Intro to Programming

18 Nov 2016 - Ms. Santos

Goals

Goal 1: You will understand how function arguments and variables work.

Goal 2: You will know how to get input from a webpage

Vocabulary

variable context

element id

Code

`document.getElementById(“name”)`

`element``.value`

`element``.innerHTML`

Big Ideas!

What is **modularization**?

Modularization is a programming technique which code is broken down into smaller self-contained pieces

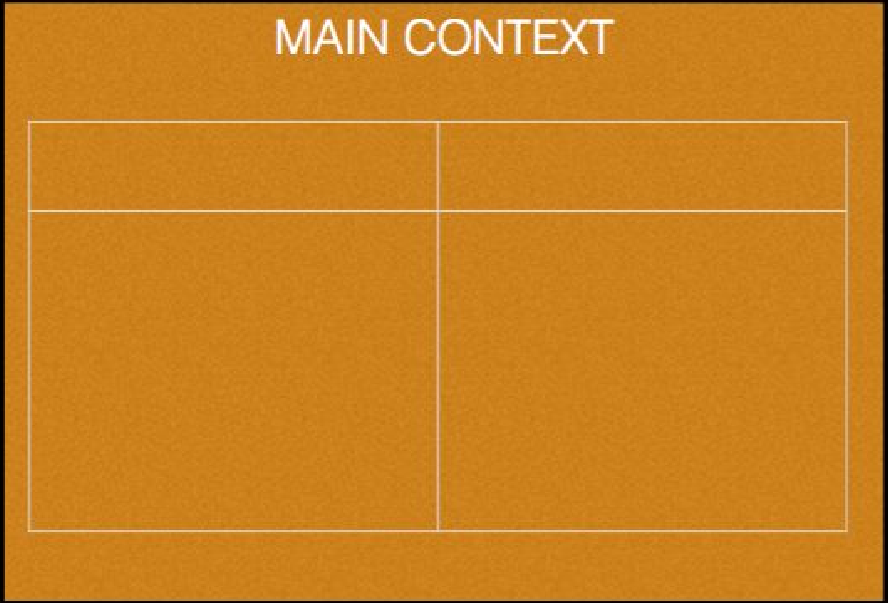
double.js

```
var value = 5;
show_double(value);
alert(value);

function show_double(num) {
    var result = num * 2;
    alert(result);
    value = “Hi!”;
}
```

double.js

Pre-processing

```
var value = 5;
show_double(value);
alert(value);

function show_double(num) {
 var result = num * 2;
 alert(result);
 value = "Hi!";
}
```



During pre-processing, the javascript interpreter creates a variable context and looks for var declarations and function definitions.

double.js

Pre-processing

```
var value = 5;
show_double(value);
alert(value);

function show_double(num) {
 var result = num * 2;
 alert(result);
 value = "Hi!";
}
```

MAIN CONTEXT

value	undefined

double.js

Pre-processing

```
var value = 5;
show_double(value);
alert(value);

function show_double(num) {
 var result = num * 2;
 alert(result);
 value = "Hi!";
}
```

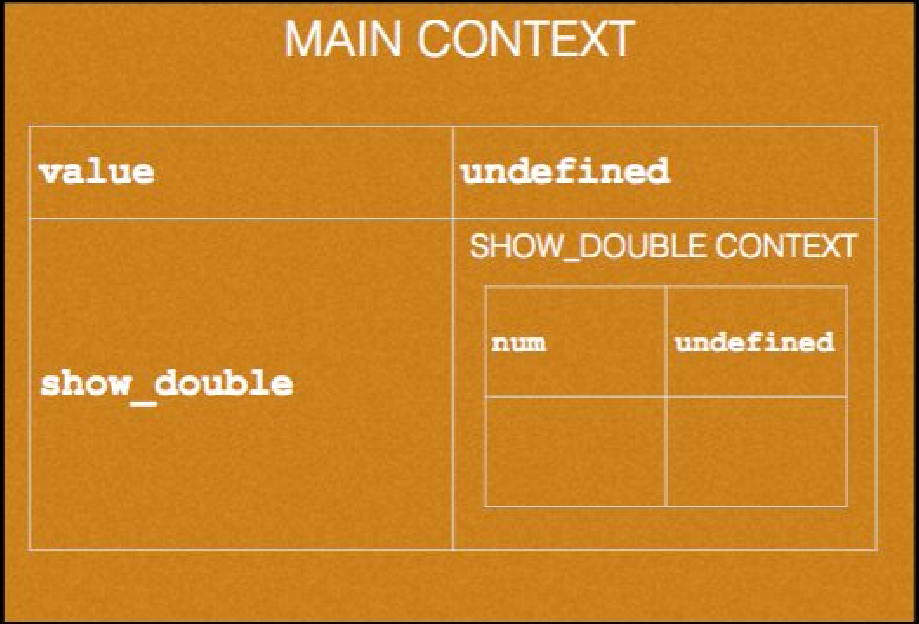
MAIN CONTEXT

value	undefined
show_double	undefined

double.js

Pre-processing

```
var value = 5;
show_double(value);
alert(value);

function show_double(num) {
 var result = num * 2;
 alert(result);
 value = "Hi!";
}
```



MAIN CONTEXT

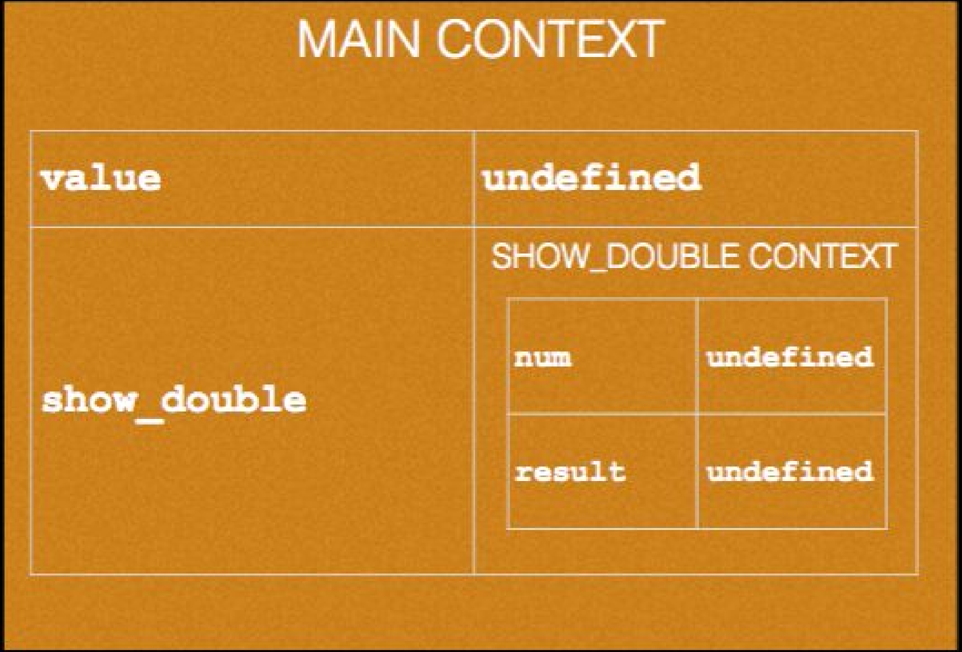
value	undefined
show_double	SHOW_DOUBLE CONTEXT num: undefined

For each function definition, a variable context is created.

double.js

Pre-processing

```
var value = 5;
show_double(value);
alert(value);

function show_double(num) {
 var result = num * 2;
 alert(result);
 value = "Hi!";
}
```



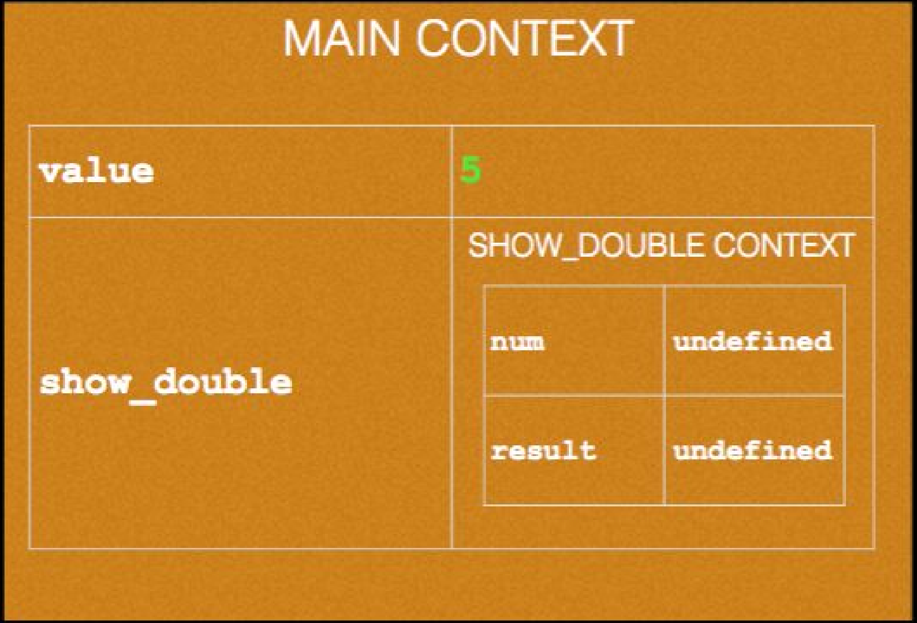
MAIN CONTEXT

<table>
<tr><td>value</td><td>undefined</td></tr>
<tr><td>show_double</td><td>SHOW_DOUBLE CONTEXT<table><tr><td>num</td><td>undefined</td></tr><tr><td>result</td><td>undefined</td></tr></table></td></tr>
</table>

double.js

Executing

```
var value = 5;
show_double(value);
alert(value);

function show_double(num) {
 var result = num * 2;
 alert(result);
 value = "Hi!";
}
```



MAIN CONTEXT

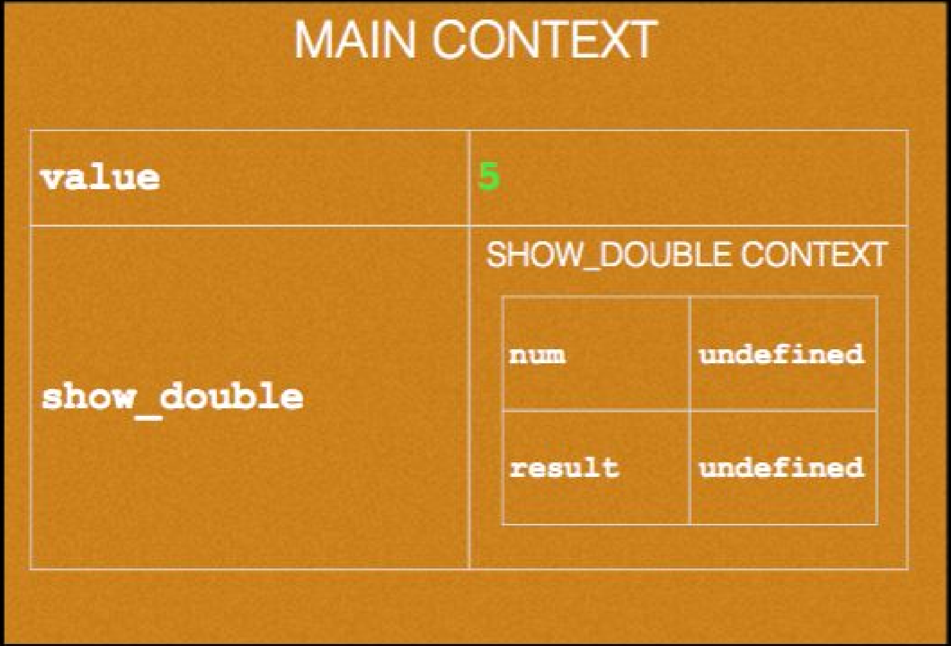
<table>
<tr><td>value</td><td>5</td></tr>
<tr><td>show_double</td><td>SHOW_DOUBLE CONTEXT<table><tr><td>num</td><td>undefined</td></tr><tr><td>result</td><td>undefined</td></tr></table></td></tr>
</table>

During execution, each line of code is run in turn. Variable assignments and function calls change the contexts.

double.js

Executing

```
var value = 5;
show_double(value);
alert(value);

function show_double(num) {
 var result = num * 2;
 alert(result);
 value = "Hi!";
}
```



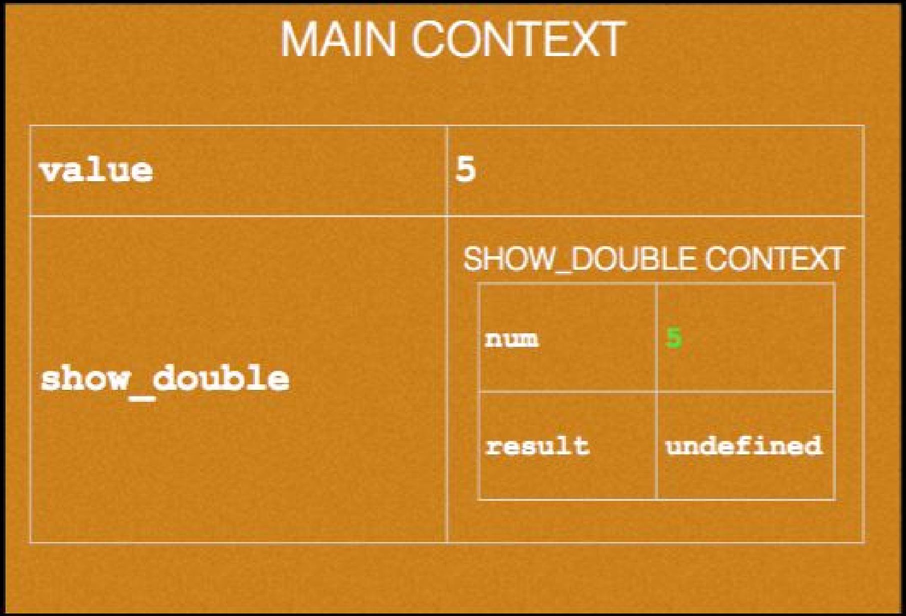
MAIN CONTEXT

<table>
<tr><td>value</td><td>5</td></tr>
<tr><td>show_double</td><td>SHOW_DOUBLE CONTEXT
<table>
<tr><td>num</td><td>undefined</td></tr>
<tr><td>result</td><td>undefined</td></tr>
</table>
</td></tr>
</table>

double.js

Executing

```
var value = 5;
show_double(value);
alert(value);

function show_double(num) {
 var result = num * 2;
 alert(result);
 value = "Hi!";
}
```



MAIN CONTEXT

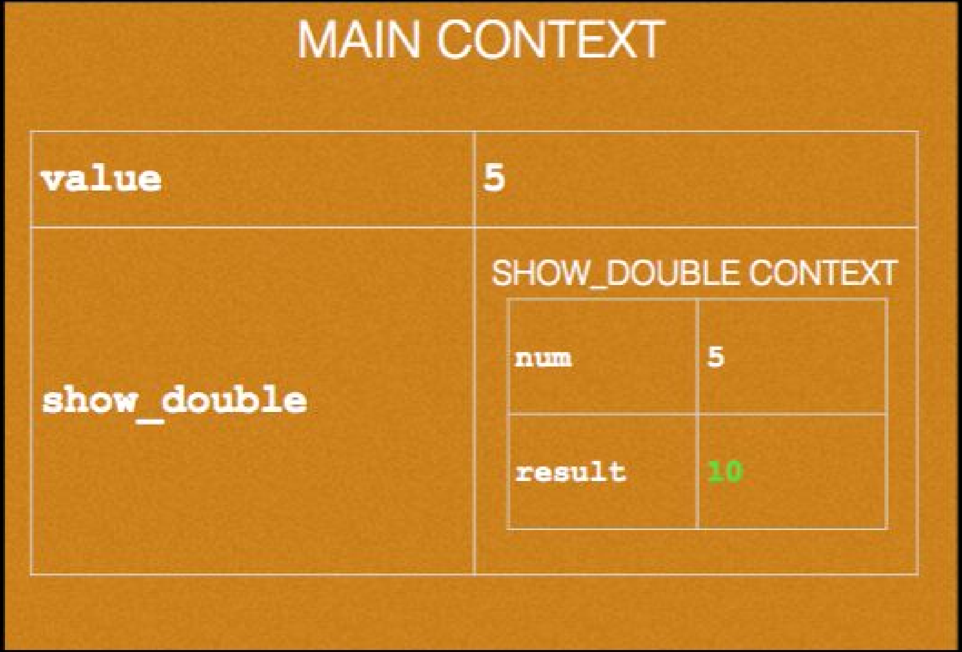
value	5
show_double	SHOW_DOUBLE CONTEXT num: 5 result: undefined

During a function call, values are passed as arguments into the function context.

double.js

Executing

```
var value = 5;
show_double(value);
alert(value);

function show_double(num) {
 var result = num * 2;
 alert(result);
 value = "Hi!";
}
```



MAIN CONTEXT

value	5
show_double	SHOW_DOUBLE CONTEXT num 5 result 10

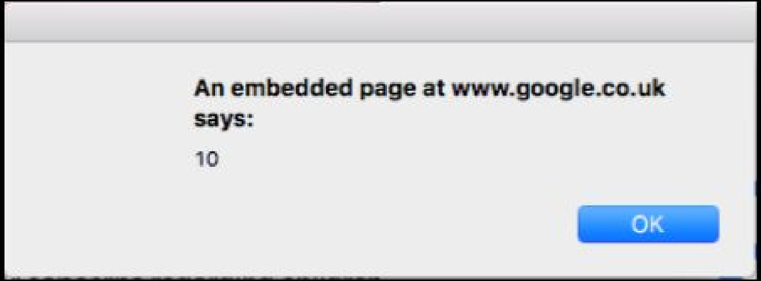
double.js

Executing

```
  var value = 5;
  show_double(value);
  alert(value);

  function show_double(num) {
    var result = num * 2;
➡   alert(result);
    value = "Hi!";
  }
```

MAIN CONTEXT

<table>
<tr><td>value</td><td>5</td></tr>
<tr><td>show_double</td><td>SHOW_DOUBLE CONTEXT
<table>
<tr><td>num</td><td>5</td></tr>
<tr><td>result</td><td>10</td></tr>
</table>
</td></tr>
</table>



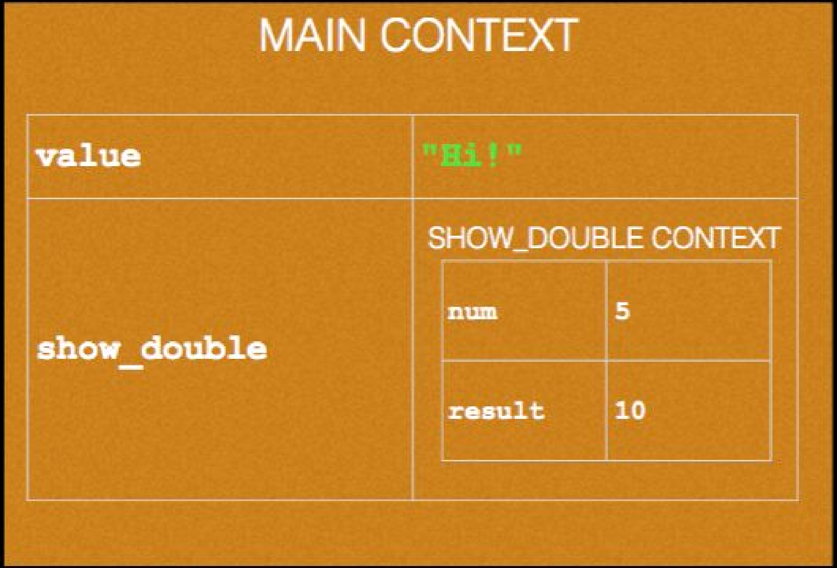
An embedded page at www.google.co.uk says:

10

double.js

Executing

```
var value = 5;
show_double(value);
alert(value);

function show_double(num) {
 var result = num * 2;
 alert(result);
 value = "Hi!";
}
```



MAIN CONTEXT

value "Hi!"

show_double

SHOW_DOUBLE CONTEXT

num 5

result 10

The value of a variable can change in any context **within which it is nested**. E.g., show_double can change the value of `value`. But the main context cannot change the value of `result`.

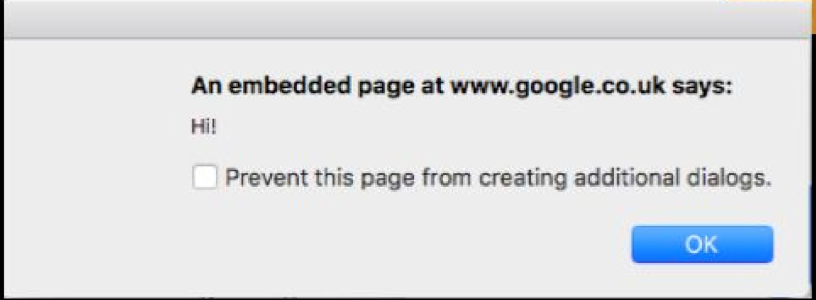
double.js

Executing

```
   var value = 5;
   show_double(value);
➡ alert(value);

   function show_double(num) {
    var result = num * 2;
    alert(result);
    value = "Hi!";
   }
```

MAIN CONTEXT

<table>
<tr><td>value</td><td>"Hi!"</td></tr>
<tr><td>show_double</td><td>SHOW_DOUBLE CONTEXT<table><tr><td>num</td><td>5</td></tr><tr><td>result</td><td>10</td></tr></table></td></tr>
</table>



An embedded page at www.google.co.uk says:

Hi!

☐ Prevent this page from creating additional dialogs.

double.js

```
var value = 5;
show_double(value);
alert(value);

function show_double(num) {
 var result = num * 2;
 alert(result);
 value = "Hi!";
}
```

void function

betterDouble.js

```
var value = 5;
var new_value = show_double(value);
alert(new_value);

function show_double(num) {
 var result = num * 2;
 alert(result);
 return "Hi!";
}
```

return function

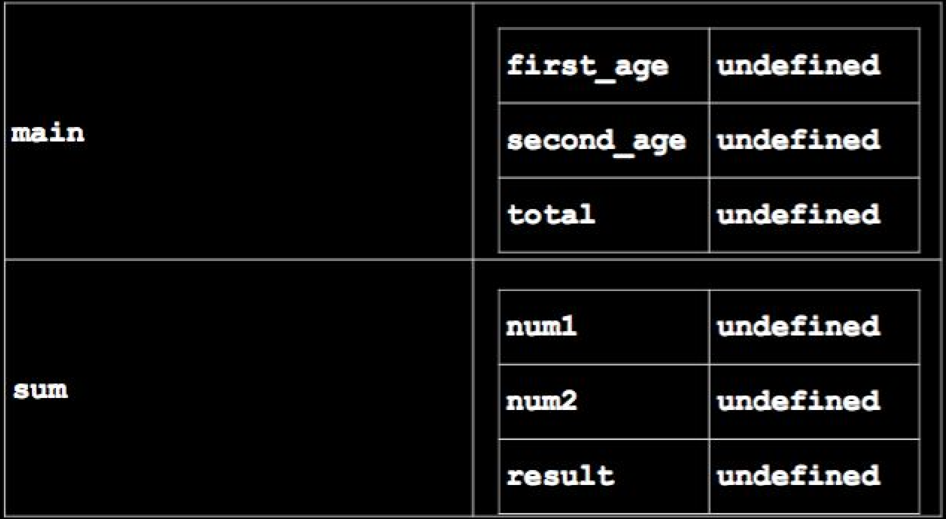
Generally, it's better to use a `return` statement rather than set variables outside a function context. Code is more modularized!

DO

Create a variable context table for the following program. Make sure you think about the order in which the pre-processor and execution work.

sumAges.js

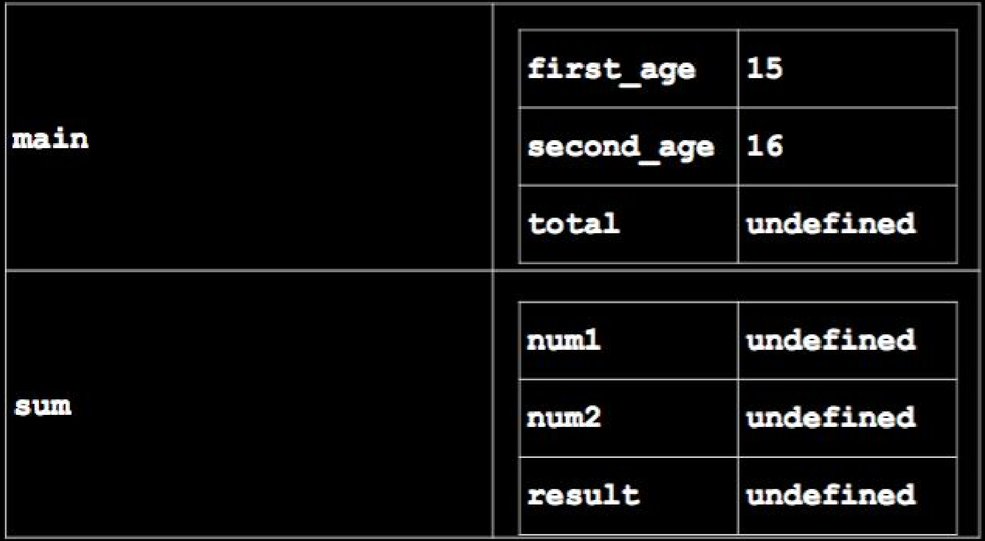
```
main();

function main() {
      var first_age = parseInt(prompt("Enter your age:"));
      var second_age = parseInt(prompt("Enter your friend's age:"));
      var total = sum(first_age, second_age);
      alert(first_age);
}

function sum(num1, num2) {
      var result = num1 + num2;
      first_age = "WHAT?";
      return result;
}
```



main	first_age	undefined
	second_age	undefined
	total	undefined
sum	num1	undefined
	num2	undefined
	result	undefined

sumAges.js

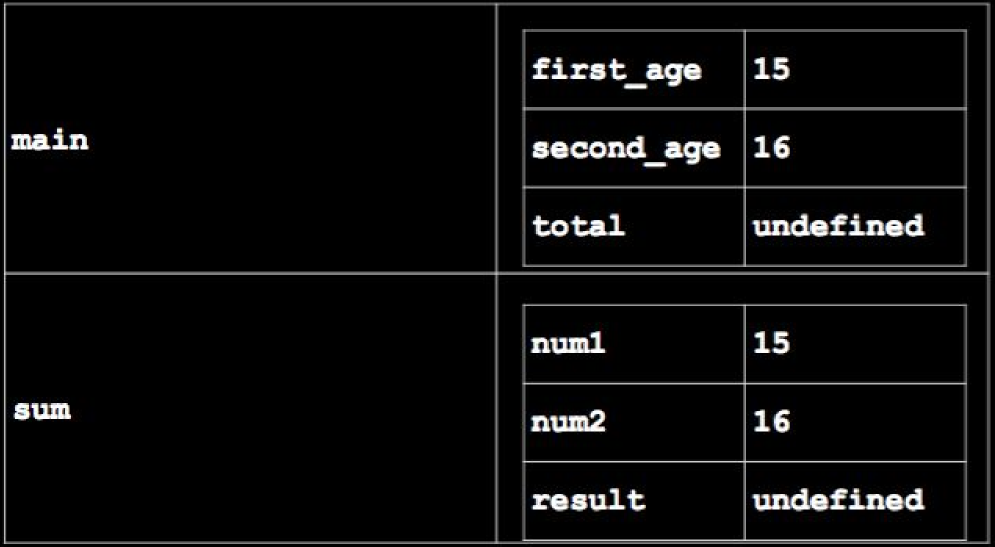
```
main();

function main() {
     var first_age = parseInt(prompt("Enter your age:"));
     var second_age = parseInt(prompt("Enter your friend's age:"));
     var total = sum(first_age, second_age);
     alert(first_age);
}

function sum(num1, num2) {
     var result = num1 + num2;
     first_age = "WHAT?";
     return result;
}
```



main	first_age	15
	second_age	16
	total	undefined
sum	num1	undefined
	num2	undefined
	result	undefined

sumAges.js

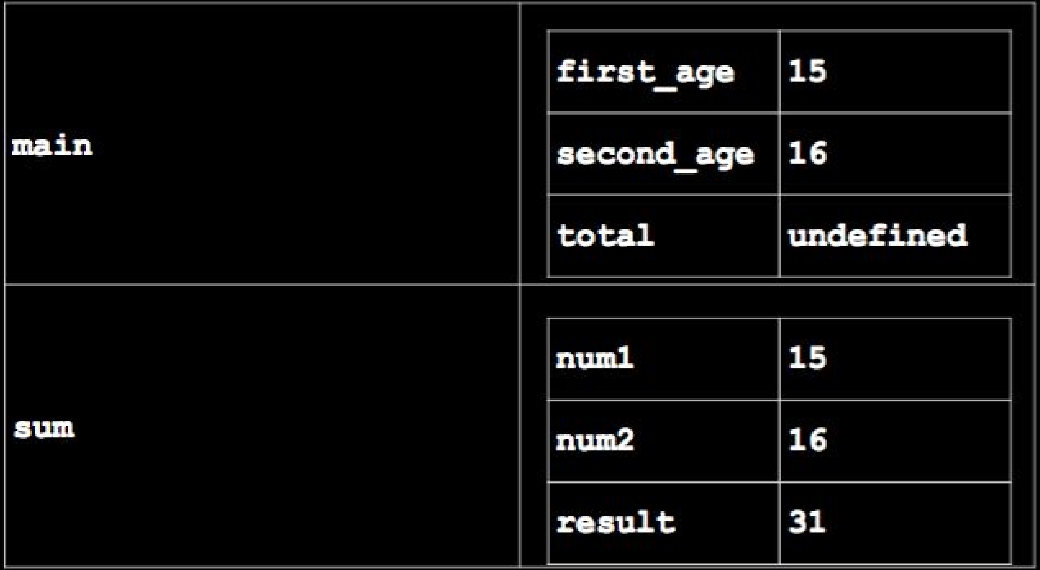
```
main();

function main() {
      var first_age = parseInt(prompt("Enter your age:"));
      var second_age = parseInt(prompt("Enter your friend's age:"));
      var total = sum(first_age, second_age);
      alert(first_age);
}

function sum(num1, num2) {
      var result = num1 + num2;
      first_age = "WHAT?";
      return result;
}
```



main	first_age	15
	second_age	16
	total	undefined
sum	num1	15
	num2	16
	result	undefined

sumAges.js

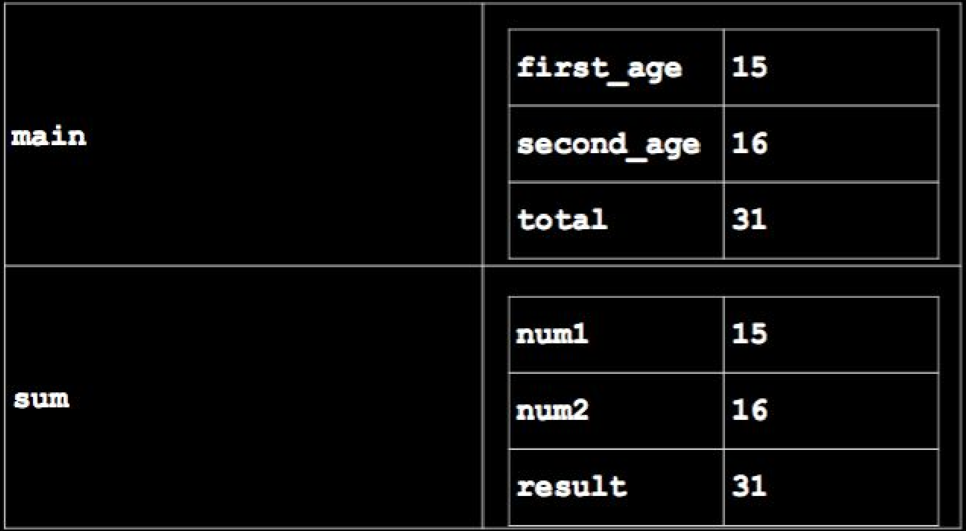
```
main();

function main() {
     var first_age = parseInt(prompt("Enter your age:"));
     var second_age = parseInt(prompt("Enter your friend's age:"));
     var total = sum(first_age, second_age);
     alert(first_age);
}

function sum(num1, num2) {
     var result = num1 + num2;
     first_age = "WHAT?";
     return result;
}
```



main	first_age	15
	second_age	16
	total	undefined
sum	num1	15
	num2	16
	result	31

sumAges.js

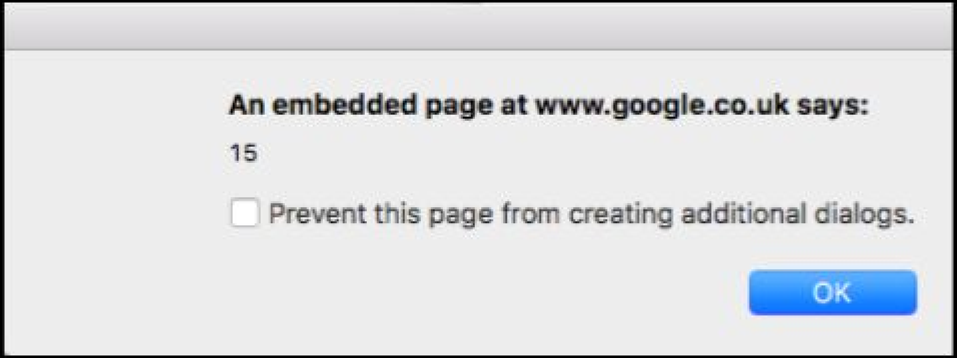
```
main();

function main() {
     var first_age = parseInt(prompt("Enter your age:"));
     var second_age = parseInt(prompt("Enter your friend's age:"));
     var total = sum(first_age, second_age);
     alert(first_age);
}

function sum(num1, num2) {
     var result = num1 + num2;
     first_age = "WHAT?";
     return result;
}
```



main	first_age	15
	second_age	16
	total	31
sum	num1	15
	num2	16
	result	31

sumAges.js

```
main();

function main() {
	var first_age = parseInt(prompt("Enter your age:"));
	var second_age = parseInt(prompt("Enter your friend's age:"));
	var total = sum(first_age, second_age);
	alert(first_age);
}

function sum(num1, num2) {
	var result = num1 + num2;
	first_age = "WHAT?";
	return result;
}
```

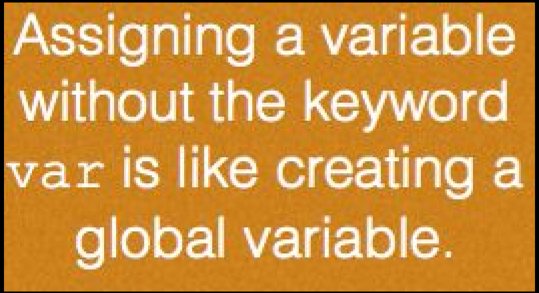


An embedded page at www.google.co.uk says:

15

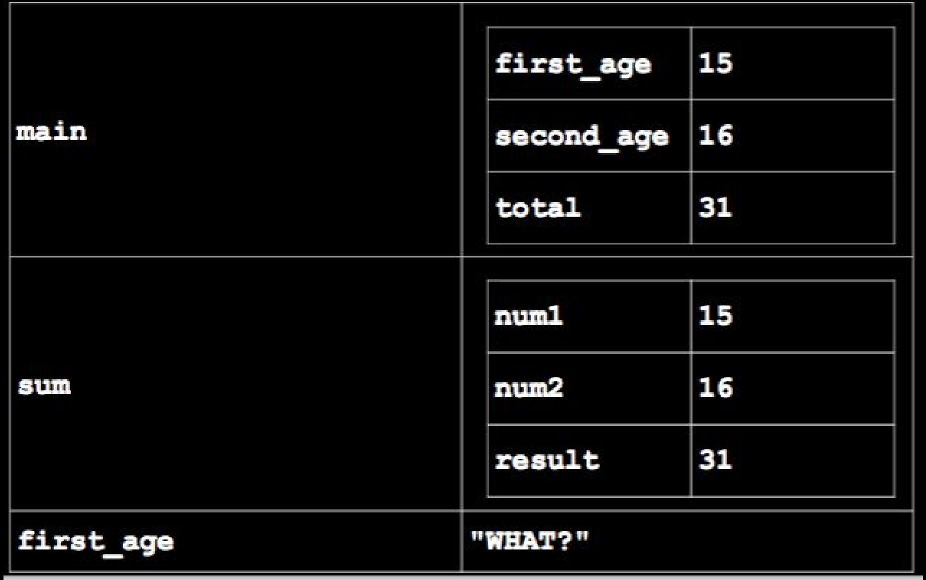
□ Prevent this page from creating additional dialogs.

sumAges.js



Assigning a variable without the keyword `var` is like creating a global variable.

```
main();

function main() {
     var first_age = parseInt(prompt("Enter your age:"));
     var second_age = parseInt(prompt("Enter your friend's age:"));
     var total = sum(first_age, second_age);
     alert(first_age);
}

function sum(num1, num2) {
     var result = num1 + num2;
     first_age = "WHAT?";
     return result;
}
```



main	first_age	15
	second_age	16
	total	31
sum	num1	15
	num2	16
	result	31
first_age	"WHAT?"	

DO

1. Run palindrome.html
2. Write comments in the code (just .js file)
3. Look out for these, can you guess what they do?

```
<p>Word 1:<input type="text" id="word1"></p>
<button onclick="checkPalindrome()">Check</button>
var textBox1 = document.getElementById(“word1”);
var word1 = textBox1.value;
response.innerHTML = message
```

`<p>Word 1:<input type="text" id="word1"></p>`	Creates a **text box element** inside a paragraph element and gives it the **id** of `word1`
`<button onclick="checkPalindrome()">Check</button>`	Creates a **button element** with the text "Check" and assigns the function checkPalindrom() to it when clicked
`var textBox1 = document.getElementById("word1");`	"Gets" the text box element with the id `word1` and saves it in the variable `textBox1`
`var word1 = textBox1.value;`	"Gets" the **text in the text box** element and saves it in the variable `word1`
`response.innerHTML = message;`	Sets the **text** in the paragraph element called `response` to `message`.

EXPLORE

1. Look at button and textbox folders.
2. Modify the code to do something fun.

DO

1. Task 4-1: kilometerConverter
2. Task 4-2: insuranceCalculator
3. Task 4-3: automobileCosts