

INFO 151

Web Systems and Services

Week 6 (T1)

Dr Philip Moore

Dr Zhili Zhao

review

- In this tutorial we will introduce:
 - Making changes to arrays and array elements
 - Arrays and strings
 - Sorting arrays
 - A brief overview of the algorithmic approach the array sorting with a worked example showing how the JavaScript elements combine to create a working program

JavaScript

- In considering JavaScript arrays we have introduced JavaScript:
 - Operators / Operands / Properties / Methods / Functions
 - These JavaScript elements are used to work with arrays
- In this tutorial we will extend JavaScript arrays including:
 - Making changes to arrays and array elements
 - Arrays and strings
 - Sorting arrays
 - A brief overview of the algorithmic approach the array sorting with a worked example showing how the JavaScript elements combine to create a working 'real-world' program

Making Changes to Array Elements

Working with Arrays

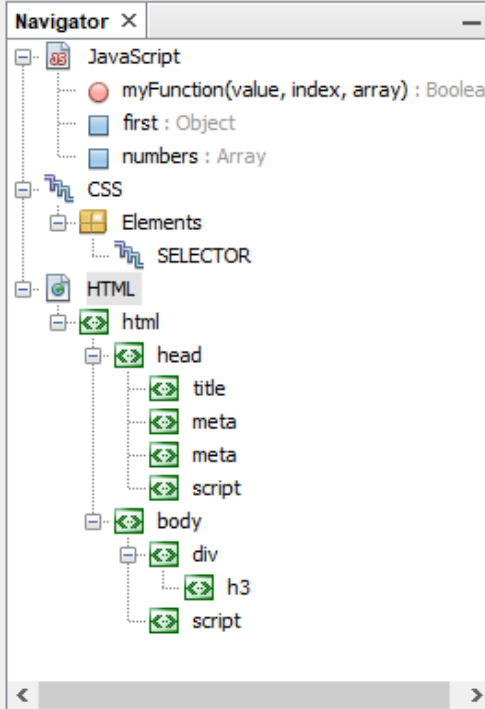
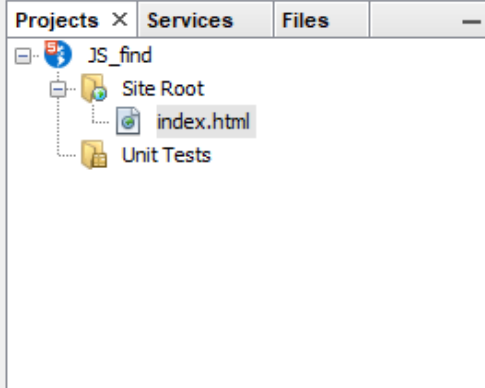
- Testing an array involves searching an array for its properties which commonly include:
 - The nature of the object: is it an *array* or an *object*
 - The *length* of the array
 - The *data values* in an array (remember in JavaScript arrays are *untyped*)
 - The data value of the *first* element
- In this tutorial we will introduce JavaScript array methods
- For example: The following slide shows (**find()**) how to find the first value in an array that passes a test function

Arrays Find

- The **Array.find()** method returns the value of the first array element that passes a test function
- This example finds (returns the value of) the first element that is larger than 18

```
var numbers = [4, 9, 16, 25, 29];  
var first = numbers.find(myFunction);  
function myFunction(value, index, array) {  
  return value > 18;  
}
```

- The **find()** method returns “25”



```
index.html x
Source History
1 <!DOCTYPE html>
2 <!--
3 This JavaScript program implement Array.find() method
4 It returns the value of the first array element that passes a test function
5 The test is to find the first data point over 18 (> 18)
6 -->
7 <html>
8   <head>
9     <title>JavaScript find(*)</title>
10    <meta charset="UTF-8">
11    <meta name="viewport" content="width=device-width, initial-scale=1.0">
12    <script>
13      function myFunction(value, index, array) {
14        return value > 18;
15      }
16    </script>
17  </head>
18  <body>
19    <div style="color:orchid">
20      <h3>JavaScript program to return the result of a test function</h3>
21    </div>
22    <script>
23      var numbers = [4, 9, 16, 25, 29];
24      var first = numbers.find(myFunction);
25      document.write("The first data point > 18 = " + first);
26    </script>
27  </body>
28 </html>
29
30
```

Apache NetBeans 11.1

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

213.0/254.0MB

Projects Services Files

JS_find

- Site Root
- index.html
- Unit Tests

Navigator

- JavaScript
 - myFunction(value, index, array) : Boolean
 - first : Object
 - numbers : Array
- CSS
- Elements
- SELECTION
- HTML
 - html
 - head
 - title
 - meta
 - meta
 - script
 - body
 - div
 - h3
 - script

index.html

Source History

```
1 <!DOCTYPE html>
2 <!--
3 This JavaScript program implement Array.find() method
4 It returns the value of the first array element that passes a test function
5 The test is to find the first data point over 18 (> 18)
6 -->
7 <html>
8   <head>
9     <title>JavaScript find(*)</title>
10    <meta charset="UTF-8">
11    <meta name="viewport" content="width=device-width, initial-scale=1.0">
12    <script>
13      function myFunction(value, index, array) {
14        return value > 18;
15      }
16    </script>
17  </head>
```

JavaScript find(*)

http://localhost:8383/JS_find/index.html

100%

JavaScript program to return the result of a test function

The first data point > 18 = 25

30:1 INS

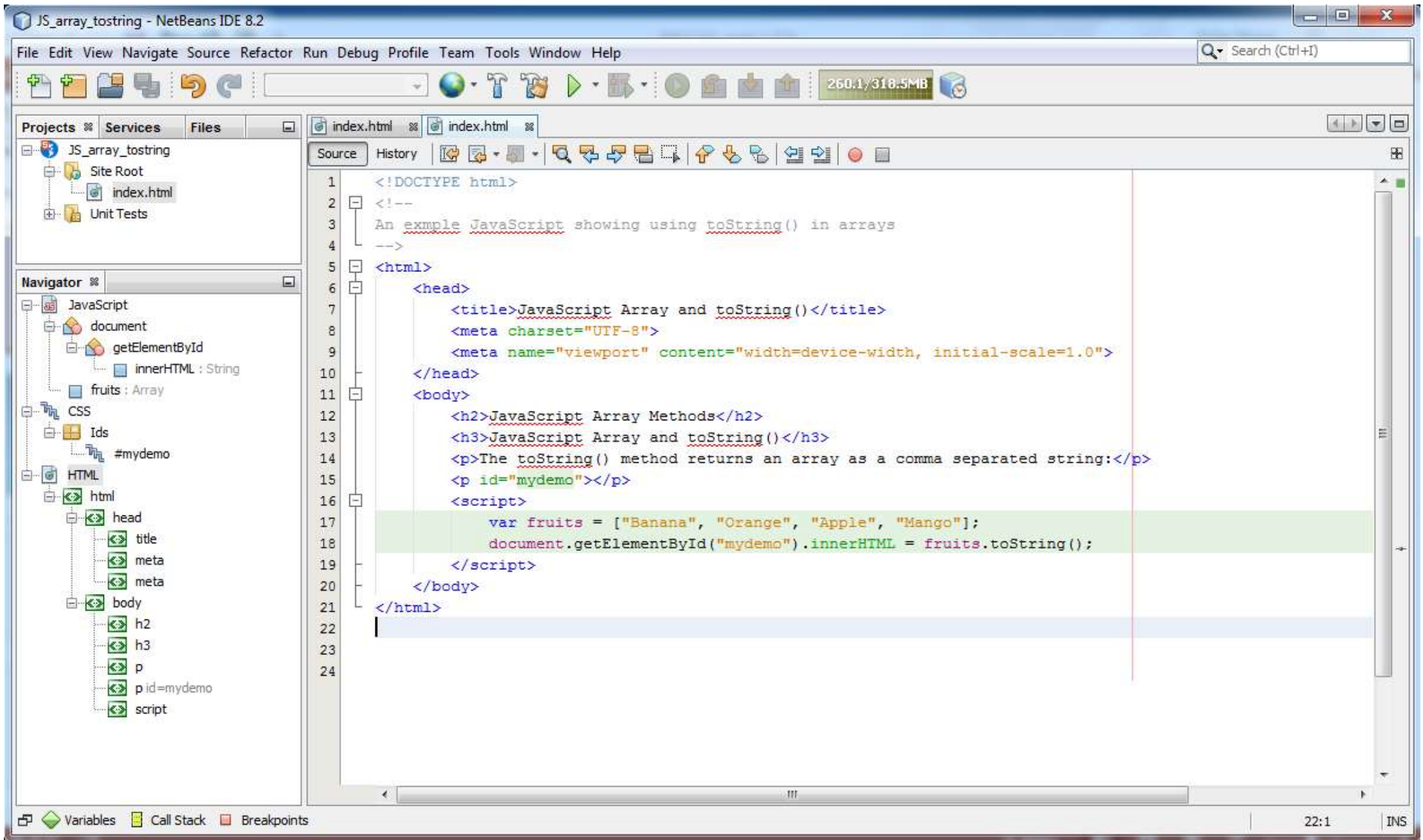
Arrays and Strings

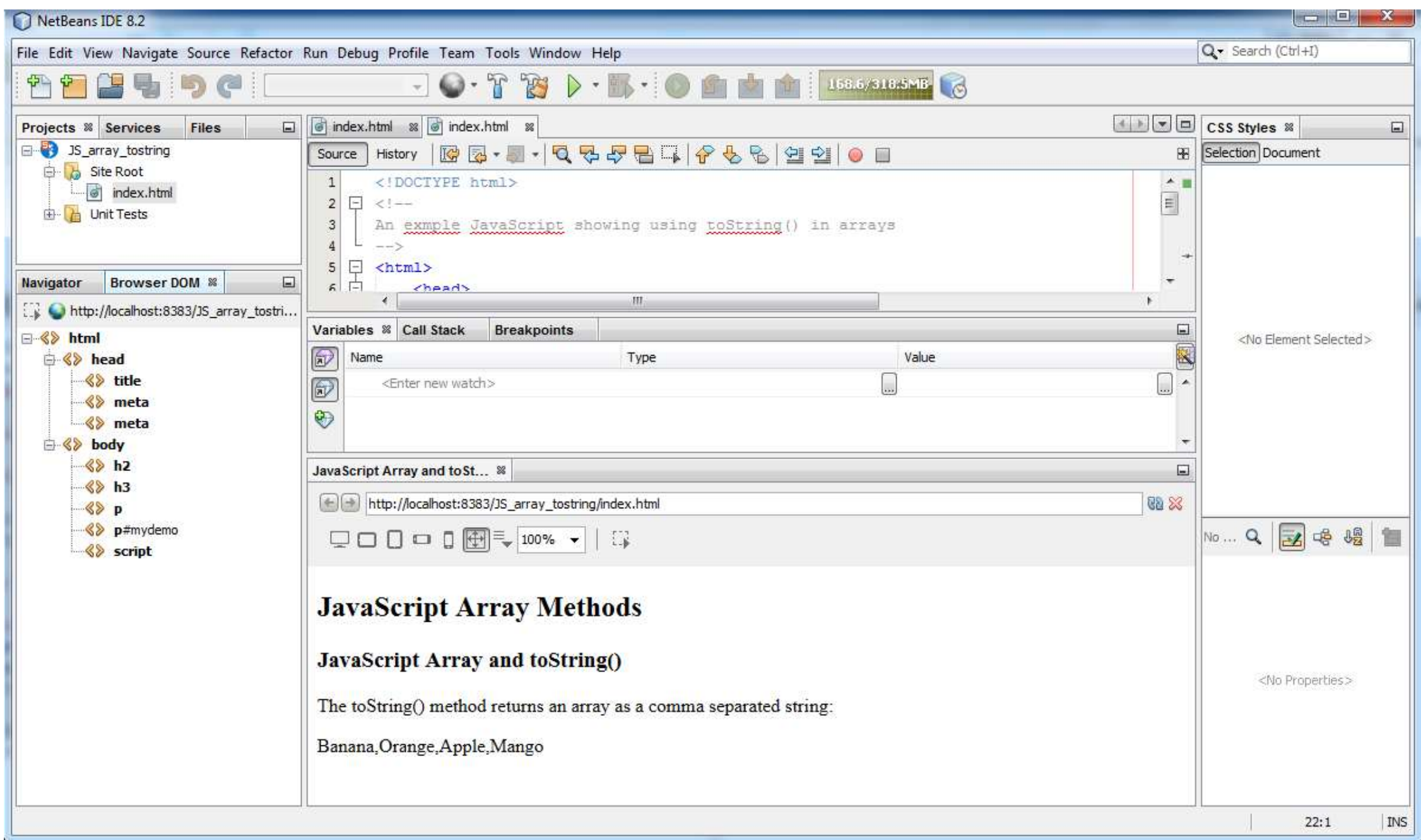
Convert an Array to a Comma Separated String

- JavaScript automatically converts an array to a comma separated string when a primitive value is expected – this is always the case for array output
- The following two examples will produce the same result (output):
 - “Banana,Orange,Apple,Mango” (a comma separated string)

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];  
document.write (fruits.toString());
```

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];  
document.write (fruits);
```





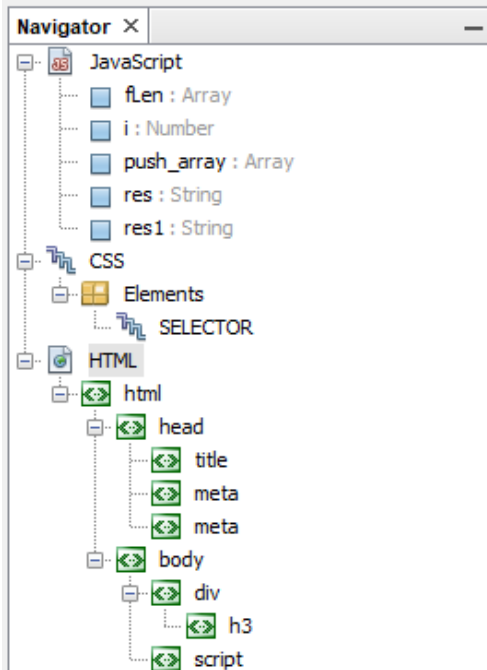
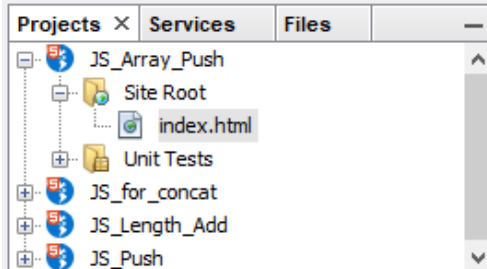
Add

Adding Elements using the **push** Property

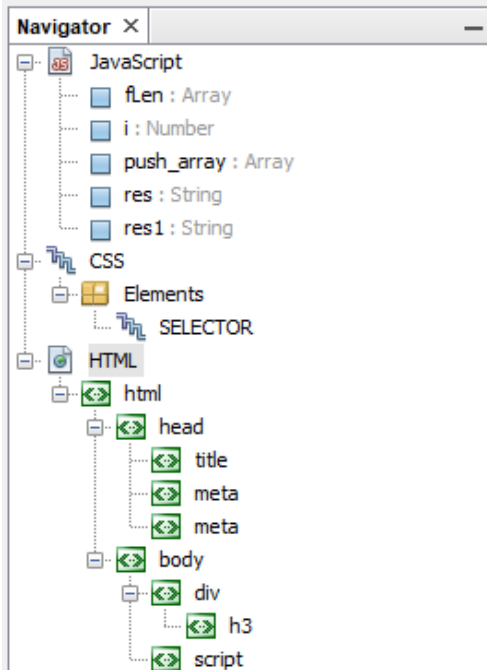
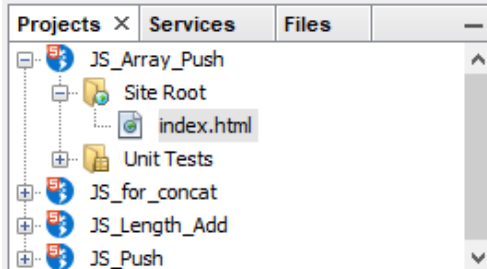
- To add a new element to the end of an existing array
 - We can use the **push** method:
- For example

```
var push_array = ["one", "two", "three", "four"];  
push_array.push("five");
```

- This method adds a new element(s) to the person array
 - We can add multiple elements to an array with this method
 - The following worked example shows the **push** method and output



```
1 <!DOCTYPE html>
2 <!--
3 A JavaScript program to add an element to an existing array
4 The program uses the push() method
5 Author: Philip Moore
6 -->
7 <html>
8   <head>
9     <title>JavaScript push() Method</title>
10    <meta charset="UTF-8">
11    <meta name="viewport" content="width=device-width, initial-scale=1.0">
12  </head>
13  <body>
14    <div style="color:yellowgreen">
15      <h3>JavaScript push() method to add an element to a JavaScript array</h3>
16    </div>
17    <script>
18      var push_array = ["one", "two", "three", "four"];
19      var fLen = push_array.length;
20      var i; var res = ""; var res1 = "";
21      for(i = 0; i < fLen.length; i++) {
22        res += (push_array[i] + ", ");
23      }
24      document.write("The original array: <br>" + res + "<br>");
25      push_array.push("five");
26      push_array[push_array.length] = "five";
27      for(i = 0; i < fLen.length; i++) {
28        res1 += (push_array[i] + ", ");
29      }
30      document.write("The new array: <br>" + res1);
31    </script>
```

```
index.html x index.html x
Source History
5 Author: Philip Moore
6 -->
7 <html>
8   <head>
9     <title>JavaScript push() Method</title>
10    <meta charset="UTF-8">
11    <meta name="viewport" content="width=device-width, initial-scale=1.0">
12  </head>
13  <body>
14    <div style="color:yellowgreen">
15      <h3>JavaScript push() method to add an element to a JavaScript array</h3>
16    </div>
17    <script>
18      var push_array = ["one", "two", "three", "four"];
19      var fLen = push_array.length;
20      var i; var res = ""; var res1 = "";
21      for(i = 0; i < fLen.length; i++) {
22        res += (push_array[i] + ", ");
23      }
24      document.write("The original array: <br>" + res + "<br>");
25      push_array.push("five");
26      push_array[push_array.length] = "five";
27      for(i = 0; i < fLen.length; i++) {
28        res1 += (push_array[i] + ", ");
29      }
30      document.write("The new array: <br>" + res1);
31    </script>
32  </body>
33 </html>
34
35
```


Projects **Services** **Files**

- JS_Array_Push
 - Site Root
 - index.html
 - Unit Tests
- JS_for_concat
- JS_Length_Add
- JS_Push

Navigator

- JavaScript
 - flen : Array
 - i : Number
 - push_array : Array
 - res : String
 - res1 : String
- CSS
 - Elements
 - SELECTOR
- HTML
 - html
 - head
 - title
 - meta
 - meta
 - body
 - div
 - h3
 - script

```

1 <!DOCTYPE html>
2 <!--
3   A JavaScript program to add an element to an existing array
4   The program uses the push() method
5   Author: Philip Moore
6 -->
7 <html>
8   <head>
9     <title>JavaScript push() Method</title>
10    <meta charset="UTF-8">
11    <meta name="viewport" content="width=device-width, initial-scale=1.0">
12  </head>
13  <body>
14    <div style="color:yellowgreen">
15      <h3>JavaScript push() method to add an element to a JavaScript array</h3>
16    </div>

```

JavaScript push() Method

http://localhost:8383/JS_Array_Push/index.html

100%

JavaScript push() method to add an element to a JavaScript array

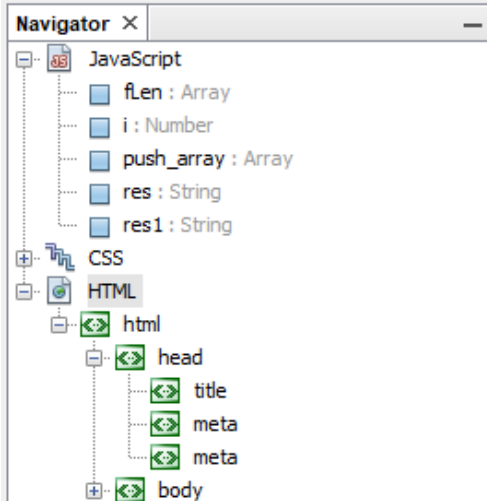
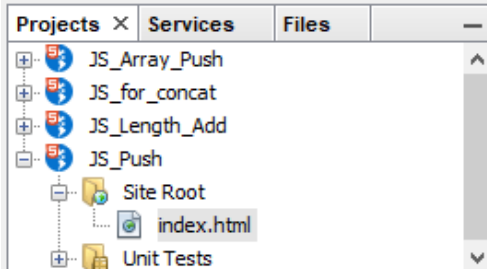
The original array:
one, two, three, four,
The new array:
one, two, three, four, five, five,

Adding Elements using the **length** Property

- The **length** property provides an easy way to append a new element(s) to an array
- For example:

```
var push_array = ["one", "two", "three", "four"];  
push_array[push_array.length]="five";  
push_array[push_array.length]="six";  
push_array[push_array.length]="seven";
```

- The JavaScript appends “five” “six” and “seven” to the end of the array

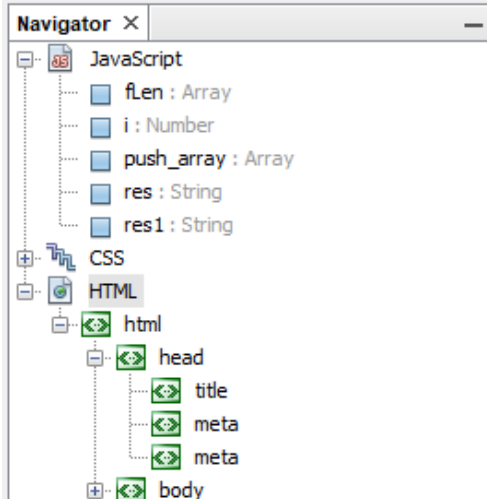
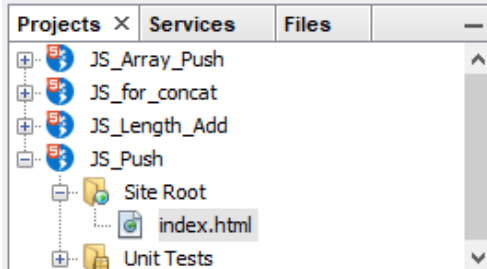


index.html

Source

History

```
1 <!DOCTYPE html>
2 <!--
3 A JavaScript program to add elements to the end of an existing array
4 The program uses the JavaScript length property
5 Author: Philip Moore
6 -->
7 <html>
8   <head>
9     <title>JavaScript length Property</title>
10    <meta charset="UTF-8">
11    <meta name="viewport" content="width=device-width, initial-scale=1.0">
12  </head>
13  <body>
14    <div style="color:yellowgreen">
15      <h3>JavaScript Push</h3>
16    </div>
17    <script>
18      var push_array = ["one", "two", "three", "four"];
19      var fLen = push_array.length;
20      var i; var res = ""; var res1 = "";
21      for(i = 0; i < fLen.length; i++) {
22        res += (push_array[i] + ", ");
23      }
24      document.write("The original array: <br>" + res + "<br>");
25      push_array[push_array.length] = "five";
26      push_array[push_array.length] = "six";
27      push_array[push_array.length] = "seven";
28      for(i = 0; i < fLen.length; i++) {
29        res1 += (push_array[i] + ", ");
30      }
```



```
index.html x
Source History
10      <meta charset="UTF-8">
11      <meta name="viewport" content="width=device-width, initial-scale=1.0">
12  </head>
13  <body>
14      <div style="color:yellowgreen">
15          <h3>JavaScript Push</h3>
16      </div>
17      <script>
18          var push_array = ["one", "two", "three", "four"];
19          var fLen = push_array.length;
20          var i; var res = ""; var res1 = "";
21          for(i = 0; i < fLen.length; i++) {
22              res += (push_array[i] + ", ");
23          }
24          document.write("The original array: <br>" + res + "<br>");
25          push_array[push_array.length] = "five";
26          push_array[push_array.length] = "six";
27          push_array[push_array.length] = "seven";
28          for(i = 0; i < fLen.length; i++) {
29              res1 += (push_array[i] + ", ");
30          }
31          document.write("The new array: <br>" + res1);
32      </script>
33
34  </body>
35 </html>
36
37
```

Projects **Services** **Files**

- JS_Array_Push
- JS_for_concat
- JS_Length_Add
- JS_Push
 - Site Root
 - index.html
 - Unit Tests

Navigator

- JavaScript
 - flen : Array
 - i : Number
 - push_array : Array
 - res : String
 - res1 : String
- CSS
- HTML
 - html
 - head
 - title
 - meta
 - meta
 - body

index.html

```

4  program uses the JavaScript length property
5  or: Philip Moore
6
7  </>
8  <head>
9      <title>JavaScript length Property</title>
10     <meta charset="UTF-8">
11     <meta name="viewport" content="width=device-width, initial-scale=1.0">
12 </head>
13 <body>
14     <div style="color:yellowgreen">
15         <h3>JavaScript Push</h3>
16     </div>
17     <script>
18         var push_array = ["one", "two", "three", "four"];
    
```

JavaScript length Propert...

http://localhost:8383/JS_Push/index.html

100%

JavaScript Push

The original array:
one, two, three, four,
The new array:
one, two, three, four, five, six, seven,

Problems in Adding Array Elements

- Adding elements with high indexes can create undefined *holes* in an array:
 - For example the following JavaScript code adds a new element [6] and the data value “six” to the numbers array

```
var numbers = ["one", "two", "three", "four"];  
numbers[6] = "six";
```
 - However: the existing array has **4 elements**
 - We have assigned “six” to numbers[6] (it should be numbers[5])
- In high-level programming languages (such as Java)
 - This would ‘**throw an array out of bounds exception**’
 - In JavaScript it will not identify the error but the program logic will be wrong

Delete

Delete and Array Element

- As JavaScript arrays are objects, elements can be deleted by using the JavaScript operator **delete**

- For example

```
var numbers = ["one", "two", "three", "four"];  
delete numbers[0];
```

- This changes the first element in **numbers** to **undefined**
- Using **delete** can result in errors because:
 - The **delete** method is designed to free up system memory rather than to adjust array sizes

Delete and Array Element

- There are better alternatives:
 - `pop()`
 - `splice()`
 - `List.splice()`
 - `Shift()`
 - `Filter()`
- The alternative methods all address different requirements
- I have provided supplementary a resource setting out details of the alternative methods and their design uses

Delete and Array Element

- The alternative methods and their uses are:
 - **splice()** remove specific elements)
 - **pop()** remove elements from the end of an array
 - **shift()** remove elements from the start of an array
 - Use **delete** to remove individual array objects
- We may also:
 - Find and remove an element of a specific value
 - Find and remove multiple elements with the same value
 - Remove elements by filtering an array

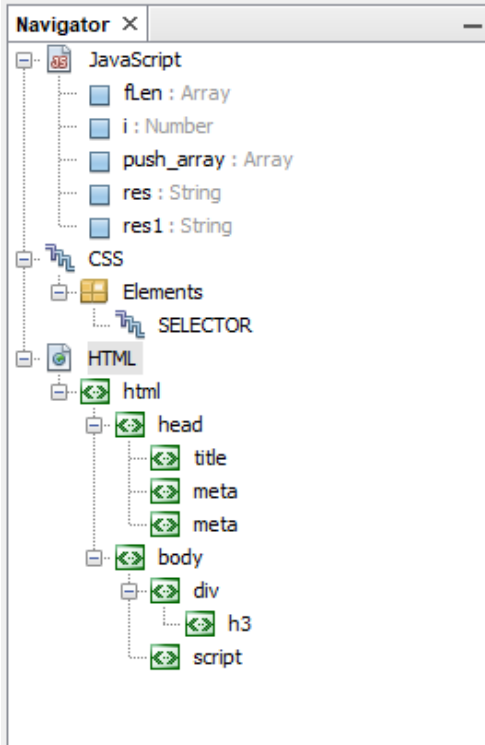
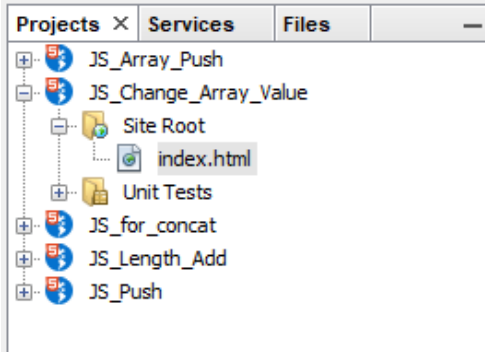
Changing Values

Changing Array Elements

- A frequent activity is to update array element value(s)
- Note: remember that JavaScript is not typed so the datatype may change resulting in a possible logic error
- The following example changes the first element of numbers[0]) from “one” to “zero”:

```
var numbers = ["one", "two", "three", "four"];  
numbers[0] = "zero";
```

- The following worked example shows the changes

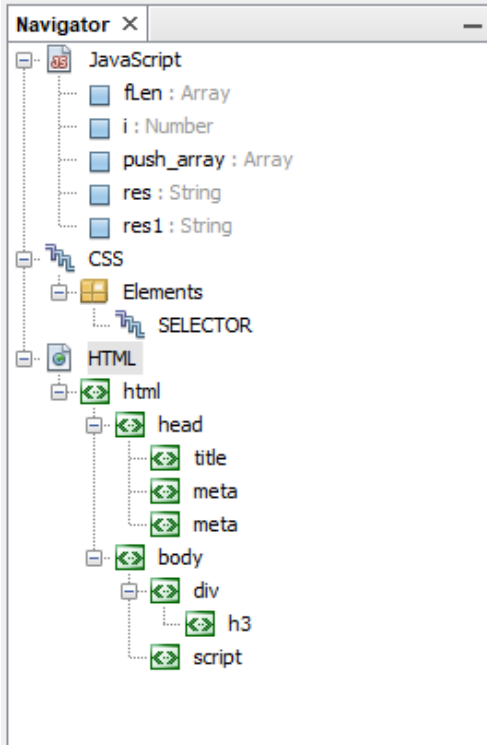
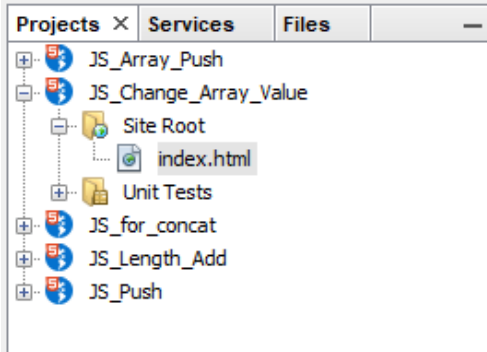


index.html x

Source

History

```
1 <!DOCTYPE html>
2 <!--
3 A JavaScript program to change the value of an element in an existing array
4 The program uses the push_array[0] = "zero"; approach
5 Author: Philip Moore
6 -->
7 <html>
8   <head>
9     <title>JavaScript push() Method</title>
10    <meta charset="UTF-8">
11    <meta name="viewport" content="width=device-width, initial-scale=1.0">
12  </head>
13  <body>
14    <div style="color:yellowgreen">
15      <h3>JavaScript push() method to add an element to a JavaScript array</h3>
16    </div>
17    <script>
18      var push_array = ["one", "two", "three", "four"];
19      var fLen = push_array.length;
20      var i; var res = ""; var res1 = "";
21      for(i = 0; i < fLen.length; i++) {
22        res += (push_array[i] + ", ");
23      }
24      document.write("The original array: <br>" + res + "<br>");
25      push_array[0] = "zero"; //change the value of the element
26      for(i = 0; i < fLen.length; i++) {
27        res1 += (push_array[i] + ", ");
28      }
29      document.write("The new array: <br>" + res1);
30    </script>
31  </body>
```



index.html x

Source

History

```
4 The program uses the push_array[0] = "zero"; approach
5 Author: Philip Moore
6 -->
7 <html>
8   <head>
9     <title>JavaScript push() Method</title>
10    <meta charset="UTF-8">
11    <meta name="viewport" content="width=device-width, initial-scale=1.0">
12  </head>
13  <body>
14    <div style="color:yellowgreen">
15      <h3>JavaScript push() method to add an element to a JavaScript array</h3>
16    </div>
17    <script>
18      var push_array = ["one", "two", "three", "four"];
19      var fLen = push_array;
20      var i; var res = ""; var res1 = "";
21      for(i = 0; i < fLen.length; i++) {
22        res += (push_array[i] + ", ");
23      }
24      document.write("The original array: <br>" + res + "<br>");
25      push_array[0] = "zero"; //change the value of the element
26      for(i = 0; i < fLen.length; i++) {
27        res1 += (push_array[i] + ", ");
28      }
29      document.write("The new array: <br>" + res1);
30    </script>
31  </body>
32 </html>
33
34
```

Projects **Services** **Files**

- JS_Array_Push
- JS_Change_Array_Value
 - Site Root
 - index.html
 - Unit Tests
- JS_for_concat
- JS_Length_Add
- JS_Push

Navigator

- JavaScript
 - flen : Array
 - i : Number
 - push_array : Array
 - res : String
 - res1 : String
- CSS
- Elements
 - SELECTOR
- HTML
 - html
 - head
 - title
 - meta
 - meta
 - body
 - div
 - h3
 - script

index.html x index.html x

```

1  <!DOCTYPE html>
2  <!--
3      A JavaScript program to change the value of an element in an existing array
4      The program uses the push_array[0] = "zero"; approach
5      Author: Philip Moore
6  -->
7  <html>
8      <head>
9          <title>JavaScript push() Method</title>
10         <meta charset="UTF-8">
11         <meta name="viewport" content="width=device-width, initial-scale=1.0">
12     </head>
13     <body>
14         <div style="color:yellowgreen">
15             <h3>JavaScript push() method to add an element to a JavaScript array</h3>
16         </div>

```

JavaScript push() Method x

http://localhost:8383/JS_Change_Array_Value/index.html

100%

JavaScript push() method to add an element to a JavaScript array

The original array:
one, two, three, four,
The new array:
zero, two, three, four,

JavaScript Program Code Structure

- We have introduced the methods to manage arrays by:
 - **Adding** elements and data values to arrays
 - **Deleting** elements and data values to arrays
 - **Changing** the data values in array elements
- The JavaScript program code structure:
 - Will follow be similar for all the tasks as will be the method of accessing arrays as shown in the previous worked examples
 - However: you must use the **appropriate methods** to make additions, deletions, and changes to arrays and array elements

Sorting Arrays

Sorting Arrays

- Sorting the elements of a JavaScript array is a frequent operation
- Arrays can be sorted in a number of ways which include:
 - *Numeric* and *alphabetical* sort
 - *Ascending* and *descending* sort
 - *Object* array sort
 - Sorting an array in *Random* order and *reversing* the order
 - Find the *highest* (max) (or) *lowest* array element value
- We will show the basic approach with the following examples
- Comprehensive details for array sort methods may be found in the course resources

JavaScript Array Properties and Methods

- In week 1 we introduced programming ‘frameworks’
- A strength of JavaScript (and Java) is the built in pre-defined API’s which implement properties and methods
- JavaScript arrays have such properties and methods
- For example:

```
var y = cars.sort();
```

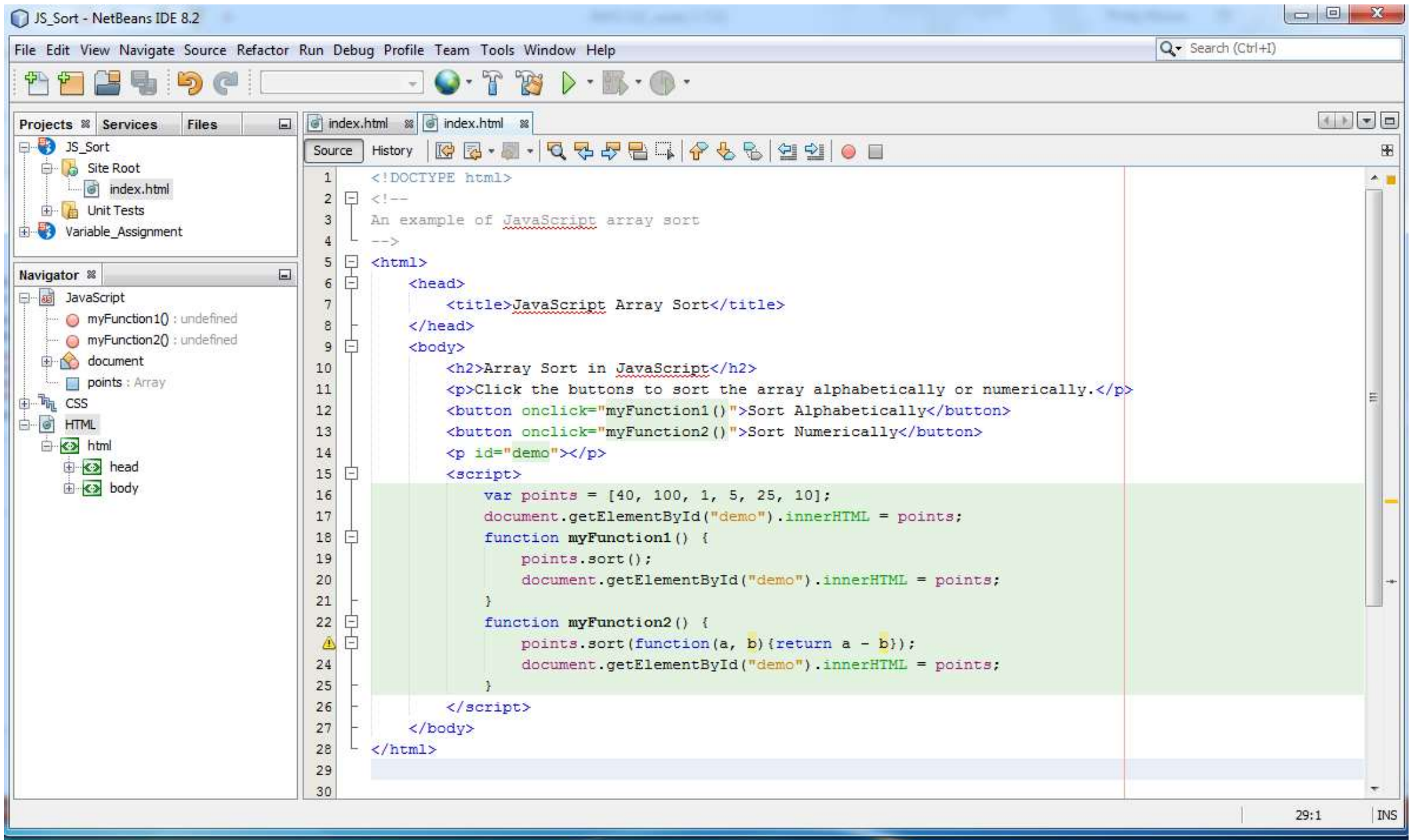
- The **sort()** method sorts arrays

Numeric Sort

Numeric Sort

- By default, the **sort()** function sorts values as **strings**
- This works well for strings ("Apple" comes before "Banana")
 - However, if numbers are sorted as **strings**, "25" is bigger than "100" because "2" is larger than "1".
- Because of this the **sort()** method will produce incorrect result when sorting numbers (the result is an alphabetical sort)
 - This can be corrected by providing a **compare function** as follows:

```
var points = [40, 100, 1, 5, 25, 10];  
points.sort(function(a, b){return a - b});
```



NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+I)

161.2/309.0MB

Projects Services Files

- JS_array_sort_asc
 - Site Root
 - index.html
 - Unit Tests
- JS_Find_High
 - Site Root

Navigator Browser DOM

- http://localhost:8383/JS_array_sort_...
 - html
 - head
 - body
 - h2
 - p
 - button
 - p#demo
 - script

Source History

```
1 <!DOCTYPE html>
2 <!--
3   An example JavaScript program to sort an array in ascending order
4 -->
```

Variables Call Stack Breakpoints

Name	Type	Value
<Enter new watch>		

JavaScript Array Sort

http://localhost:8383/JS_array_sort_asc/index.html

100%

Sort a JavaScript array in ascending order

Click the button to sort the array in ascending order.

Try it

40,100,1,5,25,10

Original numbers

CSS Styles

<No Element Selected>

<No Properties>

28:1 INS

NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+I)

226.2/329.5MB

Projects Services Files

- JS_Sort
 - Site Root
 - index.html
 - Unit Tests
 - Variable_Assignment

Navigator Browser DOM

http://localhost:8383/JS_Sort/index....

- html
 - head
 - title
 - body
 - h2
 - p
 - button
 - button
 - p#demo
 - script

Source History

```
1 <!DOCTYPE html>
2 <!--
3   An example of JavaScript array sort
4 -->
5 <html>
6   <head>
7     <title>JavaScript Array Sort</title>
```

Variables Call Stack Breakpoints

Name	Type	Value
<Enter new watch>		

JavaScript Array Sort

http://localhost:8383/JS_Sort/index.html

100%

Array Sort in JavaScript

Click the buttons to sort the array alphabetically or numerically.

Sort Alphabetically Sort Numerically

1,5,10,25,40,100

Sorted numerically

CSS Styles Selection Document

<No Element Selected>

No ...

<No Properties>

29:1 INS

Alphabetical Sort

Alphabetical Sort (strings)

- The **sort()** method sorts an array alphabetically:
- For example

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits.sort();
```

- Sorts the elements of fruits as follows

Apple, Banana, Mango, Orange

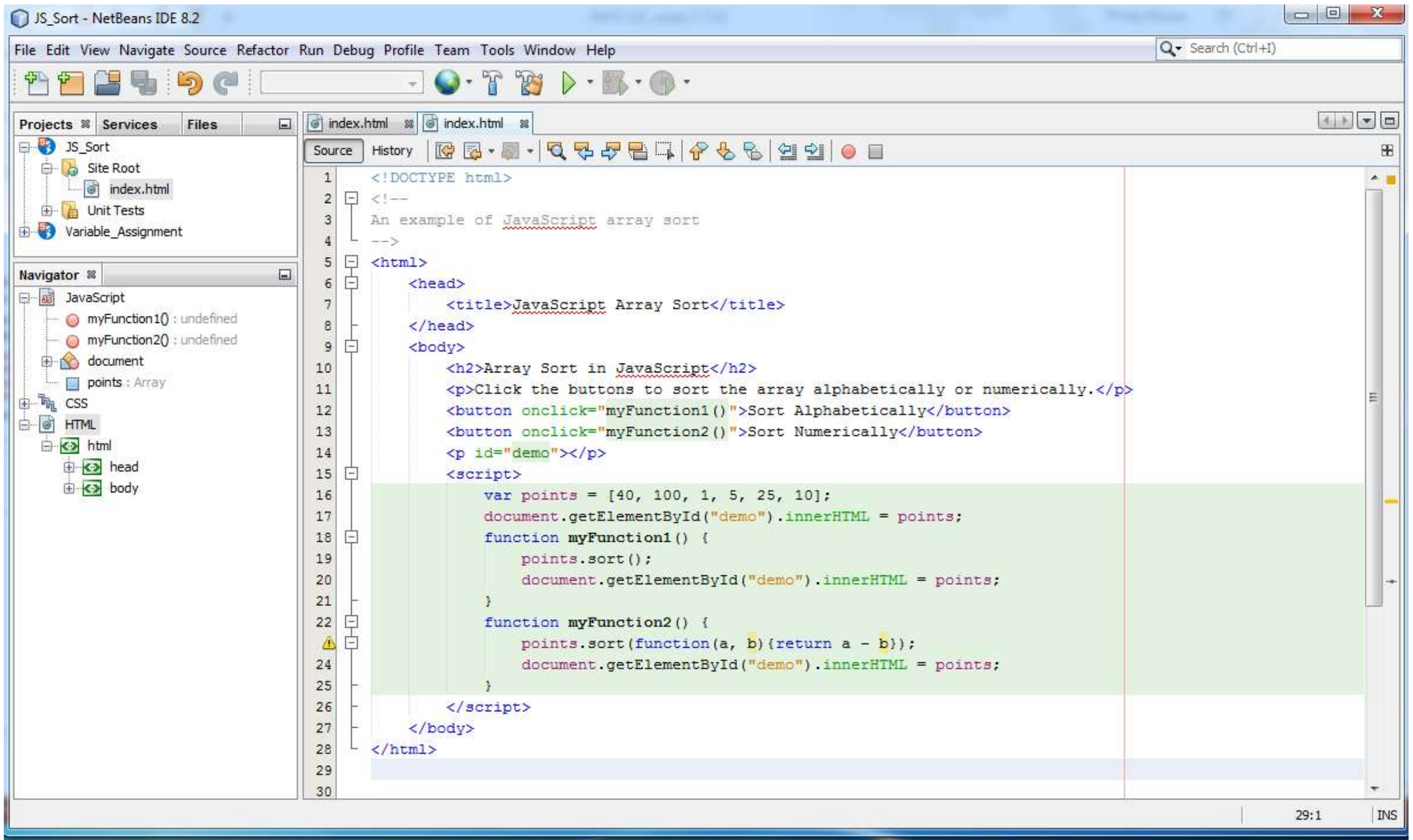
- The sort process is as expected (i.e., a, b, c, etc)
 - For numbers the process can present problems

Alphabetical Sort (numbers)

- The **sort()** method sorts an array alphabetically:
- For example consider the points array:

```
var points = [40,100,1,5,25,10];  
points.sort();
```

- Sorts the elements of the points array as follows
 - 1, 10, 100, 25, 40, 5
- The following example shows a worked an example to sort an array alphabetically



NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+I)

205.7/334.5Mb

Projects Services Files

- JS_Sort
 - Site Root
 - index.html
 - Unit Tests
 - Variable_Assignment

Navigator Browser DOM

http://localhost:8383/JS_Sort/index....

- html
 - head
 - title
 - body
 - h2
 - p
 - button
 - button
 - p#demo
 - script

index.html

```
1 <!DOCTYPE html>
2 <!--
3 An example of JavaScript array sort
4 -->
5 <html>
6 <head>
7 <title>JavaScript Array Sort</title>
```

Variables Call Stack Breakpoints

Name	Type	Value
<Enter new watch>		

JavaScript Array Sort

http://localhost:8383/JS_Sort/index.html

100%

Array Sort in JavaScript

Click the buttons to sort the array alphabetically or numerically.

Sort Alphabetically Sort Numerically

1,10,100,25,40,5

Sorted alphabetically

29:1 INS

Ascending Sort

Sort in Ascending Order

- Array sort in ascending order ()

```
var points = [40, 100, 1, 5, 25, 10];  
points.sort(function(a, b){return a - b});
```

- Following the array sort **points[0]** contains the highest value
- The following slides show a worked example of array sort in ascending order



JS_array_sort_asc - NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+I)

Projects Services Files

JS_array_sort_asc

- Site Root
- index.html
- Unit Tests

JS_Find_High

- Site Root

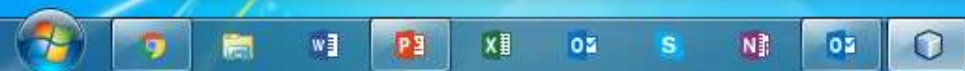
Navigator

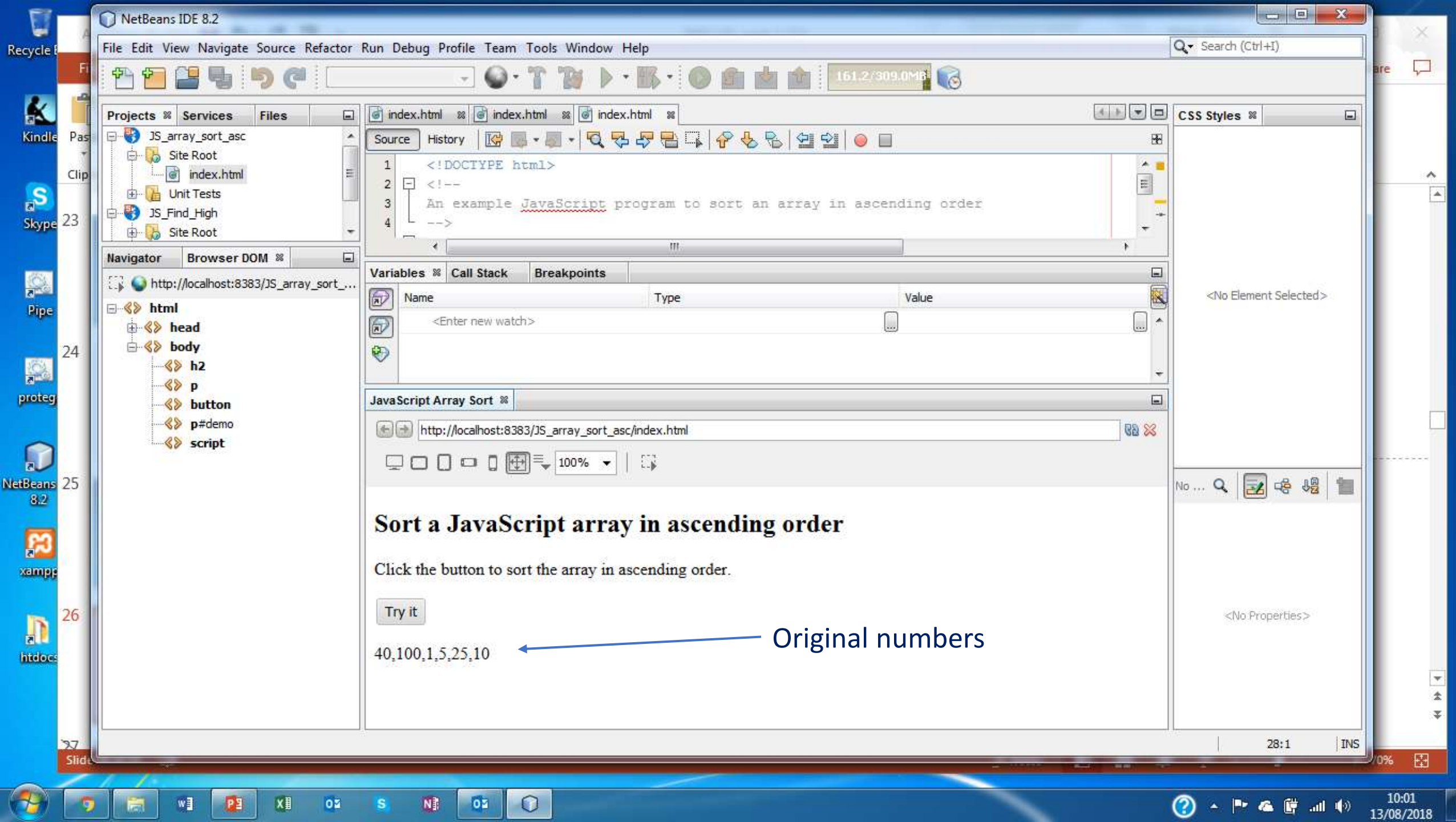
- JavaScript
 - myFunction(): undefined
 - document
 - getElementById
 - innerHTML: Array
 - points: Array
- CSS
 - Ids
 - #demo
- HTML
 - html
 - head
 - title
 - meta
 - meta
 - body
 - h2
 - p
 - button
 - p id=demo
 - script

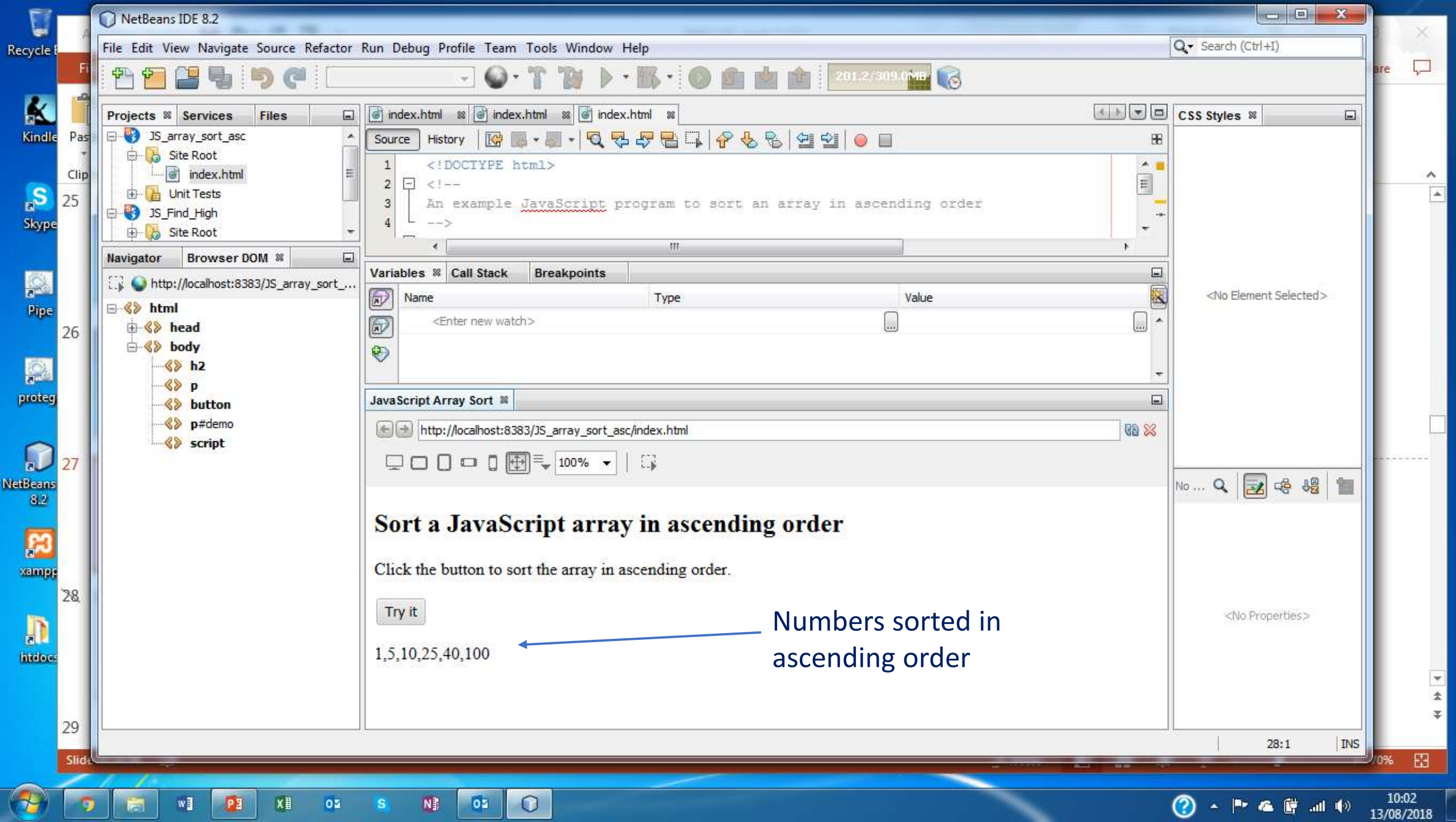
index.html

```
1 <!DOCTYPE html>
2 <!--
3   An example JavaScript program to sort an array in ascending order
4 -->
5 <html>
6   <head>
7     <title>JavaScript Array Sort</title>
8     <meta charset="UTF-8">
9     <meta name="viewport" content="width=device-width, initial-scale=1.0">
10  </head>
11  <body>
12    <h2>Sort a JavaScript array in ascending order</h2>
13    <p>Click the button to sort the array in ascending order.</p>
14    <button onclick="myFunction()">Try it</button>
15    <p id="demo"></p>
16    <script>
17      var points = [40, 100, 1, 5, 25, 10];
18      document.getElementById("demo").innerHTML = points;
19      function myFunction() {
20        points.sort(function(a, b){return a - b});
21        document.getElementById("demo").innerHTML = points;
22      }
23    </script>
24  </body>
25 </html>
```

28:1 INS







Descending Sort

Sort in Descending Order

- Array sort descending order ()

```
var points = [40, 100, 1, 5, 25, 10];  
points.sort(function(a, b){return b - a});
```

- Following the array sort **points[0]** contains the lowest value
- and [**points.length-1**] contains the highest value
- The following slides show a worked example of array sort in descending order

NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+I)

235.9/319.0MB

Projects Services Files

- JS_array_sort_asc
- JS_array_sort_desc
 - Site Root
 - index.html
 - Unit Tests
- JS_Find_High
- JS_Find_Low

Navigator Browser DOM

http://localhost:8383/JS_array_sort_...

html

- head
 - title
 - meta
 - meta
- body
 - h2
 - p
 - button
 - p#demo
 - script

index.html

```
1 <!DOCTYPE html>
2 <!--
3 An example JavaScript file showing array sort in descending order
4 -->
5 <html>
6 <head>
```

Variables Call Stack Breakpoints

Name	Type	Value
<Enter new watch>		

TODO supply a title

http://localhost:8383/JS_array_sort_desc/index.html

100%

JavaScript array sort in descending order

Click the button to sort the array in descending order.

Try it

40,100,1,5,25,10

CSS Styles

Selection Document

<No Element Selected>

No ...

<No Properties>

3:66 INS

NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+I)

280.4/319.0MB

Projects Services Files

- JS_array_sort_asc
- JS_array_sort_desc
 - Site Root
 - index.html
 - Unit Tests
- JS_Find_High
- JS_Find_Low

Navigator Browser DOM

http://localhost:8383/JS_array_sort_...

- html
 - head
 - title
 - meta
 - meta
 - body
 - h2
 - p
 - button
 - p#demo
 - script

index.html

```
1 <!DOCTYPE html>
2 <!--
3 An example JavaScript file showing array sort in descending order
4 -->
5 <html>
6 <head>
```

Variables Call Stack Breakpoints

Name	Type	Value
<Enter new watch>		

TODO supply a title

http://localhost:8383/JS_array_sort_desc/index.html

100%

JavaScript array sort in descending order

Click the button to sort the array in descending order.

Try it

100,40,25,10,5,1

Numbers sorted in descending order

CSS Styles Selection Document

<No Element Selected>

No ...

<No Properties>

3:66 INS

Bubble Sort

Bubble Sort

- The previous slides have shown the methods built into JavaScript to sort arrays
 - However: the worked example show the output is correct but the methodology `points.sort(function(a, b){return a - b});` is unclear
- In the following slides I provide:
 - A simple JavaScript Bubble Sort program to demonstrate how an array sort may work (implementations may vary between web browsers)
 - In this program you will see many functions including: `#functions` `#Random numbers` (integers) `#arrays` `#nested arrays` `#array.length` `#array processing` `#strings`. This program uses `#embedded style` and the JavaScript `#document.write` method

JS_bubble_sort - Apache NetBeans IDE 11.1

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

189.5/386.0MB

Search (Ctrl+I)

Projects Services Files

JS_bubble_sort

- Site Root
- index.html
- Unit Tests
- Important Files
 - .bowerrc
 - Gruntfile
 - bower.json
 - gulpfile
 - package.json

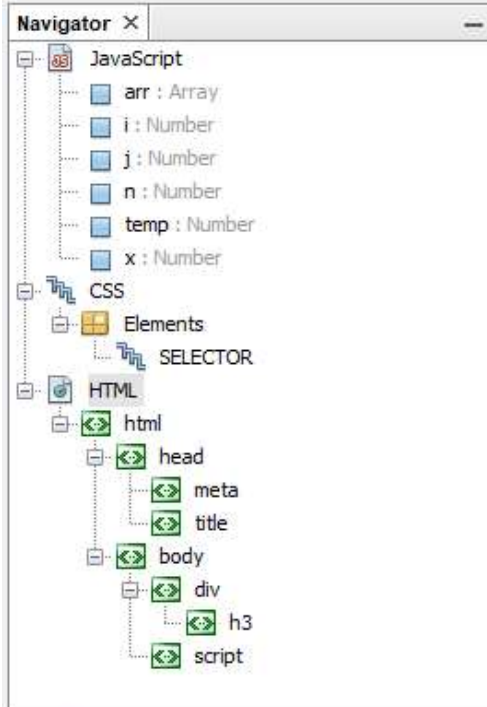
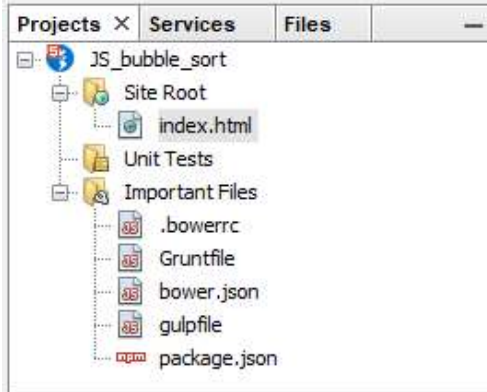
Navigator

- JavaScript
 - arr : Array
 - i : Number
 - j : Number
 - n : Number
 - temp : Number
 - x : Number
- CSS
 - Elements
 - SELECTOR
- HTML
 - html
 - head
 - meta
 - title
 - body
 - div
 - h3
 - script

index.html

Source History

```
1 <!DOCTYPE html>
2 <!--
3 A JavaScript program to demonstrate how to sort an array
4 The program shows the Bubble Sort method
5 The method is quite old but simple and effective
6 In this program you will see many functions including:
7 #functions #Random numbers (integers) #array #array.length
8 #array processing #strings #embedded style
9 This program uses JavaScript #document.write
10 Author: Philip Moore
11 -->
12 <html>
13 <head>
14   <meta charset="utf-8">
15   <title>Bubble Sort</title>
16 </head>
17 <body>
18   <div style="color:blue">
19     <h3>A basic JavaScript bubble sort implementation</h3>
20   </div>
21   <script>
22     var arr = [];
23     var i;
24     document.write("original array of random numbers <br>");
25     for(i = 0; i <= 20; i++) {
26       //var x = (Math.random() * 100 + 1);
27       var x = (Math.floor(Math.random() * 100 + 1));
28       arr[i] = x;
29       document.write(arr[i] + " * ");
30     }
31     document.write("<br>array.length: " + arr.length + "<br>");
```



```
20     </div>
21   <script>
22     var arr = [];
23     var i;
24     document.write("original array of random numbers <br>");
25     for(i = 0; i <= 20; i++) {
26         //var x = (Math.random() * 100 + 1);
27         var x = (Math.floor(Math.random() * 100 + 1));
28         arr[i] = x;
29         document.write(arr[i] + " * ");
30     }
31     document.write("<br>array.length: " + arr.length + "<br>");
32     document.write("unsorted array <br>");
33     document.write(arr + "<br>");
34     var n = arr.length; var temp = 0;
35     for(var i=0; i < n; i++){
36         for(var j=1; j < (n-i); j++){
37             if(arr[j-1] > arr[j]){
38                 //swap elements
39                 temp = arr[j-1];
40                 arr[j-1] = arr[j];
41                 arr[j] = temp;
42             }
43         }
44     }
45     document.write("sorted array");
46     document.write("<br>" + arr);
47   </script>
48 </body>
49 </html>
50
```

The screenshot shows the Apache NetBeans 11.1 IDE with the following components:

- Projects Panel:** Shows a project named 'JS_bubble_sort' with a 'Site Root' containing 'index.html', 'Unit Tests', and 'Important Files' (including .bowerrc, Gruntfile, bower.json, gulpfile, and package.json).
- Navigators Panel:** Shows the 'JavaScript' scope with variables: arr (Array), i (Number), j (Number), n (Number), temp (Number), and x (Number). It also shows the 'HTML' scope with a tree structure: html > head > meta, title > body > div > h3, script.
- Source Editor:** Displays the 'index.html' file with the following JavaScript code:

```
20 </div>
21 <script>
22   var arr = [];
23   var i;
24   document.write("original array of random numbers <br>");
25   for(i = 0; i <= 20; i++) {
26     //var x = (Math.random() * 100 + 1);
27     var x = (Math.floor(Math.random() * 100 + 1));
28     arr[i] = x;
29     document.write(arr[i] + " * ");
30   }
31   document.write("<br>array.length: " + arr.length + "<br>");
32   document.write("unsorted array <br>");
```
- Bubble Sort Panel:** Shows the URL 'http://localhost:8383/JS_bubble_sort/index.html' and a 'Bubble Sort' button. Below the button, the output of the script is displayed:

A basic JavaScript bubble sort implementation

original array of random numbers
89 * 93 * 37 * 94 * 94 * 87 * 80 * 51 * 49 * 78 * 97 * 47 * 74 * 91 * 65 * 99 * 82 * 94 * 21 * 23 * 51 *
array.length: 21
unsorted array
89,93,37,94,94,87,80,51,49,78,97,47,74,91,65,99,82,94,21,23,51
sorted array
21,23,37,47,49,51,51,65,74,78,80,82,87,89,91,93,94,94,97,99

Object Sort

Object Array Sort

- JavaScript arrays may hold a range of simple datatypes including objects
- For example
- ```
var cars = [
 {type:"Volvo", year:2016},
 {type:"Saab", year:2001},
 {type:"BMW", year:2010}];
```
- The following slide shows the object array sort

```
5 <html>
6 <head>
7 <title>TODO supply a title</title>
8 <meta charset="UTF-8">
9 <meta name="viewport" content="width=device-width, initial-scale=1.0">
10 </head>
11 <body>
12 <h2>JavaScript Array Sort</h2>
13 <p>Click the buttons to sort car objects on age.</p>
14 <button onclick="myFunction()">Sort</button>
15 <p id="demo"></p>
16 <script>
17 var cars = [{type:"Volvo", year:2016},
18 {type:"Saab", year:2001},
19 {type:"BMW", year:2010}]
20 displayCars();
21 function myFunction() {
22 cars.sort(function(a, b){return a.year - b.year});
23 displayCars();
24 }
25 function displayCars() {
26 document.getElementById("demo").innerHTML =
27 cars[0].type + " " + cars[0].year + "
" +
28 cars[1].type + " " + cars[1].year + "
" +
29 cars[2].type + " " + cars[2].year;
30 }
31 </script>
32 </body>
33 </html>
```

Projects Services Files

JS\_Array\_Length

JS\_Obj\_Array\_Sort

Site Root

index.html

Unit Tests

Navigator Browser DOM

http://localhost:8383/JS\_Obj\_Array\_...

html

head

body

h2

p

button

p#demo

script

index.html

Source History

Variables Call Stack Breakpoints

Name

TODO supply a title

http://localhost:8383/JS\_Obj\_Array\_Sort/index.html

100%

## JavaScript Array Sort

Click the buttons to sort car objects on age.

Sort

Volvo 2016  
Saab 2001  
BMW 2010

>

The screenshot shows a web browser window with the developer tools open. The browser's address bar shows the URL `http://localhost:8383/JS_Obj_Array_Sort/index.html`. The developer tools are split into two panes. The left pane contains the 'Projects' and 'Navigator' tabs. The 'Projects' tab shows a file tree with 'JS\_Array\_Length', 'JS\_Obj\_Array\_Sort', 'Site Root', 'index.html', and 'Unit Tests'. The 'Navigator' tab shows the DOM tree with the following structure:

- html
  - head
  - body
    - h2
    - p
    - button
    - p#demo
    - script

The right pane contains the 'Variables', 'Call Stack', and 'Breakpoints' tabs. The 'Variables' tab is active, showing a table with one column labeled 'Name'. Below the table is a 'TODO supply a title' message. The bottom pane shows the rendered HTML content:

## JavaScript Array Sort

Click the buttons to sort car objects on age.

Saab 2001  
BMW 2010  
Volvo 2016

>



# review

- In this tutorial we have considered:
  - Making changes to arrays and array elements
  - Arrays and strings
  - Sorting arrays
  - A brief overview of the algorithmic approach the array sorting with a worked example showing how the JavaScript elements combine to create a working program