|  |
| --- |
| **Introduction to Functions** |
|  | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Overview of Functions** | |  |      |  | | --- | | **Introduction** | |  |   A function is an assignment that JavaScript can take care of for you. It is a technique used to isolate a group of instructional code so that other parts of the application can depend on it to provide them with a reliable result. There are two main categories of functions you will be using in JavaScript: those that have already been created for you and those you create in your code.   |  | | --- | | **Practical Learning: Introducing Functions** | |  |  |  |  | | --- | --- | | 1. Start your text editor and type the following text:  |  | | --- | | <html>  <head>  <title>Henley's Department Store</title>  </head>  <body>  <h1>Henley's Department Store</h1>  <h3>Customers Order Processing</h3>  </body>  </html> |  1. Save the file as **functions1.htm** in the JavaScript Lessons folder or directory 2. Preview the file in the browser and return to your text editor |  |  | | --- | | **Function Definition** | |  |  |  | | --- | | To create a function, type the **function** keyword followed by a name for the function, followed by an opening, then a closing, parentheses, followed by an opening curly bracket “{“ and ending with a closing curly bracket “}”. Here is the syntax for creating a function:  **function** *name*(){}  The **function** keyword is required. It lets the browser know that you are creating a function. The name of the function:   * Must start with a letter or an underscore * Can contain letters, digits, and underscores in any combination * Cannot contain space * Cannot contain special characters   When naming your functions, be explicit, give names that can identify what mission the function is carrying. For example, if a function displays something, you can call it *display*. If the name of the function is a combination, you can type an underscore between them, like this: show\_window. Alternatively, you can start the first letter of the first word in lowercase and the first letter of each of the other parts in uppercase, like this: showWindow  The parentheses are required, they set apart a function from the other items of your script or web page. We will find out that they can also be used for other things.  The assignment carried by the function is delimited by the opening and the closing curly brackets {}. Everything between the bracket is part of the function. The section between the brackets is called the body of the function. A function can consist of a simple operation, an expression, or a series of more complex code. For example, you can use the**document.write()** method that we have called previously to display a string. If the assignment carried by the function is made of a single and simple line, you can write the whole function in one line. Here is an example: |  |  | | --- | | <Script Language="JavaScript">    function showAddress(){document.write("12404 Lockwood Drive #D4");}  </Script> |  |  |  |  | | --- | --- | --- | | Code written on a single line with the function name is difficult to read. Besides, most of your functions will consist of many lines of code. Therefore, it is better not only to separate the function name from its code but also to indent it, which makes it easier to read. The above function could (and should) be written as follows:   |  | | --- | | <Script Language="JavaScript">  function showAddress()  {    document.write("12404 Lockwood Drive #D4");  }  </Script> |   From our previous knowledge of HTML, we have learned so far that the browser is mainly used to give simple instructions to the browser. While the body of the HTML file is used to display items, the code we write is dealt with behind the scenes. For this reason, it can be included in the head section of an HTML file. Here is an example:    |  | | --- | | <html>  <head>  <title>JavaScript Tutorial</title>  <Script Language="JavaScript">  function showAddress()  {    document.write("12404 Lockwood Drive #D4");  }  </Script>  </head>  <body>  <h1>Graphical User Interface Objects</h1>  </body>  </html> |   A function can also have its code in the same section where it is needed as many of the functions on this site will be implemented. |      |  | | --- | | **Practical Learning: Defining Functions** | |  |  |  |  | | --- | --- | | 1. To define a function, change the file as follows:  |  | | --- | | <html>  <head>  <title>Henley's Department Store</title>  <Script Language="JavaScript">  function welcomeMessage()  {  document.write("Welcome to Henley's Department Store!");  }  </Script>  </head>  <body>  <h1>Henley's Department Store</h1>  <h3>Customers Order Processing</h3>  </body>  </html> |  1. Save the file |      |  | | --- | | **Function Calling** | |  |  |  |  | | --- | --- | | Using a function is referred to as **calling the function**. Because a JavaScript function is not part of HTML, there are some rules you should follow to call it. So far, we have been calling a function called **write** which is a member of the **document** object. As you recall, we had to start the calling of the function with the **<script>** tag. In the same way, if a function is able to carry a complete assignment, you can call it between the **<script>** and **</script>** tags.  To call the simplest JavaScript function such as the one above, type the name of the function followed by its parentheses. Here is an example:   |  | | --- | | <Script Language="JavaScript">  function showAddress()  {    document.write("12404 Lockwood Drive #D4");  }  </Script>    <Script Language="JavaScript">  showAddress();  </Script> | |  |  | | --- | | **Practical Learning: Calling Functions** | |  |  1. Call the above function as follows:  |  | | --- | | <html>  <head>  <title>Henley's Department Store</title>  <Script Language="JavaScript">  function welcomeMessage()  {  document.write("Welcome to Henley's Department Store!");  }  </Script>  </head>  <body>  <h1>Henley's Department Store</h1>  <h3>Customers Order Processing</h3>  <Script Language="JavaScript">  welcomeMessage();  </Script>  </body>  </html> |  1. Save and preview the file in your browser:   http://www.functionx.com/javascript/images/preview2.gif 2. Return to your text editor 3. Create a new file and type the following:  |  | | --- | | <html>  <head>  <title>JavaScript Tutorial</title>  </head>  <body>  <h2 font face="Georgia, Garamond, Times New Roman" color="#FF0000">  Credit Application</h2>  <font face="Verdana, Tahoma, Arial" size="3">  <form name="CreditApplication">  <table border="0" cellpadding="0" cellspacing="0">  <tr>  <td width="100">First Name:</td>  <td><input type="text" name="txtFirstName" size="14"></td>  </tr>  <tr>  <td width="100">Last Name:</td>  <td><input type="text" name="txtLastName" size="14"></td>  </tr>  <tr>  <td width="100">Full Name:</td>  <td><input type="text" name="txtFullName" size="30"></td>  </tr>  <tr>  <td width="100"><br><br></td>  <td>  <input type="button" value="Show Full Name">  </td>  </tr>  </table>  </form>  </font>  </body>  </html> |  1. Save the file as **credit1.htm** in the JavaScript Lessons folder or directory 2. Preview it in your browser:   http://www.functionx.com/javascript/images/preview9.gif 3. Return to your text editor |  |

|  |
| --- |
| **Functions and Local Variables** |
|  |

|  |  |  |  |
| --- | --- | --- | --- |
| When it receives assignments, a function is supposed to handle a specific task to produce a result. If a function is not given some items it needs, it can provide its own accessories. This means that you can declare some variables that the function needs, manipulate those variables as you wish and hand the expected result to other sections of the program or file that need it.  You declare a variable in the body of a function the same way you declare a variable in the scripts we have used so far. A variable declared in the body of a function is referred to as a local variable. Here is an example:   |  | | --- | | <Script Language="JavaScript">  function showAddress()  {    var customerAddress;  }  </Script> |   After declaring such a variable, you can initialize it and hand it to another function such as**document.write()**. Here is an example:   |  | | --- | | <Script Language="JavaScript">  function showAddress()  {    var custAddress;    custAddress = "12404 Lockwood Drive #D4";    document.write(custAddress);  }  </Script>    <Script Language="JavaScript">  showAddress();  </Script> |   A locally declared variable can also be involved in any operation as you see fit. In the following example, local variables are declared to hold different values for the calculation of a net price. Other variables are declared to hold sentences to display on the web page. The sentences are then passed to the **document.write()** method to display them:   |  | | --- | | <Script Language="JavaScript">  function calculateTheDiscount()  {    var itemPrice = 120.55;    var discountRate = 0.20; // = 20%    var discountAmount = itemPrice \* discountRate;    var netPrice = itemPrice - discountAmount;    var sentence1 = "Item Price: $" + itemPrice + "<br>";    var sentence2 = "Discount: $" + discountAmount + "<br>";    var sentence3 = "Net Price: $" + netPrice + "<br>";    document.write(sentence1);    document.write(sentence2);    document.write(sentence3);  }  </Script>  <Script Language="JavaScript">  calculateTheDiscount();  </Script> | |

|  |
| --- |
| **Practical Learning: Using a Function's Local Variables** |
|  |

|  |  |
| --- | --- |
| 1. To declare and use some local variables in a function, change the file as follows:  |  | | --- | | <html>  <head>  <title>JavaScript Tutorial</title>  <Script Language="JavaScript">  function getFullName()  {  var firstName, lastName, fullName;    firstName = document.CreditApplication.txtFirstName.value;  lastName = document.CreditApplication.txtLastName.value;  fullName = firstName + " " + lastName;    document.CreditApplication.txtFullName.value = fullName;  }  </Script>  </head>  <body>  <h2 font face="Georgia, Garamond, Times New Roman" color="#FF0000">  Credit Application</h2>  <font face="Verdana, Tahoma, Arial" size="3">  <form name="CreditApplication">  <table border="0" cellpadding="0" cellspacing="0">  <tr>  <td width="100">First Name:</td>  <td><input type="text" name="txtFirstName" size="14"></td>  </tr>  <tr>  <td width="100">Last Name:</td>  <td><input type="text" name="txtLastName" size="14"></td>  </tr>  <tr>  <td width="100">Full Name:</td>  <td><input type="text" name="txtFullName" size="30"></td>  </tr>  <tr>  <td width="100"><br><br></td>  <td>  <input type="button" value="Show Full Name" **onClick="getFullName()"**>  </td>  </tr>  </table>  </form>  </font>  </body>  </html> |  1. Save and preview the file in your browser:   http://www.functionx.com/javascript/images/preview6.gif 2. Return to your text editor |

|  |
| --- |
| **Functions and External Values** |
|  |

|  |
| --- |
| **Passing an Argument** |
|  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| When asking a function to perform an assignment, a function may need to be given the necessary value to perform its job. For example, if a function is supposed to calculate the area of a square, the function may need at least the measurement of the side to proceed. Another function that needs to calculate the price of an item after applying a discount rate would need the item’s price and the discount rate; such a function would produce the net price (price after discount).  An item or value that a function needs is called an argument. It can also be called a parameter. An argument for a function is provided between the parentheses for the function that will need it. Providing an argument to a function is referred to as **passing the argument**.  To pass an argument to a function, when implementing the function, type a variable name that represents the argument. Here is an example:  function circleArea(radius);  In the body of the function, use the argument any way you see fit. For example, its value can be involved in a calculation. Here is an example of a function that is passed an argument. The function calculates the area of a circle using the radius that is supplied:   |  | | --- | | <Script Language="JavaScript">  function circleArea(radius)  {    var area;    area = radius \* radius \* 3.14159;  }  </Script> |   To call a function that takes an argument, you must supply the argument. If you know the value of the argument, you can provide it:   |  | | --- | | <Script Language="JavaScript">  function circleArea(radius)  {    var area;    area = radius \* radius \* 3.14159;    document.write("The area of the circle is ", area);  }  </Script>  <Script Language="JavaScript">  **circleArea(20.75)**  </Script> |   You can also provide the argument using the name of the variable that holds the necessary value. The variable that is passed and the argument used by the function don’t need to have the same name but they must be of the same data type. For example, if a function is using a numeric value to perform a calculation, make sure that the variable you are providing is holding a numeric value. Here is an example of passing an argument to a function using the name of a variable:   |  | | --- | | <Script Language="JavaScript">  function circleArea(radius)  {    var area;    area = radius \* radius \* 3.14159;    document.write("The area of the circle is ", area);  }  </Script>  <Script Language="JavaScript">    var measure = 45.25;    circleArea(measure );  </Script> |   A function can also take more than one argument. In this case, provide a name for each argument between the parentheses of the function. The arguments must be separated by a comma. Here is an example:  function rectangleArea(length, height);  In the body of the function, use the variables in any necessary way. Here is an example:   |  | | --- | | <Script Language="JavaScript">  function rectangleArea(length, height)  {    var area;    area = length \* height \* 3.14159;    document.write("The area of the rectangle is ", area);  }  </Script>  <Script Language="JavaScript">    var L, H;    L = 52.05;    H = 46.55;    rectangleArea(L, H);  </Script> |   The arguments of a function don’t have to be of the same type. One of the arguments could be a natural number (**integer**) while another argument could be a **string**; yet another argument of the same function could be a decimal number or a character. If you implement a function that takes different types of arguments, when calling the function and supplying the arguments, provide the arguments following the order of the types of arguments supplied. Here is an example of a function that takes different types of arguments:   |  | | --- | | <Script Language="JavaScript">  function rectangleArea(length, height, shape)  {    var area;    area = length \* height \* 3.14159;    document.write("The area of the ", Shape, " is ", area);  }  </Script>  <Script Language="JavaScript">    var L, H, S;    L = 52.05;    H = 46.55;    S = "rectangle";    rectangleArea(L, H, S);  </Script> | |

|  |
| --- |
| **Returning a Value** |
|  |

|  |
| --- |
| When a function has performed an assignment, it may provide a value that other functions would need, for any reason you judge necessary. When a function produces a value an makes it available to other functions, such a function is said to **return** a value.  To define a function that returns a value, just before the closing curly bracket, type the**return** keyword followed by the value that the function returns. The value should be a valid and expected type. Because JavaScript is not a type-checking language, meaning it hardly checks anything you are doing, it is your responsibility to make sure that your function returns the right value.  Here is a function that returns a value:  function rectangleArea(length, height)  {  var area;  area = length \* height \* 3.14159;  return area;  }  Even if a function returns a value, it can still be called like any other function. If a function returns a value, its name can be assigned to a variable to give a value to, or change the value of, the variable. Here is an example:  <Script Language="JavaScript">  function rectangleArea(length, height)  {  var area;  area = length \* height \* 3.14159;  return area;  }  function displayArea()  {  var l, h, rectArea;  l = 52.05;  h = 46.55;  rectArea = rectangleArea(l, h);  document.write("The area of the is ", rectArea);  }  </Script>  <Script Language="JavaScript">  displayArea();  </Script> |