



Arrays:

Indexing a Collection of Items


D.A. Clements



Just a thought...




© Scott Adams, Inc./Dist. by UFS, Inc.




Arrays

- Indexing
 - * Creating and using lists, or arrays
- Processing an array
 - * Element by element
- Array methods
 - * Quick work with lists



Creating and using lists, or arrays


INDEXING



What is an Array?

- An indexed list of items, or elements
 - * Indexed means each element in the list has a number, or index

1. George Washington	14. James Buchanan	28. Warren Harding
2. John Adams	15. Abraham Lincoln	29. Calvin Coolidge
3. Thomas Jefferson	16. Andrew Johnson	30. Herbert Hoover
4. James Madison	17. Ulysses S. Grant	31. Franklin D. Roosevelt
5. James Monroe	18. Rutherford B. Hayes	32. Harry S. Truman
6. John Quincy Adams	19. James Garfield	33. Dwight Eisenhower
7. Andrew Jackson	20. Chester Arthur	34. John Kennedy
8. Martin Van Buren	21. Grover Cleveland	35. Lyndon Johnson
9. William Harrison	22. Benjamin Harrison	36. Richard Nixon
10. John Tyler	23. Grover Cleveland	37. Gerald Ford
11. James Polk	24. William McKinley	38. James Carter
12. Zachary Taylor	25. Theodore Roosevelt	39. Ronald Reagan
13. Millard Fillmore	26. William H. Taft	40. George H. W. Bush
14. Franklin Pierce	27. Woodrow Wilson	41. William Clinton
		42. George W. Bush



Indexing

- Process of creating a sequence of names by associating a base name with a number (like Apollo 13 or Henry VIII)
 - * Each indexed item is called an element of the base-named sequence
- Index Syntax
 - * index number is enclosed in square brackets []
- Iterations can be used to refer to all elements of a name
 - * **A[j]** for successive iterations over **j** referring to different elements of **A**



Indexing (cont'd)

- *Index Origin*
 - * The point at which indexing begins (the least index)
 - * In life, the first element may begin with 1, or have no number (Queen Elizabeth)
 - * JavaScript *always* uses index origin 0



Rules for Arrays

- Arrays are variables initialized by `new Array (<number of elements>);`
- <number of elements> is number of items in ;
- Array indexing begins at 0
- Greatest index is <number of elements> - 1
- Number of elements is array length
- Index values range from 0 to (length - 1)



Array Reference Syntax

- Array reference is array name together with index enclosed in brackets (non-negative integer or expression or variable that resolves to non-negative integer)

`array[i]`



Array Reference Syntax

- The World-Famous Iteration, or 0-origin loop iteration, is perfect for looping through arrays
 - * Start at 0
 - * Increment by 1 to process every element in the array
 - Use the incrementing variable as the index for the array element
 - * End when you reach the last element in the array



Element by element

PROCESSING AN ARRAY



for Loops Rule

- The World-Famous Iteration for looping through an array:


```
for ( i = 0; i < fruits.length; i++ )
{
    alert(fruits[i]);
}
```
- `.length` is a built-in JavaScript property that always gives you the length of an array
 - * Length of an array is the number of elements



Demonstration

- Looping through the fruits array



Processing elements in an array

```
var i, text="";           //declare iteration and other variables
var fruits = new Array(
    'lemons','apples','mangoes','tangerines','kumquats',
    'cantaloupe','peaches','grapefruit','raspberries');
alert("Total number of fruits is " + fruits.length);
for (i=0; i<fruits.length; i++)
{
    text += i + '. ' + fruits[i] + '<br />';
}
document.write("<h1>Elements of Fruits Array:</h1><p>" +
text + "</p>");
```



Array Methods: **.push**

- Add more fruits to the array with **.push**

```
var i, text="";           //declare iteration and other variables
var fruits = new Array(
    'lemons','apples','mangoes','tangerines','kumquats','cantaloupe',
    'peaches','grapefruit','raspberries');
fruits.push('bananas','oranges','pears');
alert("Total number of fruits is " + fruits.length);
for (i=0; i<fruits.length; i++)
{
    text += i + '. ' + fruits[i] + '<br />';
}
document.write("<h1>Elements of Fruits Array:</h1><p>" + text + "</p>");
```



for Loops Rule!

After adding more elements to our array, does our for loop still work?

```
for ( i = 0; i < fruits.length; i++
)
{
    alert(fruits[i]);
}
```

- Yes! **fruits.length** still takes us to the end of the fruits array—whatever its length.



Array Methods: **push**

- Verify it by looping through the expanded fruits array



Quick work with lists

ARRAY METHODS



Array Methods = Possibilities!

- **push**
 - * adds elements to the array
 - fruits.push('bananas','nectarines','apples');
- **pop**
 - * pulls the last element off of the array
 - fruits.pop();
- **concat**
 - * combines several arrays into one
 - * Note: copies of the arrays are used
 - * The original arrays remain and are unaffected
 - fruits.concat(citrus,stoneFruit,berries);



Array Methods = Possibilities!

- **join**
 - * combines all elements into a string, separated by commas or as specified: fruits.join(,);
- **sort**
 - * sorts the elements in the array
 - fruits.sort(); //always ascending
- **reverse**
 - * reverses the elements in an array
 - * Used with sort to sort descending
 - fruits.sort(); //sorts into ascending order
 - fruits.reverse(); //reverses to descending



Array Methods = Possibilities!

- **toString**
 - * converts the array to a string
 - fruits.toString();



Array Method: sort

FIT100 Sort with **.sort**

- * Ascending only (A-Z, 0-9)

```
var i, text=""; //declare iteration and other variables
var fruits = new Array(
    'lemons','apples','mangoes','tangerines','kumquats','cantaloupe',
    'peaches','grapefruit','raspberries');
fruits.push('bananas','oranges','pears');
fruits.sort();
alert("Total number of fruits is " + fruits.length);
for (i=0; i<fruits.length; i++)
{
    text += i + '. ' + fruits[i] + '<br />';
}
document.write("<h1>Elements of Fruits Array:</h1><p>" + text + "</p>");
```



Array Sort

- Demonstration



Sort in Descending Order

FIT100 Reverse the sort with **.reverse**

```
var i, text=""; //declare iteration and other variables
var fruits = new Array(
    'lemons','apples','mangoes','tangerines','kumquats','cantaloupe',
    'peaches','grapefruit','raspberries');
fruits.push('bananas','oranges','pears');
fruits.sort();
fruits.reverse();
alert("Total number of fruits is " + fruits.length);
for (i=0; i<fruits.length; i++)
{
    text += i + '. ' + fruits[i] + '<br />';
}
document.write("<h1>Elements of Fruits Array:</h1><p>" + text + "</p>");
```



Array Method: **reverse**

- Demonstration



End papers...

- Why is programming fun?
 - Second is the pleasure of making things that are useful to other people. Deep within, we want others to use our work and to find it helpful. In this respect the programming system is not essentially different from the child's first clay pencil holder "for Daddy's office."

Source: Frederick P. Brooks, Jr. *The Mythical Man-Month: Essays on Software Engineering*