۱۹-۵ متغیرهای محلی و سراسری

در همه زبان‌های برنامه‌نویسی هنگامی که نامی از متغیرها به میان می‌آید، در مورد طول عمر آن‌ها هم بحث می‌گردد؛ منظور از طول عمر یک متغیر، زمانی است که از تعریف متغیر تا از بین رفتن آن سپری می‌شود.

فرض کنید متغیری را در ابتدای کد جاوا اسکریپت تعریف نموده‌اید، طول عمر این متغیر، از زمان تعریف تا زمان بسته شدن مرورگر خواهد بود و لذا در ادامه کد نمی‌توانید متغیر دیگری را به همین نام تعریف نمایید چون باعث تداخل خواهد شد. به چنین متغیرهایی، سراسری گفته می‌شود چون در سرتاسر برنامه وجود دارند و قابل استفاده هستند.

حال فرض کنید متغیری را درون یک تابع تعریف کنیم. طبیعتاً طول عمر این متغیر از زمان شروع فراخوانی تابع تا اتمام اجرای آن خواهد بود و لذا تعریف متغیرهای هم‌نام درون تابع‌های مجزا ایرادی ندارد. این نوع متغیرها، محلی نامیده می‌شود چون در یک محدوده خاص قابل دسترس می‌باشند.

```
<script type="text/javascript">
function1
{
        var name="Vahid";
        document.write(name);
}
function2
{
        var name="Saeed";
        document.write("Your Name is: " + name);
}

var name1="Ali";
document.write(name1);

var name2="Reza"
document.write(name2);
</script>
```

متغیرهای محلی

متغیرهای سراسری

۱۹-۶ انواع داده‌ای[1] در جاوا اسکریپت

هر زبان برنامه‌نویسی یا اسکرپیت‌نویسی برای اجرای عملیات موردنظر با مجموعه‌ای از مقادیر کار می‌کند. این مقادیر که دارای نوع و محدوده خاصی هستند باید در طول برنامه به‌گونه‌ای ذخیره شوند تا بتوان عملیات

1 . Data Types

موردنیاز برای محاســبه، مقداردهی یا خواندن مقادیر را بدون بروز اشکال انجام داد. لذا در هر زبان مجموعه‌ای از انواع‌داده‌ای تعریف می‌شود که کدنویس باید با ویژگی‌های هر یک از آن‌ها آشنا باشد تا برای ذخیره‌سازی و بازیابی مقادیر با مشکل مواجه نشود.

انواع داده‌ای یا DataTypeهای موجود در جاوا اسکریپت را می‌توان به دو دسته تقسیم‌بندی کرد:

- **انواع داده‌ای اولیه (Primitive)**
- **انواع داده‌ای ارجاعی (Reference)**

۱-۶-۱۹ انواع داده‌ای اولیه

همان‌گونه که پیش از این در بخش «انتســاب مقادیر به متغیرها» دانســتید، داده‌ها را می‌توان در سه نوع متغیر منطقی، عددی و رشته‌ای ذخیره نمود. این‌ها همان انواع داده‌ای اولیه هستند که در این بخش جزییات مربوط به هریک را مرور خواهیم کرد.

نوع داده‌ای منطقی یا boolean: در این نوع داده‌ای می‌توان مقادیر «درست» یا «نادرست» را ذخیره نموده و در طول برنامه عمدتاً برای بررسی حالت‌های شرطی مورد استفاده قرار می‌گیرد. مقدار درست با true و مقدار نادرست با false تعیین می‌گردد مانند:

```
var bChecked=true;
```

نوع داده‌ای عددی یا number: این نوع داده‌ای می‌تواند حاوی اعداد صحیح یا اعشاری باشد مانند:

```
var iNum=-12;
var iCount=23;
var fNum=13.542;
var iBase8 = 070;
```

نوع داده‌ای رشــته‌ای یا string: برای ذخیره‌ســازی رشته‌ها از این نوع‌داده‌ای استفاده می‌شود. برای مثال وقتی عبارت زیر را درون برنامه می‌نویسید:

```
var sText="Hello!";
```

درون حافظه ســاختاری به صورت زیر تشــکیل می‌شــود و هریک از نویســه‌ها درون یکی از خانه‌ها قرار می‌گیرند. برای مثال حرف o در خانه‌ای با نمایه ۴ قرار خواهد گرفت.

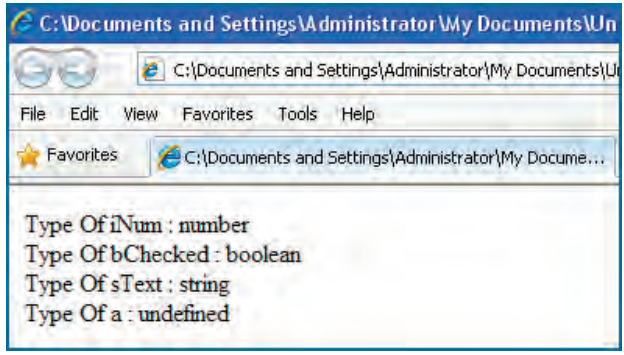
H	e	l	l	o	!
0	1	2	3	4	5

حال می‌خواهیم با تابعی آشنا شویم که یک متغیر را گرفته و نوع داده‌ای آن را برمی‌گرداند؛ این تابع جزو توابع پیش‌ساخته جاوا اسکریپت است و به صورت typeof(var) نوشته می‌شود.

مثال Example

خروجی کد زیر چیست؟

```
<body>
<script type="text/javascript">
var iNum=10, bChecked=false, sText="This is a js Code" , a ;
document.write("Type Of iNum: " + typeof(iNum) + "<br/>");
document.write("Type Of bChecked: " + typeof(bChecked) + "<br/>");
document.write("Type Of sText: " + typeof(sText) + "<br/>");
document.write("Type Of a: " + typeof(a) + "<br/>");
</script>
</body>
```



بررسی کد:

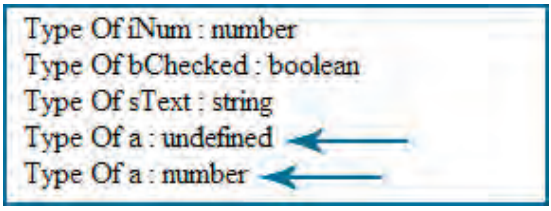
در این کد با اســتفاده از تابع typeof() نوع داده‌ای متغیرهای تعریف شــده برگردانده شده که نتیجه برای متغیرهای iNum، bChecked و sText مطابق انتظار است. اما برای متغیر a، نوع undefined به معنی «تعریف نشده» برگردانده شده است. دلیل این مسأله را باید در مقداری که به متغیر نسبت داده شده جستجو کرد.

همان‌گونه که در کد فوق می‌بینید، متغیر a تعریف شده اما هیچ مقداری به آن منتسب نشده بنابراین نوع داده‌ای آن برای برنامه مجهول مانده است.

مثال Example

قطعه کد زیر را به انتهای کد مثال قبل اضافه و نتیجه را در مرورگر مشاهده نمایید.

```
a=iNum;
document.write("Type Of a: " + typeof(a) + "<br/>");
```



بررسی کد:

همان‌طور که در این مثال می‌بینید، نوع داده‌ای متغیر a ابتدا undefined اســت اما وقتی متغیر iNum را به آن نسبت می‌دهیم، نوع داده‌ای آن همانند این متغیر می‌شود و با اجرای دستور typeof()، عبارت number برگردانده می‌شود.

۲-۶-۱۹ انواع داده‌ای ارجاعی

وقتی صحبت از انواع داده‌ای ارجاعی به میان می‌آید، مفاهیم شیءگرایی و در رأس همه آن‌ها مفهوم کلاس (Class) خودنمایی می‌کند. انواع داده‌ای ارجاعی در جاوا اسکریپت عبارتند از:

- **کلاس Boolean**: برای ایجاد یک شیء منطقی
- **کلاس Number**: برای ایجاد شیء عددی
- **کلاس String**: برای ایجاد شیء عددی

در بخش قبل با انواع داده‌ای اولیه و روش تعریف آن‌ها با اســتفاده از کلمه کلید var آشــنا شــدید. حال می‌خواهیم چگونگی تعریف یک نوع داده‌ای ارجاعی را فرابگیریم. در حالت کلی برای تعریف یک شیء از نگارش زیر استفاده می‌شود:

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

var مشـخص می‌کند که در حال تعریف یک متغیر هسـتیم و کلمه کلیدی new هم یک نمونه از کلاس object را ایجاد می‌کند تا به این ترتیب شیء o ایجاد شود. مزیت استفاده از انواع داده‌ای ارجاعی این است که به راحتی می‌توانیم به متدها و خصوصیت‌های تعریف شـده برای آن‌ها دسترسی پیدا کرده و عملیات موردنظر را به سادگی انجام دهیم.

قصد داریم یک شیء عددی ایجاد نموده و آن را مقدار دهی کنیم.

```
var n=new Number();
n=100;
```

مقداردهی اولیه به اشیاء را می‌توانیم در حین تعریف هم به صورت زیر انجام دهیم:

```
var n=new Number(100);
```

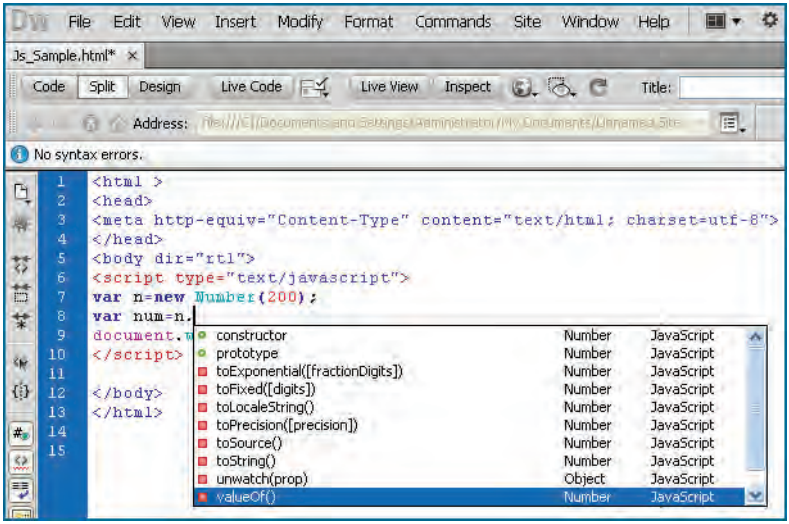
اکنون شیء عددی ایجاد و با عدد ۱۰۰ مقداردهی شده است و می‌توانیم با کد زیر، مقدار آن را روی صفحه بنویسیم:

```
document.write(n);
```

مقدار یک شیء عددی را درون یک متغیر عددی قرار دهید.

ابتدا یک شـیء عددی ایجاد و آن را با یک عدد مقداردهی می‌کنیم. حال برای انتسـاب مقدار این شـیء بـه متغیر عددی از متد ()valueof اسـتفاده می‌کنیـم. هنگامی که این کـد را درون محیط Dreamweaver می‌نویسید، با نوشتن نام شیء و تایپ نقطه، لیستی از متدها و خصوصیت‌های آن شیء ظاهر می‌شود.

```
<script type="text/javascript">
var n=new Number(200);
var num=n.valueOf();
document.write(num);
</script>
```



با نمایش صفحه فوق در مرورگر عدد ۲۰۰ نمایش داده می‌شود.

مثال Example

کدی بنویسید که ابتدا یک نام را ذخیره نموده و سپس نام را به همراه طول آن را روی صفحه وب بنویسد.

```
<script type="text/javascript">
var s=new String("Ali Salari");

document.write("طول نام "+ s + " برابر است با " + s.length + " نویسه ");
</script>
```

بررسی کد:

در این قطعه کد، ابتدا یک شـیء رشـته‌ای تعریف و با عبارت "Ali Salari" مقداردهی می‌شـود. سـپس محتوای رشته و نیز طول آن روی صفحه نمایش داده می‌شود. برای استخراج طول رشته از خصوصیت length

اســتفاده شــده که جز خصوصیت‌های تعریف شده توسط جاوا اسکریپت برای کلاس String است. بدون وجود چنیــن خصوصیتی مجبور بودیم با نگارش یک قطعه کد، طول آن را محاســبه کنیم و این‌جاســت که مزایای استفاده از روش شیءگرایی برای ایجاد برنامه آشکار می‌شود.

نتیجه اجرای کد زیر چیست ؟

```
<script type="text/javascript">
var b1=new Boolean(1)
var b2= b1.valueOf();
document.write(b2);
</script>
```

عبارت true روی صفحه ظاهر می‌شود چون شیء منطقی b1، با عدد ۱ (معادل true) مقداردهی شده است و سپس این مقدار درون متغیر منطقی b2 قرار گرفته و نهایتاً مقدار b2 روی صفحه نوشته شده است.

آیا اجرای کد زیر با خطا مواجه می‌شود؟

```
<body dir="rtl">
<script type="text/javascript">
var name="Hamid Razavi";
document.write(name.length);
</script>
```

در نگاه اول ممکن اســت این‌طور به نظر برســد که یک متغیر رشته‌ای (نه شــیء رشته‌ای) تعریف شده و ســپس از خصوصیت length آن اســتفاده شده است در صورتی‌که خصوصیت‌ها و متدها مربوط به انواع داده‌ای ارجاعی اســت نه اولیه! بنابراین اجرای کد باید با خطا مواجه شــود اما نکته این‌جاست که جاوا اسکریپت حاوی سازوکاری درونی برای تبدیل ضمنی متغیرها به اشیاء است لذا شما می‌توانید یک متغیر رشته‌ای تعریف نموده و در آن از متدها و خصوصیت‌های اشیاء رشته‌ای استفاده نمایید.

۱۹-۷ تبدیل انواع داده‌ای

هر زبان برنامه‌نویســی قدرتمندی باید دارای سازوکاری کارآمد برای تبدیل انواع داده‌ای به یک‌دیگر باشد.

برای نمونه ممکن است هنگام بررسی مقادیر وارد شده در یک فرم وب، شماره ملی کاربر را به صورت رشته‌ای دریافت کنید اما لازم باشد برای ارزیابی آن، رشته دریافت شده تبدیل به عدد شود. به این کار تبدیل انواع داده‌ای یا Data Type Convertion گفته می‌شود.

۱-۷-۱۹ تبدیل به رشته

برای تبدیل اعداد یا مقادیر منطقی به رشته، می‌توانید از متد ()toString که برای اشیاء عددی (و به تبع آن متغیرهای عددی) و نیز متغیرهای منطقی تعریف شده استفاده نمایید.

کد زیر چه مقداری را نمایش می‌دهد؟

```
<script type="text/javascript">
var n=100;
var s=n.toString();
document.write(typeof(s));
</script>
```

عبارت string را برمی‌گرداند چون عدد n با استفاده از متد ()toString تبدیل به یک متغیر رشته‌ای شده و در s ذخیره گردیده است بنابراین اکنون نوع داده‌ای s، عبارت است از string.

خروجی کد زیر چیست؟

```
<script type="text/javascript">
var b =new Boolean(0);
document.write(b.toString());
</script>
```

عبارت false روی صفحه نوشته می‌شود چون شیء منطقی b با مقدار صفر (معادل false) مقداردهی شده است و سپس محتوای آن توسط متد ()toString به رشته تبدیل و روی صفحه نمایش داده شده است.

کد زیر چه عبارتی را برمی‌گرداند؟

```
<script type="text/javascript">
```

```
var n =7;
document.write(n.toString(2));
</script>
```

وقتی در پرانتز متد ()toString عدد ۲ درج شود، نتیجه به صورت دودیی نمایش داده می‌شود و لذا به جای عدد ۷، معادل دودویی آن یعنی ۱۱۱ نمایش داده خواهد شد. عدد ۱۶ هم برای نمایش هگزادسیمال استفاده می‌شود.

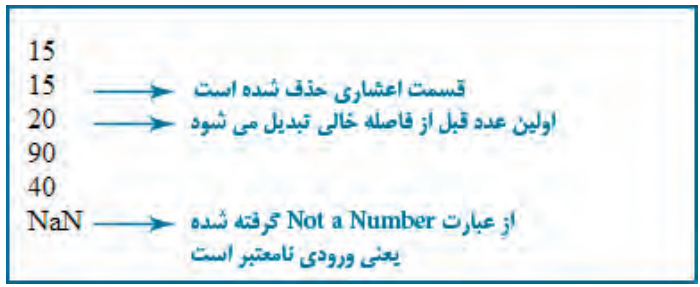
۲-۷-۱۹ تبدیل به عدد

در جاوا اسکریپت برای تبدیل مقادیر رشته‌ای به عدد صحیح از تابع parseInt(string) استفاده می‌شود. تابع[1] که در ادامه کتاب با روش نگارش و فراخوانی آن بیش‌تر آشنا خواهید شد، حاوی مجموعه‌ای از دستورات است که صفر، یک یا چند ورودی را پذیرفته و حداکثر یک خروجی را برمی‌گرداند.

کد زیر چه مقادیری را برمی‌گرداند؟

```
<script type="text/javascript">
document.write(parseInt("15") + "<br />");
document.write(parseInt("15.18") + "<br />");
document.write(parseInt("20 10 30") + "<br />");
document.write(parseInt(" 90 ") + "<br />");
document.write(parseInt("40 Books") + "<br />");
document.write(parseInt("Number 40") + "<br />");
</script>
```

مقادیر زیر برگردانده می‌شود:



1 . Function

چنان‌چه می‌خواهید رشـــته موردنظر را به عدد اعشـــاری تبدیل کنید باید به جای تابع ()parseInt از تابع ()parseFloat استفاده نمایید.

با اجرای کد زیر چه اعدادی روی صفحه نمایش داده می‌شود؟

```
<script type="text/javascript">
document.write(parseFloat("20") + "<br />");
document.write(parseFloat("15.33") + "<br />");
document.write(parseFloat("34.5 11 18") + "<br />");
</script>
```

اعداد 20، 15.33 و 34.5 روی صفحه وب ظاهر می‌گردد.

۳-۷-۱۹ استفاده از توابع تبدیل[1]

روش دیگر برای تبدیل انواع داده‌ای به یک‌دیگر از توابع تبدیل است. در جاوا اسکریپت سه تابع تبدیل زیر تعریف شده‌اند که مقدار دریافتی را به نوع‌داده‌ای متناظر تبدیل می‌کنند:

- Boolean (value)
- Number (value)
- String(value)

نتیجه اجرای کد زیر چیست؟

```
<script type="text/javascript">
var a1 = Boolean(1);
var a2 = Number(false);
var a3 = String(122.33);
document.write(a1 + "<br />");
document.write(a2 + "<br />");
document.write(a3 + "<br />");
</script>
```

1 . Cast Functions

عبارت‌های true، 0 و 122.33 نمایش داده می‌شود.

```
true    ⟶  عدد ۱ توسط تابع Boolean() تبدیل به مقدار منطقی true شده است
0       ⟶  مقدار منطقی false توسط تابع Number() تبدیل به عدد ۰ شده است
122.33  ⟶  عدد درون پرانتز توسط تابع String() تبدل به رشته شده است
```

۱۹-۸ شناخت عمل‌گرها

در جاوا اسکریپت همانند هر زبان برنامه‌نویسی یا اسکریپت‌نویسی دیگری، تعدادی عملگر[1] وجود دارد که بر روی عملوند(ها)[2]، عملیات خاصی را انجام می‌دهند. این عملیات می‌تواند انجام یک محاسبه ریاضی، اجرای یک مقایسه و ... باشد.

۱۹-۸-۱ عمل‌گرهای ریاضی

از عملگرهای ریاضی برای انجام محاسبات ریاضی ساده نظیر جمع، ضرب، محاسبه باقی‌مانده و ... استفاده می‌شود. در جدول زیر عملگرهای ریاضی قابل استفاده در جاوا اسکریپت را مشاهده می‌کنید.

مقدار اولیه متغیر X در همه ردیف‌ها برابر با عدد ۶ است.

عملگر	عملیات	مثال	نتیجه
+	جمع	Y=X+2	Y=8
-	منها	Y=X-2	Y=4
*	ضرب	Y=X*3	Y=18
/	تقسیم	Y=X/4	Y=1.5
%	باقی‌مانده	Y=X%5	Y=1
++	افزایش یک‌واحدی	Y=++X	Y=7
--	کاهش یک‌واحدی	Y=--X	Y=5

1 . Operator
2 . Operand

مثال Example

کدی بنویسید که هنگام باز شدن صفحه وب، یک عدد از کاربر گرفته شود و رقم سمت راست آن در یک پنجره پیغام به نمایش در بیاید.

```
<html ><head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
</head>
<body>
<script type="text/javascript">
var n= window.prompt(یک عدد صحیح وارد کنید:);
window.alert("رقم سمت راست عدد وارد شده: " + n%10);
</script>
</body></html>
```

نتیجه اجرای کد را در تصویر زیر می‌بینید.

بررسی کد:

ابتدا با اســتفاده از متد window.prompt() پنجره ورود عدد ظاهر می‌گردد. هنگامی که کاربر عدد را وارد و روی دکمه OK کلیک می‌کند، مقدار وارد شــده درون متغیر n ذخیره می‌شــود. ســپس با اســتفاده از متد window.alert() باقی‌مانده این عدد بر ۱۰ (یعنی رقم سمت راست) به همراه یک پیغام نمایش داده می‌شود.

کد زیر چه نتیجه‌ای را برمی‌گرداند؟

```
<script type="text/javascript">
var x=11,y=15;
y=++x;
x=(y*x)/4;
document.write(x + "<br/>");
document.write(y + "<br/>");
</script>
```

مقدار اولیه x برابر با ۱۱ و y برابر با ۱۵ است. دستور x++ مقدار متغیر x را یک‌واحد افزایش می‌دهد و بنابراین برابر با ۱۲ می‌شود. همین مقدار هم در متغیر y قرار می‌گیرد و مقدار قبلی آن یعنی ۱۵ پاک می‌شود. حال حاصل‌ضرب x*y/4 (یعنی 12*12/4) درون x قرار می‌گیرد. بنابراین خروجی کد فوق

```
12
36
```

است.

خروجی کد زیر چیست؟

```
<script type="text/javascript">
var x=11,y=0, z=0;
y=x++;
z=++x;
++z;
document.write("X= " + x + "<br/>");
document.write("Y= " + y + "<br/>");
document.write("Z= " + z + "<br/>");
</script>
```

بررسی کد:

وقتی عملگرهای ++ و – بعد از یک متغیر قرار می‌گیرند، ابتدا عملیات انتساب با مقدار فعلی متغیر انجام می‌شود و سپس عمل افزایش یک‌واحدی انجام خواهد شد.

بنابراین در کد

```
y=x++;
```

ابتدا مقدار x (یعنی ۱۱) درون y قرار می‌گیرد و سپس مقدار x یک واحد افزایش می‌یابد. اکنون مقدار x برابر است با ۱۲ . در ادامه، دستور

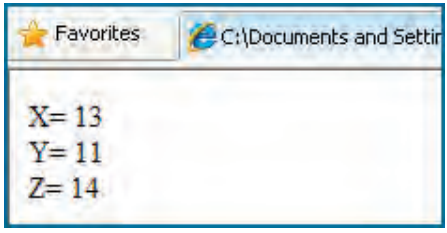
```
z=++x;
```

ابتدا مقدار x را یک‌واحد افزایش داده و سپس عدد ۱۳را درون z قرار می‌دهد.

نهایتا هم دستور

```
++z;
```

باعث افزایش یک‌واحدی مقدار z می‌شود. لذا خروجی کد فوق به صورت زیر است:



عملگر + این توانایی را نیز دارد که رشته‌ها را به یک‌دیگر متصل نماید. برای نمونه خروجی کد زیر:

```
var s1="Java" , s2="Script";
document.write(s1+s2);
```

عبارت JavaScript خواهد بود.

در فصول بعد هنگام بررسی اشیاء پیش‌ساخته جاوا اسکریپت، این کار را از طریق متدهای شیء String انجام خواهیم داد.

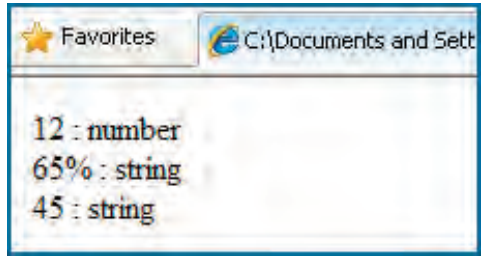
مثال example

نتیجه اجرای کد زیر چیست؟

```
var x;
x=6+6;
document.write(x + ": " + typeof(x) + "<br/>");
x="6"+"5"+"%";
document.write(x + ": " + typeof(x) + "<br/>");
x=4+"5";
```

```
document.write(x + ": " + typeof(x) + "<br/>");
```

خروجی زیر روی صفحه وب ظاهر می‌شود.



نکته قابل توجه در کد این است که وقتی یک عدد با یک رشته جمع می‌شود، خروجی از نوع رشته خواهد بود.

۲-۸-۱۹ عمل‌گرهای انتساب

پیش از این با عملگر = که برای انتساب یک مقدار به یک متغیر مورد استفاده قرار می‌گیرد آشنا شدید. در جدول زیر با روش‌های دیگری از کاربرد این عملگر که برای خلاصه‌نویسی دستورات مورد استفاده قرار می‌گیرد آشنا می‌شوید.

مقدار اولیه متغیر X در همه ردیف‌ها برابر با عدد ۶ و مقدار Y برابر با ۱۲ است.

عملگر	مثال	دستور معادل	نتیجه
=	Y=X		Y=6
=+	Y+=X	Y=Y+X	Y=18
=-	Y-=X	Y=Y-X	Y=6
=*	Y*=X	Y=Y*X	Y=72
=/	Y/=X	Y=Y/X	Y=2
=%	Y%=X	Y=Y%X	Y=0

کد زیر چه نتیجه‌ای را برمی‌گرداند؟

```
<script type="text/javascript">
var x=4,y=5, z=6;
z%=x;
y+=z++;
document.write("Y= " + y + "<br/>");
document.write("Z= " + z + "<br/>");
</script>
```

دستور z%=x به صورت z=z%x تفسیر شده و عدد ۲ به عنوان باقی‌مانده تقسیم z بر x درون z قرار می‌گیرد. حال عبارت y+=z++ به صورت y=y+z++ در می‌آید یعنی ابتدا z با y جمع شده و درون y قرار می‌گیرد و سپس z یک واحد اضافه می‌شود. بنابراین خروجی به صورت زیر است:

```
Y = 7
Z = 3
```

اگر در مثال قبل دستور y+=z++ به صورت y+=++z نوشته شود چه تغییری در نتایج ایجاد خواهد شد؟

این‌بار نتایج زیر برگردانده می‌شود چون z قبل از جمع شدن با y یک‌واحد اضافه شده است

```
Y=8
Z = 3
```

خروجی کد زیر چیست ؟

```
var x=4,y=5, z=6;
x=y=z;
document.write("X= " + x + " , ");
document.write("Y= " + y + " , ");
document.write("Z= " + z );
```

مقدار z در y و مقدار y در x قرار می‌گیرد بنابراین عبارت زیر روی صفحه وب ظاهر خواهد شد.

X=6 , Y=6 , Z=6

۳-۸-۱۹ عمل‌گرهای مقایسه‌ای

عملگرهای مقایسه‌ای، دو عملوند را با هم مقایسه نموده و یک خروجی منطقی (true یا false) برمی‌گردانند. جدول صفحه بعد، عملگرهای منطقی را در زبان جاوا اسکریپت همراه با کاربرد آن‌ها نشان می‌دهد.

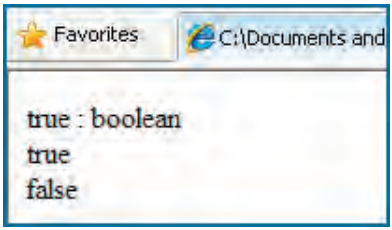
مقدار اولیه متغیر X در همه ردیف‌ها برابر با عدد ۶ است.

عملگر	توضیح	مثال	نتیجه
==	معادل است با	X==5	false
		X=="6"	true
===	از نظر نوع و مقدار معادل است با	X===6	true
		X==="6"	false
!=	معادل نیست با	X!=5	true
>	بزرگ‌تر است از	X>3	true
<	کوچک‌تر است از	X<3	false
>=	بزرگ‌تر یا مساوی است	X>=6	true
<=	کوچک‌تر یا مساوی است	X<=5	false

کد زیر چه مقادیری را برمی‌گرداند؟

```
<script type="text/javascript">
var x=10;
var result = (x*2) >= 20;
document.write(result + ": " + typeof(result) + "<br/>");
document.write(x==10);
document.write("<br/>");
document.write(x==="10");
</script>
```

نتایج زیر هنگام نمایش صفحه وب ظاهر می‌شود:



از آن‌جا که عبارت 20=<(x*2) صحیح است، مقدار منطقی true در متغیر result قرار می‌گیرد. همچنین به دلیل صحیح بودن عبارت x==10 مقدار true برای درج توسط متد write ارسال می‌گردد. علاوه بر این چون عدد ۱۰ توسط عملگر === با رشته ۱۰ مقایسه شده، نتیجه نادرست است. بدیهی است اگر عبارت"10"===x به عبارت x===10 تبدیل شود، نتیجه این عبارت هم صحیح خواهد بود.

۴-۸-۱۹ عمل‌گرهای منطقی

این نوع عملگرها که معمولاً برای بررسی نتیجه نهایی چند مقایسه مورد استفاده قرار می‌گیرند، یک خروجی منطقی (true یا false) برمی‌گردانند. در جدول زیر روش به‌کارگیری آن‌ها توضیح داده شده است.

مقدار اولیه متغیر X در همه ردیف‌ها برابر با عدد ۶ و مقدار Y برابر با ۱۰ است.

عملگر	کارکرد	مثال	نتیجه
&&	«و» منطقی	X<5 && Y>9	true
||	«یا» منطقی	X>=7 || Y<=9	false
!	«نه» منطقی	(x==y)!	true

خروجی کد زیر چیست؟

```
<script type="text/javascript">
```

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

خروجی به صورت زیر است

```
true
false
```

چون عبارت x>y نادرست است و «یا»ی منطقی آن با عبارت نادرست (y*2)<10 نادرست خواهد بود. همچنین عبارت !z درست و طبیعتاً «و» منطقی آن با عبارت درست y>x، درست است.

کد زیر چه نتیجه‌ای را برمی‌گرداند؟

```
<script type="text/javascript">
var a,b,c;
a=(15%7==1);
b=(14!=2*7);
c= !((a && b) || a)
document.write(c==true);
</script>
```

بررسی کد:

از آن‌جا که باقی‌مانده ۱۵ بر ۷ برابر یک است بنابراین: a=ture

عبارت 7*2=!14هم نادرست است لذا: b=false

a && b نادرست است اما «یا»ی منطقی آن با a صحیح می‌شود. اما چون قبل از پرانتز عمل‌گر ! قرار گرفته بنابراین: c=false

در پایان هم، متد write نتیجه مقایسه c را با true بر می‌گرداند که طبیعتاً false خواهد بود.

۱۹-۹ تعریف و فراخوانی تابع

تا این‌جای کتاب با روش نوشتن کدهای جاوا اسکریپت و اجرای آن‌ها در حین بارگذاری صفحه آشنا شدید. حال اگر بخواهیم قطعه کدی، هم‌زمان با باز شدن صفحه به اجرا در نیاید و اجرای آن مشروط به وقوع یک رویداد مانند کلیک شدن دکمه باشد، آیا راه‌حلی وجود دارد؟

خوشبختانه پاسخ مثبت است. شما می‌توانید کد موردنظر را درون یک تابع بنویسید و فراخوانی تابع را منوط به وقوع یک رویداد کنید. این کار علاوه بر این‌که اجرای عملیات‌های متنوع و پیچیده را امکان‌پذیر می‌سازد، یک مزیت مهم دیگر هم دارد. عملیات‌ها، درون توابع پیاده‌سازی می‌شوند و در زمان موردنظر فراخوانی و اجرا می‌گردند؛ استفاده از این روش، کدهای نوشته شده را منظم‌تر خواهد کرد و ردگیری خطاهای احتمالی به سادگی انجام خواهد شد. علاوه بر این اگر لازم باشد عملیاتی درون یک صفحه چندین بار اجرا شود، لازم نیست کدها چندین بار تکرار شوند؛ بلکه صرفاً عملیات فراخوانی انجام خواهد گرفت.

تعریف تابع همانند سایر کدهای جاوا اسکریپت می‌تواند در بخش سرصفحه، بدنه صفحه وب و یا درون فایل خارجی با پسوند js صورت گیرد. برای ایجاد یک تابع باید مراحل زیر را انجام دهید:

- کلمه کلیدی function را بنویسید. توجه داشته باشید که این عبارت حتماً باید با حروف کوچک انگلیسی نوشته شود.
- نام منحصربه‌فردی را برای تابع تایپ کنید. سعی کنید نام تابع با کارکرد آن تناسب داشته باشد تا در بررسی کد دچار سردرگمی نشوید.
- پارامترهایی که تابع دریافت می‌کند را مشخص نمایید.
- بدنه تابع را درون علامت‌های { } بنویسید.
- مقداری را که باید توسط تابع برگردانده شود (در صورت وجود) تعیین نمایید.

می‌خواهیم تابعی بنویسیم که طول و عرض یک مستطیل را دریافت کند و مساحت آن را برگرداند. کد زیر را درون ویرایش‌گر متنی وارد کنید.

```
<html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<script type="text/javascript">
function CalculateArea(width,height)
{
        var s= width*height;
        return (s);
}
</script>
```

تعریف تابع

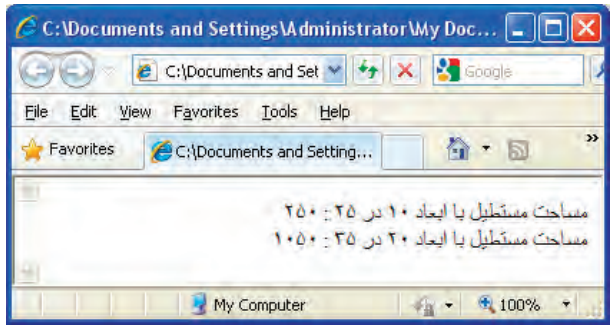
```
</head>
<body dir="rtl">
<script type="text/javascript">

document.write("مساحت مستطیل با ابعاد ۱۰ در ۲۵: " + CalculateArea(10,25));
document.write("<br/>");
document.write("مساحت مستطیل با ابعاد ۲۰ در ۳۵: " + CalculateArea(30,35));
</script>
</body>
</html>
```

فراخوانی تابع

فراخوانی تابع

هنگام باز کردن صفحه درون مرورگر، عبارت زیر نوشته می‌شود.



مثال Example

در مثال قبل، هنوز هم فراخوانی تابع، هم‌زمان با بارگذاری صفحه انجام می‌شود. قصد داریم با تغییراتی در کد، این کار را مشروط به کلیک شدن یک دکمه کنیم. کد را به صورت زیر تغییر دهید:

```
<html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<script type="text/javascript">
function CalculateArea(width,height)
{
    var s= width*height;
    return (s);
}
function WriteResults()
{
```

```
        document.write("مساحت مستطیل با ابعاد ۱۰ در ۲۵: " + CalculateArea(10,25));
        document.write("<br/>");
        document.write("مساحت مستطیل با ابعاد ۲۰ در ۳۵: " + CalculateArea(30,35));
}
</script>
</head>
<body dir="rtl">
<input type="button" value="کلیک کنید" onClick="WriteResults()" />
</body>
</html>
```

در ایـــن صفحـــه، تابع دیگری با نام WriteResults() تعریف گردیده و وظیفه نوشـــتن عباراتی که در مثال قبل مشـــاهده کردید به آن واگذار شده است. همان‌طور که مشـــاهده می‌کنید، تابع می‌تواند فاقد پارامتر ورودی باشد و ضمناً مقداری را برنگرداند؛ این توابع عمدتاً برای انجام کاری خاص مثل نوشتن یک متن مورد استفاده قرار می‌گیرند.

حال به بدنه صفحه دقت کنید. یک کنترل از نوع دکمه روی صفحه قرار گرفته و مشخصه onClick آن با نام تابع مقداردهی شده است. منظور از کد "onClick="functionName()" این است که وقتی رویداد onClick دکـمه به وقوع پیـوســت (یعنی دکـمه کلیک شـــد)، تابـعی که نام آن مشـخص شـــده فراخـوانی شود. تـابع WriteResults برای محاسبه مساحت‌ها، دوبار تابع CalculateArea را فراخوانی می‌کند؛ بنابراین فراخوانی تابع از درون تابع دیگر امکان‌پذیر است.

این‌بار پس از نمایش صفحه وب، شما صرفاً یک دکمه را خواهید دید و برای فراخوانی تابع و مشاهده نتایج باید روی این دکمه کلیک نمایید.

مثال Example

صفحه‌ای ایجاد کنید تا کاربر با کلیک روی یک دکمه، پیغام مناسبی را دریافت کند و پنجره بسته شود.

```
<html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<script type="text/javascript">
function CloseWindow()
{
    window.alert("منتظر دیدار مجدد شما هستیم");
    window.close();
}
</script>
</head>
<body dir="rtl">
<input type="button" value="بستن" onClick="CloseWindow()" />
</body>
</html>
```

در این مثال، رویداد OnClick دکمه با نام تابع CloseWindow مقداردهی شده است و وقتی این تابع فراخوانی شود، کاربر پیغامی را مشاهده خواهد کرد و پس از کلیک روی دکمه OK، پنجره مرورگر بسته خواهد شد.

نکته قابل توجهی که در این مثال‌ها با آن آشنا شدید، مفهوم رویداد بود. هر یک از کنترل‌هایی که بر روی صفحه وب قرار می‌گیرند دارای رویدادهای مخصوص به خود هستند. خوشبختانه این رویدادها توسط جاوا اسکریپت شناسایی می‌شوند و با استفاده از کدهای جاوا اسکریپت می‌توانید پاسخ‌های مناسبی را برای وقوع این رویدادها ایجاد نمایید.

برای نمونه می‌توانید تابعی را به رویداد onMouseOver یک پاراگراف نسبت دهید تا وقتی کاربر، اشاره‌گر ماوس را روی نوشته‌های پاراگراف برد، عملیاتی مثل تغییر رنگ یا جابه‌جایی نوشته‌ها انجام شود. به این ترتیب می‌توان صفحاتی را ایجاد کرد که با کاربر تعامل برقرار نموده و صفحات وب را از محیطی ساده و یکنواخت به محیطی پویا تبدیل نمایند.

فعلاً برای پیگیری مثال‌های خود به رویداد onClick دکمه‌ها بسنده می‌کنیم اما در فصل‌های بعدی کتاب، رویدادهای کنترل‌های صفحات وب را به صورت دقیق‌تر و با ذکر مثال بررسی خواهیم نمود تا توانایی شما برای کدنویسی جاوا اسکریپت به حد مطلوب برسد.

چکیده‌ی فصل

- متغیر، مخزنی برای ذخیره داده‌ها در طول اجرای برنامه است.
- برای نام‌گذاری متغیرها نباید از کلمات کلیدی یا رزرو شده استفاده نمود.
- برای مقداردهی به متغیرها از عملگر = استفاده می‌شود.
- انواع داده‌ای در جاوا اسکریپت به دو دسته اولیه و ارجاعی تقسیم‌بندی می‌شوند.
- برای تبدیل رشته به عدد صحیح می‌توان از متد ()parseInt و برای تبدیل عدد به رشته از متد ()toString. استفاده نمود.
- برای انجام عملیات‌های موردنظر می‌توان از عملگرهای ریاضی، انتساب، مقایسه‌ای و منطقی استفاده نمود.
- تعریف تابع در برنامه باعث صرفه‌جویی در زمان کدنویسی و نیز تسهیل اشکال‌زدایی برنامه می‌شود.

پرسش‌ها و تمرین‌ها

۱. آیا استفاده از کلیدواژه var برای تعریف متغیرها اجباری است؟

۲. کدام‌یک از موارد زیر برای نام‌گذاری متغیرهای جاوا اسکریپت معتبر نیست؟

- _12a
- var
- 2abc
- &sss
- num
- return

۳ .تفاوت متغیرهای محلی و سراسری را توضیح دهید.

۴. انواع داده‌ای اولیه را با ذکر مثال شرح دهید.

۵. تابع typeof چه کاربردی دارد؟

۶. کد زیر چه مقداری را برمی‌گرداند؟ چرا؟

```
<script type="text/javascript">
var b =new Boolean(100);
document.write(b.toString());
</script>
```

۷. کدی بنویسید که سال تولد کاربر را دریافت نموده و سن وی را نمایش دهد.

۸. کد زیر چه مقادیری را برمی‌گرداند؟

```
<script type="text/javascript">
var x=15,y=2, z=6;
y=x--;
z=++x;
++z;
document.write("X= " + x + "<br/>");
document.write("Y= " + y + "<br/>");
document.write("Z= " + z + "<br/>");
</script>
```

۹. خروجی کد زیر چیست؟

```
var x=15,y=11,z=true;
document.write((x<y) || (y*3)>38);
document.write("<br/>");
document.write(!(z && y>x));
```

۱۰. صفحه‌ای عبارت «خروج» ایجاد کنید تا وقتی کاربر اشاره‌گر ماوس را روی آن می‌برد، پنجره مرورگر بسته شود.

فصل بیستم

کنترل
روند اجرای برنامه

هدف‌های رفتاری

پس از مطالعه این فصل از فراگیر انتظار می‌رود:

۱. روش کار با عبارت‌های شرطی را فرابگیرد.

۲. با نحوه کار حلقه‌های تکرار آشنا شود.

۳. با استفاده از عبارت‌های شرطی و حلقه‌های تکرار، روند اجرای برنامه را کنترل نماید.

فصل

20

کنترل روند اجرای برنامه

کلیات

در همه زبان‌های برنامه‌نویسی و اسکریپت‌نویسی، روش‌هایی برای کنترل روند برنامه پیش‌بینی شده تا کدنویس بتواند عملیات خاصی را به تعداد دفعات مشخصی انجام دهد یا ادامه یک عملیات را منوط به تحقق یک شرط نماید. به همین دلیل یادگیری روش استفاده از «عبارت‌های شرطی» و نیز «حلقه‌های تکرار»، نقش تعیین‌کننده‌ای در افزایش مهارت و توانایی برنامه‌نویس خواهد داشت. در این فصل این دو مفهوم را با ذکر مثال‌های متعدد و گویا مرور خواهیم کرد.

۲۰-۱ عبارت‌های شرطی

در هنگام نگارش کد با موقعیت‌هایی مواجه خواهید شد که باید بسته به درست یا نادرست بودن یک یا چند شرط، عملیات خاصی را انجام دهید. فرض کنید سن کاربر را از وی دریافت کرده‌اید و می‌خواهید تعیین کنید که این کاربر در کدام‌یک از رده‌های سنی نوجوان، جوان، میان‌سال و سالمند قرار می‌گیرد. در این موقعیت باید مقدار ورودی را با بازه‌های سنی مقایسه نموده و در صورت تطبیق با هر یک از شروط، مقدار خاصی را برگردانید.

برای مدیریت حالت‌های شرطی در برنامه از سه دستور زیر استفاده می‌شود:

- if
- if…else
- switch

۱-۱-۲۰ دستور if

در شرایطی که لازم است یک شرط بررسی شود و در صورت درستی شرط، قطعه کدی اجرا گردد از دستور if استفاده می‌شود. نگارش کلی این دستور به صورت زیر است:

```
if (condition)
{
    execute code if condition is true
}
```

کدی بنویسید که مقدار یک متغیر عددی را خوانده و اگر کوچک‌تر از ۱۰ است پیغام مناسبی را روی صفحه نمایش دهد.

```
var n=9;
if (n<10)
{
document.write( n + " از ۱۰ کوچکتر است ");
}
```

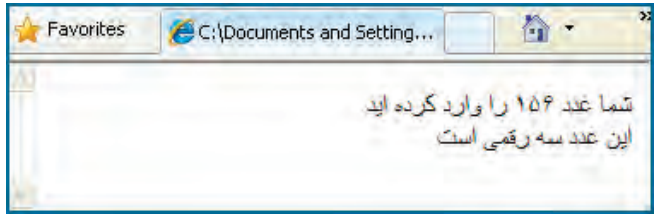
اگر متغیر x را با عدد ۲۰ مقداردهی کنیم چه نتیجه‌ای چاپ می‌شود؟

هیچ عبارتی روی صفحه نمایش داده نمی‌شود چون شرط نادرست است و عبارت درون علامت‌های { } به اجرا در نمی‌آید.

مثال Example

کدی بنویسید که یک عدد را از کاربر دریافت نموده و اگر عدد سه رقمی است، پیغام مناسبی را روی صفحه بنویسد.

```
<script type="text/javascript">
var n=window.prompt("عدد موردنظر را وارد کنید, ۰" );
if (n>=100 && n<=999)
{
document.write("شما عدد " + n + " را وارد کرده اید " + "<br/>");
document.write("این عدد سه رقمی است");
}
</script>
```



در این کد با استفاده از عمگر منطقی &&، درستی هر دو شرط بررسی شده است.

مثال Example

اگر در مثال قبلی علامت‌های { } را برداریم چه اتفاقی می‌افتد؟

```
1  if (n>=100 && n<=999)
2  document.write("شما عدد " + n + " را وارد کرده اید " + "<br/>");
3  document.write("این عدد سه رقمی است");
```

همان‌طور که قبلاً اشاره شد، اگر شرط دستور if صحیح باشد، تمامی دستورات موجود در بین علامت‌های { } به اجرا درخواهد آمد اما اگر سهواً یا عمداً این علامت‌ها را وارد نکنید، فقط اولین دستورِ بعد از if به عنوان

کدی که باید اجرا شود تلقی خواهد شد و سایر کدها جزو بدنه برنامه در نظر گرفته می‌شوند. بنابراین در صورت صحیح بودن شـرط، خط ۲ اجرا می‌گردد و خط ۳ هم (که حالا جزو دسـتور if نیسـت) در هر صورت به اجرا درخواهد آمد. بنابراین حتی اگر عدد دو رقمی هم وارد کنید، عبارت «این عدد سـه رقمی اسـت» روی صفحه نوشته می‌شود و لذا منطق برنامه نادرست خواهد بود.

کدی بنویسید که یک عدد را از ورودی دریافت کند و زوج یا فرد بودن آن را مشخص نماید.

```
<body dir="rtl">
<script type="text/javascript">

var n=window.prompt("عدد موردنظر را وارد کنید","۰" );
if (n%2==0)

document.write("عدد وارد شده زوج است");
if (n%2!=0)

document.write("عدد وارد شده فرد است");
</script>
</body>
```

بررسی کد:

- عدد وارد شده توسط کاربر درون متغیر n ذخیره می‌شود.
- دستور if اول، باقی‌مانده این عدد را محاسبه و با صفر مقایسه می‌کند، اگر برابر با صفر بود، اعلام می‌کند که عدد وارد شده زوج است.
- دستور if دوم هم در صورتی که باقی‌مانده عدد بر دو مخالف صفر باشد، فرد بودن آن را اعلام خواهد کرد.
- توجه داشته باشید که در هر یک از دستورات if فقط یک سطر کد وجود دارد بنابراین قرار دادن علامت { }، تأثیری در نتیجه نهایی نخواهد داشت.

۲-۱-۲۰ دستور if…else

دستور if…else شکل کامل‌تر دستور if است و با استفاده از آن می‌توان در صورت نادرست بودن شرط هم مجموعه‌ای از کدها را اجرا نمود. نگارش کلی این دستور به صورت زیر است:

```
if (condition)
{
```

```
    execute code if condition is true
}
else
{
    execute code if condition is false
}
```

کدی بنویسید که یک عدد را از ورودی دریافت کند و زوج یا فرد بودن آن را مشخص نماید

```
<script type="text/javascript">
var n=window.prompt("عدد موردنظر را وارد کنید","۰" );
if (n%2==0)
document.write("عدد وارد شده زوج است");
else
document.write("عدد وارد شده فرد است");
</script>
```

کدی بنویسید که نام کاربر را بخواند، در صورتی که نام وارد شده بین ۳ تا ۱۵ نویسه است، آن را بنویسد، در غیراین‌صورت پیغامی مبنی بر نامعتبر بودن نام وارد شده ظاهر نماید.

```
var s=window.prompt("نام خود را وارد کنید، طول نام باید بین ۳ تا ۱۵ نویسه باشد");
if (s.length>=3 && s.length<=15)
{
document.write("نام وارد شده: " + s);
}
else
{
window.alert("نام وارد شده نامعتبر است");
}
```

مثال Example

صفحـــه‌ای حاوی یک دکمه ایجاد کنید تا کاربر با کلیک روی آن، پنجـــره‌ای حاوی دکمه‌های OK و Cancel مشاهده نماید. سپس در صورت کلیک روی هر یک از این دو دکمه، پیغامی مبنی بر کلیک شدن دکمه ظاهر گردد.

```
<html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<script type="text/javascript">
function show_confirm()
{
var result=window.confirm("یکی از دکمه ها را فشار دهید");
if (result==true)
  {
  window.alert("OK را کلیک کردید");
  }
else
  {
  window.alert("Cancel را کلیک کردید");
  }
}
</script>
</head>
<body>
<input type="button" onclick="show_confirm()" value="نمایش پنجره تأیید" />
</body>
</html>
```

بررسی کد:

- در این کد تابعی به نام show_confirm() ایجاد شـــده که پس از فراخوانی با اســـتفاده از متد window.confirm() پنجره تأیید را به همراه یک پیغام نشان می‌دهد.
- در ایـــن نـــوع پنجره، چنان‌چه کاربر روی دکمه OK کلیک کند، مقدار منطقی true و در صورت کلیک روی دکمه Cancel، مقدار false برگردانده می‌شود.
- این مقدار در متغیر result ذخیره و با اســـتفاده از دســـتور if…else ارزیابی می‌گردد تا پیغام مناســـب

برای کاربر نمایش داده شود.

- درون بدنه صفحه وب هم دکمه‌ای قرار داده شـده و مشـخصه onClick آن با نام تابع مقداردهی شده است تا با کلیک کاربر روی دکمه، فراخوانی تابع انجام شود.

دسـتور if...else را می‌توان به صورت زنجیره‌ای از شـرط‌ها نیز مورد استفاده قرار داد تا بتوان حالت‌های مختلف را مورد بررسی قرار داد و متناسب با هر حالت، عملیات موردنظر را انجام دارد. نگارش حالت زنجیره‌ای به صورت زیر است:

```
if (condition_1)
{
    execute code if condition_1 is true
}
else if (condition_2)
{
    execute code if condition_2 is true
}
else
{
    execute code if condition_1 and condition_2 are false
}
```

کدی بنویسید که سن کاربر را دریافت نموده و بسته به حالت‌های زیر، پیغام مناسب را چاپ کند.

سن	پیغام
کمتر از ۱۰ سال	خوش آمدید
بین ۱۰ و ۱۸	از بخش بازی‌های وب‌سایت دیدن نمایید
بین ۱۸ تا ۳۰	از فروشگاه آن‌لاین ما بازدید فرمایید
بیش‌تر از ۳۰	خوش آمدید

```
<script type=”text/javascript”>
```

```
var n=window.prompt("کاربر عزیز، سن خود را وار نمایید");
var msg="";
if (n>=10 && n<=18)
{
    msg="از بخش بازی های وب سایت دیدن نمایید";
}
else if (n>18 && n<30)
{
    msg="از فروشگاه آنلاین ما بازدید فرمایید";
}
else
{
    msg="خوش آمدید";
}
window.alert(msg);
</script>
```

در این کد برای جلوگیری از تکرار بی‌مورد متد alert، پیغام مناسب درون متغیر msg ذخیره شده و پس از بررسی شرط‌ها و مقداردهی با پیغام مناسب، متغیر برای متد ارسال شده است.

۳-۱-۲۰ دستور switch

وقتی تعداد شرط‌هایی که می‌خواهیم متناظر با آن‌ها عملیات خاصی را انجام دهیم زیاد می‌شود بهتر است به جای استفاده از if های متعدد، از دستور switch استفاده نماییم. نگارش کلی دستور switch به صورت زیر است.

```
switch (expression)
{
case value_1:
execute code if expression is equal to value_1
break;
case value_2:
execute code if expression is equal to value_2
break;
case value_3:
execute code if expression is equal to value_3
break;
```

```
default:
execute code if expression is not equal to value_1, value_2 and value_3
}
```

برای اســتفاده از این دســتور، ابتدا مقداری که باید مورد مقایسه قرار گیرد به جای expression قرار داده می‌شــود. ســپس برای هر مقایسه، کلمه کلیدی case و سپس مقدار موردنظر درج می‌شود. پس از علامت : نیز مجموعه دســتوراتی که باید در صورت معادل بودن expression و مقدار اجرا گردند قرار می‌گیرند و دســتور break هم اعلام می‌کند که در صورت صحیح بودن شــرط، نیازی به بررســی سایر موارد نیست. نهایتاً چنان‌چه نتیجه هیچ‌یک از مقایسه‌ها صحیح نباشد، کدی که پس از default قرار گرفته به اجرا در خواهد آمد.

اســکریپتی بنویســید که یک پیش‌شماره را از کاربر دریافت نموده و مشخص نماید این پیش‌شماره مربوط به کدام‌یک از شهرهای کشور است.

```
<body dir="rtl">
<script type="text/javascript">
var code=window.prompt("کد شهر موردنظر را وارد کنید");
var city;
switch (code)
{
    case "021":
        city="تهران";
        break;
    case "0311":
        city="اصفهان";
        break;
    case "0411":
        city="تبریز";
        break;
    case "0511":
        city="مشهد";
        break;
    defult:
        city="سایر شهرها";
```

```
}
document.write("کد وارد شده متعلق است به " + city);
</script>
</body>
```

۲۰-۲ حلقه‌های تکرار

حلقه‌های تکرار در زبان‌های برنامه‌نویسی اهمیت فوق‌العاده‌ای دارند چون اغلب اوقات لازم است عملیاتی در برنامه به تعداد دفعات مشخصی تکرار شود یا تکرار آن‌قدر ادامه یابد تا شرط خاصی محقق گردد. در زبان جاوا اسکریپت برای ایجاد حلقه‌های تکرار از دستورات while، do…while و for استفاده می‌شود که هر یک کاربرد خاصی دارند.

۲۰-۲-۱ دستور while

شکل کلی نگارش این دستور به صورت زیر است:

```
while (condition)
{
    execute code while condition is true
}
```

دستور while یک شرط را دریافت می‌کند و تا زمانی که این شرط درست است، کدهای بین علامت { } را اجرا خواهد کرد. در صورت نادرست شدن شرط، کنترل برنامه به خارج از حلقه منتقل می‌شود.

کدی بنویسید که عبارت «خوش آمدید» را پنج بار روی صفحه بنویسد.

```
<body>
<script type="text/javascript">
var i=0;
while (i<5)
{
    document.write("خوش آمدید <br/>");
    ++i;
}
</script>
</body>
```

بررسی کد:

- برای ایجاد حلقه‌های تکرار، ابتدا باید یک شمارنده (مانند i) تعریف نمایید. سپس شرط اتمام حلقه while را عبور مقدار i از سقف مجاز قرار دهید. درون بدنه حلقه هم دستورات موردنظر را نوشته و ضمناً فراموش نکنید که مقدار شمارنده را تغییر دهید.
- در این کد، ابتدا شمارنده i با عدد صفر مقداردهی می‌شود.
- از آن‌جا که شرط حلقه یعنی (i<5) صحیح است، یک‌بار عبارت «خوش آمدید» روی صفحه نوشته خواهد شد و ضمناً مقدار i یک‌واحد افزایش می‌یابد.
- مجدداً اجرای برنامه به ابتدای حلقه برمی‌گردد. هنوز هم شرط صحیح است و بنابراین دستورات اجرا می‌شود. این‌کار تا جایی ادامه می‌یابد که مقدار i به عدد ۵ می‌رسد و این‌بار شرط 5<5 نادرست است؛ لذا اجرای حلقه به پایان می‌رسد.

کدی بنویسید که ۵ کادر متنی (TextBox) روی صفحه قرار داده و شناسه آن‌ها را با مقادیر txt1، txt2 و ... نام‌گذاری نماید.

```
<script type="text/javascript">
var i=1;
while (i<6)
{
    document.write("<input type=\"text\" id=txt" + i + "/><br/>");
    ++i;
}
</script>
```

بررسی کد:

- در این کد شمارنده i با عدد یک مقداردهی شده است و با هر بار اجرای حلقه، عبارت سازنده کادر متنی روی صفحه تولید می‌شود و شناسه آن با عبارت txt بعلاوه مقدار شمارنده مقداردهی می‌شود.
- توجه داشته باشید که برای ایجاد یک کادر متنی باید کد

```
<input type="text" id="…" />
```

وارد صفحه وب شود. اما اگر در ورودی متد write علامت " را قرار دهیم به معنی ابتدا یا انتهای رشته ورودی خواهد بود. برای جلوگیری از این مشکل و نادیده گرفته شدن این علامت به عنوان ابتدا و انتهای رشته، باید قبل از آن‌ها نویسه \ را قرار دهیم. به این ترتیب علامت‌های " به عنوان بخشی از رشته تلقی خواهند شد.

کد مثال قبل را به گونه‌ای تغییر دهید که این کار با کلیک شدن یک دکمه و از طریق تابع انجام شود.

```
<html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<script type="text/javascript">
function CreateTextBox(n)
{
var i=1;
while (i<=n)
{
      document.write("<input type=\"text\" id=txt" + i + "/><br/>");
      ++i;
}
}
</script>
</head>
<body>
<input type="button" value="کلیک کنید" onClick="CreateTextBox(5)" />
</body>
</html>
```

بررسی کد:

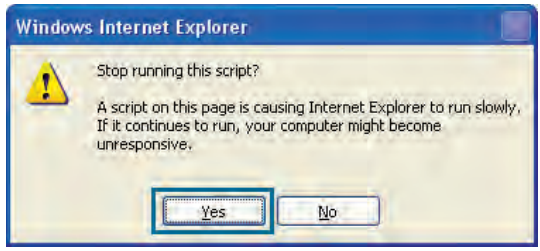
- در ایــن مثــال تابعی با نام CreateTextBox ایجاد کرده‌ایم که یک عدد دریافت می‌کند و به تعداد آن، کادر متنی روی صفحه قرار می‌دهد.
- سپس رویداد onClick دکمه‌ای که روی صفحه ایجاد کرده‌ایم با این تابع مقداردهی شده و ضمناً عدد ۵ برای تابع ارسال می‌شود.

اجرای کد زیر چه نتیجه‌ای دربر دارد؟

```
<script type="text/javascript">
```

```
var i=1;
while (2<3)
{
      document.write( i +"<br/>");
      ++i;
}
</script>
```

از آن‌جا که شـرط while همواره صحیح است، اجرای دستورات حلقه هیچ‌گاه متوقف نمی‌شود و اصطلاحاً یک «حلقه بی‌پایان» ایجاد می‌شـود. در این حالت مرورگر پیغامی مبنی بر تقاضای توقف اجرای اسـکریپت را مطـرح خواهد کرد که با کلیک روی دکمه Yes می‌توانید به ایـن حلقه بی‌انتها پایان دهید. کلیک روی دکمه No باعث می‌شود اعداد صحیح بزرگ‌تر از یک روی صفحه وب نوشته شوند.



۲۰-۲-۲ دستور do…while

کارکرد این دسـتور شـباهت زیادی به دسـتور while دارد، با این تفاوت که بررسی شرط در انتهای حلقه انجام می‌گیرد و در نتیجه، کدِ درون حلقه حداقل یک‌بار به اجرا در می‌آید. نگارش کلی این دسـتور به صورت زیر است:

```
do
{
execute code
}
while (condition)
```

مثال ۱: نتیجه اجرای کد زیر چیست؟

```
<script type="text/javascript">
var i=0;
while (i<0)
```

```
{
    document.write( i +"<br/>");
    ++i;
}
do
{
    document.write( i +"<br/>");
    ++i;
}
while (i<0)
</script>
```

حلقه while اجرا نمی‌شـود چون شـرط i<0 نادرست است و لذا مقدار i هم تغییر نمی‌کند. اما حلقه …do while یک‌بار اجرا می‌شـود و عدد صفر روی صفحه نوشـته می‌شود اما با افزایش یک‌واحدی i، دیگر شرط i<0 صحیح نیست و بنابراین، اجرای اسکریپت به انتها می‌رسد.

کدی بنویسید که یک عدد غیرصفر (n) و یک رشته (s) را از کاربر بگیرد و s را n بار روی صفحه بنویسد.

```
<body>
<script type="text/javascript">
var n= window.prompt("یک عدد بزرگ تر از صفر وارد کنید");
var s= window.prompt("یک عبارت متنی وارد نمایید");
if (n>0)
{
    do
    {
        document.write(s + "<br/>");
        --n;
    }
    while (n>0)
}
else
{
```

```
        window.alert("عدد وارد شده معتبر نیست");
}
</script>
```

بررسی کد:

- عدد n و رشته s از ورودی خوانده می‌شود،
- اگر n بزرگ‌تر از صفر باشد، یک‌بار روی صفحه نوشته می‌شود و سپس از مقدار n یک‌واحد کسر می‌شود و به همین ترتیب اجرای حلقه ادامه می‌یابد.
- چنان‌چه عدد صفر یا کمتر باشد، پیغام خطا برای کاربر نمایش داده می‌شود.

کدی بنویسید که اسامی افراد را به صورت تک‌تک خوانده و نهایتاً همه آن‌ها را روی صفحه نمایش دهد. نشانه اتمام اسامی، ورود رشته "end" است.

```
<script type="text/javascript">
var name="", s="";
name= window.prompt("یک نام را وارد نمایید");
do
{
        s = s + name + "<br/>";
        name= window.prompt("یک نام را وارد نمایید");
}
while (name!="end")
document.write(s);
</script>
```

بررسی کد:

- ابتدا متغیرهای name و s تعریف و با یک رشته خالی ("") مقداردهی می‌شوند چون اگر به آن‌ها مقدار اولیه داده نشود، مقدار پیش‌فرض undefined را می‌پذیرند.
- سپس یک نام از ورودی خوانده شده و حلقه شروع می‌شود.
- از s برای نگه‌داری کلیه اسامی وارد شده استفاده می‌شود و هر بار، نام وارد شده به مقدار فعلی s افزوده می‌شود و مجدداً در s ذخیره می‌گردد.

از آن‌جا که شـرط اتمام حلقه، وارد شـدن رشـته "end" اسـت بنابراین عمل خواندن نام را در انتهای حلقه انجام می‌دهیم تا شـرط بررسـی شـود و در صورت مخالف بودن با مقدار "end" کنترل برنامه به ابتدای حلقه منتقل شود.

۳-۲-۲۰ دستور for

در مثال ۳ دستور do…while مشاهده کردید که می‌توان از این دستور برای عملیات‌های تکراری که تعداد دفعات آن‌ها ممکن است بسته به شرایط تغییر کند استفاده کرد؛ چرا که معلوم نبود کاربر قصد دارد چند نام وارد کند.

در این میان از دستور for برای تکرار یک عملیات به تعداد دفعات مشخص استفاده می‌گردد و نگارشی به صورت زیر دارد:

```
for(a;b;c)
{
    execute code
}
```

- a: در این بخش از دستور، مقداردهی اولیه شمارنده انجام می‌شود.
- b: در این قسـمت، شـرطی قرار داده می‌شود که اگر نتیجه آن درست باشد، حلقه ادامه پیدا می‌کند و در غیراین‌صورت، به اتمام می‌رسد.
- c: هر بار که اجرای دستورات حلقه به پایان می‌رسد، دستور موجود در این بخش نیز به اجرا در می‌آید. عمدتاً دستور تغییر شمارنده در این بخش صورت می‌گیرد.

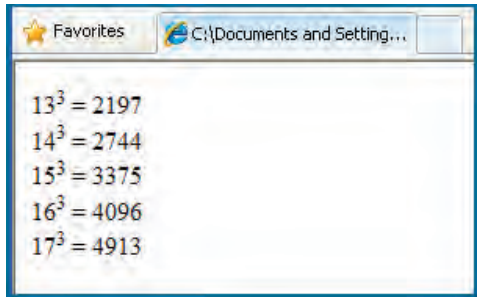
کدی بنویسید که اعداد ۱ تا ۱۰ را روی صفحه بنویسد.

```
<script type="text/javascript">
for (i=1;i<=10;++i)
{
    document.write(i + "<br/>");
}
</script>
```

مثال Example

کدی بنویسید که توان سوم اعداد بین ۱۳ تا ۱۷ را نشان دهد.

```
<script type="text/javascript">
for (i=13;i<=17;++i)
{
      document.write(i +"<sup>3</sup> = " + i*i*i + "<br/>");
}
</script>
```



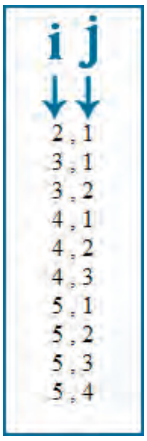
بررسی کد:

در این مثال با اســتفاده از برچسب که برای نمایش توان اعداد استفاده می‌شود، توان سوم شمارنده i (یعنی مقدار i*i*i) نمایش داده شده است.

مثال Example

خروجی کد زیر چیست؟

```
for (i=2;i<=5;++i)
{
      j=1;
      while (j<i)
      {
            document.write(i +" , " + j + "<br/>");
            ++j;
      }
}
```



بررسی کد:

- در این کد، ابتدا متغیر i با عدد ۲ مقداردهی می‌شود.
- سپس کدهای درون حلقه for به اجرا درمی‌آید. لذا j با عدد ۱ مقداردهی شده و به دلیل درست بودن شرط j<i عبارت 2,1 روی صفحه نوشته می‌شود.
- در ادامه مقدار j به ۲ افزایش می‌یابد و از آن‌جا که دیگر شـرط j<i صدق نمی‌کند، حلقه while پایان یافته و کنترل به for منتقل می‌گردد.
- ایـن بـار i یک‌واحد افزایش یافتـه و مراحل قبل تکرار می‌گردد. این روش اسـتفاده از حلقه‌ها، کاربرد حلقه‌های تودرتو نام دارد.

نتیجه اجرای عبارت کد زیر چیسـت؟ اگر علامت‌های { } برداشـته شـوند چه تغییری در نتیجه حاصل می‌شود؟

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

بـا جـرای کد فوق نتیجه زیر در خروجی ظاهر می‌شـود چون متغیر i از ۱ تـا ۳ تغییر می‌کند و در هر بار اجرای حلقه دوبار نوشته می‌شود.

1
1
2
2
3
3

اما وقتی آکولادها برداشته می‌شود و کد به صورت زیر درمی‌آید:

```
for (i=1;i<=3;++i)
document.write(i+ "<br/>");
document.write(i+ "<br/>");
```

صرفاً اولین دسـتور write جزو دستورات حلقه محسوب می‌شود (مانند آن‌چه در مورد if مشاهده کردید) و لذا خروجی به صورت زیر خواهد بود.

1
2
3
4

اعداد ۱ تا ۳ توسـط دسـتور write اول و عدد ۴ توسط دسـتور write دوم نوشته شده است. این قاعده در مورد همه دستوراتی که کدهای آن‌ها درون آکولاد قرار می‌گیرند مانند while هم صادق است.

مـثال Example

با اجرای کد زیر چند بار کلمه JavaScript روی صفحه نوشته می‌شود؟

```
<script type="text/javascript">
for (i=1;i<=4;++i)
for (j=1;j<=3;++j)
document.write("JavaScript <br/>");
</script>
```

۱۲ بار! چون حلقه اول چهار بار و حلقه دوم در هر بار اجرای حلقه اول، سه بار به اجرا در می‌آید. در این مثال هم for دوم تنها دستورِ for محسوب می‌شود و نیازی به گذاشتن آکولاد نیست.

۴-۲-۲۰ دستور break

دستور break برای خروج از حلقه تکرار کاربرد دارد؛ یعنی درون حلقه‌ای مانند while چنان‌چه دستور break اجرا شود، صرف‌نظر از مقادیر و درستی یا نادرستی شروط، کنترل برنامه به خارج از حلقه منتقل می‌شود. کاربرد دیگر این دستور، همان‌گونه که در بخش‌های قبل مشاهده کردید، پایان دادن به بررسی شرط‌ها در دستور switch بود.

خروجی کد زیر چیست؟

```
<script type="text/javascript">
var i=0;
while (true)
{
    document.write(i++ + "<br/>");
    if (i==5)
    break;
}
</script>
```

اعداد صفر تا چهار روی صفحه نوشته می‌شود و وقتی مقدار i برابر با ۵ شد، دستور break اجرا و حلقه به پایان می‌رسد.

کدی بنویسید که اسامی افراد را به صورت تک‌تک خوانده و نهایتاً همه آن‌ها را روی صفحه نمایش دهد. نشانه اتمام اسامی، ورود رشته "end" است.

```
<script type="text/javascript">
var s="" , name="";
while (true)
{
```

```
        name= window.prompt("نام موردنظر را وارد کنید");
        if (name=="end")
        break;
        s= s + name + "<br/>";
    }
    document.write(s);
    </script>
```

کدی بنویسید که تعدادی عدد بزرگ‌تر از صفر را از ورودی خوانده و تعداد اعداد زوج را نمایش دهد. ورود عدد صفر نشان‌دهنده اتمام ورود اعداد است.

```
<script type="text/javascript">
var cnt=0 , n;
while (true)
{
    n= window.prompt("عدد موردنظر را وارد کنید");
    if (n==0)
    break;
    if (n%2==0)
    cnt++;
}
document.write("تعداد اعداد زوج وارد شده: " + cnt);
</script>
```

بررسی کد:

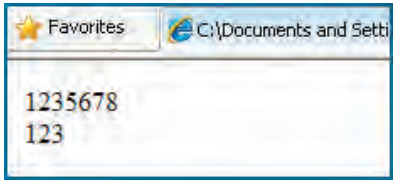
- ابتدا شمارنده cnt با صفر مقداردهی می‌شود.
- شـــرط حلقه while همواره true اســـت بنابراین تکرار حلقه آن‌قدر ادامه می‌یابد تا دستور break; اجرا شود.
- در هر بار اجرای حلقه، عدد n خوانده می‌شود؛ اگر این عدد برابر با صفر بود، حلقه خاتمه می‌یابد.
- اگر n زوج باشـــد، یک‌واحد به مقدار شمارنده افزوده می‌شود. نهایتاً، متد write مقدار شمارنده را روی صفحه درج خواهد کرد.

۵-۲-۲۰ دستور continue

وجود این دستور درون حلقه، باعث می‌شود دستورات پس از آن نادیده گرفته شده و کنترل برنامه به ابتدای حلقه منتقل شود. در این حالت، مانند روند طبیعی اجرای حلقه، شرط مورد بررسی قرار می‌گیرد و در صورت صحیح بودن، اجرای حلقه ادامه پیدا می‌کند؛ در غیراین‌صورت، اجرای حلقه پایان می‌پذیرد.

خروجی کد زیر چیست؟

```
<script type="text/javascript">
for (i=1;i<=8;i++)
{
      if (i==4)
      continue;
      document.write(i);
}
document.write("<br/>");
for (i=1;i<=8;i++)
{
      if (i==4)
      break;
      document.write(i);
}
</script>
```



بررسی کد:

- در حلقه for اول، اعداد ۱ تا ۸ شمارش شده و روی صفحه نوشته می‌شوند، فقط وقتی شمارنده برابر با ۴ می‌شود، دستور continue اجرا شده و در نتیجه کنترل برنامه به ابتدای حلقه منتقل می‌شود، لذا عدد ۴ نوشته نخواهد شد.
- در حلقه for دوم، اعداد ۱ تا ۴ شمارش می‌شوند، اما وقتی مقدار شمارنده برابر با ۴ می‌شود، اجرای حلقه متوقف می‌گردد لذا فقط اعداد ۱تا ۳ روی صفحه درج خواهند شد.

فصل

20

کنترل روند اجرای برنامه

چکیده‌ی فصل

- در زبان‌های برنامه‌نویسی و اسکریپت‌نویسی برای کنترل روند برنامه از «عبارت‌های شرطی» و نیز «حلقه‌های تکرار» استفاده می‌شود.
- برای مدیریت حالت‌های شرطی از دستورات if ، if … else و switch استفاده می‌گردد.
- برای انجام عملیات‌های تکراری در جاوا اسکریپت از حلقه‌های while، do…while و for استفاده می‌شود.
- می توانیم اجرای توابع را مشروط به وقوع رویدادی از عناصر صفحه کنیم؛ برای نمونه رویداد onClick عنصر دکمه.
- دستور break باعث اتمام حلقه می‌شود.
- دستورات بعد از contiune نادیده گرفته می‌شوند و کنترل برنامه به ابتدای حلقه منتقل می‌شود.

پرسش‌ها و تمرین‌ها

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

۲. کدی بنویسید که تعدادی عددِ بزرگ‌تر از صفر را از کاربر دریافت نموده و میانگین آن‌ها را روی صفحه بنویسد. ورود عدد صفر نشانه اتمام اعداد است.

۳. خروجی کد زیر چیست؟

```
<script type="text/javascript">
var i=5;
while (--i>1)
document.write(i + "<br/>");
</script>
```

۴. کد زیر چه نتایجی را برمی‌گرداند؟

```
<script type="text/javascript">
for (i=0;i<=25;i=i+2)
document.write(i + “<br>”);
</script>
```

۵. کدی بنویسید که با استفاده از حلقه‌های تودرتو شکل زیر را روی صفحه نمایش دهد.

```
*
**
***
****
*****
```

فصل بیست و یکم

کار با
اشیاء جاوا اسکریپت

هدف‌های رفتاری

پس از مطالعه این فصل از فراگیر انتظار می‌رود:

۱. با ماهیت اشیاء پیش‌ساخته در جاوا اسکریپت آشنا شود.

۲. روش دستیابی به خصوصیت‌ها و استفاده از متدها را فرا بگیرد.

۳. توانایی کار با اشیاء متداول را کسب نماید.

فصل

21

کار با اشیاء جاوا اسکریپت

کلیات

همان‌گونه که در فصل‌های پیشـین این کتاب توضیح داده شـد، جاوا اسکریپت یک زبان اسکریپت‌نویسی شـیءگرا محسوب می‌شـود و با بهره‌گیری از این خاصیت می‌توانید اشـیاء موردنظر را ایجاد و در طول برنامه از آن‌ها اسـتفاده کنید. شـیئی که در جاوا اسـکریپت می‌سازید در واقع نوعی از داده اسـت که مطابق با نیاز خود ایجاد کرده‌اید و در سـاماندهی اطلاعات و پردازش آن‌ها نقش تعیین‌کننده‌ای دارد. علاوه بر این در جاوا اسکریپت تعدادی شیء پیش‌ساخته[1] وجود دارد که در این فصل با روش استفاده از آن‌ها آشنا خواهید شد.

۲۱-۱ تعریف یک شیء (مطالعه آزاد)

پیش از آن‌که با اشـیاء پیش‌ساخته جاوا اسکریپت آشنا شوید بهتر است در عمل، روش ایجاد و استفاده از یک شیء را فرابگیرید تا چنان‌چه ابهامی پیرامون این مفهوم در ذهن شما وجود دارد برطرف شود.

فرض کنید در حال ایجاد برنامه‌ای برای یک فروشگاه هستیم که با اطلاعات کالاها و مشتریان کار می‌کند، بنابراین می‌توانیم در این برنامه، کلاس (مفهوم) مشتری را ایجاد و از روی آن اشیاء موردنظر را بسازیم. طبیعتاً هـر شـیء تعدادی خصوصیت (مانند نام، نـام خانوادگی و ...) دارد که همان Propertyها هسـتند و نیز تعداد عملکرد یا متد (Method) خواهد داشت.

- برای ایجاد کلاس، از کلمه کلیدی function استفاده می‌شود.
- نام موردنظر برای کلاس را وارد کنید.
- مقادیر لازم برای مقداردهی به این شیء را درون پرانتز وارد نمایید.

1 . Built-in

- کلمه کلیدی this را تایپ و پس از آن یک نقطه درج کنید. حالا خصوصیت موردنظر برای شیء را وارد نمایید. سپس مقدار متناظر با این خصوصیت را که در تعریف کلاس قید شده به آن منتسب کنید. این کار را برای همه خصوصیت‌های شیء تکرار نمایید.
- اکنون نوبت به متدهای شیء می‌رسد. کلمه کلیدی this را وارد و پس از درج نقطه، نام متد را وارد نمایید. حال نام متد را به این مقدار منتسب نمایید.
- حال متدهای تعریف شده برای کلاس را در خارج از آن پیاده‌سازی نمایید.

```
function customer(n,l,a)
{
    //properties
    this.name=n;
    this.lastname=l;
    this.age=a;

    //methods
    this.changeLastname=changeLastname;
    this.increaseAge=increaseAge;
}

function changeLastname(new_l)
{
    this.lastname=new_l;
}

function increaseAge()
{
    this.age= this.age +1;
}
```

در کد فوق، ابتدا کلاس مشتری تعریف شده که حاوی سه خصوصیت نام، نام خانوادگی و سن است. سپس دو متد برای آن پیاده‌سازی گردیده که changeLastname، نام خانوادگی جدیدی را به مشتری نسبت می‌دهد و increaseAge، سن وی را یک‌سال افزایش می‌دهد. اکنون باید از این کلاس تعریف شده، درون برنامه استفاده نماییم.

- برای ایجاد شیء از روی کلاس، کلمه کلید var و نام شیء نوشته می‌شود.
- پس از علامت انتساب، نام کلاس با مقادیر لازم برای مقداردهی به شیء ساخته شده درون پرانتز و به همان ترتیبی که در تعریف کلاس وجود داشت قید می‌گردد.

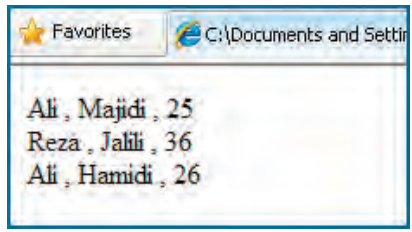
اکنون شیء، ساخته شده و می‌توان از خصوصیات یا متدهای آن استفاده کرد.

```
<body>
<script type=”text/javascript”>
var c1=new customer(“Ali”,”Majidi”,25);
var c2=new customer(“Reza”,”Jalili”,36);
document.write(c1.name + “ , “ + c1.lastname + “ , “ + c1.age );
document.write(“<br/>”);
document.write(c2.name + “ , “ + c2.lastname + “ , “ + c2.age );
c1.changeLastname(“Hamidi”);
c1.increaseAge();
document.write(“<br/>”);
document.write(c1.name + “ , “ + c1.lastname + “ , “ + c1.age );
</script>
</body>
```

در کد فوق، شیء c1 ساخته شده و خصوصیت‌های نام، نام خانوادگی و سن وی با مقادیر Ali، Majidi و 25 مقداردهی شده است. شیء c2 هم از روی کلاس مشتری ایجاد و با مقادیر Reza، Jalili و 36 مقداردهی گردیده است. اکنون می‌توانیم این مقادیر را خوانده و روی صفحه نمایش دهیم.

حال با فراخوانی متدهای تعریف شده برای کلاس مشتری قصد داریم مقادیر ذخیره شده در شیء c1 را تغییر دهیم. متد changeLastname("new name") نام جدید را جایگزین نام قبلی می‌کند و متد increase-Age باعث افزایش یک‌واحدی سن می‌شود. حال وقتی مجدداً خصوصیات شیء c1 را روی صفحه می‌نویسیم، متوجه تغییر مقادیر می‌شویم.

نتیجه اجرا مجموعه کدهای درج شده در این بخش به صورت زیر است:



۲۱-۲ اشیاء پیش‌ساخته جاوا اسکریپت

در جاوا اسکریپت تعدادی شیء پیش‌ساخته وجود دارد که استفاده از آن‌ها باعث تسهیل در انجام عملیات موردنظر می‌شود. این اشیاء دارای تعدادی خصوصیت هستند که اطلاعاتی را در مورد داده‌ها در اختیار شما قرار

می‌دهند و ضمناً با اســتفاده از متدهــای آن‌ها می‌توانید عملیات موردنظر را به ســادگی انجام دهید. برای مثال همان‌گونه در فصول گذشــته مشــاهده کردیده استفاده از خصوصیت length شــیء String، طول رشته را برای شما محاسبه می‌کند. در صورت عدم استفاده از این خصوصیت مجبور هستید متد محاسبه طول رشته را شخصاً بنویسید که طبیعتاً کار زمان‌بری است.

۱-۲-۲۱ شیء String

این شیء برای ذخیره‌سازی عبارت‌های متنی، دست‌کاری آن‌ها و نیز استخراج اطلاعاتی در مورد آن‌ها استفاده می‌شود.

برای تعریف یک شیء String می‌توانید از نگارش زیر استفاده نمایید:

```
var txt = new String("My Text!");
```

یا به شکل ساده‌تری از عبارت زیر را به کار ببرید:

```
var txt = "My Text!";
```

مهم‌ترین خصوصیت این شیء، length است که طول رشته را برمی‌گرداند، بنابراین، نتیجه اجرا دستور زیر، درج عدد ۸ روی صفحه است.

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

پرکاربردترین متدهای این شیء را در جدول زیر مشاهده می‌کنید.

متد	کاربرد
charAt(index)	نویسه موجود در نمایه داده شده را برمی‌گرداند.
contact(str1,str2,…)	چند رشته را دریافت نموده و آن‌ها را به رشته جاری متصل می‌کند. عملکرد این متد همانند عملگر + است.
replace(substring,newstring)	در کل رشته، عبارت newstring را جایگزین substring می‌کند.
search(str)	عبارت str را درون رشته جستجو می‌کند و محل شروع آن را برمی‌گرداند. عدد ۱- نشانه عدم وجود عبارت در رشته است.
split()	رشته به مجموعه‌ای از زیررشته‌ها تفکیک می‌کند.
substr(start,length)	بخشی از رشته را برمی‌گرداند که از نمایه start شروع شده و به طول length ادامه می‌یابد.
toLowerCase()	رشته را به حروف کوچک تبدیل می‌کند.
toUpperCase()	رشته را به حروف بزرگ تبدیل می‌کند.

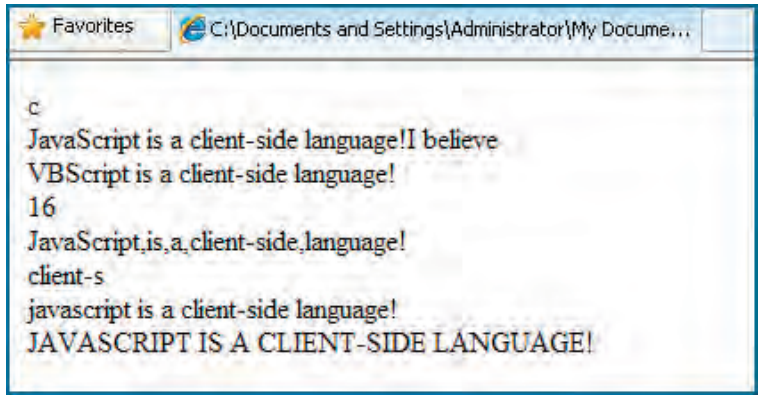
مثال Example

کد زیر چه مقادیری را برمی‌گرداند؟

```
<script type="text/javascript">
var str1=new String("JavaScript is a client-side language!");
var str2="I believe";
document.write(str1.charAt(5) + "<br/>");
document.write(str1.concat(str2,"<br/>"));
document.write(str1.replace("JavaScript","VBScript") + "<br/>");
document.write(str1.search("client") + "<br/>");
document.write(str1.split(" ") + "<br/>");
document.write(str1.substr(16,8) + "<br/>");
document.write(str1.toLowerCase() + "<br/>");
document.write(str1.toUpperCase() + "<br/>");
</script>
```

نکته

نمایه (index) نویسه‌ها از صفر شروع می‌شود، بنابراین در یک رشته، نویسه سوم دارای نمایه ۲ است.



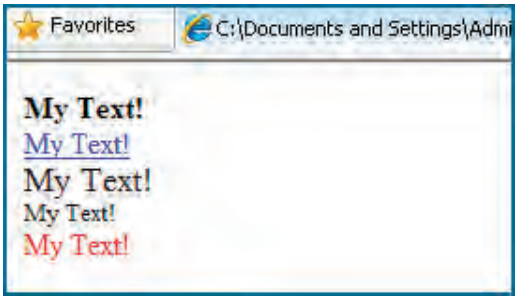
برای شـــیء String متدهای دیگری نیز تعریف شـــده که برای دســـت‌کاری ظاهر رشته‌ها در صفحات وب کاربرد دارند و همانند عناصر HTML عمل می‌کنند. تعدادی از این متدها را درون جدول بعد مشاهده می‌کنید.

متد	کاربرد
big()	اندازه فونت را بزرگ می‌کند.
blink()	متن را در حالت چشمک‌زن نمایش می‌دهد (در مرورگرهای IE، Chrome و Safari) عمل نمی‌کند.
bold()	متن را به صورت توپر نمایش می‌دهد.
fontcolor()	متن را با رنگ مشخص شده نشان می‌دهد.
Italics()	متن را به صورت مایل نمایش می‌دهد.
link()	متن را به صورت پیوند در می‌آورد.
small()	اندازه فونت را کوچک می‌کند.

مثال Example

نتیجه اجرای کد زیر چیست؟

```
<script type="text/javascript">
var str1=new String("My Text!");
document.write(str1.bold() + "<br/>");
document.write(str1.link("http://www.google.com") +"<br/>");
document.write(str1.big() + "<br/>");
document.write(str1.small() + "<br/>");
document.write(str1.fontcolor("red") + "<br/>");
</script>
```



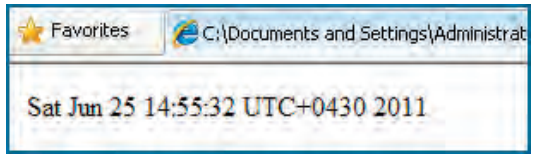
۲۱-۲-۲ شیء Date

با استفاده از این شیء می‌توانید زمان و تاریخ جاری را به دست آورده و عملیات موردنظر روی آن را پیاده‌سازی کنید. برای استخراج زمان و تاریخ جاری و قرار دادن آن درون یک شیء متغیر از دستور زیر استفاده می‌شود:

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

کدی بنویسد که زمان و تاریخ جاری را روی صفحه نمایش دهد.

```
var now=new Date();
document.write(now);
```



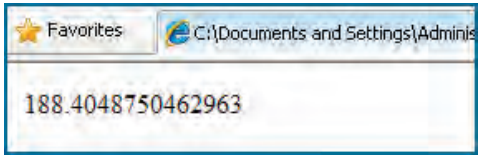
نحوه نمایش زمان و تاریخ به ما یادآوری می‌کند که برای استخراج زمان در قالب موردنظر باید از متدهای پیش‌بینی شده برای این شیء استفاده نماییم. متدهای پرکاربرد را در این جدول مشاهده می‌کنید.

متد	کاربرد
getDate()	شماره روز را در ماه برمی‌گرداند که عددی بین ۱ تا ۳۱ است.
getDay()	شماره روز را در هفته برمی‌گرداند که عددی بین ۰ تا ۶ است.
getMonth()	شماره ماه را به صورت عددی بین ۰ تا ۱۱ برمی‌گرداند.
getFullYear()	سال را به صورت چهاررقمی از تاریخ استخراج می‌کند.
getSeconds()	ثانیه را از شیء Date جدا می‌کند.
getMinutes()	دقیقه را از شیء Date استخراج می‌کند.
getHours()	عددی بین ۰ تا ۲۳ را به عنوان ساعت برمی‌گرداند.
getTime()	زمان سپری شده از روز ۱۹۷۰/۱/۱ را برحسب میلی‌ثانیه نمایش می‌دهد.

مثال Example

کدی بنویسید که نشان دهد چند روز تا پایان سال ۲۰۱۱ میلادی مانده است.

```
<script type="text/javascript">
var now=new Date();
var endOfYear=new Date(2011,11,31);
var msRemaining= endOfYear - now;
var daysRemaining = msRemaining/86400000;
document.write(daysRemaining);
</script>
```



بررسی کد:

- زمان جاری با ایجاد یک نمونه از شیء Date در متغیر now ذخیره می‌شود.
- انتهای سال میلادی به عنوان مقادیر اولیه برای شیء Date فرستاده می‌شود تا شیئی با زمان موردنظر ایجاد و در متغیر endOfYear ذخیره شود.
- زمان جاری از متغیر حاوی انتهای سال کسر می‌گردد و نتیجه درون متغیر msRemaining ذخیره می‌گردد. این مقدار برحسب میلی‌ثانیه است و باید آن را تبدیل کرد.
- با تقسیم عدد فوق بر ۸۶۴۰۰۰۰۰ که حاصل‌ضرب ۱۰۰۰×۶۰×۶۰×۲۴ است می‌توان تعداد روزهای باقی‌مانده را استخراج کرد و نمایش داد.

مثال Example

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

```
<script type="text/javascript">
var now=new Date();
```

```
var dayNum=now.getDay();
var day;
switch (dayNum)
{
    case 0:
    day = "یکشنبه";
    break;
    case 1:
    day = "دوشنبه";
    break;
    case 2:
    day = "سه شنبه";
    break;
    case 3:
    day = "چهارشنبه";
    break;
    case 4:
    day = "پنجشنبه";
    break;
    case 5:
    day = "جمعه";
    break;
    case 6:
    day = "شنبه";
    break;
}
document.write("امروز: " + day);
</script>
```

بررسی کد:

- متد ()getDay شماره روز را از تاریخ استخراج می‌کند.
- در تقویم میلادی، یکشنبه با عدد صفر نشان داده می‌شود.
- با استفاده از دستور switch، شماره روز بررسی شده و مقدار متناظر به متغیر day نسبت داده می‌شود.
- این مقدار توسط دستور write روی صفحه نوشته می‌شود.

مثال Example

کدی بنویسید که زمان جاری را نمایش دهد.

```
<script type="text/javascript">
var now=new Date();
document.write(now.getHours() + ":" + now.getMinutes() +":" + now.getSeconds());
</script>
```

در این مثال، زمانی که روی صفحه نمایش داده می‌شود، زمان بارگذاری صفحه است و برای مشاهده زمان فعلی باید صفحه را به صورت دستی، تازه‌سازی (Refresh) کرد. آیا راهی وجود دارد تا زمان به صورت خودکار روی صفحه تغییر کند؟ بله! کد زیر را امتحان کنید.

```
<html>
<head>
<script type="text/javascript">
function startTime()
{
    var today=new Date();
    var h=today.getHours();
    var m=today.getMinutes();
    var s=today.getSeconds();
    // اگر عدد کوچکتر از ۱۰ است یک صفر به ابتدای آن اضافه کن //
    m=checkTime(m);
    s=checkTime(s);
    document.getElementById('txt').innerHTML=h+":"+m+":"+s;
    t=setTimeout('startTime()',500);
}
function checkTime(i)
{
```

```
if (i<10)
  {
      i="0" + i;
  }
return i;
}
</script>
</head>

<body onload="startTime()">
<div id="txt"></div>
</body>
</html>
```

بررسی کد:

- در این کد تابعی به نام startTime تعریف شـده که همانند مثال قبل، زمان جاری را در قالب مناسـب استخراج می‌کند.
- از تابع checkTime برای تبدیل ثانیه‌ها و دقیقه‌های تک‌رقمی به دو رقمی استفاده می‌شود. این کار با افزودن یک صفر به سمت چپ آن‌ها انجام می‌شود.
- رویداد setTimeOut هر نیم ثانیه (۵۰۰ میلی‌ثانیه) یک‌بار تابع تولید زمان را فراخوانی می‌کند.
- درون صفحه، لایه‌ای با شناسه txt قرار داده شده و با استفاده از دستور .('document.getElementById('txt' innerHTML زمان درون آن درج می‌شود.
- تابع startTime در رویداد onload برچسب body (هنگام بارگذاری این برچسب) فراخوانی می‌شود.

در فصل‌های آینده مطالب بیش‌تری را در مورد رویدادها فراخواهید گرفت.

۲۱-۲-۳ شیء Math

با اسـتفاده از این شیء می‌توانید محاسـبات ریاضی موردنیاز را در زبان جاوا اسکریپت پیاده‌سازی نمایید. این شیء نیز همانند سایر اشیاء دارای تعدادی خصوصیت و تعدادی متد است که از میان خصوصیت‌ها می‌توان به PI اشاره نمود که عدد پی را برمی‌گرداند. متدهای پرکاربرد این شیء هم در جدول زیر فهرست شده‌اند.