Form Input and Output  
Form Input  
  
You can obtain user input by using a form input text box, a form button, and the onclick event handler. This is the preferred way for handling multiple input. Instead of invoking multiple prompts, you can obtain the input from the form, using dot syntax.  
  
For example:  
  
<head>  
<script type="text/javascript">  
/\* <![CDATA[ \*/  
function process()  
{  
var number1, number2, n1, n2, sum;  
number1 = document.myform.num1.value;  
number2 = document.myform.num2.value;  
n1 = parseInt(number1);  
n2 = parseInt(number2);  
sum = n1 + n2;  
alert("The sum of the numbers is " + sum); }  
/\*]]>\*/  
</script>  
</head>  
<body>  
<div align="center">  
<form name="myform" action="">  
First Number: <input type="text" name="num1" size="10" />  
<br /><br />  
Second Number: <input type="text" name="num2" size="10" />  
<br /><br />  
<input type="button" onclick="process()" value="SUBMIT" />  
<input type="reset" value="RESET" />  
</form>  
</div>  
</body>   
  
[Displayed](http://fog.ccsf.cc.ca.us/~srubin/form1.html).  
  
Explanations:  
  
BODY SECTION:  
  
The form statement is given a name, in this case 'myform', so that the script can access the values of the form's elements. The input text statements also have distinct names so that their values can be accessed by the script. The input button functions to invoke the script's function, named 'process', via the onclick event handler.  
  
THE SCRIPT:  
  
The script contains the user-declared function process(), whose content is required to be enclosed within braces. After declaring the variables to be used in the function, we retrieve the value of the input text boxes and assign them to variables. Namely:  
  
number1 = document.myform.num1.value;  
number2 = document.myform.num2.value;   
  
This statement means to assign the value of the field whose name is 'num1', in the form whose name is 'myform', to the variable 'number1'.  
  
It should be noted that:  
number1 = myform.num1.value;  
works in IE but NOT in Firefox. Firefox REQUIRES it to be  
number1 = document.myform.num1.value;  
  
After the string values are converted to integers via parseInt statements, n1 + n2 is calculated and assigned to the variable 'sum'. Finally an alert is used to display the desired result.  
  
Another way to obtain a value from a form input text box is to use the getElementById method. Assume that the form input text box is:  
  
<input type="text" id="num1" size="10" />  
  
Notice that an 'id' attribute replaces the 'name' atribute. In the script's function, we assign the variable 'number1' to the value of the input text field whose name is 'num1'.  
  
number1 = document.getElementById("num1").value  
  
This is the same as: number1 = document.myform.num1.value;  
  
So given that our form contains:  
  
<input type="text" id="num1" size="10" />  
<input type="text" id="num2" size="10" />  
  
Our entire script may look like this:  
  
<script type="text/javascript">  
/\* <![CDATA[ \*/  
function process()  
{  
var number1, number2, n1, n2, sum;  
number1 = document.getElementById("num1").value;  
number2 = document.getElementById("num2").value;  
n1 = parseInt(number1);  
n2 = parseInt(number2);  
sum = n1 + n2;  
alert("The sum of the numbers is " + sum); }  
/\*]]>\*/  
</script>  
</head>   
  
Note that when using the getElementById method as in the above example, you should not include the name of the form. The following would cause a syntax error:  
  
number1 = document.myform.getElementById("num1").value;  
Form Output  
  
One way to output the results from a script is to use document.write statements within the body tags. The problem with this method is that it usually leads to writing to a new page. A better approach is to output the results, using either form input text boxes or a form textarea. For example:  
  
<head>  
<script type="text/javascript">  
/\* <![CDATA[ \*/  
function process()  
{  
var number1, number2, n1, n2, sum;  
number1 = document.myform.num1.value;  
number2 = document.myform.num2.value;  
n1 = parseInt(number1);  
n2 = parseInt(number2);  
sum = n1 + n2;  
document.myform.result.value = sum;  
}  
/\*]]>\*/  
</script>  
<body>  
<div align="center">  
<form name="myform" action="">  
First Number:<input type="text" name="num1" size="10" />  
<br /><br />  
Second Number:<input type="text" name="num2" size="10" />  
<br /><br />  
Sum is: <input type="text" name="result" size="10" />  
<br /><br />  
<input type="button" onclick="process()" value="SUBMIT" />  
<input type="reset" value="RESET" />  
</form>  
</body>   
  
[Displayed](http://fog.ccsf.cc.ca.us/~srubin/form2.html).  
  
Notice that:  
document.myform.result.value = sum;  
actually outputs the value of 'sum' into the form text box whose name is 'result'. Alternatively, you may choose to ouptut the results into a form textarea. For example:  
  
<head>  
<script type="text/javascript"> /\*  
<![CDATA[ \*/  
function process()  
{  
var number1, number2, n1, n2, sum;  
number1 = document.myform.num1.value;  
number2 = document.myform.num2.value;  
n1 = parseInt(number1);  
n2 = parseInt(number2);  
sum = n1 + n2;  
document.myform.result.value = ("The sum of the numbers is " + sum);  
}  
/\*]]>\*/  
</script>  
<body>  
<div align="center">  
<form name="myform" action="">  
First Number: <input type="text" name="num1" size="10" />  
<br /><br />  
Second Number:  
<input type="text" name="num2" size="10" />  
<br /><br />  
Results: <textarea rows="4" cols="40" name="result"></textarea>  
<br /><br />  
<input type="button" onclick="process()" value="SUBMIT" />  
<input type="reset" value="RESET" />  
</form>  
</body>   
  
[Displayed](http://fog.ccsf.cc.ca.us/~srubin/form3.html).  
  
The advantage of using a textarea is that you can display multiple lines of output. For example, using the last example, suppose I change:  
  
document.myform.result.value = ("The sum of the numbers is " + sum);  
to the following:  
  
document.myform.result.value = ("The inputted numbers were " + n1 + ", " +  
n2 + "\nThe sum of the integers was " + sum);   
  
[Displayed](http://fog.ccsf.cc.ca.us/~srubin/form4.html).  
  
Notice the use of \n to cause a line break in the outputted string. Also notice that when you break up a long string over two or more lines that the concatenation symbol + must end all lines, except the last. Notice that the following can also be used:  
  
document.getElementById("result").value = ("The inputted numbers were " + n1 +  
", " + n2 + "\nThe sum of the integers was " + sum);  
  
  
You can also add CSS style rules to form elements for an interesting visual effect. For example:  
  
<style type="text/css">  
textarea {font-family: verdana; font-size: 22px;}  
</style>   
  
[Displayed](http://fog.ccsf.cc.ca.us/~srubin/form5.html).