Java Script

فصل نهم: شروع کار با Java Script

همانطور که قبلا گفتیم برنامه‌های تحت وب به دو بخش تقسیم می‌شوند:

- برنامه‌های استاتیک (Static)
- برنامه‌های دینامیک (Dynamic)

برنامه‌های استاتیک برنامه‌هایی هستند که صفحات آنها از متن، تصاویر، صوت و ... تشکیل شده‌اند و احیانا یک یا چند لینک به سایر آدرسها در آن قرار گرفته است. در واقع این نوع صفحات با کاربر تعاملی ندارند و صرفا کاربر می‌تواند موارد داخل این صفحات را مشاهده نماید. (در مورد این برنامه‌ها به تفضیل شرح دادیم)

برنامه‌های دینامیک برنامه‌هایی هستند که داده‌ای را از کاربر دریافت، و آن را مورد پردازش قرار می‌دهند، و در انتها نتیجه آن را در اختیار کاربر قرار می‌دهند. خود برنامه‌های دینامیک به دو بخش تقسیم می‌شوند:

- برنامه‌های سمت کاربر (Client side)
- برنامه‌های سمت سرور (Server side)

برنامه‌های Client side آنهایی هستند که پردازش روی داده‌ها در سمت کاربر توسط Web Browser انجام می‌شود. برای نوشتن اینگونه برنامه‌ها از زبانهایی چون VB Script,Java Script,... استفاده می‌شود.

برنامه‌های Server side آنهایی هستند که پردازش روی داده‌ها در سمت سرور انجام می‌شود. بدینگونه که داده‌ها از سمت کاربر به سمت سرور فرستاده می‌شود(به عنوان مثال با Submit کردن فرم سمت کاربر). سپس در سمت سرور عمل پردازش روی داده‌ها انجام شده و نتیجه پردازش به سمت کاربر فرستاده می‌شود. برای نوشتن اینگونه برنامه‌ها در سمت سرور از زبانهایی چون ASP.Net,PHP,... استفاده می‌شود.

آشنایی با Java Script

Java Script یک زبان برنامه‌نویسی سمت سرور می‌باشد. این زبان توسط شرکت Netscape اختراع شد و به عنوان اولین زبان اسکریپت نویسی ارایه شد. شرکت میکروسافت نیز برای اینکه از شرکت Netscape عقب نیافتد نسخه دیگری مشابه این زبان را با عنوان Jscript تولید نموده است که خیلی‌ها فکر می‌کنند همان Java Script است. از زبان Java Script برای کارهای زیر می‌توان استفاده نمود:

- اعتبارسنجی فرمهای HTML قبل از ارسال به سمت سرور
- شناسایی مرورگر کاربران
- ایجاد انواع منوها
- ...

کدهای نوشته در Java Script توسط کاربرهای دیگر نیز قابل مشاهده هستند. بدین معنا که هر کاربری می‌تواند کدهای نوشته شده در صفحه HTML را ببیند. به عنوان مثال:

View -> Source (Intertnet Explorer)

Right Click On Loaded Page -> Click On View Page Source item (Firefox)

بنابراین توصیه می‌شود از آن برای برنامه‌نویسی های مهم چون اتصال به Database و ... استفاده نشود

کدهای Java Script هم می‌توانند درون یک صفحه HTML بصورت ترکیبی با کدهای دیگر قرار گیرند و یا در یک فایل جداگانه با پسوند js.

تفاوتهای Java و Java Script

Java Script مربوط به شرکت Netscpae بوده ولی Java متعلق به شرکت Oracle است.

Java Script یک زبان اسکریپت نویسی است اما Java یک زبان برنامه نویسی است.

Java Script توسط مفسر تفسیر می‌گردد اما Java توسط کامپایلر، کامپایل می‌شود.

بطورکلی: Java Script در صفحات وب تاثیرگذار است اما برنامه‌های Java برای خلق برنامه‌های پیچیده بکار می‌رود.

اصول نوشتن کدهای Java Script

Java Script یک زبان حساس به متن است. در ضمن فاصله بین تگها باید متناسب باشد در غیر اینصورت کار نخواهد کرد. بنابراین زمان نوشتن دستورات خیلی باید دقت کرد. محل قرارگیری کدهای Java Script بین تگهای HTML می‌باشد. برای اینکه مفسر مرورگر بداند کجا باید کدهای Java Script اجرا و در کجا به آنها خاتمه دهد باید از تگهای <Script></Script> استفاده نمود.

شناسه‌های تگ Java Script :

language: مشخص کننده زبان اسکریپت نویسی است.

type: برای زبان Java Script باید مقدار آن را برابر text/javascript قرار داد.

بطور کلی محل نوشتن کدهای Java Script در سه بخش صورت می‌گیرد:

۱ - در بخش head ۲-در بخش body ۳-بصورت یک فایل خارجی

نحوه نوشتن کدهای Java Script

الف) درون تگ head: از این مورد زمانی باید استفاده کنید که مایلید کدهایی که وارد می‌کنید قبل از بقیه کدها اجرا شود. به مثال زیر دقت کنید:

```
<html>
<head>
        <script language="javascript" type="text/javascript">
                function message()
                {
                        alert ("Welcome to Farhad and Farzad imany site.");
                }
        </script>
</head>
<body onload="message()">
<h1> Hello Everybody </h1>
</body>
</html>
```

زمان فراخوانی این صفحه ابتدا پنجره‌ای باز شده و پیام زیر را نشان می‌دهد:

Welcome to Farhad and Farzad imany site.

ب) درون تگ body: از این مورد زمانی باید استفاده کنید که مایلید کدهایی که وارد می‌کنید در بین برنامه مورد استفاده قرار گیرد. به مثال زیر دقت کنید:

```
<html>
<body>
        <script language="javascript" type="text/javascript">
                document.write("Hello Everybody");
        </script>
</body>
</html>
```

زمان فراخوانی این صفحه پیام زیر داخل صفحه دیده خواهد شد:

Hello Everybody

می‌توانید از دستورات HTML داخل اسکریپتهای Java Script استفاده نمائید. مانند مثال زیر:

```
<html>
<body>
        <script language="javascript" type="text/javascript">
                document.write("<h1 align='center'> Hello Everybody </h1>");
        </script>
</body>
</html>
```

ج) بصورت فایل خارجی: از این مورد زمانی باید استفاده کنید که مایلید کدهایی که وارد می‌کنید در یک فایل خارجی با پسوند js باشد و در جایی که از فایل HTML نیاز بود آن را فراخوانی نمائید. این برای زمانی مفید است که قصد دارید از کدهای خود در چندین صفحه استفاده نمایید.به مثال زیر توجه کنید:

ابتدا دستور زیر را در یک فایل با نام message.js ذخیره می‌کنیم:

```
document.write("<h1 align='center'> Hello Everybody </h1>");
```

حال برای فراخوانی message.js در فایل HTML خود بصورت زیر عمل می‌کنیم:

```
<html>
<body>
        <script language="javascript" type="text/javascript" src="message.js">
        </script>
</body>
</html>
```

فصل دهم: متغیرها و دستورات شرطی در Java Script

متغیرها

برای تعریف یک متغیر در Java Script بصورت زیر عمل می‌کنیم:

```
Var نام متغیر;
```

دقت کنید که نام متغیر نباید با یک عدد شروع شود و همچنین هیچگونه فاصله‌ای هم نباید داشته باشد. در ضمن همانگونه که گفته شد در Java Script حالت حساس بودن به حروف وجود دارد. بنابراین متغیری با نام z با متغیری با نام Z کاملا تفاوت دارد.

اگر متغیری دارای مقدار متنی باشد، باید متن را داخل کوتیشن قرار داد. به مثالهای زیر توجه کنید:

```
Var x;
Var t="iran";
```

مقدار دهی متغیر می‌توان بصورت جداگانه صورت گیرد. به مثال توجه کنید:

```
Var x;
X=10;
```

اگر ابتدا به متغیر مقداری بدهید و بعد آن را تعریف کنید، مقدار قبلی آن نگاه داشته می‌شود:

```
X=10;
Var x;
```

انواع متغیرها

بطور کلی دو نوع متغیر وجود دارد که تفاوت آنها در محل مورد استفاده می‌باشد:

متغیرهای Global: در ابتدای اسکریپت تعریف شده و مقداردهی می‌شوند و سپس در همه جای کد قابل استفاده می‌باشند.

متغیرهای Local: فقط در یک قسمت از اسکریپت کاربرد داشته و معمولا در توابع استفاده می‌شوند.

انواع عملگر

عملگرهای ریاضی Arithmetic Operators

عملگر	توصیف	مثال	نتیجه
+	جمع	x = 2, y = 2 x+y	4
-	تفریق	x = 5 , y = 3 x - y	2
*	ضرب	x = 5 , y = 4 x * y	20
/	تقسیم	x = 20 , y = 4 x / y	5
%	خارج قسمت	x = 19 , y = 4 x % y	3
++	افزودن	x = 5 x++	x = 6
--	کم کردن	x = 5 x--	x = 4

عملگرهای انتصاب Assignment Operators

عملگرها	مثال	برابر است با
=	x = y	x = y
+ =	x + = y	x = x + y
- =	x - = y	x = x - y
* =	x * = y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y

عملگرهای مقایسه ای

عملگرها	توصیف	مثال
==	مقادیر اگر برابر باشند نتیجه درست خواهد بود.	x = 5 , y = 5 x == y
===	مقایسه بین مقادیر و جنس متغیر که در مثال روبرو نتیجه غلط بازگردانده می شود	x = 5 y = "5" x === y
!=	اگر مقادیر برابر نباشند نتیجه درست است	5 != 8
>	بزرگتر	8 > 5
<	کوچکتر	8 < 10
>=	بزرگتر یا مساوی که در این مثال نتیجه غلط است	5 >= 8
<=	کوچکتر یا مساوی که در این مثال نتیجه درست است	5 <= 8

عملگرهای منطقی

- && عملگری که به معنای ترکیب (و) دو مقدار است (x < 10 && y > 1)
- || عملگری که به معنای ترکیب (یا) دو مقدار است (x == 5 || y == 5)
- ! عملگری که به معنای (مخالف) است ! (x == y)

تاریخ و زمان در Java Script

آبکت new Date(): برای نمایش تاریخ و ساعت جاری بصورت زیر عمل کنید:

```
<html>
<body>
	<script language="javascript" type="text/javascript">
	document.write (new Date());
	</script>
</body>
</html>
```

حال اگر خواستیم جزییات مختلف این آبجکت نظیر سال و ماه و ... را استخراج کنیم بصورت زیر عمل می‌کنیم:

```
<html>
<body>

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

		d= new Date();
		document.write(d+"<br>");
		document.write(d.getFullYear()+"<br>");
		document.write(d.getMonth()+"<br>");
		document.write(d.getDay()+"<br>");
		document.write(d.getHours()+"<br>");
		document.write(d.getMinutes()+"<br>");
		document.write(d.getSeconds());

	</script>

</body>
</html>
```

اگر خواستید سال تاریخ جاری را صرفا تنظیم کنید بصورت زیر عمل کنید:

```
<html>
<body>

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

	d=new Date();
	d.setFullYear(2007);
	document.write(d);

	</script>

</body>
</html>
```

دستورات شرطی در Java Script

ساختار دستور بصورت زیر است:

```
If (شرط)
    {
        بلوک دستوراتی که در صورت برقراری شرط باید اجرا گردند
    }
else
    {
        بلوک دستوراتی که در صورت عدم برقراری شرط باید اجرا گردند
    }
```

مثال) برنامه‌ای بنویسید که اگر ساعت جاری سیستم از ۱۰ کوچکتر باشد پیام Good Morning را نشان دهد و در غیر اینصورت پیام Good Afternoon را نشان دهد.

```
<html>
<body>

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

    d=new Date();
    t=d.getHours()

    if (t<10)
            {
            document.write("Good Morning");
            }
    else
            {
            document.write("Good Afternoon");
            }
    </script>

</body>
</html>
```

Ifهای تودرتو در Java Script

اگر خواستید چند شرط در if مورد بررسی قرار گیرد از elseif استفاده می‌گردد. به مثال توجه کنید:

```
<html>
<body>

    <script language="javascript" type="text/javascript">
    t=21;

    if (t>=0 && t<=5)
    {
            document.write("greater than or equal to 0 and less than or equal 5");
    }

    else if (t>5 && t<=10)
    {
            document.write("greater than 5 and less than or equal to 10");
    }

    else if (t>10 && t<=15)
    {
            document.write("greater than 10 and less than or equal to 15");
    }
    else if (t>15 && t<=20)
    {
            document.write("greater than 15 and less than or equal to 20");
    }
    else
    {
            document.write("undefine range");
    }
</script>

</body>
</html>
```

switch در Java Script

همانطور که در بالا ملاحظه کردید استفاده از else if خوانایی برنامه را پایین می‌آورد. در این موارد بهتر است از ساختار switch استفاده گردد. به مثال زیر توجه کنید:

```
<html>
<body>
      <script language="javascript" type="text/javascript">
       var day=new Date().getDay();

         switch (day)
         {
            case 0:
              document.write("Today its Sunday");
              break;

            case 1:
              document.write("Today its Monday");
              break;

            case 2:
              document.write("Today its Tuesday");
              break;

            case 3:
              document.write("Today its Wednesday");
              break;

            case 4:
              document.write("Today its Thursday");
              break;

            case 5:
              document.write("Today its Friday");
              break;

            case 6:
              document.write("Today its Saturday");
              break;

           default:
              document.write("Error in processing");
         }

</script>

</body>
</html>
```

در این مثال خروجی دستور var day=new Date().getDay(); که شماره روز تاریخ جاری است در متغیر day قرار می‌گیرد. بر اساس مقدار day نام روز هفته چاپ می‌گردد. در Java Script عدد صفر معرف یکشنبه و عدد یک معرف دوشنبه و به همین صورت تا آخر ادامه می‌یابد.

پنجره‌های هشدار در Java Script

نوع اول) پنجره صرفا با ظاهر کردن یک پیام فعال می‌شود و کاربر با زدن OK این پنجره را می‌بندد:

```
<html>
<head>
	<script language="javascript" type="text/javascript">
		function message()
		{
			alert ("Welcome to Farhad and Farzad imany site.");
		}
	</script>
</head>
<body>
<input type="button" onclick="message()" value="show alert window">
</body>
</html>
```

نوع دوم) پنجره با ظاهر کردن یک پیام فعال می‌شود و کاربر با زدن OK یا Cancel می‌تواند روند ادامه عملیات را تعیین کند:

```
<html>
<head>
	<script language="javascript" type="text/javascript">
		function disp_confirm()
		{
			var r=confirm("Press a button!")
			if (r==true)
			{
				alert("You pressed OK!")
			}
			else
			{
				alert("You pressed Cancel!")
			}
		}
	</script>
</head>
<body>
<input type="button" onclick="disp_confirm()" value="Display a confirm box" />
</body>
</html>
```

نوع سوم) پنجره با ظاهر کردن یک پیام فعال می‌شود و کاربر می‌تواند مقداری را وارد نماید و سپس بر اساس مقدار وارده تصمیم گیری شود:

```
<html>
<head>
	<script language="javascript" type="text/javascript">
		function disp_confirm()
		{
			var r=prompt("What is your name?","Guest");
			alert ("Hello "+r);
		}
	</script>
</head>
<body>
<input type="button" onclick="disp_confirm()" value="Display a confirm box" />
</body>
</html>
```

فصل یازدهم: سایر دستورات در Java Script

توابع پویا در Java Script

تا کنون توابعی که ذکر شدند صرفا برای انجام یک عمل خاص بود و نمی‌توانستیم به آنها انعطاف بدهیم. حال می‌خواهیم تابعی بنویسیم که سه عدد مختلف را به آن بدهیم و تابع آنها را در هم ضرب و نتیجه را نشان دهد. برای اینکار:

```
<html>
<head>
	<script language="javascript" type="text/javascript">
		function delta(a,b,c)
		{
			return a*b*c;
		}
	</script>
</head>
<body>
<input type="button" onclick="document.write(delta(2,3,4))" value="Display a confirm box" />
</body>
</html>
```

حلقه‌ها در Java Script

هرگاه بخواهیم یک سری کد به دفعات مشخص تکرار شوند و یا بخواهیم یک سری کد تا زمانی که شرط مورد نظر ما برقرار است اجرا شود باید از حلقه‌ها استفاده کرد. بطور کلی در Java Script دو نوع حلقه داریم:(حلقه for - حلقه while)

مثال) برنامه‌ای با استفاده از حلقه for بنویسید که اعداد یک تا ده را روی صفحه زیر هم نشان دهد:

```
<html>
<head>
	<script language="javascript" type="text/javascript">
		function ring1()
		{
			var i=1;
				for (i=1;i<=10;i++)
				{
					document.write(i);
					document.write("</br>");
				}
		}
	</script>
</head>
<body>
<input type="button" onclick="ring1()" value="run"/>
</body>
</html>
```

مثال) برنامه‌ای با استفاده از حلقه while بنویسید که اعداد یک تا ده را روی صفحه زیر هم نشان دهد:

```
<html>
<head>
	<script language="javascript" type="text/javascript">
		function ring1()
		{
			var i=1;
				while (i<11)
				{
					document.write(i);
					document.write("</br>");
					i++;
				}
		}
	</script>
</head>
<body>
<input type="button" onclick="ring1()" value="run"/>
</body>
</html>
```

این حلقه شرط را در ابتدا تست کرده و تا زمانی که برقرار باشد کار را ادامه خواهد داد. یک نوع دیگر این حلقه بصورت زیر است که خواهید دید. این حلقه بدون توجه به شرط که درانتهای آن است دستکم یکبار اجرا می‌شود. به مثال دقت کنید:

```
<html>
<head>
	<script language="javascript" type="text/javascript">
		function ring1()
		{
			var i=1;
				do
				{
					document.write(i);
					document.write("</br>");
					i++;
				}
				while (i<11);

		}
	</script>
</head>
<body>
<input type="button" onclick="ring1()" value="run"/>
</body>
</html>
```

در این حلقه اگر مقدار متغیر i را برابر 11 قرار دهید خواهید دید که عدد 11 روی صفحه نمایش داده خواهد شد و بعد حلقه توقف خواهد نمود. یعنی شرط را در انتها تست می‌کند.

آشنایی با دستور break

به مثال زیر توجه کنید. مقدار i برابر با 1 است. شرط حلقه نیز بر این موضوع استوار است تا زمانی که مقدار متغیر i از صفر بیشتر است به کار خود ادامه بده. به این حلقه، حلقه بینهایت می‌گویند و هیچگاه متوقف نخواهد شد. وظیفه دستور break کنترل این موضوع در میان حلقه است و از افتادن حلقه به چرخه بینهایت جلوگیری می‌کند.

```
<html>
<head>
	<script language="javascript" type="text/javascript">
		function ring1()
		{
			var i=1;
				while (i>0)
				{
					document.write(i);
					document.write("<br>");
					i++;
					if (i==11) break;
				}
		}
	</script>
</head>
<body>
<input type="button" onclick="ring1()" value="run"/>
</body>
</html>
```

آشنایی با دستور continue

فرض کنید می‌خواهیم حلقه‌ای بنویسیم که اعداد یک تا ده را به استثنای عدد سه روی صفحه نشان دهد. در آن صورت باید از continue استفاده کرد. به مثال دقت کنید:

```
<html>
<head>
	<script language="javascript" type="text/javascript">
		function ring1()
		{
			var i=1;
				for (i=1;i<=11;i++)
				{
					if (i==3) continue;
				document.write(i);
				document.write("<br>");
				}
		}
	</script>
</head>
<body>
<input type="button" onclick="ring1()" value="run"/>
</body>
</html>
```

دستور continue هر جای حلقه باشد عملیات به ابتدای حلقه باز می‌گردد.

آرایه‌ها

از آرایه زمانی استفاده می‌کنیم که قصد داریم داخل یک متغیر چندین مقدار را ذخیره نمائیم. طرز تعریف آرایه بصورت زیر است:

```
var i = new array();
```

مثال) برنامه ای بنویسید که با استفاده از حلقه for اعداد صفر تا ده را در آرایه پر کرده و با یک حلقه for دیگر آنها را نمایش دهد.

```
<html>
<body>

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

        var myarr = new Array();

        for (i=0;i<11;i++) myarr[i]=i

        for (j=0;j<11;j++) document.write(myarr[j]+"</br>")

</script>

</body>
</html>
```

متدهای آرایه‌ها

متد length: نشان دهنده تعداد خانه‌های آرایه است.

```
<html>
<body>

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

        var myarr = new Array();

        for (i=0;i<11;i++) myarr[i]=i

        document.write(myarr.length);

</script>

</body>
</html>
```

متد concat: برای نمایش دو آرایه در امتداد همدیگر مورد استفاده قرار می‌گیرد:

```
<html>
<body>
<script language="javascript" type="text/javascript">
    var a1=["Reza","Hasan"];
    var a2=["Sara","Shabnam"];

    var a3=a1.concat(a2);

    document.write (a3);
</script>
</body>
</html>
```

متد join: زمانی که محتویات یک آرایه را با document.write (array name); نمایش می‌دهیم، بین آیتمها از علامت ویرگول استفاده می‌شود. اگر بخواهیم این علامت را ما تعیین کنیم از این متد بهره خواهیم گرفت:

```
<html>
<body>
<script language="javascript" type="text/javascript">
    var a1=["Reza","Hasan"];

    document.write (a1.join("&"));

</script>
</body>
</html>
```

متد pop: با بکارگیری این متد آخرین عنصر آرایه از آن بیرون کشیده می‌شود و طول آرایه یک واحد کم می‌شود

```
<html>
<body>
<script language="javascript" type="text/javascript">
    var a1=["Reza","Hasan"];

    document.write (a1.pop());

</script>
</body>
</html>
```

متد push: با بکارگیری این متد یک عنصر به آخر آرایه اضافه می‌شود و تعداد خانه‌های آرایه را نشان می‌دهد.

```
<html>
<body>

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

    var a1=["Reza","Hasan"];

    document.write (a1.push("Hamid"));

    document.write (a1);

</script>

</body>
</html>
```

متد shift: با بکارگیری این متد اولین عنصر آرایه از آن بیرون کشیده می‌شود و طول آرایه یک واحد کم می‌شود

```
<html>
<body>

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

    var a1=["Reza","Hasan"];

    document.write (a1.shift());


</script>

</body>
</html>
```

متد unshift: با بکارگیری این متد یک عنصر به اول آرایه اضافه می‌شود و تعداد خانه‌های آرایه را نشان می‌دهد.

```
<html>
<body>

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

    var a1=["Reza","Hasan"];

    document.write (a1.unshift("Hamid"));

    document.write (a1);

</script>

</body>
</html>
```

متد unshift در Internet Explorer کار نمی‌کند. برای دیدن نتیجه از firefox استفاده کنید

متد reverse: بالعکس نمودن محتویات آرایه وظیفه این متد است.

```
<html>
<body>

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

    var a1=["Reza","Hasan"];

    document.write (a1.reverse());

</script>

</body>
</html>
```

متد slice: نقطه ای را دریافت و از آن نقطه به بعد آرایه را بیرون می‌کشد و نمایش می‌دهد.

```
<html>
<body>

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

    var a1=["Reza","Hasan","Ali","Sara","Ahmad","Bahman"];

    document.write (a1.slice(3));

</script>

</body>
</html>
```

متد sort: مرتب سازی آرایه را برعهده دارد

```
<html>
<body>

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

    var a1=["Reza","Hasan","Ali","Sara","Ahmad","Bahman"];

    document.write (a1.sort());

</script>

</body>
</html>
```

فصل دوازدهم: آشنایی با توابع رشته‌ای

تابع length: برای استخراج طول رشته بکار می‌رود.

```
<html>
<body>

<script type="text/javascript">

    var txt = "Farhad and Farzad Imany";

    document.write(txt.length);

</script>

</body>
</html>
```

تابع charAt**:** برای استخراج یک کاراکتر با شماره اندیس معین از یک رشته متنی بکار می‌رود.

```
<html>
<body>

<script type="text/javascript">

    var txt = "Farhad and Farzad Imany";

    document.write(txt.charAt(2));

</script>

</body>
</html>
```

تابع concat**:** برای ترکیب دو رشته بکار می‌رود.

```
<html>
<body>

<script type="text/javascript">

    var str1 = "Farhad and ";
    var str2 = "Farzad Imany";

    document.write(str1.concat(str2));

</script>
</body>
</html>
```

تابع indexOf**:** نقطه شروع یک زیررشته در رشته را برمی‌گرداند.

```
<html>
<body>

<script type="text/javascript">

    var str1 = "Farhad and Farzad Imany";

    document.write (str1.indexOf("Farzad"));

</script>
</body>
</html>
```

تابع lastIndexOf**:** نقطه شروع یک زیررشته در رشته را برمی‌گرداند. (اگر آن زیررشته چندبار در متن باشد نقطه شروع آخرین زیررشته را بیرون خواهد داد.)

```
<html>
<body>

<script type="text/javascript">

    var str="Hello planet earth, you are a great planet.";

    document.write (str.lastIndexOf("planet"));


</script>
</body>
</html>
```

تابع substr**:** برای بیرون کشیدن یک زیررشته از رشته اصلی با دادن نقطه شروع و تعداد برش مورد استفاده قرار می‌گیرد.

```
<html>
<body>

<script type="text/javascript">

    var str="Hello planet earth, you are a great planet.";

    document.write (str.substr(6,6));

</script>
</body>
</html>
```

تابع toLowerCase**:** تبدیل کل رشته به حروف کوچک.

```
<html>
<body>

<script type="text/javascript">

    var str="Hello planet earth, you are a great planet.";

    document.write(str.toLowerCase());

</script>
</body>
</html>
```

تابع toUpperCase**:** تبدیل کل رشته به حروف بزرگ.

```
<html>
<body>

<script type="text/javascript">

    var str="Hello planet earth, you are a great planet.";

    document.write(str.toUpperCase());

</script>
</body>
</html>
```