JavaScript Arrays

Mohamed Mukthar Ahmed

JavaScript Arrays

Outline

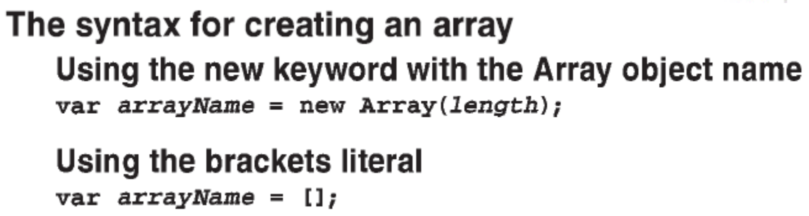
- JavaScript Arrays
- Creating an Array
- Assigning values & Referencing
- Add & Delete Array Elements
- Traversing an Array
- The for-in Loop
- Common Methods of Array object
- Code Analysis

JavaScript Arrays

- An **array** is an object that contains one or more items called **elements**
- Elements can be primitive data type or an object
- The **length** of the array is the number of elements in the array
- One way to create an array is to use the **new** keyword, the name of array object and a set of parentheses that contains a length parameter.
  - If you don't specify the length, array doesn't contain elements
  - If you specify a length, each element is set to undefined
- The other way to create an array is to code a set of brackets with or without a list of elements

JavaScript Arrays

- An **array** is an object that contains one or more items called **elements**
- The other way to create an array is to code a set of brackets with or without a list of elements
- To refer to the elements in an array use an **index**. Indexing start from **zero** (0)



The syntax for creating an array

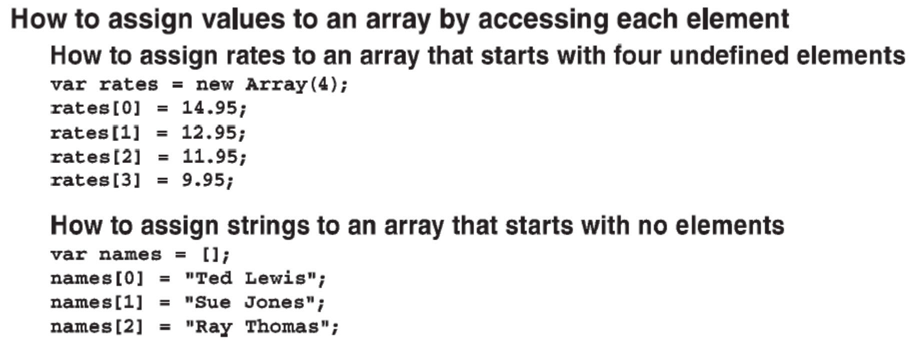
Using the new keyword with the Array object name

```
var arrayName = new Array(length);
```

Using the brackets literal

```
var arrayName = [];
```

JavaScript Arrays



How to assign values to an array by accessing each element

How to assign rates to an array that starts with four undefined elements

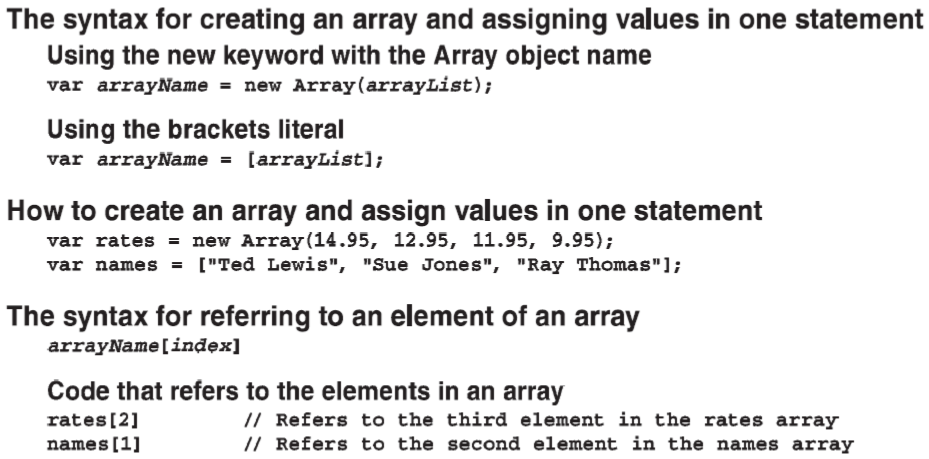
```
var rates = new Array(4);
rates[0] = 14.95;
rates[1] = 12.95;
rates[2] = 11.95;
rates[3] = 9.95;
```

How to assign strings to an array that starts with no elements

```
var names = [];
names[0] = "Ted Lewis";
names[1] = "Sue Jones";
names[2] = "Ray Thomas";
```

Mohamed Mukthar Ahmed

JavaScript Arrays



The syntax for creating an array and assigning values in one statement

Using the new keyword with the Array object name

```
var arrayName = new Array(arrayList);
```

Using the brackets literal

```
var arrayName = [arrayList];
```

How to create an array and assign values in one statement

```
var rates = new Array(14.95, 12.95, 11.95, 9.95);
var names = ["Ted Lewis", "Sue Jones", "Ray Thomas"];
```

The syntax for referring to an element of an array

```
arrayName[index]
```

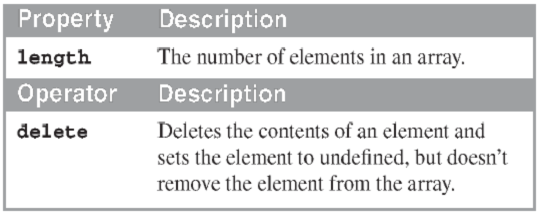
Code that refers to the elements in an array

```
rates[2]        // Refers to the third element in the rates array
names[1]        // Refers to the second element in the names array
```

Mohamed Mukthar Ahmed

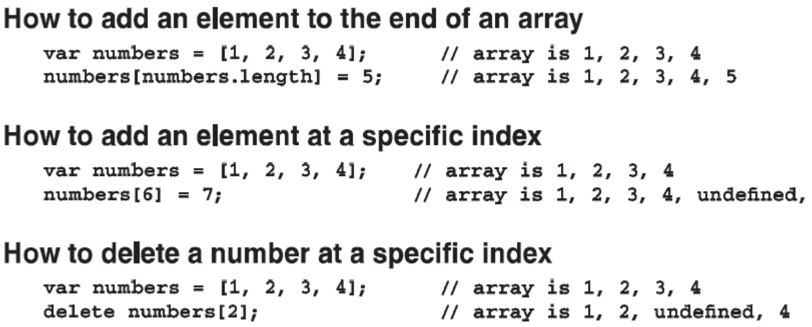
Add & Delete Array Elements

- To add an element to the end of the array, use the length property as index and assign a value
- If you add an element at a specific index that isn't the next one in the sequence, undefined elements are added
- To delete the value of an element in an array use the **delete** operator



Property	Description
`length`	The number of elements in an array.
Operator	**Description**
`delete`	Deletes the contents of an element and sets the element to undefined, but doesn't remove the element from the array.

Add & Delete Array Elements



How to add an element to the end of an array

```
var numbers = [1, 2, 3, 4];       // array is 1, 2, 3, 4
numbers[numbers.length] = 5;      // array is 1, 2, 3, 4, 5
```

How to add an element at a specific index

```
var numbers = [1, 2, 3, 4];     // array is 1, 2, 3, 4
numbers[6] = 7;                 // array is 1, 2, 3, 4, undefined,
```

How to delete a number at a specific index

```
var numbers = [1, 2, 3, 4];       // array is 1, 2, 3, 4
delete numbers[2];                // array is 1, 2, undefined, 4
```

Traversing an Array

- To traverse an array use the **for** loop

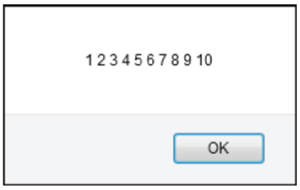
Code that puts the numbers 1 through 10 into an array

```
var numbers = [];
for (var i = 0; i < 10; i++) {
    numbers[i] = i + 1;
}
```

Code that displays the numbers array created above

```
var numbersString = "";
for (var i = 0; i < numbers.length; i++) {
    numbersString += numbers[i] + " ";
}
alert (numbersString);
```

The message that’s displayed



```
1 2 3 4 5 6 7 8 9 10
```

Traversing an Array

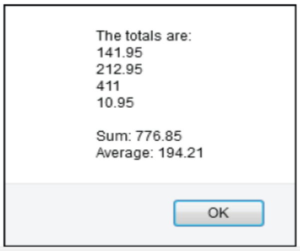
Code that computes the sum and average of an array of totals

```
var totals = [141.95, 212.95, 411, 10.95];
var sum = 0;
for (var i = 0; i < totals.length; i++) {
    sum += totals[i];
}
var average = sum / totals.length;
```

Code that displays the totals array, the sum, and the average

```
var totalsString = "";
for (var i = 0; i < totals.length; i++) {
    totalsString += totals[i] + "\n";
}
alert ("The totals are:\n" + totalsString + "\n" +
       "Sum: " + sum.toFixed(2) + "\n" + "Average: " +
                                      average.toFixed(2) );
```

The message that’s displayed



```
The totals are:
141.95
212.95
411
10.95

Sum: 776.85
Average: 194.21
```

The for-in Loop

- The **for-in** loop also helps us in traversing an array
- Moreover, it access only those elements in an array that are defined

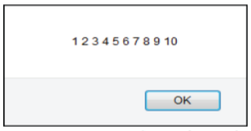
The syntax of a for-in loop

```
for (var elementIndex in arrayName) {
    // statements that access the elements
}
```

A for-in loop that displays the numbers array in a message box

```
var numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
var numbersString = "";
for (var index in numbers) {          // The start of the for-in loop
    numbersString += numbers[index] + " ";
}
alert(numbersString);
```

The message that's displayed

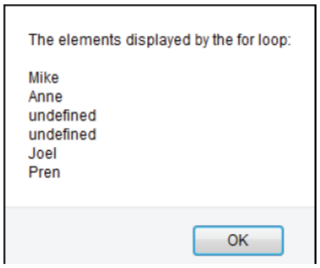


```
1 2 3 4 5 6 7 8 9 10
```

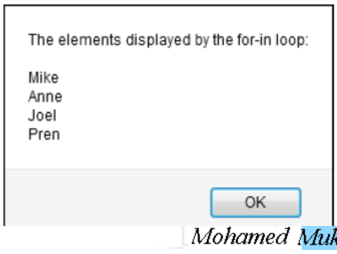
Code that shows the difference between for and for-in loops

```
var names = ["Mike", "Anne", "Ray"];
names[4] = "Joel";                    // Array is Mike, Anne, Ray, undefined, Joel
names[names.length] = "Pren";         // Pren is added to the array
delete names[2];                      // Ray is deleted from the array

var namesString1 = "The elements displayed by the for loop:\n\n";
for (var i = 0; i < names.length; i++) {
    namesString1 += names[i] + "\n"; }    // Includes undefined elements

var namesString2 = "The elements displayed by the for-in loop:\n\n";
for (var i in names) {
    namesString2 += names[i] + "\n"; }    // Omits undefined elements

alert (namesString1);
alert (namesString2);
```

The messages that are created by the for and the for-in loops



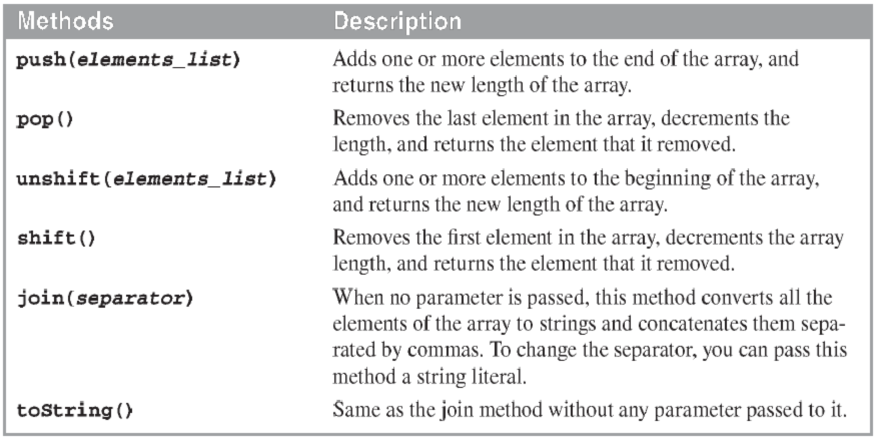
```
The elements displayed by the for loop:

Mike
Anne
undefined
undefined
Joel
Pren
```



```
The elements displayed by the for-in loop:

Mike
Anne
Joel
Pren
```

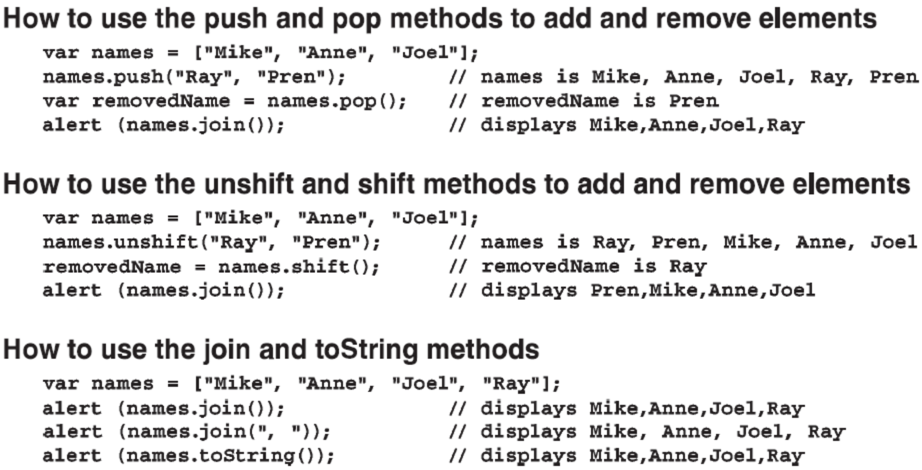
Common Methods of Array Object

- The common methods of an Array object are



Methods	Description
`push(elements_list)`	Adds one or more elements to the end of the array, and returns the new length of the array.
`pop()`	Removes the last element in the array, decrements the length, and returns the element that it removed.
`unshift(elements_list)`	Adds one or more elements to the beginning of the array, and returns the new length of the array.
`shift()`	Removes the first element in the array, decrements the array length, and returns the element that it removed.
`join(separator)`	When no parameter is passed, this method converts all the elements of the array to strings and concatenates them separated by commas. To change the separator, you can pass this method a string literal.
`toString()`	Same as the join method without any parameter passed to it.

Common Methods of Array Object



How to use the push and pop methods to add and remove elements

```
var names = ["Mike", "Anne", "Joel"];
names.push("Ray", "Pren");          // names is Mike, Anne, Joel, Ray, Pren
var removedName = names.pop();      // removedName is Pren
alert (names.join());               // displays Mike,Anne,Joel,Ray
```

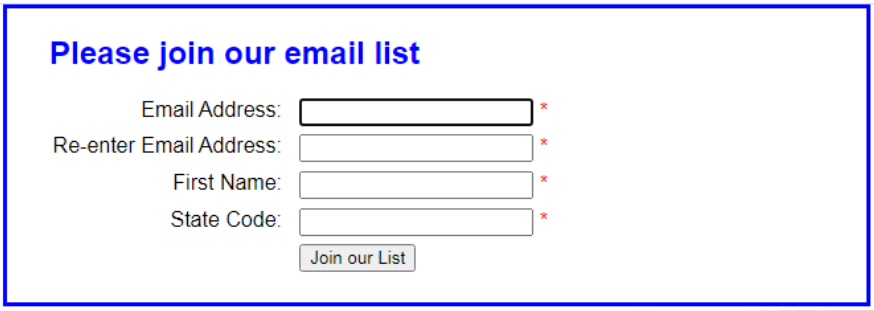
How to use the unshift and shift methods to add and remove elements

```
var names = ["Mike", "Anne", "Joel"];
names.unshift("Ray", "Pren");       // names is Ray, Pren, Mike, Anne, Joel
removedName = names.shift();        // removedName is Ray
alert (names.join());               // displays Pren,Mike,Anne,Joel
```

How to use the join and toString methods

```
var names = ["Mike", "Anne", "Joel", "Ray"];
alert (names.join());               // displays Mike,Anne,Joel,Ray
alert (names.join(", "));           // displays Mike, Anne, Joel, Ray
alert (names.toString());           // displays Mike,Anne,Joel,Ray
```

Email List Application – v2



Please join our email list

Email Address: *

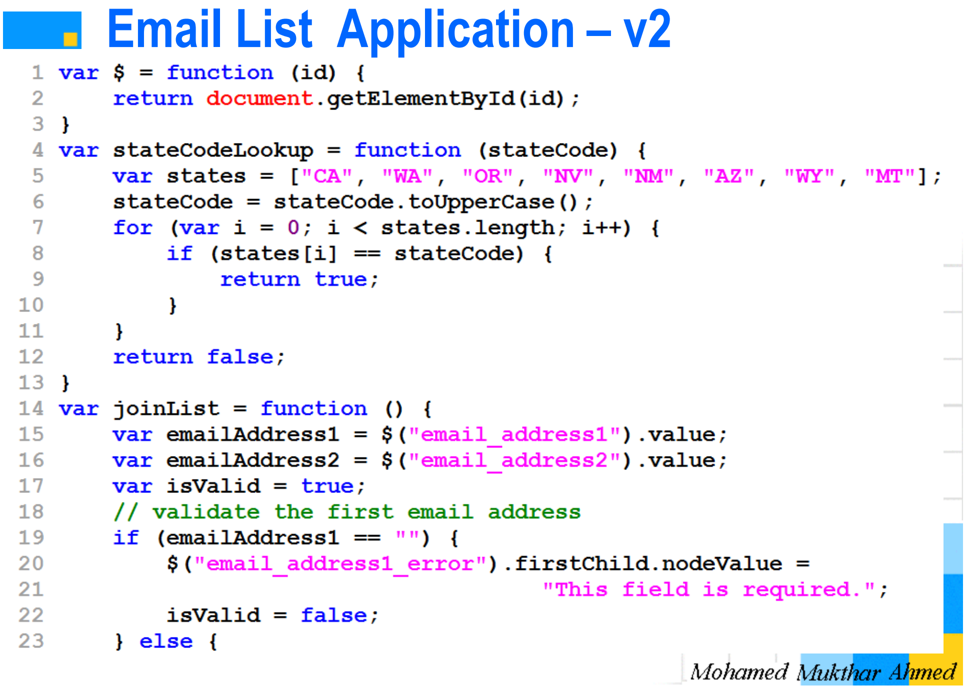
Re-enter Email Address: *

First Name: *

State Code: *

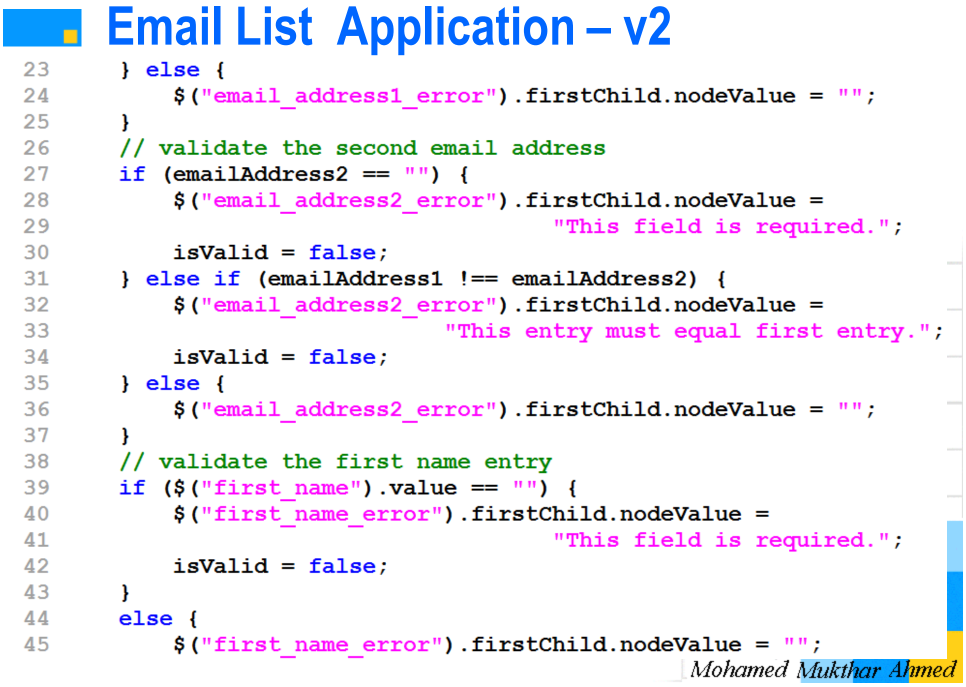
Email List Application – v2

```
<body>
    <section>
        <h1>Please join our email list</h1>
        <form id="email_form" name="email_form" action="join.html" method="get">
            <label for="email_address1">Email Address:</label>
            <input type="text" id="email_address1" name="email_address1">
            <span id="email_address1_error">*</span><br>

            <label for="email_address2">Re-enter Email Address:</label>
            <input type="text" id="email_address2" name="email_address2">
            <span id="email_address2_error">*</span><br>

            <label for="first_name">First Name:</label>
            <input type="text" id="first_name" name="first_name">
            <span id="first_name_error">*</span><br>

            <label for="state_code">State Code:</label>
            <input type="text" id="state_code" name="state_code">
            <span id="state_code_error">*</span><br>

            <label>&nbsp;</label>
            <input type="button" id="join_list" value="Join our List">
        </form>
    </section>
```



Email List Application – v2

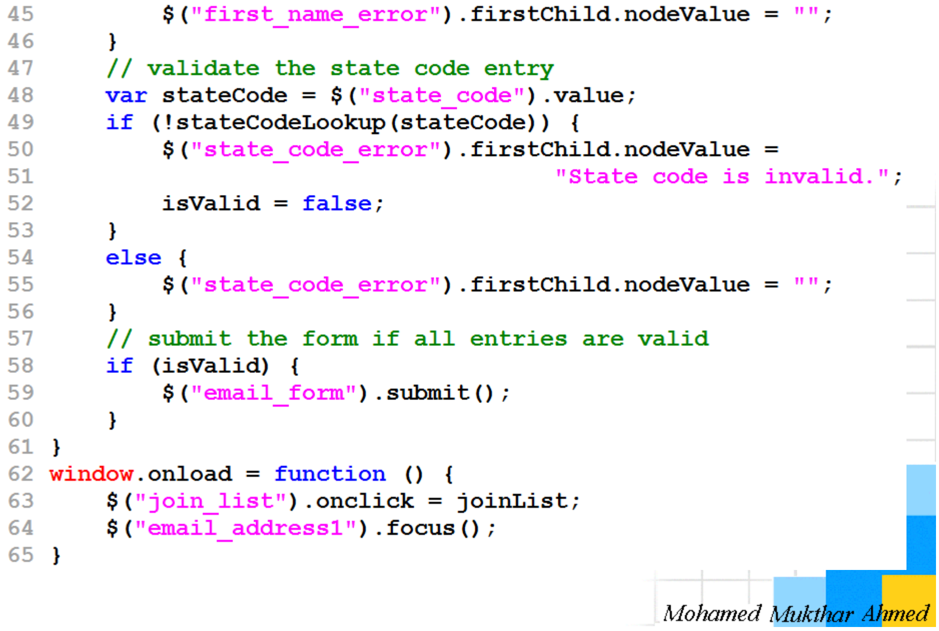
```
var $ = function (id) {
    return document.getElementById(id);
}
var stateCodeLookup = function (stateCode) {
    var states = ["CA", "WA", "OR", "NV", "NM", "AZ", "WY", "MT"];
    stateCode = stateCode.toUpperCase();
    for (var i = 0; i < states.length; i++) {
        if (states[i] == stateCode) {
            return true;
        }
    }
    return false;
}
var joinList = function () {
    var emailAddress1 = $("email_address1").value;
    var emailAddress2 = $("email_address2").value;
    var isValid = true;
    // validate the first email address
    if (emailAddress1 == "") {
        $("email_address1_error").firstChild.nodeValue =
                                    "This field is required.";
        isValid = false;
    } else {
```



Email List Application – v2

```
    } else {
        $("email_address1_error").firstChild.nodeValue = "";
    }
    // validate the second email address
    if (emailAddress2 == "") {
        $("email_address2_error").firstChild.nodeValue =
                                    "This field is required.";
        isValid = false;
    } else if (emailAddress1 !== emailAddress2) {
        $("email_address2_error").firstChild.nodeValue =
                            "This entry must equal first entry.";
        isValid = false;
    } else {
        $("email_address2_error").firstChild.nodeValue = "";
    }
    // validate the first name entry
    if ($("first_name").value == "") {
        $("first_name_error").firstChild.nodeValue =
                                    "This field is required.";
        isValid = false;
    }
    else {
        $("first_name_error").firstChild.nodeValue = "";
```

Email List Application – v2



```
        $("first_name_error").firstChild.nodeValue = "";
    }
    // validate the state code entry
    var stateCode = $("state_code").value;
    if (!stateCodeLookup(stateCode)) {
        $("state_code_error").firstChild.nodeValue =
                                    "State code is invalid.";
        isValid = false;
    }
    else {
        $("state_code_error").firstChild.nodeValue = "";
    }
    // submit the form if all entries are valid
    if (isValid) {
        $("email_form").submit();
    }
}
window.onload = function () {
    $("join_list").onclick = joinList;
    $("email_address1").focus();
}
```