بسم الله الرحمن الرحیم

آموزشگاه تحلیل داده

تخصصی ترین مرکز برنامه نویسی و دیتابیس در ایران

تعریف های تابع (function definition)

مدرس : مهندس افشین رفوآ

دوره آموزش Jquery

دوره آموزش JavaScript

تعریف های تابع (function definition)

برای ***تعریف*** یک تابع جدید در جاوا اسکریپت از کلیدواژه ی *function* استفاده می کنیم.

برای تعریف تابع می توان از **عبارت** (expression) و یا **اعلان** (declaration) استفاده کرد.

Function declaration

در مباحث اولیه ی این سری آموزشی با نحوه ی **اعلان** (declare) تابع آشنا شدید. ساختار نگارشی (syntax) تعریف تابع به ترتیب زیر بود:

```
function functionName(parameters) {
  code to be executed
}
```

توابع اعلان شده، بلافاصله پس از اعلان اجرا نمی شوند. این توابع برای استفاده در آینده ذخیره شده و به مجرد فراخوانی (invoke) اجرا می شوند.

مثال:

```
<!DOCTYPE html>
<html>
<body>
  <p>This example calls a function which performs a calculation, and returns the result:</p>
  <p id="demo"></p>
  <script>
    function myFunction(a, b) {
      return a * b;
    }
    document.getElementById("demo").innerHTML = myFunction(4, 3);
  </script>
</body>
</html>
```

توجه: جاوااسکریپت برای جدا سازی دستورات قابل اجرا از هم، از کاراکتر نقطه ویرگول استفاده می کند. از آن جایی که تعریف تابع (function declaration) یک دستور قابل اجرا محسوب نمی شود، آن را با کاراکتر نقطه ویرگول خاتمه نمی دهیم.

استفاده از عبارت برای تعریف تابع

می توان به وسیله ی expression، یک تابع جدید اعلان یا تعریف کرد.

عبارت تابع (function expression) را می توان در یک متغیر ذخیره کرد:

```
<!DOCTYPE html>
<html>
<body>
  <p>A function can be stored in a variable:</p>
  <p id="demo"></p>
  <script>
    var x = function (a, b) { return a * b };
    document.getElementById("demo").innerHTML = x;
  </script>
</body>
</html>
```

پس از اینکه عبارت تابع در متغیر ذخیره شد، می توان آن متغیر را به صورت تابع مورد استفاده قرار داد:

```
<!DOCTYPE html>
<html>
<body>
  <p>
    After a function has been stored in a variable,
    the variable can be used as a function:
  </p>
  <p id="demo"></p>
```

```
  <script>
    var x = function (a, b) { return a * b };
    document.getElementById("demo").innerHTML = x(4, 3);
  </script>
</body>
</html>
```

تابع بکار رفته در مثال پیش، در واقع یک ***تابع ناشناس*** (anonymous function= تابع بدون اسم) بود.

توابعی که در متغیر ذخیره می شوند، نیازی به اسم ندراند. همیشه برای فراخوانی آن ها باید از اسم متغیر استفاده شود.

تابع مثال فوق به یک نقطه ویرگول ختم می شود زیرا که تابع یاد شده جزئی از یک دستور قابل اجرا می باشد.

تابع توکار ( ) function ـ یک سازنده (constructor)

همان طور که در مثال های قبلی شاهد بودید، توابع در زبان جاوا اسکریپت با استفاده از کلیدواژه ی function اعلان می شوند.

روش دیگری نیز برای اعلان تابع جدید وجود دارد که آن استفاده از تابع سازنده ی توکار خود زبان جاوا اسکریپت به نام Function( ) است.

مثال:

```
<!DOCTYPE html>
<html>
<body>
  <p>JavaScrip has an built-in function constructor.</p>
  <p id="demo"></p>
  <script>
    var myFunction = new Function("a", "b", "return a * b");
    document.getElementById("demo").innerHTML = myFunction(4, 3);
  </script>
</body>
</html>
```

البته استفاده از تابع مزبور کاملا اختیاری است. مثال بالا را اینگونه نیز می توان نوشت:

```
<!DOCTYPE html>
<html>
<body>
  <p id="demo"></p>
```

```
  <script>
    var myFunction = function (a, b) { return a * b }
    document.getElementById("demo").innerHTML = myFunction(4, 3);
  </script>
</body>
</html>
```

نکته: در اغلب موارد می توان از بکار بردن واژه ی کلیدی new خودداری کرد.

Function hoisting_قرار دادن تعریف تابع در بالای scope

در فصل های گذشته با مفهوم hoisting (یک رفتار پیش فرض جاوا اسکریپت) آشنا شدید.

Hoisting یک رفتار پیش فرض زبان است طی آن declaration (**تعریف**) تابع به بالای scope (دامنه) جاری انتقال داده می شود.

این رفتار پیش فرض برای تعریف متغیرها و توابع اعمال می شود.

به خاطر وجود این قابلیت، می توان توابع را پیش از اعلان مورد استفاده قرار داد:

```
myFunction(5);

function myFunction(y) {
  return y * y;
}
```

توابعی که با expression (عبارت) تعریف می شوند، از این قابلیت بهره نمی برند.

توابع خود فراخوان (Self-Invoking Functions)

می توان کاری کرد که function expression ها " خود فراخوان " شوند.

یک عبارت خود فراخوان (self-invoking expression) به صورت خودکار و بدون اینکه صدا زده شود، اجرا (راه اندازی) می شود.

هر گاه " () " پس از عبارت تابع (function expression) قرار بگیرد، عبارت تابع به خودی خود اجرا می شود.

یک function declaration (اعلان تابع) قابلیت خود فراخوانی را ندارد. برای اینکه نشان دهیم این یک عبارت تابع است که خود را صدا می زند، باید در هر دو طرف تابع مورد نظر پرانتز درج کنیم:

```
<!DOCTYPE html>

<html>
<body>
  <p>Functions can be invoked automatically without being called:</p>
  <p id="demo"></p>
  <script>
    (function () {
      document.getElementById("demo").innerHTML = "Hello! I called myself";
    })();
  </script>
</body>
</html>
```

تابع بکار رفته در این مثال، یک تابع ناشناس و بدون اسم است خودش خود را صدا می زند.

استفاده از توابع به عنوان مقدار

توابع جاوا اسکریپت را می توان به صورت مقدار مورد استفاده قرار داد:

```
<!DOCTYPE html>
<html>
<body>
  <p>Functions can be treated as values:</p>
  <p>x = myFunction(4,3) or x = 12</p>
  <p>In both cases, x becomes a number with the value of 12.</p>
  <p id="demo"></p>
  <script>
    function myFunction(a, b) {
      return a * b;
    }
    var x = myFunction(4, 3);
    document.getElementById("demo").innerHTML = x;
  </script>
</body>
</html>
```

همچنین می توان توابع جاوا اسکریپت را عبارات بکار برد:

```
<!DOCTYPE html>
<html>
<body>
  <p>Functions can be used in expressions.</p>
  <p id="demo"></p>
  <script>
    function myFunction(a, b) {
      return a * b;
    }
```

```
    var x = myFunction(4, 3) * 2;
    document.getElementById("demo").innerHTML = x;
  </script>
</body>
</html>
```

توابع نوعی شی هستند!

عملگر typeof یک تابع را به عنوان خروجی برای توابع دیگر باز می گرداند.

توابع جاوا اسکریپت را بهتر می توان تحت عنوان شی تعریف کرد، به این خاطر که هم دارای خواص هستند و هم متد.

خاصیت arguments.length در واقع تعداد آرگومان های دریافت شده را به هنگام فراخوانی تابع مورد نظر ، باز می گرداند:

```
<!DOCTYPE html>
<html>
<body>
  <p>The arguments.length property returns the number of arguments received by the function:</p>
  <p id="demo"></p>
  <script>
    function myFunction(a, b) {
      return arguments.length;
    }
    document.getElementById("demo").innerHTML = myFunction(4, 3);
  </script>
</body>
</html>
```

متد toString() یک تابع را به صورت رشته برمی گرداند:

```
<!DOCTYPE html>
<html>
<body>
  <p>The toString() method returns the function as a string:</p>
  <p id="demo"></p>
  <script>
    function myFunction(a, b) {
      return a * b;
    }
    document.getElementById("demo").innerHTML = myFunction.toString();
  </script>
</body>
</html>
```

نکته: تابعی که به عنوان خاصیت برای یک شی تعریف می شود، متد آن شی خوانده می شود.

تابعی که به منظور ایجاد اشیا جدید طراحی شده، object constructor (سازنده ی شی) نامیده می شود.