Functions and Expressions

Functions

- Functions refer to a related group of JavaScript statements that are executed as a single unit.
- The syntax for defining a function is:

  function name_of_function(parameters {

  statements;

  }

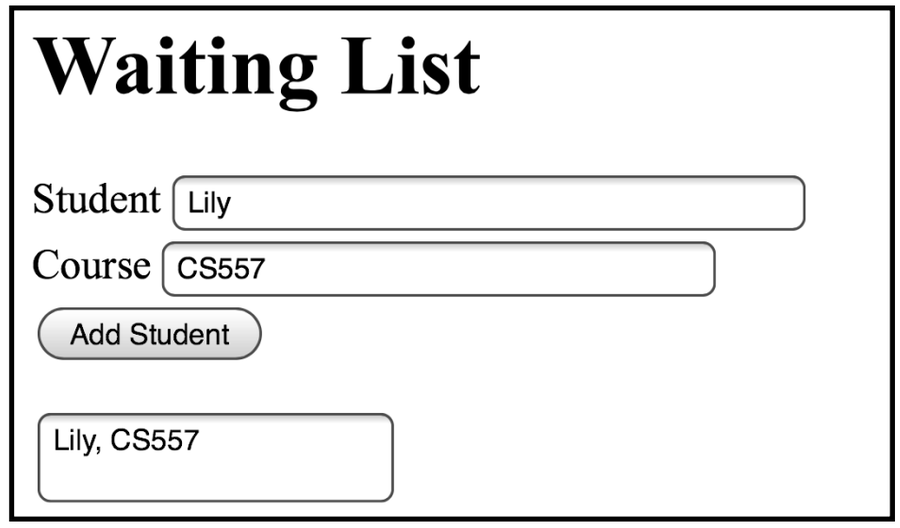
  - A parameter is a variable that is used within a function.
  - You do not need to include the var keyword.
- To execute a function, you must call it from elsewhere in you program. The JavaScript code that calls a function is referred to as a function call and consists of-the function name followed by parentheses that contains **arguments** or **actual parameters**.
  - Passing arguments is to send arguments to the parameters of a called function.

Example of using a simple function

```
<html>
    <head>
        <title>function example 1</title>
        <script type="text/javascript">
        function printStudentNames(student1, student2, student3) {
            document.write("<p>" + student1 + "</p>");
            document.write("<p>" + student2 + "</p>");
            document.write("<p>" + student3 + "</p>");
        }
        </script>
    </head>
    <body>
        <h1>function example 1</h1>
        <script type="text/javascript">
            printStudentNames("Peter", "Lily", "Nancy");
        </script>
    </body>
</html>
```

Example of Waiting List

```
<html>
    <head>
        <title>Waiting List</title>
        <script type="text/javascript">
            function addStudent(){
                var studentInfo = document.newStudent.studentName.value + ", ";
                studentInfo += document.newStudent.course.value;
                document.newStudent.students.value = studentInfo;
            }
        </script>
    </head>
    <body>
        <h1>Waiting List</h1>
        <form action="" method="post" name="newStudent">
          <p>
            Student
            <input type="text" name="studentName"
            style="width: 240px" /><br />
            Course
            <input type="text" name="course"
            style="width: 210px" /><br />
            <input type="button" value="Add Student" onclick="addStudent()" />
          </p>
          <p>
            <textarea name="students"></textarea>
          </p>
          </form>
    </body>
</html>
```



Results from a function

- In many cases, you may want your program to receive the results from a called function and then use those results in other code.
  - You can return a value from a function to a calling statement by assigning the calling statement to a variable.
  - To return a value to a returnValue variable, the code must include a return statement within a function.
- For example, the following script contains the findMax() function, which compares two numbers and returns the biggest number to the calling statement.

```
function findMax(a, b) {
    var max;
     If (a > b)
         max  = a;
     else
         max = b;
     return max;
}
```

Example of Waiting List using a return statement

```
<html>
    <head>
        <title>Waiting List</title>
        <script type="text/javascript">
            function addStudent(name, course){
                var studentInfo = name + ", ";
                studentInfo += course;
                return studentInfo;
            }
        </script>
    </head>
    <body>
        <h1>Waiting List</h1>
        <form action="" method="post" name="newStudent">
        <p>
         Student
         <input type="text" name="studentName"
         style="width: 240px" /> 
         <br />
         Course
         <input type="var text" name="course"
         style="width: 210px" /><br />
         <input type="button" value="Add Student"
                onclick="var tempStudents = addStudent(
                    document.newStudent.studentName.value,
                    document.newStudent.course.value);
                    document.newStudent.students.value = tempStudents;" />
        </p>
        <p>
         <textarea name="students"></textarea>
        </p>
        </form>
    </body>
</html>
```

Variable Scope

- A **variable scope** is where in your program a declared variable can be used. A variable's scope can be either global or local.
  - A **global variable** is one that is declared outside a function and is available to all parts of your program.
  - A **local variable** is declared inside a function and is only available within the function in which it is declared.
- You must use the var keyword when you declare a local variable. However, the var keyword is optional for declaring a global variable.

Example of using Global and Local Variables

```
<html>
    <head>
        <title>Shipping Charges</title>
        <script type="text/javascript">
            var productPrice = 99.00;
            function applyShipping() {
                shippingCharge = 9.95;
                var totalPrice = productPrice + shippingCharge;
                document.write("<p>The product price is $" + productPrice + "<br />");
                document.write("The shipping charge is $" + shippingCharge + "<br />");
                document.write("The product price plus shipping is $" + totalPrice +
"</p>");
}
        </script>
    </head>
    <body>
        <h1>Shipping Charge</h1>
        <script type="text/javascript">
            applyShipping();
            document.write("<p>The product price is $" + productPrice + "<br />");
            document.write("The shipping charge is $" + shippingCharge + "<br />");
            document.write("The product price plus shipping is $" + totalPrice + "</
p>");
        </script>
    </body>
</html>
```

Shipping Charge

The product price is $99
The shipping charge is $9.95
The product price plus shipping is $108.95

The product price is $99
The shipping charge is $9.95

Common Built-in JavaScript Functions

Function	Description
eval(string)	Evaluates expressions contained within strings
isFinite(number)	Determines whether a number is finite
isNaN(number)	Determines whether a value is the special value NaN (Not a Number)
parseFloat(string)	Converts string literals to floating-point numbers
parseInt(string)	Converts string literals to integers

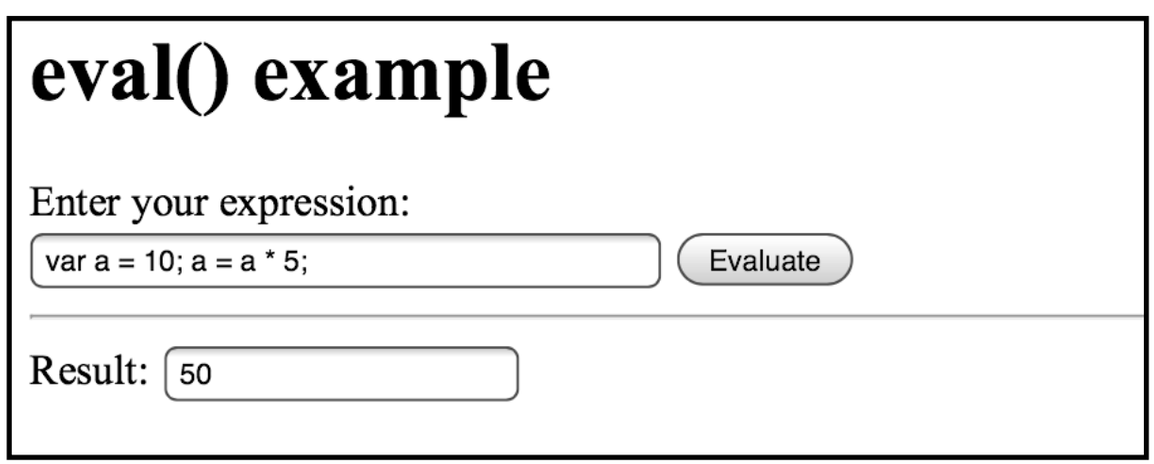
```
<html>
    <head>
        <title>Built-in Functions</title>
    </head>
    <body>
        <h1>Built-in Functions</h1>
        <script type="text/javascript">
            var result = eval("5 + 6");
            document.write("result = " + result + "<br\>");
            var a = 0;
            document.write("isFinite(result / a ) = " + isFinite(result / a) + "<br\>");
            document.write("isNaN(a) = " + isNaN(a) + "<br\>");
            var b = parseFloat("5.55");
            document.write("b = " + b + "<br\>");
            var c = parseInt("5.55");
            document.write("c = " + c + "<br\>")
        </script>
    </body>
</html>
```

Built-in Functions

result = 11
isFinite(result / a) = false
isNaN(a) = false
b = 5.55
c = 5

Example of Using eval()

```
<html>
    <head>
        <title>eval() example</title>
        <script type="text/javascript">
            function execute() {
                //alert("test");
                var exp = document.myForm.expression.value;
                var result = eval(exp);
                document.myForm.result.value = result;
            }
        </script>
    </head>
    <body>
        <h1>eval() example</h1>
        <form method="post" action="" name="myForm">
        Enter your expression: <br />
        <input type="text" name="expression" value="" style="width: 240px" />
        <input type="button" value="Evaluate" onclick="execute()"/>
        <hr />
        Result:
        <input type="text" name="result" value="" />
        </form>
    </body>
</html>
```



Data Types

- A data type is the specific category of information that a variable contains.
- Data types that can be assigned only a single value are called primitive types.
- JavaScript supports the five primitive data types described

Data Types	Description
Number	Positive or negative numbers with or without decimal places, or number written using exponential notation
Boolean	A logical value of true or false
String	Text such as “Hello World”
Undefined	A variable that has never had a value assigned to it, has not been declared, or does not exist
Null	An empty value

Demonstrating undefined and null

```
<html>
    <head>
        <title>Undefined and Null</title>
    </head>
    <body>
        <h1>Undefined and Null</h1>
        <script type="text/javascript">
            var price;
            document.write("<p>The price is $" + price + ".</p>");
            price = 85.2;
            document.write("<p>The price is $" + price + ".</p>");
            price = null;
            document.write("<p>The price is $" + price + ".</p>");
        </script>
    </body>
</html>
```

Undefined and Null

The price is $undefined.

The price is $85.2.

The price is $null.

More on data types

- JavaScript is a **loosely typed programming language**. The JavaScript interpreter automatically determines what type of data is stored in a variable and assigns the variable's data type accordingly.
- An **integer** is a positive or negative number with no decimal places.
  - Range: -9007199254740990 (-253) to 9007199254740990 (253)
- A **floating-point number** is a number that contains decimal places or that is written in exponential notation.
  - For example, the number 200,000,000,000 can be written in exponential notation as 2.0e11
  - Range: ±1.7976931348623157 x 10308 to ± 5 x 10—324.
- A **Boolean** value is a logical value of true or false. In JavaScript, you can only use the words true and false to indicate Boolean values.
  - When you attempt to use a Boolean variable in a mathematical operation, JavaScript converts it to an integer value of 1 or 0.

Strings

- A JavaScript string is a text string contains zero or more characters surrounded by double or single quotation marks.
- Empty strings are valid values for literal strings and are not con- sidered to be null or undefined.
  - var name = "";
- When you need to include a quoted string within a literal string surrounded by double quotation marks, you surround the quoted string with single quotation marks, or vice versa.
  - `document.write("The reserved word is called 'keyword'.");`
  - `document.write('The reserved word is called "keyword".');`
- The concatenation operator (+) is used to combine two strings.

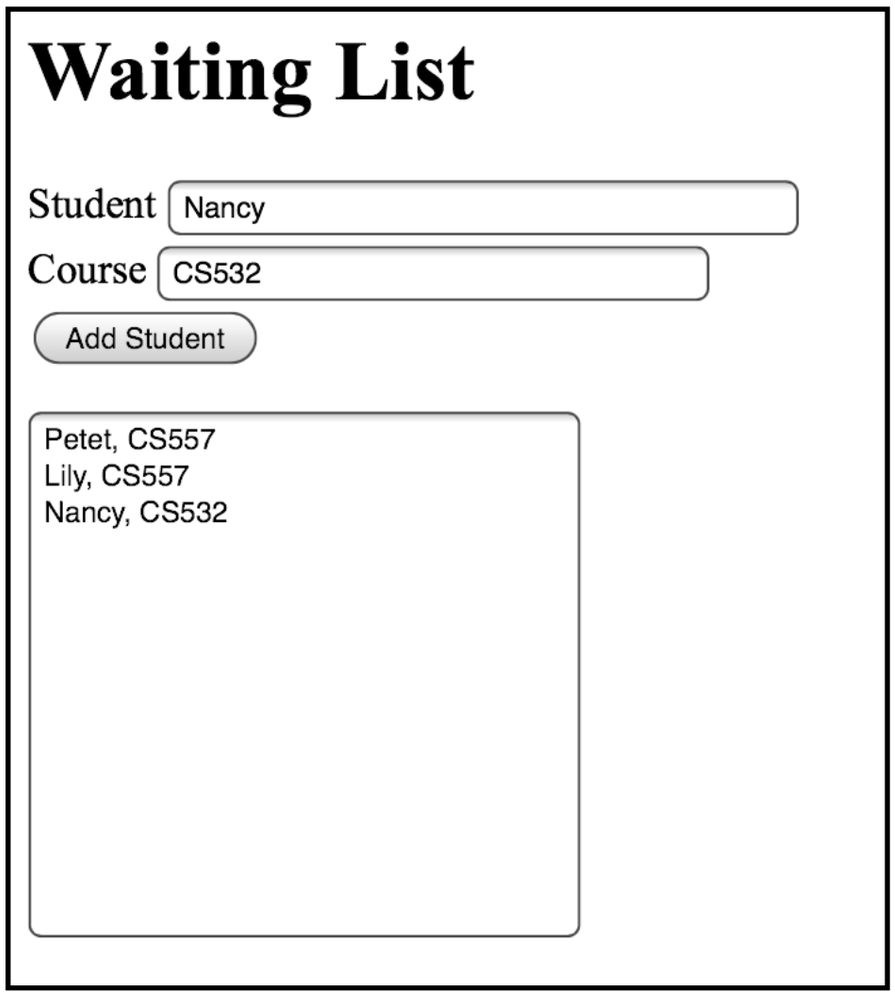
```
var name = "Peter";
var subject = "C++";
var course = name + " is teaching " + subject;
```

- the compound assignment operator (+=) to combine two strings.

```
var name = "Peter";
var subject = "C++";
var subject2 = "Java";
var course = name + " is teaching " + subject;
course += " and " + subject2;
```

Enhanced Waiting List

```
<html>
     <head>
          <title>Waiting List</title>
          <script type="text/javascript">
               function addStudent(name, course){
                    var studentInfo = name + ", ";
                    studentInfo += course + "\r";
                    return studentInfo;
               }
          </script>
     </head>
     <body>
          <h1>Waiting List</h1>
          <form action="" method="post" name="newStudent">
          <p>
           Student
           <input type="text" name="studentName"
           style="width: 240px" /> 
           <br />
           Course
           <input type="var text" name="course"
           style="width: 210px" /><br />
           <input type="button" value="Add Student"
                   onclick="var tempStudents = addStudent(
                        document.newStudent.studentName.value,
                        document.newStudent.course.value);
                        document.newStudent.students.value += tempStudents;" />
          </p>
          <p>
           <textarea style="width: 210px; height: 200px"name="students"></textarea>
          </p>
          </form>
     </body>
</html>
```



Operators

Category	Operators	Description
Arithmetic	addition (+) subtraction (-) multiplication (*) division (/) modulus (%) increment (++)	Used for performing mathematical calculations
Assignment	assignment (=) compound addition assignment (+=) compound subtraction assignment (–=) compound multiplication assignment (*=) compound division assignment (/=) compound modulus assignment (%=)	Assigns values to variables
Comparison	equal (==) strict equal (===) not equal (!=) strict not equal (!==) greater than (>) less than (<) greater than or equal (>=) less than or equal (<=)	Compares operands and returns a Boolean value
Logical	And (&&) Or (||) Not (!)	Used for performing Boolean operations on Boolean operands
String	concatenation operator (+) compound assignment operator (+=)	Performs operations on strings
Special	property access (.) array index ([]) function call (()) comma (,) conditional expression (?:)	Used for various purposes and do not fit within other operator categories

Example of Cost Estimate

```
<html>
    <head>
        <title>Cost Estimate</title>
        <script type="text/javascript">
            var guestsCost = 0;
            var limousinesCost = 0;
            var liveMusicCost = 0;
            var flowersCost = 0;
            var totalEstimate = 0;
            var liveMusic = false;
            var flowers = false;
            function calcGuests() {
                totalEstimate -= guestsCost;
                guestsCost = document.details.numGuests.value * 65;
                totalEstimate += guestsCost;
                document.estimate.cost.value = "$" + totalEstimate;
            }
            function calcLimousines() {
                totalEstimate -= limousinesCost; limousinesCost =
                document.details.numLimousines.value * 65; totalEstimate += limousinesCost;
                document.estimate.cost.value = "$" + totalEstimate;
            }
            function addMusic() {
                (liveMusic == false) ? liveMusicCost = 500 : liveMusicCost = 0; totalEstimate += liveMusicCost;
                liveMusic = true;
                document.estimate.cost.value = "$" + totalEstimate;
            }
            function removeMusic() {
                (liveMusic == true) ? liveMusicCost = -500 : liveMusicCost = 0;
                totalEstimate += liveMusicCost; liveMusic = false;
                document.estimate.cost.value = "$" + totalEstimate;
            }
            function addFlowers() {
                (flowers == false) ? flowersCost = 400 : flowersCost = 0; totalEstimate += flowersCost;
                flowers = true;
                document.estimate.cost.value = "$" + totalEstimate;
            }

            function removeFlowers() {
                (flowers == true) ? flowersCost = -400 : flowersCost = 0;
                totalEstimate += flowersCost; flowers = false;
                document.estimate.cost.value = "$" + totalEstimate;
            }

        </script>
    </head>
```

Example of Cost Estimate

```
<body>
        <h1>Cost Estimate</h1>
        <form method="post" action="" name="details">
        <table>
            <tr>
                <td>Guests<br />($65 each)</td>
                <td><input type="text" name="numGuests" size="3" onchange="calcGuests()" /></td>
            </tr>
            <tr>
                <td>Limousines<br />($125 each)</td>
                <td><input type="text" name="numLimousines" size="3" onchange="calcLimousines()" /></td>
            </tr>
            <tr>
                <td>Live music<br />($500)</td>
                <td>
                    <input type="radio" name="music" onchange="addMusic()" />Yes
                    <input type="radio" name="music" checked="checked" onclick="removeMusic()" />No
                </td>
            </tr>
            <tr>
                <td>Flowers<br />($400)</td>
                <td>
                    <input type="radio" name="flowers" onchange="addFlowers()" />Yes
                    <input type="radio" name="flowers" checked="checked" onchange="removeFlowers()"/>No
                </td>
            </tr>
        </table>
        </form>
        <hr />
        <form method="post" action="" name="estimate">
            <p>
                Estimated total cost:
                <input type="text" name="cost" size="5" value="0" />
            </p>
        </form>
    </body>
</html>
```