی (3.14)
متد	Pow(x,y)	عدد x به توان عدد y
	Random()	تولید عدد تصادفی بین ۰ و ۱

	Round(x)	گرد کردن عدد x به نزدیکترین عدد صحیح
	Floor(x)	گرد کردن عدد x به کوچکترین عدد صحیح
	Ceil(x)	گرد کردن عدد x به بزرگترین عدد صحیح
	توابع مثلثاتی	شامل ... sin, cos, tan, asin, acos, atan,

مثال۱: تابعی که مساحت دایره به شعاع r را محاسبه کرده و برگشت می‌دهد.

```
function circle(r)
{
 s= Math.PI *Math.pow(r,2);     // πr²
 return s;
}
```

مثال۲: کدی بنویسید که یک عدد تصادفی بین ۰ تا ۶ تولید کند و نمایش دهد.

```
n=Math.random()*6
n=Math.round(n);
document.write(n);
```

مثال۳: تولید سه عدد تصادفی بین ۱ و ۱۸

```
for(i=1;i<=3;++i)
{
  n=1+Math.random()*17
  n=Math.round(n);
  document.write("<br/>"+n);
}
```

مثال۴: قطعه کد زیر ۵ عدد تصادفی بین ۵۰۰ تا ۱۰۰۰ تولید می‌کند.

```
i=1;
while (i<=5)
{
   n=Math.random()*1000;
   n=Math.round(n);
   if (n<500)
     continue;
  document.write("<br/>"+n);
  ++i;
}
```

مثال۵: تابعی که یک عدد تصادفی بین هر دو عدد دلخواه min و max تولید می‌کند.

```
function tasadofi(min,max)
{
   n= min+(Math.random()*(max-min));
   n=Math.round(n);
   return n;
}
```

مثال۶: خروجی قطعه کد زیر چیست؟

```
document.write(Math.floor(Math.PI)); // عدد پی (۳,۱۴) به سمت کوچکترین عدد صحیح گرد می‌شود یعنی ۳
```

شیء Array

متغیر ویژه‌ای است که می‌تواند در یک لحظه چندین مقدار را در خود نگه‌داری کند.

تعریف آرایه با حالت‌های مختلف

```
var   a=new Array();              // تعریف آرایه بدون مقدار اولیه
var   b=new Array(4);             // تعریف آرایه بدون مقدار اولیه و به طول ۴
var   c=new Array(12,5,16,3,19);  // تعریف آرایه با مقدار اولیه و به طول ۵
```

آرایه c بصورت زیر در حافظه ذخیره می‌شود.

12	5	16	3	19	←آرایه
0	1	2	3	4	←اندیس

برای قرار دادن مقادیر در آرایه بصورت زیر عمل می کنیم:

```
نام آرایه[اندیس]=مقدار;
```

دستور زیر باعث می‌شود خانه اول آرایه a مقدار بگیرد.

```
a[0]="ali";
```

برای نمایش محتوای آرایه c بصورت زیر عمل می‌کنیم:

```
document.write(c);          //12,5,16,3,19
document.write(c[2]);       //16
```

متدها و خصوصیات آرایه

موضوع	نام	کاربرد
خصوصیت	length	طول آرایه را بر می‌گرداند.
متد	sort()	محتوای آرایه را مرتب می‌کند.
	function(a,b){return a-b}	مقایسه اعداد در متد sort()

مثال۱: برنامه ای بنویسید که ۵ عدد از کاربر دریافت کرده و آنها را در آرایه قرار دهد سپس آنها را از آخر به اول نمایش دهد.

```
var a=new Array(5);
for(i=0;i<=4;++i)
  a[i]=window.prompt("یک عدد وارد کنید");
for(i=4;i>=0;--i)
  document.write(a[i]+",");
```

مثال۲: کدی بنویسید که ۲۰ عدد تصادفی بین ۰ و ۱۰۰۰ تولید و در آرایه قرار دهد سپس آرایه را مرتب روی صفحه نمایش دهد.

```
var a=new Array(20);
for(i=0;i<=19;++i)
  a[i]=Math.round(Math.random()*1000);
a.sort(function(a,b){return a-b});
 document.write(a);
```

متد sort() مرتب سازی را بر اساس ترتیب اعداد اجام می‌دهد و به همین خاطر عددی مثل ۲۰ قبل از ۵ قرار می‌گیرد بخاطر اینکه در ترتیب اعداد ۲ قبل از ۵ است برای رفع این مشکل و مرتب سازی صحیح اعداد از متد زیر درون پرانتز متد sort() استفاده می‌شود. این متد اعداد را با هم مقایسه می‌کند و ترتیب آنها درست می‌شود.

```
function(a,b){return a-b}
```

در متد فوق اگر بجای a-b از b-a استفاده شود مرتب سازی نزولی انجام می‌شود.

مثال۳: کدی بنویسید که نشان دهد امروز چه روزی از هفته است.

```
Var a=new Array("یکشنبه","دوشنبه","سه‌شنبه","چهارشنبه","پنجشنبه","جمعه","شنبه");
Var d=new Date();
Document.write(a[d.getDay]);
```

فصل ششم: طراحی فرم و اعتبارسنجی آن

تابع (function)

تعریف تابع

در برنامه های بزرگ، برنامه به قسمتهای کوچکتر تقسیم می شود که هر قسمت وظیفه بخشی از برنامه را انجام می دهد. به هر یک از این قسمتها یک تابع گفته می شود.

مزایای استفاده از تابع

۱. امکان نوشتن برنامه بصورت گروهی فراهم می‌شود.
۲. تسهیل اشکالزدایی برنامه
۳. صرفه‌جویی در زمان کدنویسی
۴. از تکرار کدنویسی جلوگیری می‌شود.

تابع را می‌توان در بخش بدنه صفحه (body)، سرصفحه (header) و یا درون فایل خارجی با پسوند js قرار داد و در زمان لازم آنرا فراخوانی کرد تا اجرا شود.

نحوه‌ی تعریف تابع

```
function   نام تابع(لیست پارامترها)
{
 بدنه تابع
}
```

نکته: مقداری که باید توسط تابع برگردانده شود (در صورت وجود) توسط دستور return تعیین می‌شود.

مثال: تابع زیر دو عدد بعنوان طول و عرض مستطیل دریافت می‌کند و مساحت آنرا برگشت می‌دهد.

```
function  rect(h,w)
{
   var a;
    a=h*w;
   return (a)
}
```

نحوه‌ی فراخوانی تابع

تابع همزمان با باز شدن صفحه اجرا نمی‌شود و اجرای آن مشروط به وقوع یک رویداد (مانند کلیک شدن دکمه) خواهد بود.

مثال۱: دستور زیر باعث اجرای تابع فوق می‌شود.

```
<input   type="button"   value="مساحت"   onclick="rect(5,8)"/>
```

با تگ input یک دکمه ایجاد شده است و با کلیک روی دکمه رویداد onclick باعث اجرای تابع می شود.

مثال۲: با کلیک روی پیوند زیر تابع فوق اجرا می‌شود.

```
<a    href="javascript:rect(5,8)">    مساحت مستطیل    </a>
```

آشنایی با رویدادها

هر یک از کنترل‌هایی که روی صفحه وب قرار می گیرند دارای رویدادهای مخصوص به خود هستند. یکی از ویژگی‌های جاوااسکریپت شناسایی رویداد و پاسخ مناسب نسبت به آن است. در زیر چند نمونه رویداد ذکر شده است:

موضوع	نام رویداد	زمان وقوع
ماوس	onclick	کلیک با ماوس
	ondblclick	دابل کلیک با ماوس
	onmouseover	قرار گرفتن اشاره‌گر ماوس روی کنترل
	onmousedown	فشار دادن دکمه ماوس
	onmouseup	رها کردن دکمه ماوس
صفحه کلید	onkeydown	فشردن کلید از صفحه کلید
	onkeyup	رها کردن کلید از صفحه کلید
	onkeypress	فشردن و رها کردن کلید از صفحه کلید
سایر موارد	onchange	تغییر مقدار کنترل
	onfocus	در اختیار گرفتن تمرکز (هنگام ورود به کنترل)
	onblur	از دست دادن تمرکز (هنگام ترک یک کنترل)
	Onselect	انتخاب عنصر
	Onload	هنگام بارگذاری صفحه(در تگ body نوشته می‌شود)

مفهوم تمرکز (focus)

هنگامیکه با ماوس یا کلید tab یک عنصر یا کنترل انتخاب می‌شود تمرکز به آن منتقل می‌شود و رویداد onfocus روی می‌دهد. هنگامی هم که با استفاده از کلید tab به سراغ عنصر دیگری می‌روید یا با ماوس روی عنصر دیگری کلیک می‌کنید و در واقع عنصر اول را ترک می‌کنید رویداد onblur آن عنصر رخ می‌دهد.

مثال۱: قطعه کد زیر تعداد کاراکترهای وارد شده در کادر متنی را همزمان با تایپ نمایش می‌دهد.

```
<script type="text/javascript">
function len()
{
   document.getElementById("p1").innerHTML=document.getElementById("txt1").value.length;
}
</script>
<input   type="text"    id="txt1"   onkeyup="len()" />
<p  id="p1" ></p>
```

مثال۲: در قطعه کد زیر با کلیک روی دکمه، جمع اعداد نمایش داده می‌شود.

```
function add()
```

```
{
	n1=document.getElementById("txt1").value
	n2=document.getElementById("txt2").value
	document.getElementById("p1").innerHTML=Number(n1)+Number(n2)
}
</script>
<input  type="text"   id="txt1"  />
<input  type="text"   id="txt2"  />
<p  id="p1" ></p>
<input  type="button"  value="+"   onclick="add()"/>
```

مثال ۳: در ابتدای بارگذاری صفحه پیغام خوش‌آمدگویی نمایش داده می‌شود.

```
<head>
<script type="text/javascript">
function msg()
{
	window.alert("به دنیای جاوااسکریپت خوش آمدید");
}
</script>
</head>
<body onload="msg()">
</body>
```

رویداد onload مربوط به تگ body است و همزمان با ورود صفحه به حافظه ram اجرا می شود.

طراحی فرم و اعتبارسنجی آن

برای طراحی فرم از تگ <form> استفاده می‌شود.

```
<form     name=”form1”>
…
</form>
```

نکته: برای دسترسی به فرم و کنترل های روی آن با استفاده از ویژگی name آنها را نامگذاری می‌کنیم.

اجزای فرم

کنترل‌های روی فرم مانند کادر متنی، دکمه و … با تگ <input/> ایجاد می‌شوند و با ویژگی type نوع کنترل مشخص می‌شود.

```
<input    type=”نوع کنترل ” name=”نام کنترل” />
```

مقادیر مربوط به ویژگی type:

Text : کادر متنی

Password : کادر متنی برای دریافت رمز

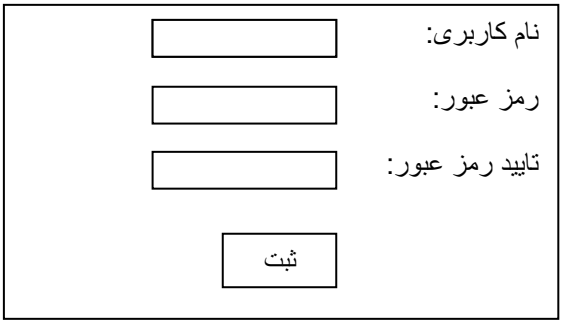
Button : دکمه

Checkbox : کادر انتخاب

Radio : دکمه رادیویی

و …

مثال: یک فرم بصورت زیر طراحی کنید.




```
<form name="form1">
	<input type="text" name="text1" />نام کاربری<br/>
```

```
        <input type="password" name="text2" />رمز عبور<br/>
        <input type="password" name="text3" />تایید رمز عبور<br/>
        <input type="button" name="button1" value="ثبت"  /><br/>
</form>
```

اعتبارسنجی فرم

برای دسترسی به داده های وارد شده از طریق فرم می‌توان به صورت زیر عمل کرد:

Document.نام فرم.نام کنترل.value

مثال۱: قطعه کد زیر داده‌های وارد شده در فرم بالا را بر اساس شرایط زیر کنترل می‌کند:

- طول نام کاربری باید بین ۶ تا ۱۲ حرف باشد.
- طول رمز بیشتر از ۶ کاراکتر باشد
- رمز و تایید آن یکسان باشد.

```
<script type="text/javascript">
msg=""
function error()
{
        u=document.form1.text1.value;
        if (u.length<6 || u.length>12)
          msg+="نام کاربری باید ۶ تا ۱۲ نویسه داشته باشد"+"<br/>"
         p=document.form1.text2.value;
         if (p.length<6)
          msg+="طول رمز عبور نباید از ۶ نویسه کمتر باشد"+"<br/>"
         cp=document.form1.text3.value;
        if (p!=cp)
          msg+="رمز عبور با تایید آن یکسان نیست "+"<br/>"
        if (msg=="")
         return true
        else
        {
                document.getElementById("p1").innerHTML=msg;
                msg=""
                return false
        }
}
</script>
<form name="form1">
        <table>
                <tr>
                        <td><label>نام کاربری:</label></td>
                        <td><input type="text" name="text1" /></td>
                </tr>
                <tr>
                        <td><label>رمز:</label></td>
                        <td><input type="password" name="text2" /></td>
                </tr>
                <tr>
                        <td><label>تایید رمز:</label></td>
                        <td><input type="password" name="text3" /></td>
                </tr>
                <tr>
                        <td><input type="button" name="button1" value="ثبت" onclick="error()" /></td>
                        <td><input type="button" name="button2" value="جدید" /></td>
                </tr>
        </table>
</form>
<p id="p1" style="color:red"></p>
```

مثال۲: در قطعه کد زیر با ترک کادر متنی اعتبارسنجی آن انجام می‌شود.

```
<script type=”text/javascript”>
function error()
{
	var u=document.frm1.txt1.value;
	if (u.length<6 || u.length>12)
	{
		window.alert("نام کاربری معتبر نیست");
		document.frm1.txt1.focus();
	}
}
</script>
<form action="" method="" name="frm1">
<input type="text" name="txt1" onblur="error()" />
</form>
```

دکمه رادیویی

با استفاده از قطعه کد زیر، روی فرم یک دکمه رادیویی برای مدرک تحصیلی با مقادیر دیپلم، فوق دیپلم و لیسانس در نظر گرفته شده است.

```
<label><input type="radio" name="radio1" value="1" />دیپلم</label><br />
<label><input type="radio" name="radio1" value="2"  />فوق دیپلم</label><br />
<label><input type="radio" name="radio1" value="3"  />لیسانس</label><br />
```

مهمترین خصوصیت دکمه رادیویی checked است که مشخص می‌کند آیا دکمه رادیویی انتخاب شده یا نه.

قطعه کد اعتبارسنجی آن در تابع error() وارد می‌شود:

```
m=document.form1.radio1;// قرار دادن گزینه‌های مربوط به دکمه رادیویی در آرایه
a=false;
for(i=0;i<m.length;++i)
{
	if(m[i].checked)
	  a=true
}
if (a==false)
    msg+="مدرک تحصیلی خود را انتخاب کنید"+"<br/>"
```

کادر انتخاب (checkbox)

توابعی برای انتخاب و لغو انتخاب کادرهای انتخاب روی فرم

```
function SelectAll()
{
s=document.form1.Check1
for(i=0;i<s.length;++i)
	document.form1.Check1[i].checked=true
}
function Deselect()
{
s=document.form1.Check1
for(i=0;i<s.length;++i)
	document.form1.Check1[i].checked=false
}
```

لیست

برای ایجاد لیست از تگ <select> استفاده می‌شود.

برای انتخاب گزینه‌های لیست از تگ <option> استفاده می‌شود.

```
   <select name="list">
	<option value="0" selected="selected">انتخاب شهر</option>
	<option value="1">تهران</option>
	<option value="2">مشهد</option>
	<option value="3">تربت حیدریه</option>
    </select>
```

خصوصیت selectedIndex گزینه انتخاب شده را مشخص می‌کند.

قطعه کد اعتبارسنجی آن در تابع error() وارد می‌شود.

```
l=document.form1.list; // قرار دادن گزینه‌های مربوط به لیست در آرایه
i=document.form1.list.selectedIndex;
if (l.options[i].value==0)
  msg+="محل سکونت خود را انتخاب کنید"+"<br/>"
```

پرش خودکار

در ابتدای نمایش صفحه وب برای پرش خودکار مکان نما به اولین کادر متنی بصورت زیر عمل می‌کنیم:

```
<body  onload="document.form1.text1.focus()"> //پرش به کادر متنی اول
```

و همچنین در کادرهای متنی که طول آنها از قبل مشخص است می‌توان هنگام ورود داده در کادر متنی کاری کرد که پرش خودکار مکان نما به کادر بعدی انجام شود.

مثال: در فرم زیر تاریخ تولد کاربر بصورت پرش خودکار دریافت می‌شود.

```
<form  name="form1">
<label>:تاریخ تولد</label>
<input  type="text" id="text1" size="2" maxlength="2" onkeyup="if(this.value.length== 2) form1.text2.focus()"/>-
<input  type="text" id="text2" size="2" maxlength="2" onkeyup="if(this.value.length== 2) form1.text3.focus()"/>-
<input  type="text" id="text1" size="4" maxlength="4" onkeyup="if(this.value.length== 4) form1.button1.focus()"/>
<input  type="button" id="button1" value="ثبت"/>
</form>
```

در قطعه کد فوق فرض شده است که اگر طول کادر متنی مربوط به روز تولد حداکثر ۲ کاراکتر وارد شد پرش به کادر متنی ماه تولد انجام شود. برای کادرهای متنی بعدی نیز به همین صورت عمل می‌کنیم.

عبارات منظم

عبارات منظم ترکیب‌های خاصی از حروف و علائم هستند که برای جستجو و مقایسه‌ی رشته‌ها استفاده می‌شوند. در داده‌هایی مانند کد ملی افراد و شماره پلاک خودرو که از الگوی خاصی پیروی می‌کنند می‌توان اعتبارسنجی آنها را با استفاده از عبارات منظم انجام داد. مثلا برای کد ملی که از ۱۰ رقم تشکیل شده است می‌توان از عبارت منظم ^[0-9]{10}$ استفاده کرد.

تعریف شئ از نوع عبارت منظم:

```
var  r=new  RegExp("^[0-9]{10}$");
```

متد test() : برای مقایسه داده‌ها با عبارت منظم بکار می رود

مثال: با فرض اینکه کد ملی در کادر متنی با نام text1 وارد شده است بصورت زیر می توان اعتبارسنجی آن را کنترل کرد.

```
c=document.form1.text1.value
if (r.test(c)==false)
   window.alert("کد ملی معتبر نیست")
```