INFO 151
Web Systems and Services

Week 5 (T1)

Dr Philip Moore
Dr Zhili Zhao

Objects and OOP
Selection
Iteration

Sources of Resources

- The sources of information and resources for JavaScript may be found at:
  - The w3schools.com web-site
    - https://www.quanzhanketang.com/
  - The recommended course text book:
    - Sams Teach Yourself PHP, MySQL & JavaScript All in One SIXTH EDITION

Overview

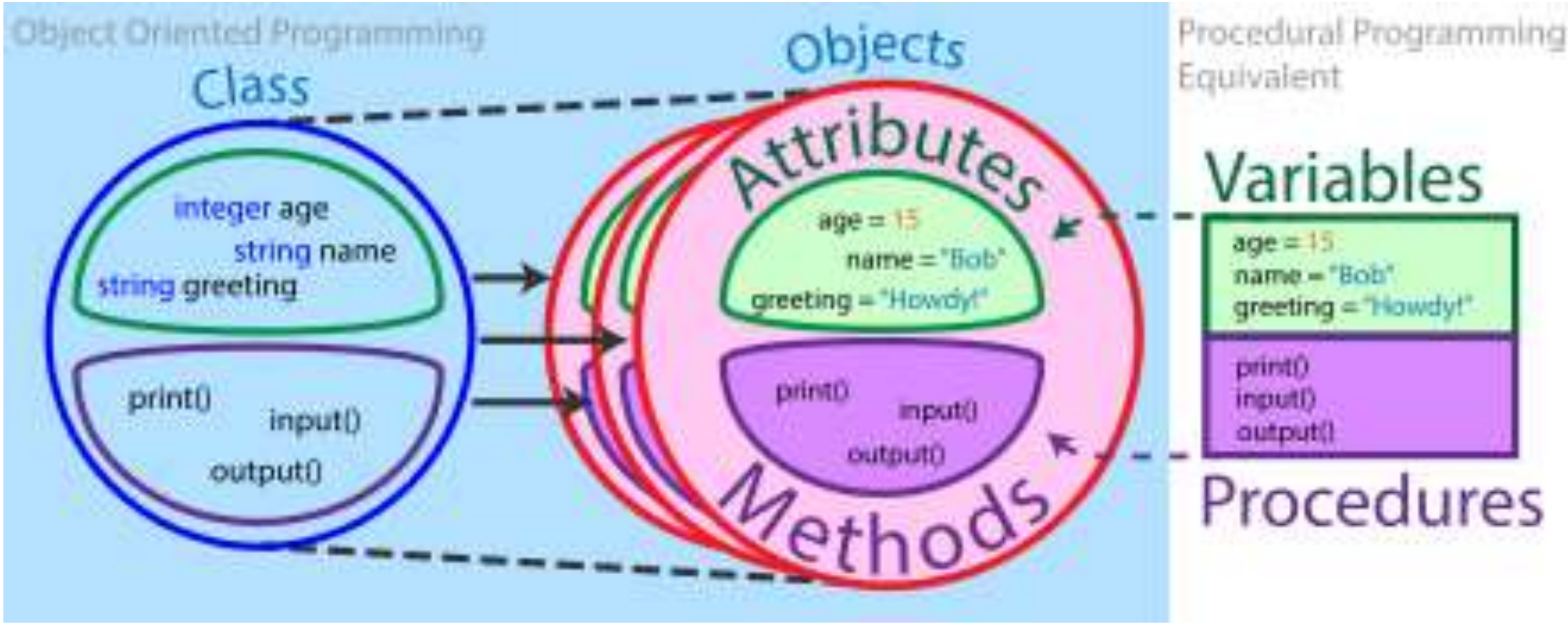
- In this session we will introduce:
  - Objects and Object-Oriented Programming (OOP)
  - Selection:
    - Using conditional operators to select from a range of alternatives
  - Iteration:
    - Using loops to repeat a program run until a condition is satisfied or a termination criteria is reached

Objects
and
Object-Orientated Programming

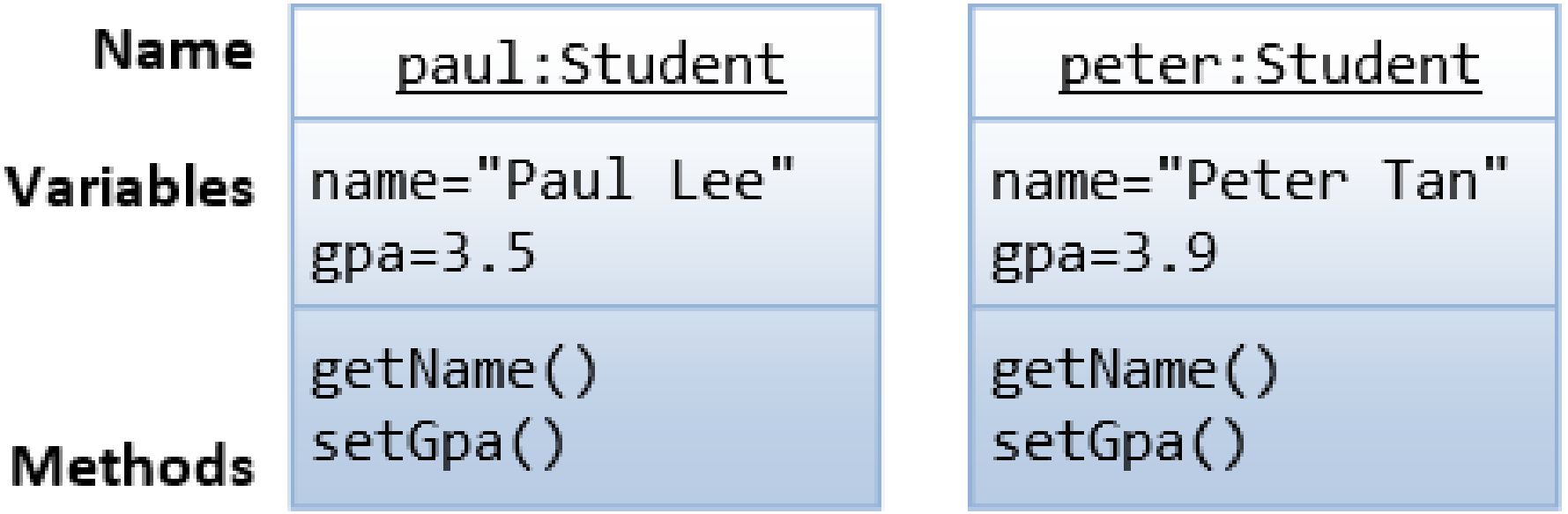
Object-Oriented Programming

- Object-oriented programming (OOP) is a programming paradigm based on the concept of *objects* where objects:
  - Hold data in the form of *attributes*
  - Hold program code in the form of *methods*
- A feature of an object is the capability for the *methods* to *access* and *modify* the data and properties of the object with which they are associated
  - Objects have a notion of **`this`** (or self)

Object-Oriented Programming

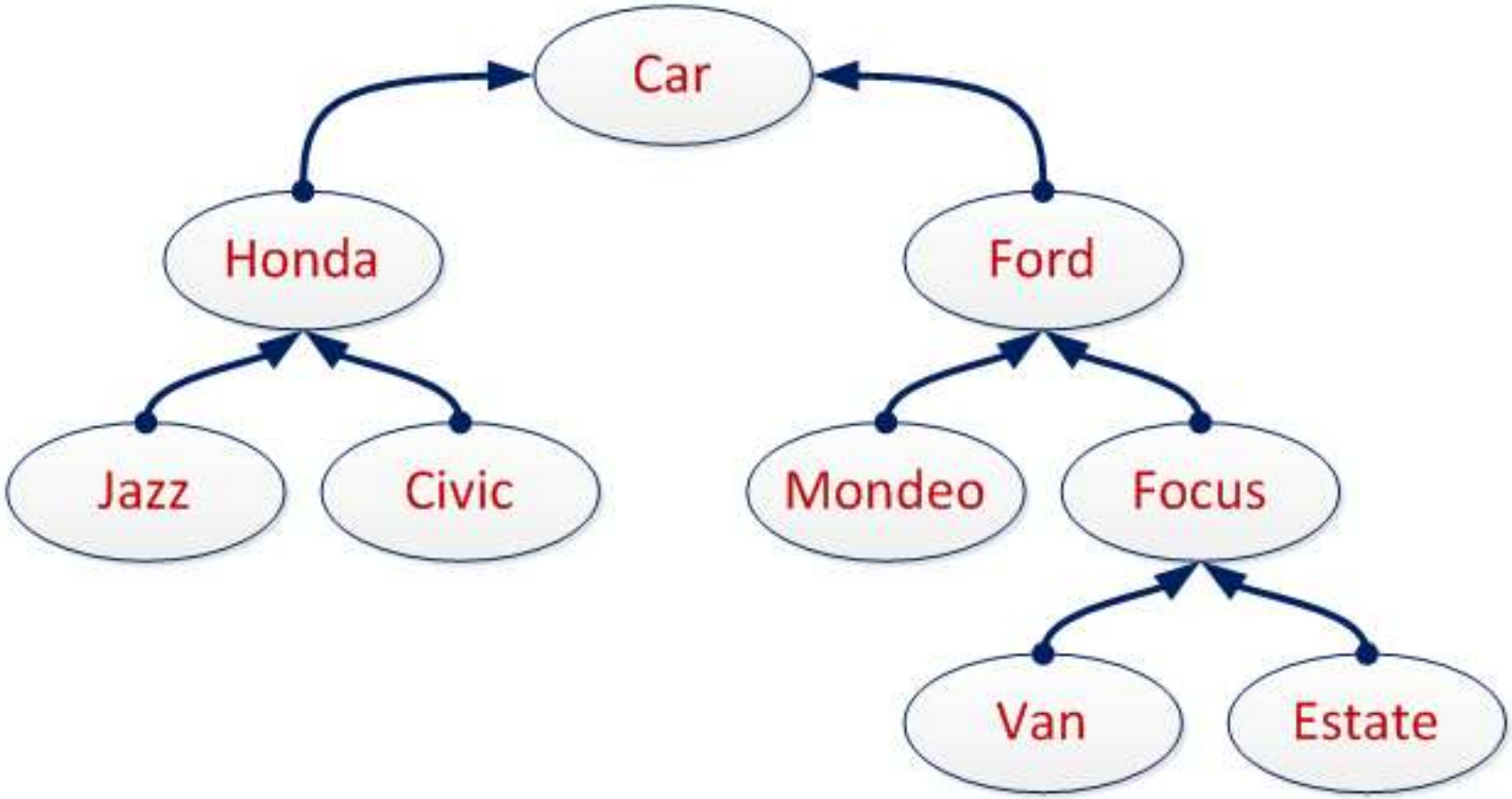


Objects – Student

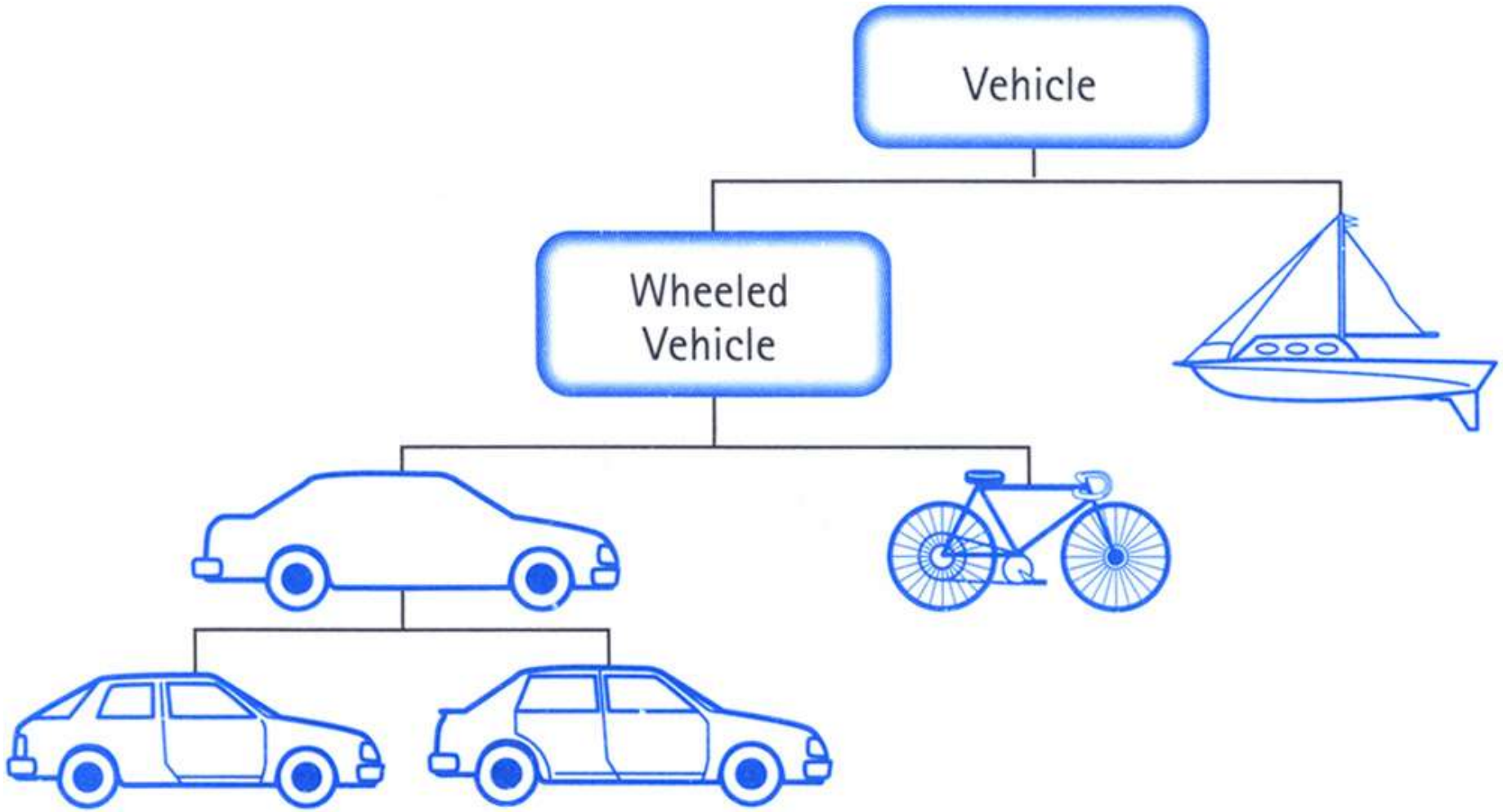


Two instances - paul and peter - of the class Student

Objects – Vehicles



Objects – Transport



Object-Oriented Programming

- An object-oriented program
  - Is created using well encapsulated objects
  - Messages are passed between objects in the program run
  - The messages create and manage the interactions between objects
- For example
  - Messages include calls to functions and the passing of parameters with the returned result

Creating Objects

- Objects in JavaScript are created using the `new` operator
- A new object with no properties is created as follows

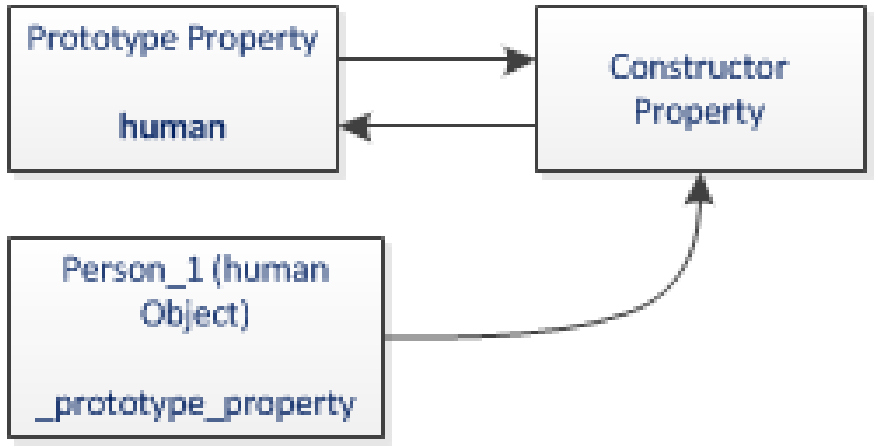
```
var x = new Object();
```

- Predefined constructors that are members of a class of objects
  - Properties and methods are automatically defined (this approach uses a JavaScript API
  - For example: a date object that represents the current time is created as follows

```
var now = new Date();
```

Object Prototype Property

- When a function is created in JavaScript a `prototype property` to the function
- This *prototype* property is an object (called as a *prototype* object) and has a *constructor* property by default
- The *constructor* property points back to the function on which `prototype object` is a property
- Access the function's prototype property is achieved using the syntax
  `functionName.prototype`
- For example: the person1 object (created using the Human constructor function) has a `__prototype__ property` which points to the prototype object of the constructor function

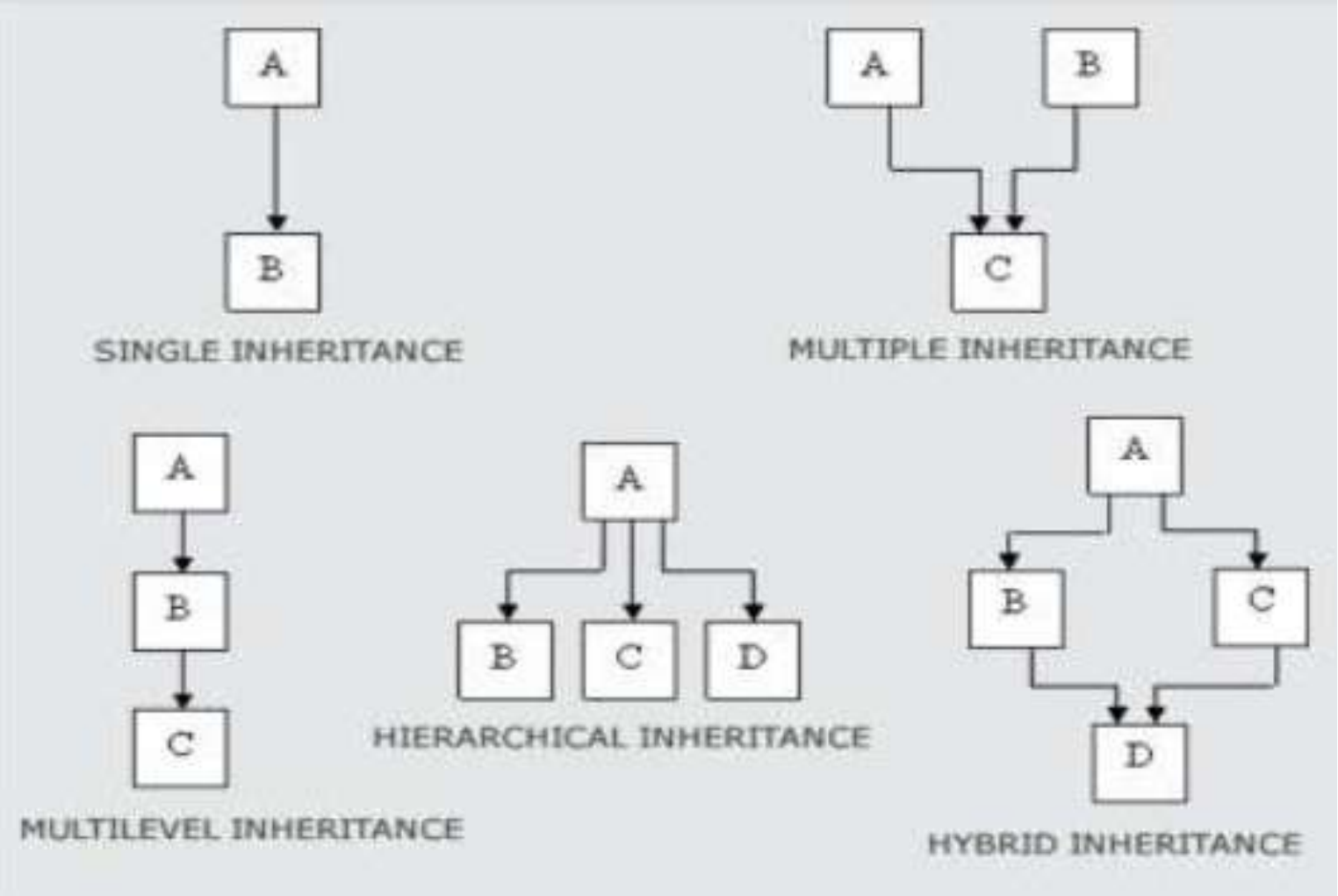


Object Orientation
and
Inheritance

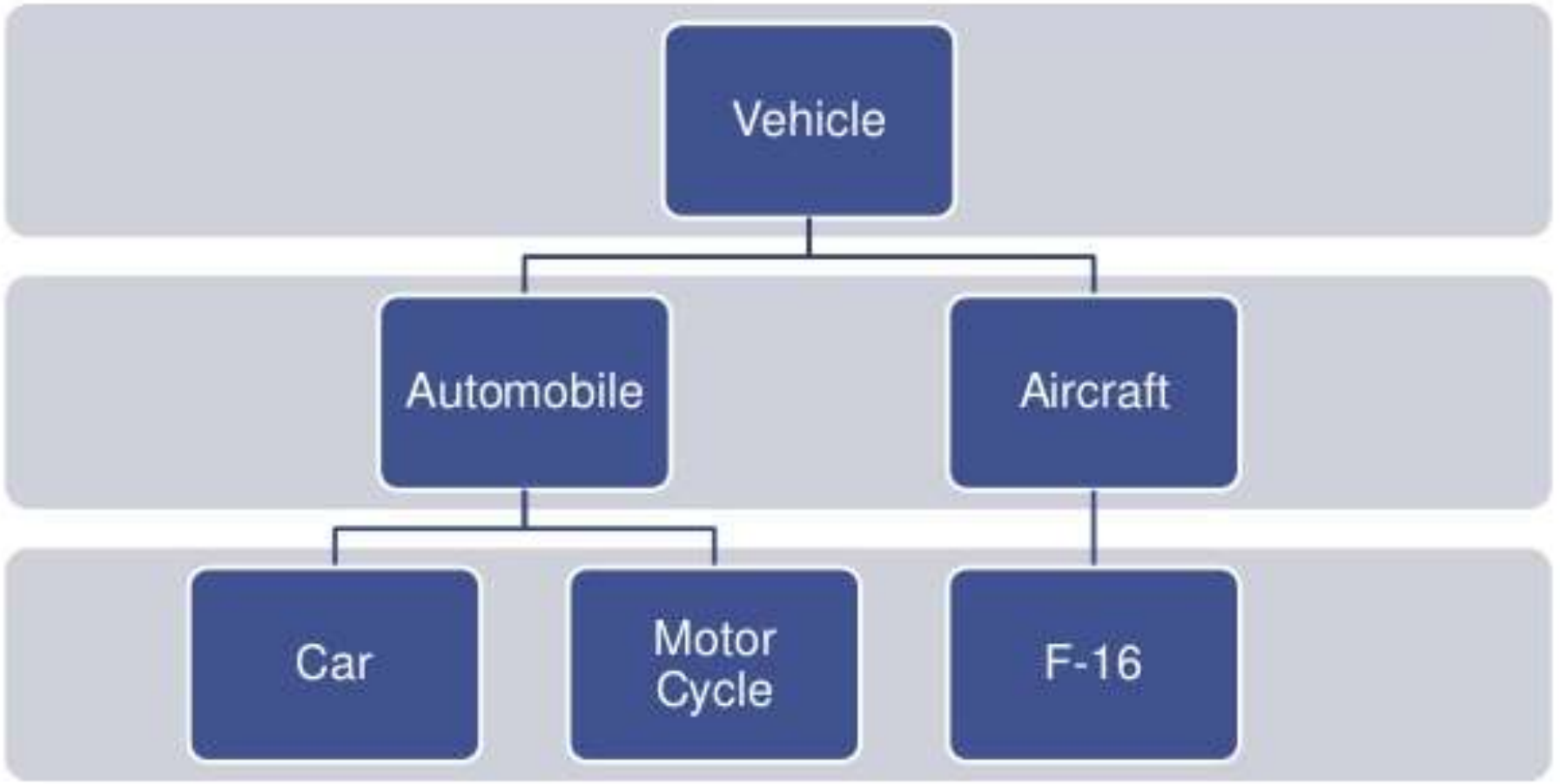
Object-Oriented Programming

- In OOP:
  - Computer programs are designed around the concept of creating interactions between *objects*
- In OOP and JavaScript:
  - There is the concept of inheritance where a *child* (object) inherits the attributes of the *parent* (object) and may add additional attributes
  - When an inherited function is executed the value of **`this`** points to the *inheriting object*

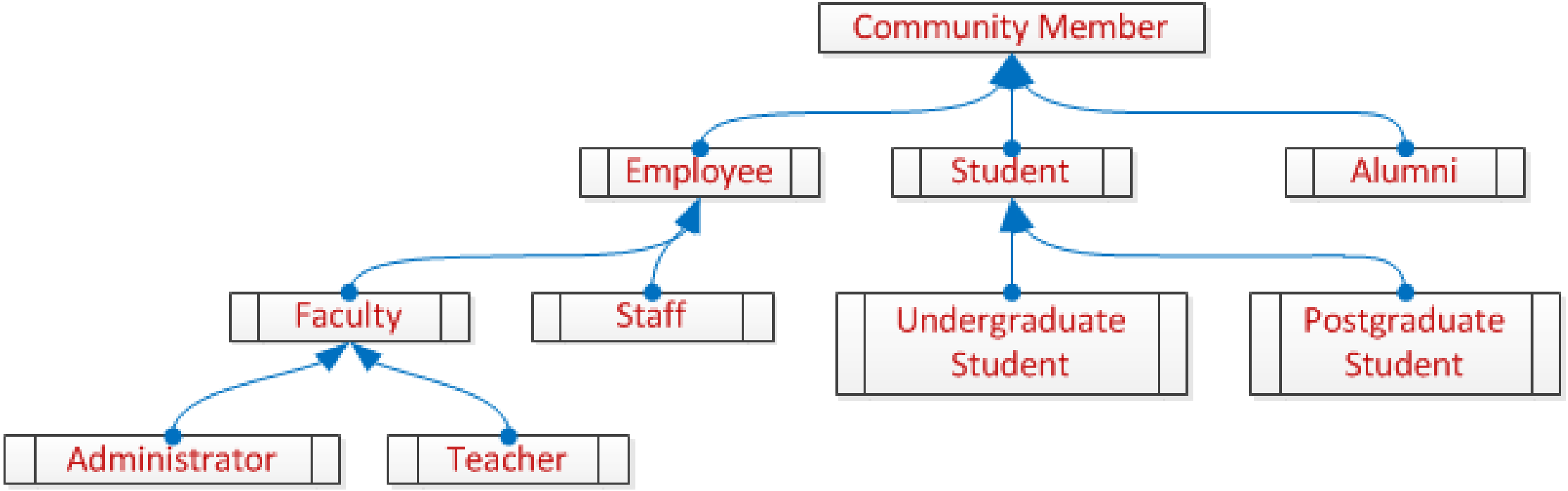
Types of Inheritance



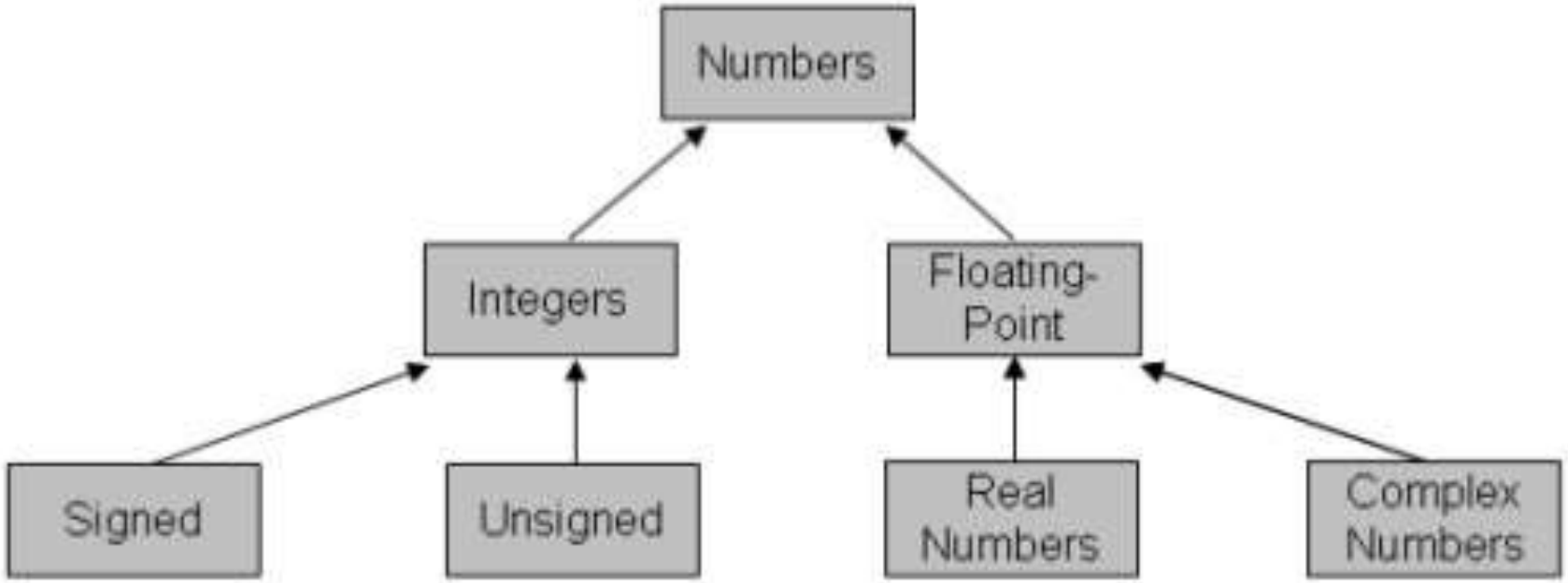
An Inheritance Tree



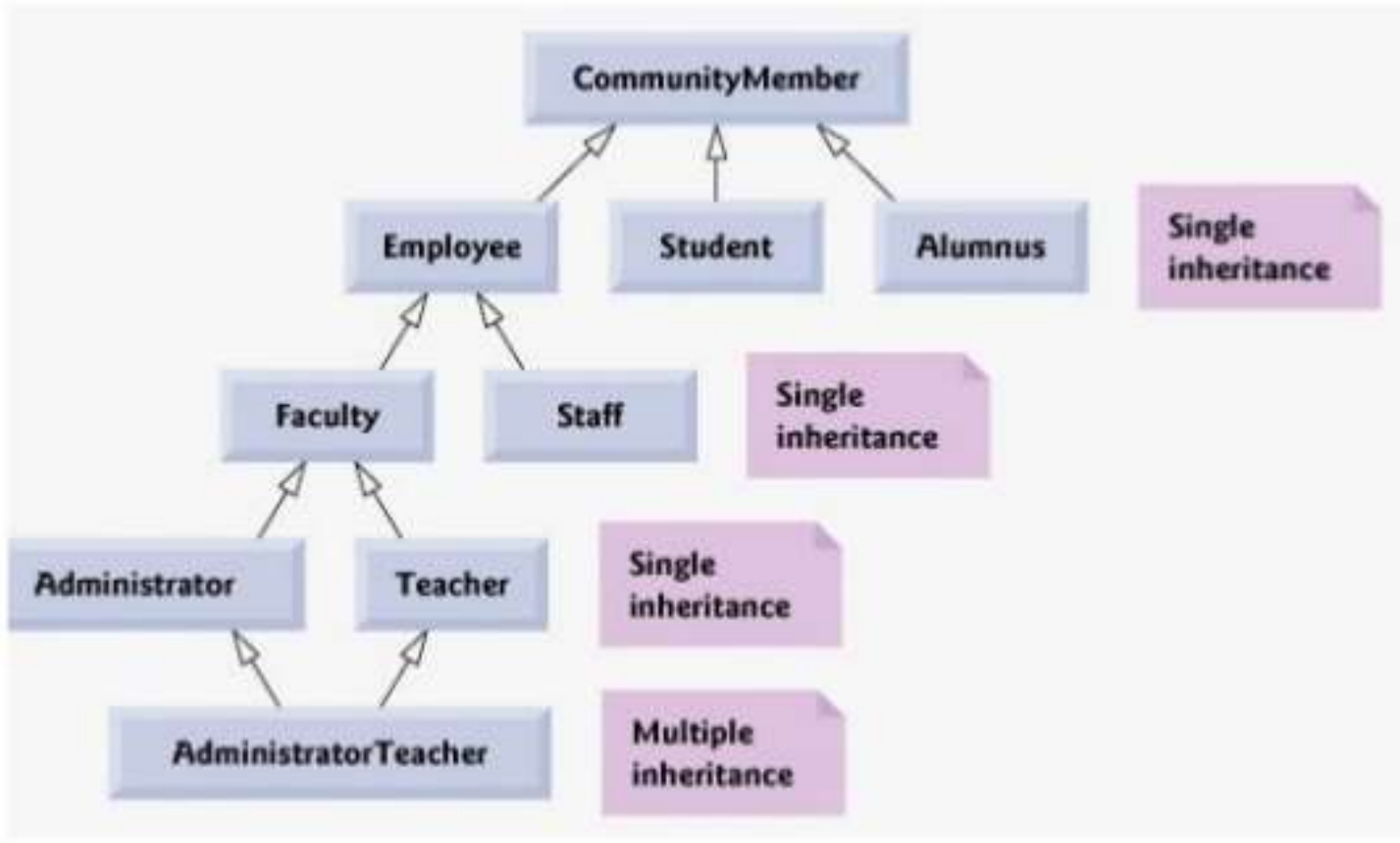
Inheritance Example



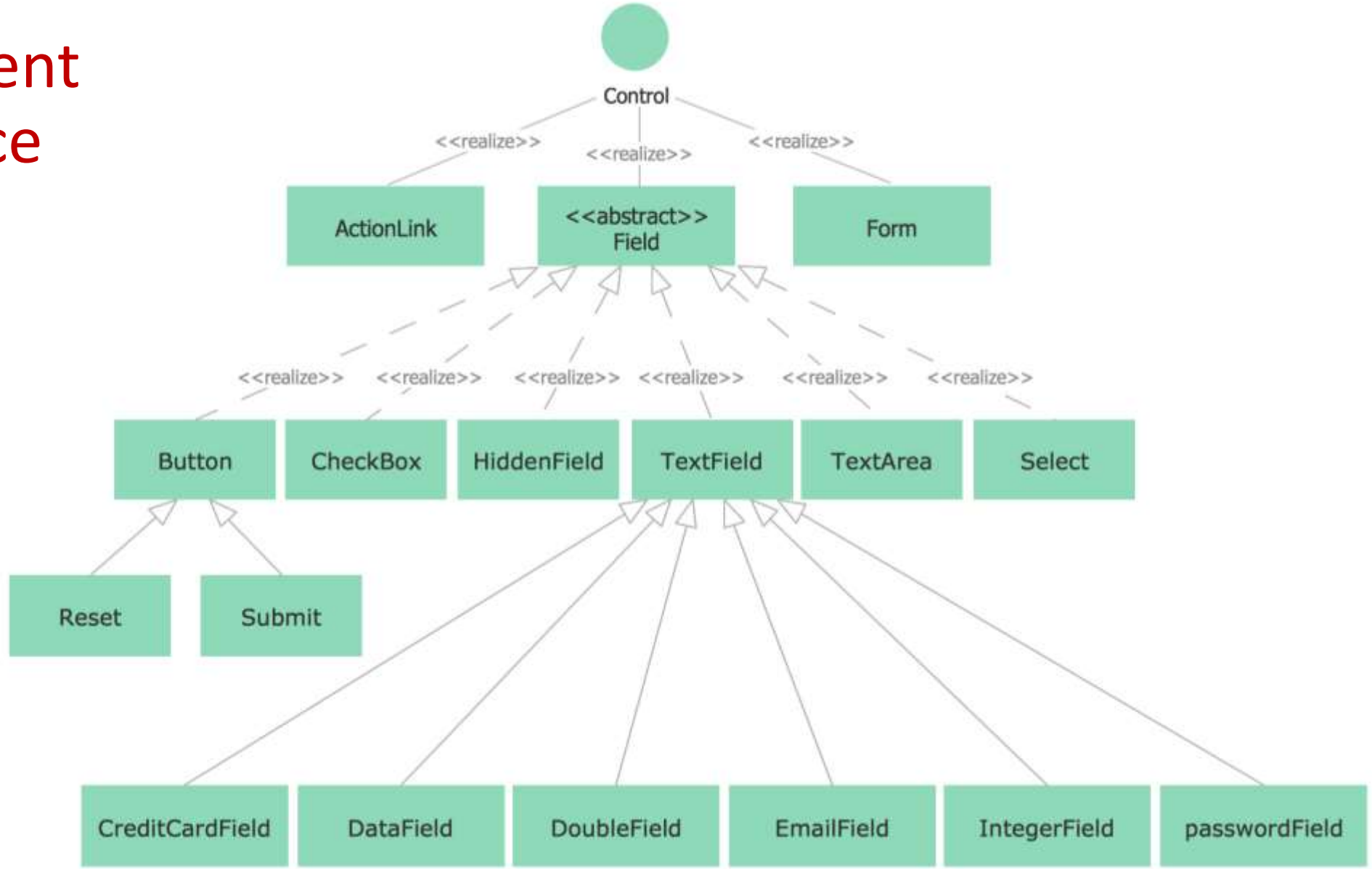
Inheritance – Numbers



Inheritance and Hierarchy



Form Element Inheritance



Selection

Selection Operators

- Selection forms an essential function is all computer programming
- Selection uses a range of operations:
  - `if`
  - `if … else`
  - `if … else if … else`
  - `Switch`
- The following slides demonstrate these operations

`if` Selection

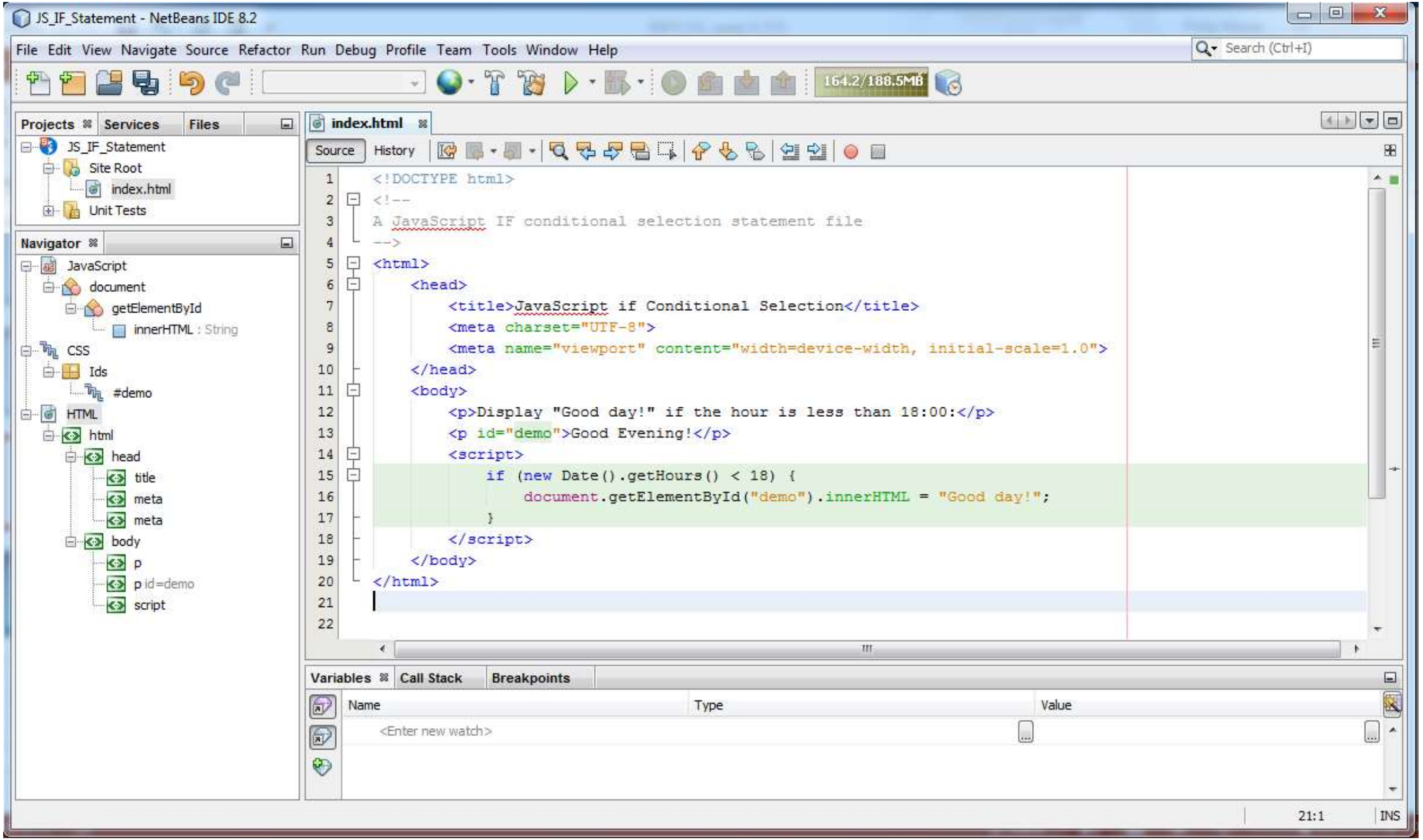
if and **if … else**

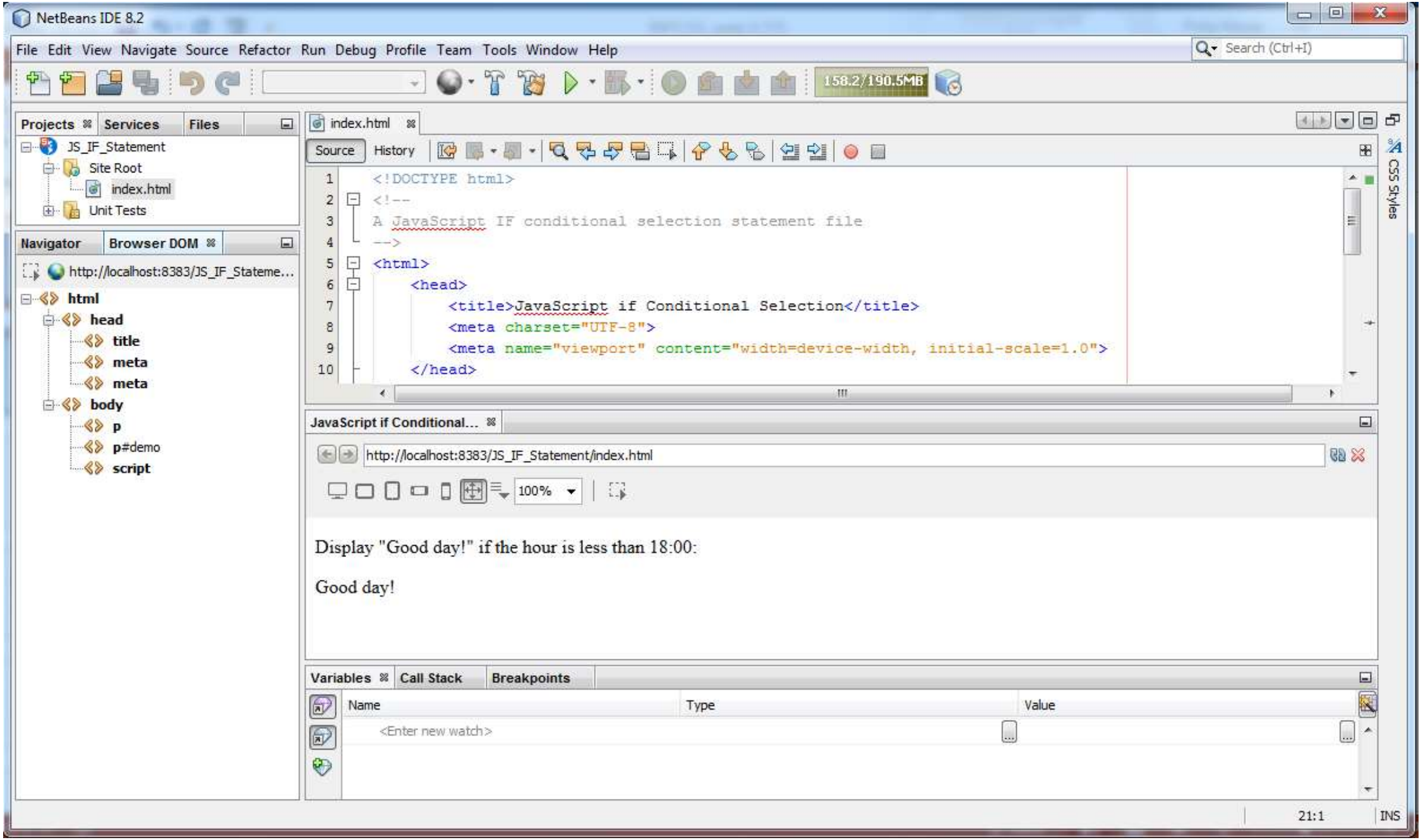
- The following **if … else** conditional statements execute a statement ***IFF*** an expression is ***true***

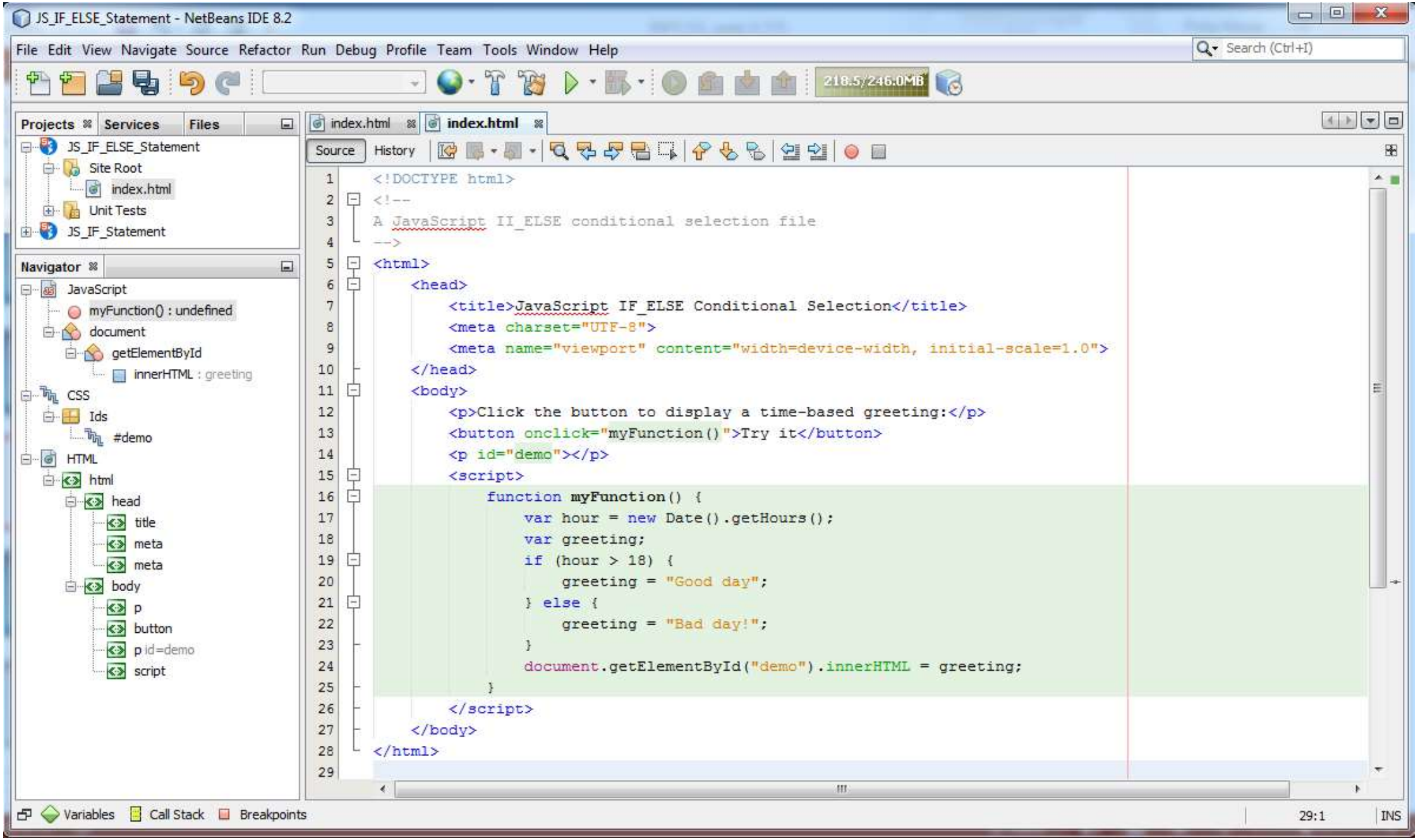
```
if( expression )
    statement
```

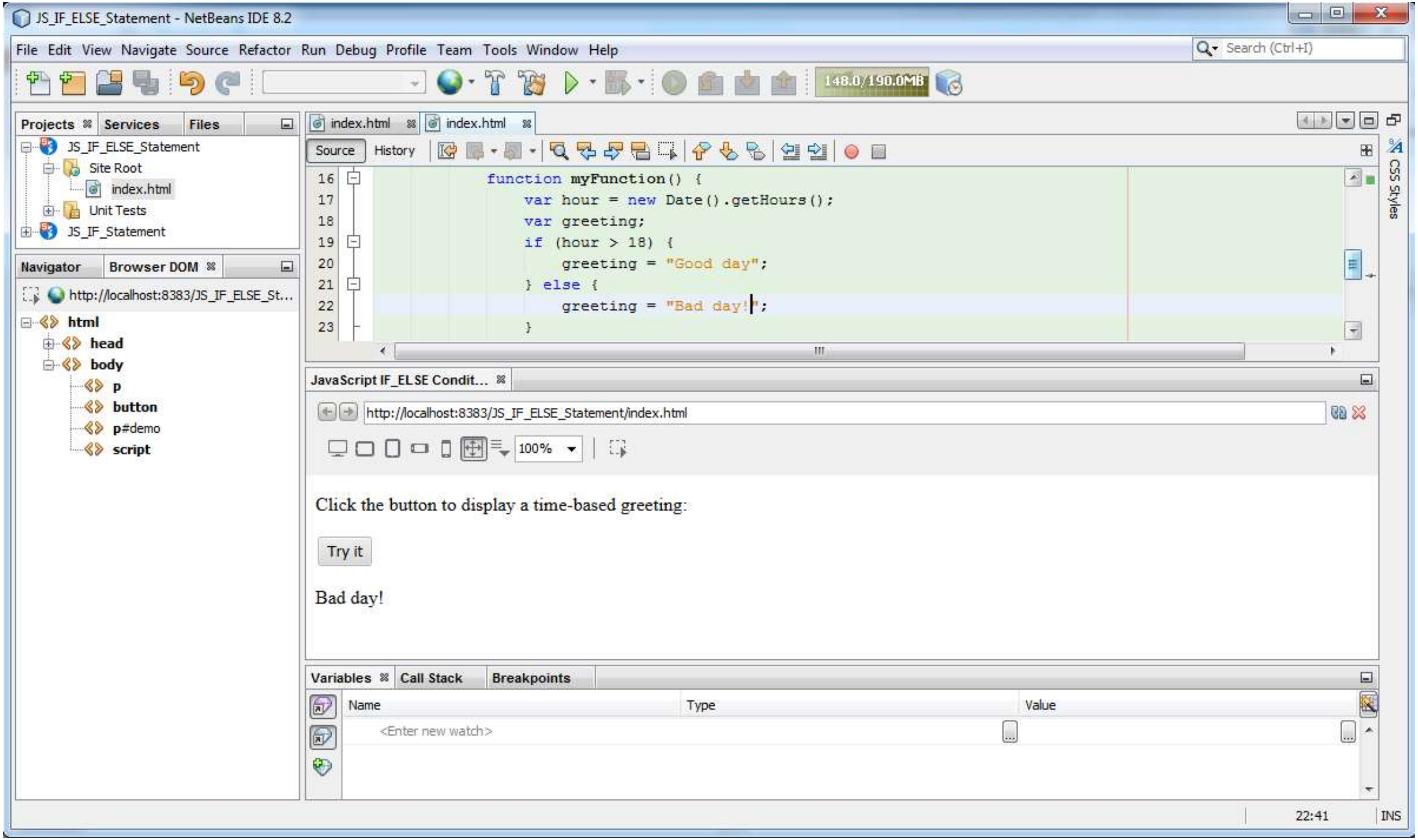
- The following **if … else** conditional statements execute a statement ***IFF*** an expression is ***false***

```
if(expression)
        statement 1
else
        statement 2
```


JS_IF_Statement - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
164.2/188.5MB
Projects
Services
Files
JS_IF_Statement
Site Root
index.html
Unit Tests
Navigator
JavaScript
document
getElementById
innerHTML : String
CSS
Ids
#demo
HTML
html
head
title
meta
meta
body
p
p id=demo
script
index.html
Source
History
1 <!DOCTYPE html>
2 <!--
3 A JavaScript IF conditional selection statement file
4 -->
5 <html>
6 <head>
7 <title>JavaScript if Conditional Selection</title>
8 <meta charset="UTF-8">
9 <meta name="viewport" content="width=device-width, initial-scale=1.0">
10 </head>
11 <body>
12 Display "Good day!" if the hour is less than 18:00:
13 Good Evening!
14 <script>
15 if (new Date().getHours() < 18) {
16 document.getElementById("demo").innerHTML = "Good day!";
17 }
18 </script>
19 </body>
20 </html>
21
22
Variables
Call Stack
Breakpoints
Name
Type
Value
<Enter new watch>
21:1
INS


NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
158.2/190.5MB
Projects
Services
Files
JS_IF_Statement
Site Root
index.html
Unit Tests
Navigator
Browser DOM
http://localhost:8383/JS_IF_Stateme...
html
head
title
meta
meta
body
p
p#demo
script
index.html
Source
History
1 <!DOCTYPE html>
2 <!--
3 A JavaScript IF conditional selection statement file
4 -->
5 <html>
6 <head>
7 <title>JavaScript if Conditional Selection</title>
8 <meta charset="UTF-8">
9 <meta name="viewport" content="width=device-width, initial-scale=1.0">
10 </head>
CSS Styles
JavaScript if Conditional...
http://localhost:8383/JS_IF_Statement/index.html
100%
Display "Good day!" if the hour is less than 18:00:
Good day!
Variables
Call Stack
Breakpoints
Name
Type
Value
<Enter new watch>
21:1
INS


JS_IF_ELSE_Statement - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
218.5/246.0MB
Projects
Services
Files
JS_IF_ELSE_Statement
Site Root
index.html
Unit Tests
JS_IF_Statement
Navigator
JavaScript
myFunction() : undefined
document
getElementById
innerHTML : greeting
CSS
Ids
#demo
HTML
html
head
title
meta
meta
body
p
button
p id=demo
script
index.html
index.html
Source
History
1 <!DOCTYPE html>
2 <!--
3 A JavaScript II_ELSE conditional selection file
4 -->
5 <html>
6 <head>
7 <title>JavaScript IF_ELSE Conditional Selection</title>
8 <meta charset="UTF-8">
9 <meta name="viewport" content="width=device-width, initial-scale=1.0">
10 </head>
11 <body>
12 Click the button to display a time-based greeting:
13 <button onclick="myFunction()">Try it</button>
14
15 <script>
16 function myFunction() {
17 var hour = new Date().getHours();
18 var greeting;
19 if (hour > 18) {
20 greeting = "Good day";
21 } else {
22 greeting = "Bad day!";
23 }
24 document.getElementById("demo").innerHTML = greeting;
25 }
26 </script>
27 </body>
28 </html>
29
Variables
Call Stack
Breakpoints
29:1
INS

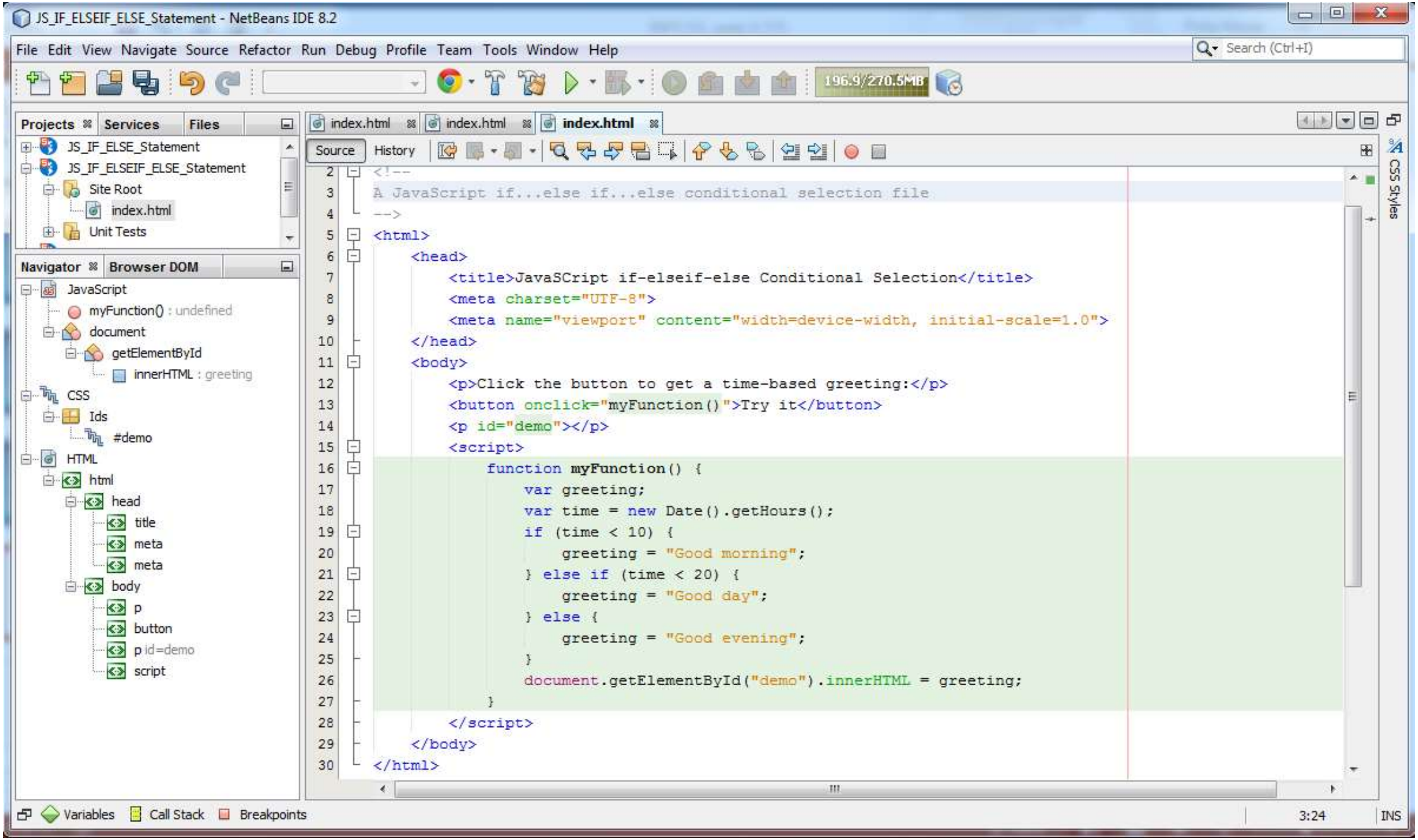

JS_IF_ELSE_Statement - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
148.0/190.0MB
Projects
Services
Files
JS_IF_ELSE_Statement
Site Root
index.html
Unit Tests
JS_IF_Statement
Navigator
Browser DOM
http://localhost:8383/JS_IF_ELSE_St...
html
head
body
p
button
p#demo
script
index.html
index.html
Source
History
16 function myFunction() {
17 var hour = new Date().getHours();
18 var greeting;
19 if (hour > 18) {
20 greeting = "Good day";
21 } else {
22 greeting = "Bad day!";
23 }
CSS Styles
JavaScript IF_ELSE Condit...
http://localhost:8383/JS_IF_ELSE_Statement/index.html
100%
Click the button to display a time-based greeting:
Try it
Bad day!
Variables
Call Stack
Breakpoints
Name
Type
Value
<Enter new watch>
22:41
INS

if … else if … else

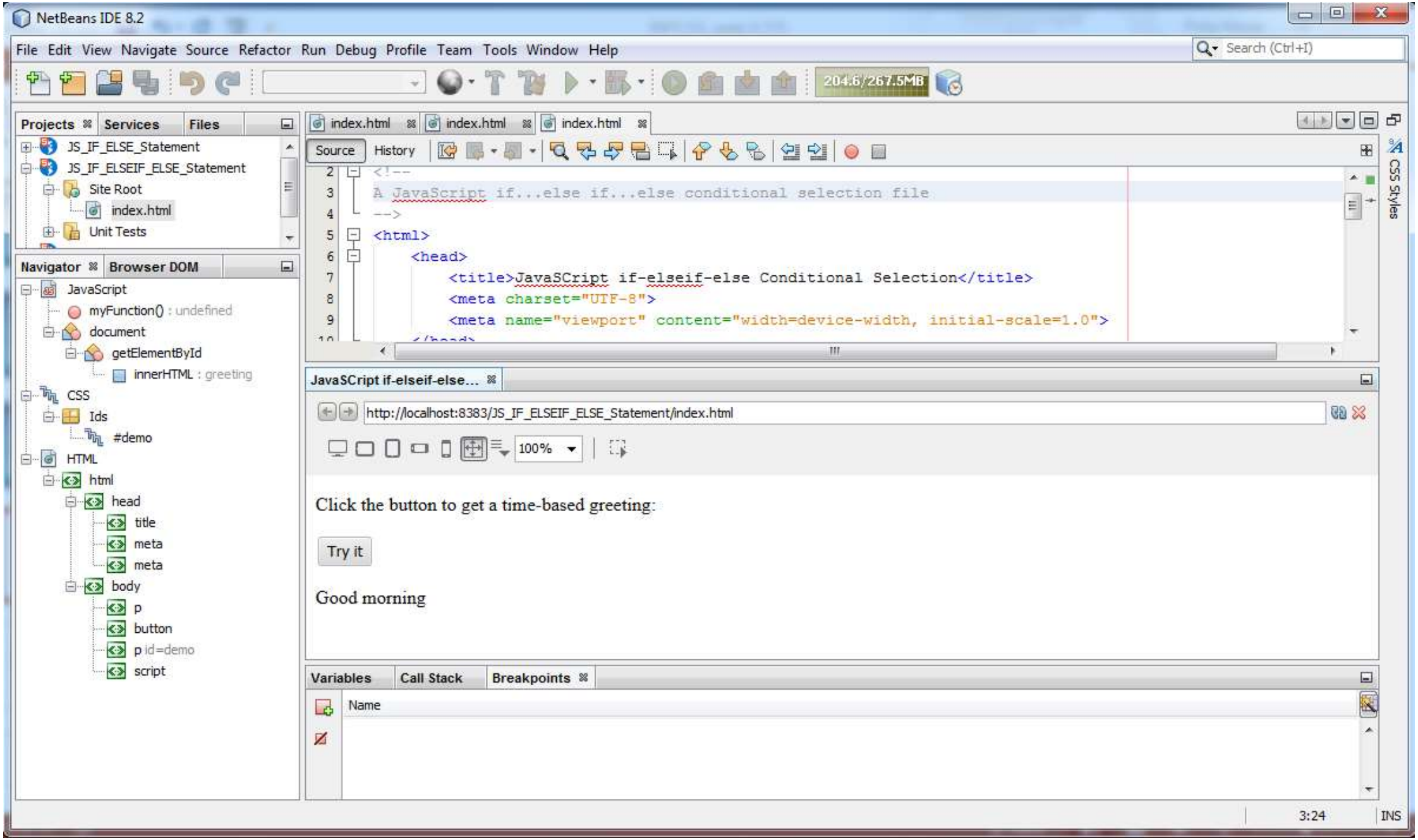
- Any else clause may be combined with a nested **else … if** statement to produce an **else…if** statement as follows:

```
if(expression)
        statement 1
else if
        statement 2
else
        statement 3
```

- The following slides show working examples of the conditional selection operators


JS_IF_ELSEIF_ELSE_Statement - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
196.9/270.5MB
Projects
Services
Files
JS_IF_ELSE_Statement
JS_IF_ELSEIF_ELSE_Statement
Site Root
index.html
Unit Tests
Navigator
Browser DOM
JavaScript
myFunction() : undefined
document
getElementById
innerHTML : greeting
CSS
Ids
#demo
HTML
html
head
title
meta
meta
body
p
button
p id=demo
script
index.html
index.html
index.html
Source
History
CSS Styles
<!--
A JavaScript if...else if...else conditional selection file
-->
<html>
<head>
<title>JavaScript if-elseif-else Conditional Selection</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
Click the button to get a time-based greeting:
<button onclick="myFunction()">Try it</button>

<script>
function myFunction() {
var greeting;
var time = new Date().getHours();
if (time < 10) {
greeting = "Good morning";
} else if (time < 20) {
greeting = "Good day";
} else {
greeting = "Good evening";
}
document.getElementById("demo").innerHTML = greeting;
}
</script>
</body>
</html>
Variables
Call Stack
Breakpoints
3:24
INS


NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
204.6/267.5MB
Projects
Services
Files
JS_IF_ELSE_Statement
JS_IF_ELSEIF_ELSE_Statement
Site Root
index.html
Unit Tests
Navigator
Browser DOM
JavaScript
myFunction() : undefined
document
getElementById
innerHTML : greeting
CSS
Ids
#demo
HTML
html
head
title
meta
meta
body
p
button
p id=demo
script
index.html
index.html
index.html
Source
History
<!--
A JavaScript if...else if...else conditional selection file
-->
<html>
<head>
<title>JavaScript if-elseif-else Conditional Selection</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
CSS Styles
JavaScript if-elseif-else...
http://localhost:8383/JS_IF_ELSEIF_ELSE_Statement/index.html
100%
Click the button to get a time-based greeting:
Try it
Good morning
Variables
Call Stack
Breakpoints
Name
3:24
INS

`if` Selection

- In considering the examples shown it is important to note:
  - In all the examples block curly brackets `{…}` have been used
- For `if` statements:
  - The `{ … }` may be omitted
- For (`if … else`) and (`if … else if … else`) blocks:
  - The block curly brackets `{ … }` **MUST** be used
  - If the block curly brackets `{ … }` are omitted there will be an **ERROR**
- It is good programming practice to always use `{ … }`
  - To improve the readability of the program code and reduce possible errors

`Switch` Selection

Switch Conditional Selection

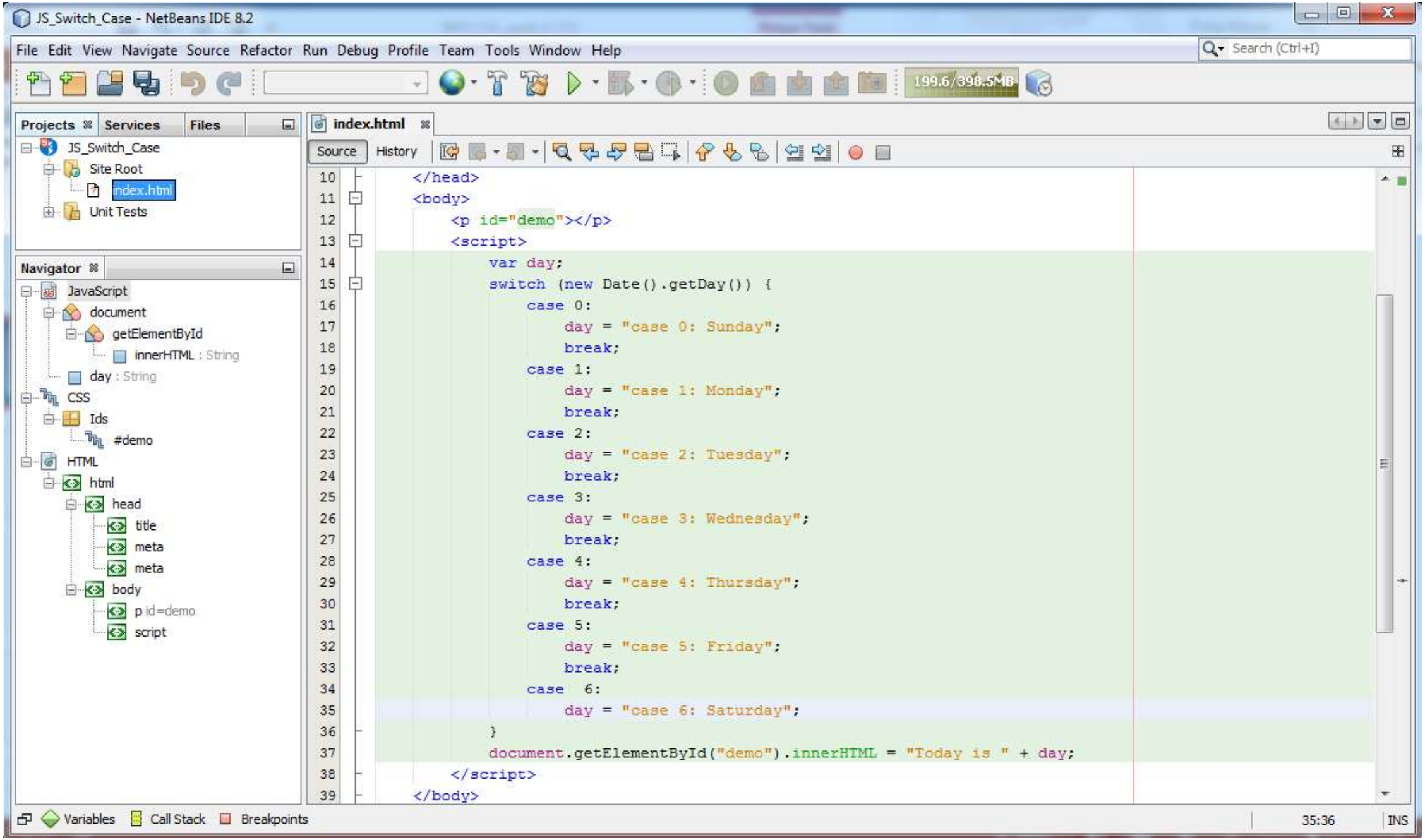
- A switch statement is:
  - A multi-way branch
  - It evaluates an expression and then jumps to a statement that is **`labelled`** with a case clause that matches the value of the expression
  - Where no matching case **`label`** is found the switch statement jumps to the statement (if any) with the **`default label`**
- The **`switch`** syntax is as follows:

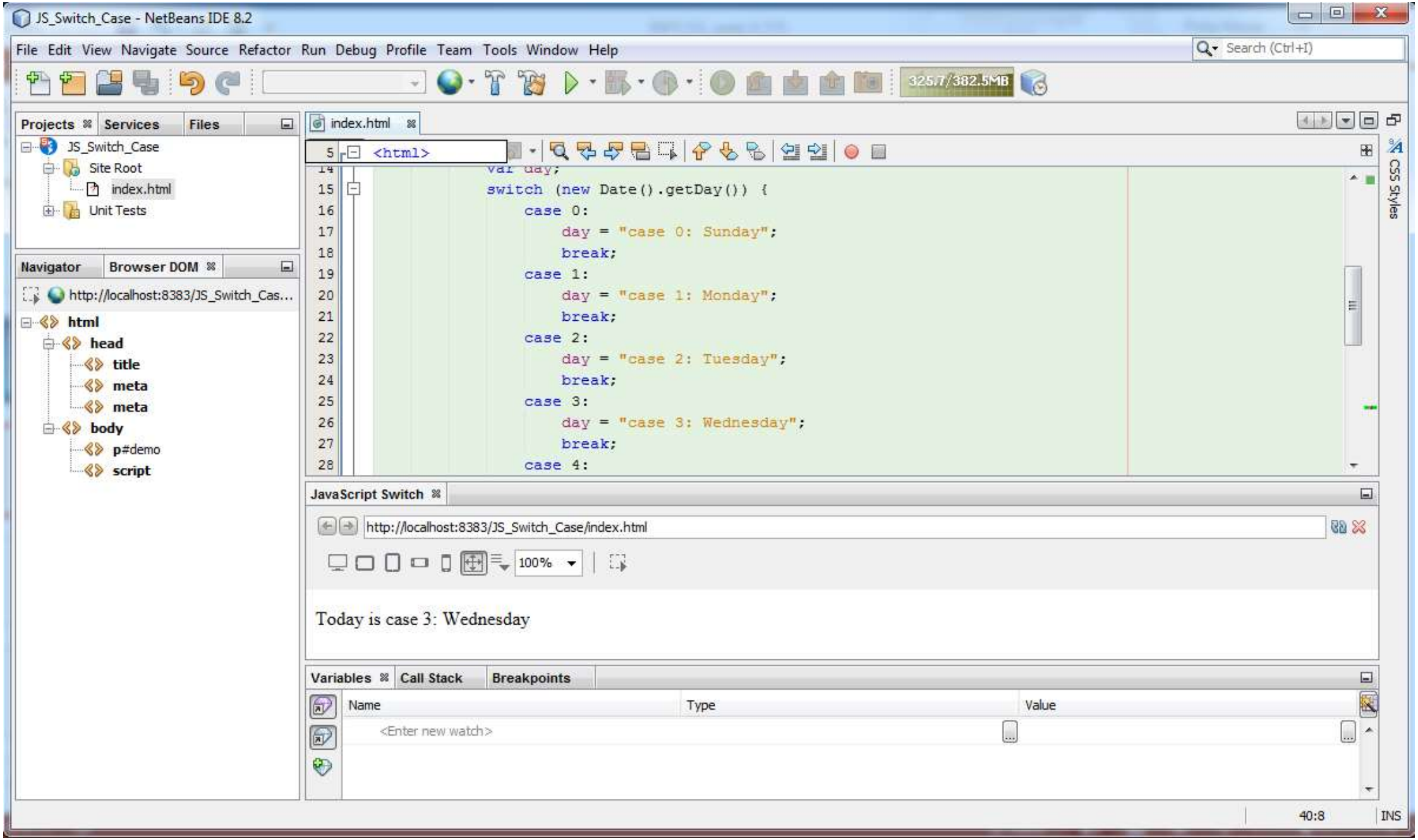
```
switch ( expression ) {
        case constant expression: statements
        [case constant expression: statements]
        [ … ]
        default: statements
}
```

The Switch Syntax

```
switch(expression) {
    case n:
        code block
        break;
    case n:
        code block
        break;
    default:
        code block
}
```

- Important:
  - The *first case* which matches the expression will be executed
  - The *default case* must always be *final* case


JS_Switch_Case - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
199.6/398.5MB
Projects
Services
Files
JS_Switch_Case
Site Root
index.html
Unit Tests
Navigator
JavaScript
document
getElementById
innerHTML : String
day : String
CSS
Ids
#demo
HTML
html
head
title
meta
meta
body
p id=demo
script
index.html
Source
History
10 </head>
11 <body>
12
13 <script>
14 var day;
15 switch (new Date().getDay()) {
16 case 0:
17 day = "case 0: Sunday";
18 break;
19 case 1:
20 day = "case 1: Monday";
21 break;
22 case 2:
23 day = "case 2: Tuesday";
24 break;
25 case 3:
26 day = "case 3: Wednesday";
27 break;
28 case 4:
29 day = "case 4: Thursday";
30 break;
31 case 5:
32 day = "case 5: Friday";
33 break;
34 case 6:
35 day = "case 6: Saturday";
36 }
37 document.getElementById("demo").innerHTML = "Today is " + day;
38 </script>
39 </body>
Variables
Call Stack
Breakpoints
35:36
INS

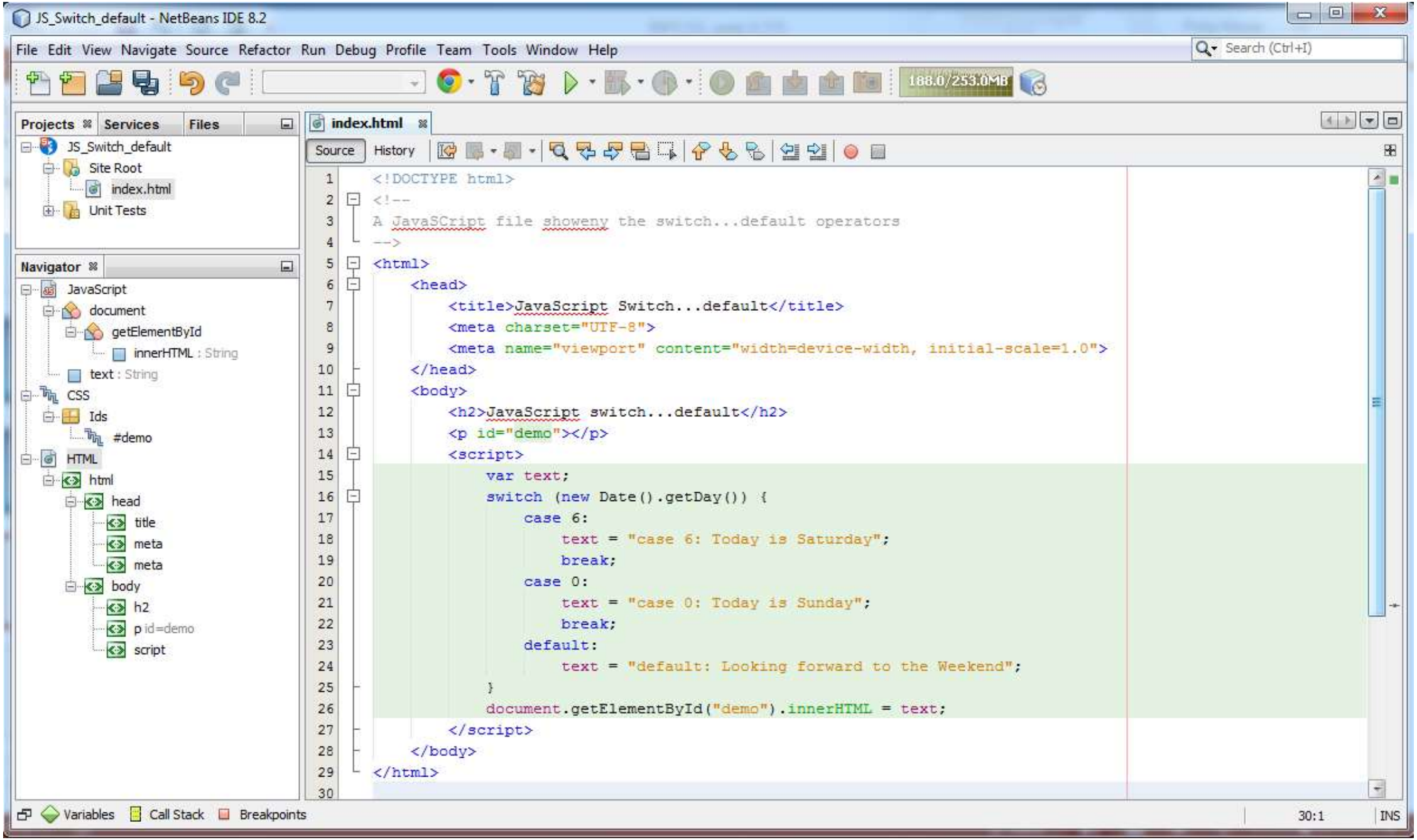

JS_Switch_Case - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
325.7/382.5MB
Projects
Services
Files
JS_Switch_Case
Site Root
index.html
Unit Tests
Navigator
Browser DOM
http://localhost:8383/JS_Switch_Cas...
html
head
title
meta
meta
body
p#demo
script
index.html
5 <html>
14 var day;
15 switch (new Date().getDay()) {
16 case 0:
17 day = "case 0: Sunday";
18 break;
19 case 1:
20 day = "case 1: Monday";
21 break;
22 case 2:
23 day = "case 2: Tuesday";
24 break;
25 case 3:
26 day = "case 3: Wednesday";
27 break;
28 case 4:
CSS Styles
JavaScript Switch
http://localhost:8383/JS_Switch_Case/index.html
100%
Today is case 3: Wednesday
Variables
Call Stack
Breakpoints
Name
Type
Value
<Enter new watch>
40:8
INS

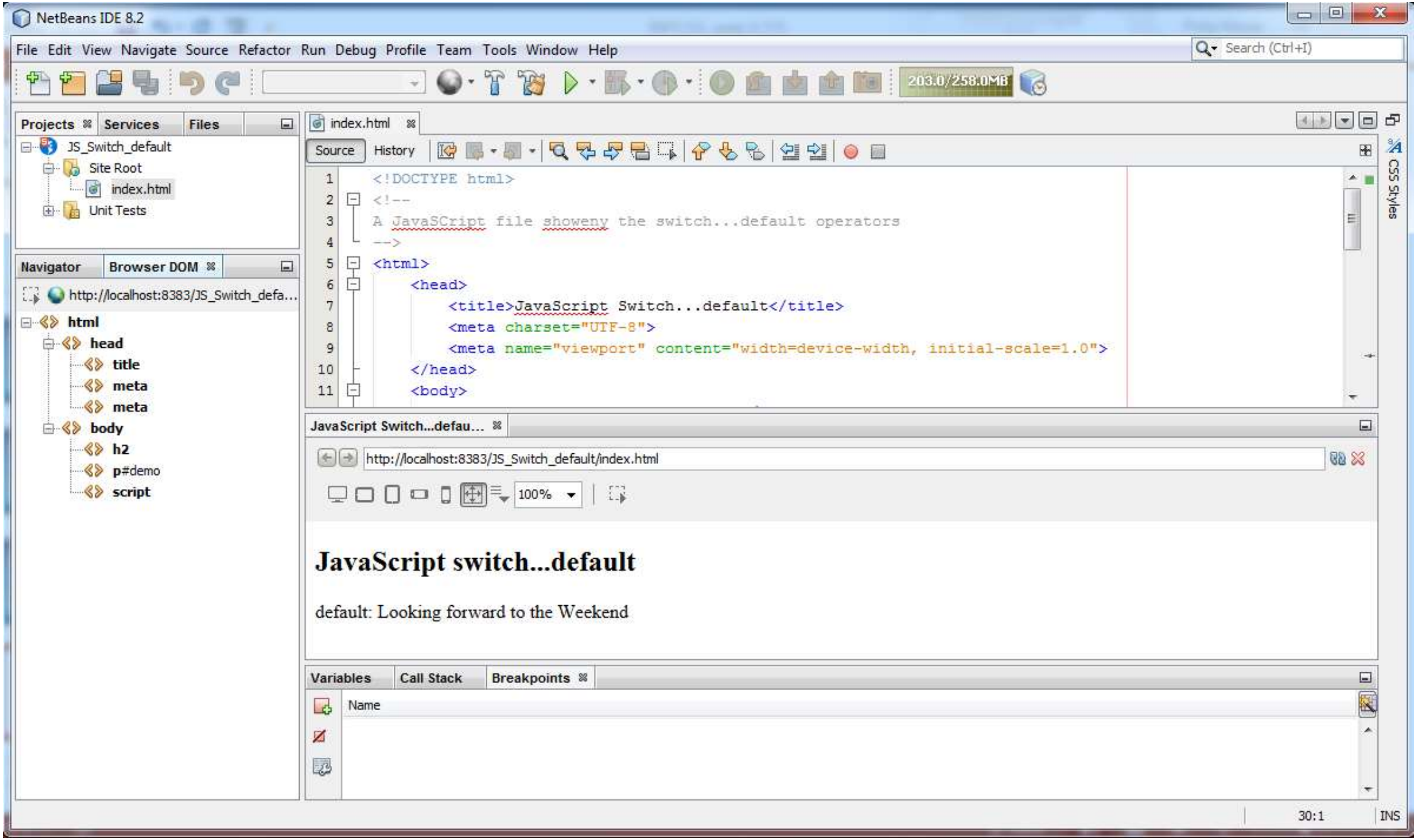
The Switch Case `break` Operator (1)

- A switch statement may include the **`break`** operator
- When JavaScript reaches a **`break`** operator
  - It breaks out of the switch block and the code processing
  - Proceeds to the next line of JavaScript code (or)
  - Returns the result
- The **`break`** operator improves the efficiency of the code as it stops the execution redundant testing of following cases inside the **`switch`** block

The Switch `break` Operator (2)

- The efficiency of the program is improved because:
  - When an *expression* has been satisfied the evaluation is complete there is no need for more switch blocks to be evaluated
- In computer programs:
  - The `switch` statement can
    - Output text
    - Call (and in some languages such as Java create) functions
    - Create objects
  - Any legal (in a programming language) statement may be implemented in a `switch` block


JS_Switch_default - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
188.0/253.0MB
Projects
Services
Files
JS_Switch_default
Site Root
index.html
Unit Tests
Navigator
JavaScript
document
getElementById
innerHTML : String
text : String
CSS
Ids
#demo
HTML
html
head
title
meta
meta
body
h2
p id=demo
script
index.html
Source
History
1 <!DOCTYPE html>
2 <!--
3 A JavaSCript file showeny the switch...default operators
4 -->
5 <html>
6 <head>
7 <title>JavaScript Switch...default</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 switch...default</h2>
13
14 <script>
15 var text;
16 switch (new Date().getDay()) {
17 case 6:
18 text = "case 6: Today is Saturday";
19 break;
20 case 0:
21 text = "case 0: Today is Sunday";
22 break;
23 default:
24 text = "default: Looking forward to the Weekend";
25 }
26 document.getElementById("demo").innerHTML = text;
27 </script>
28 </body>
29 </html>
30
Variables
Call Stack
Breakpoints
30:1
INS

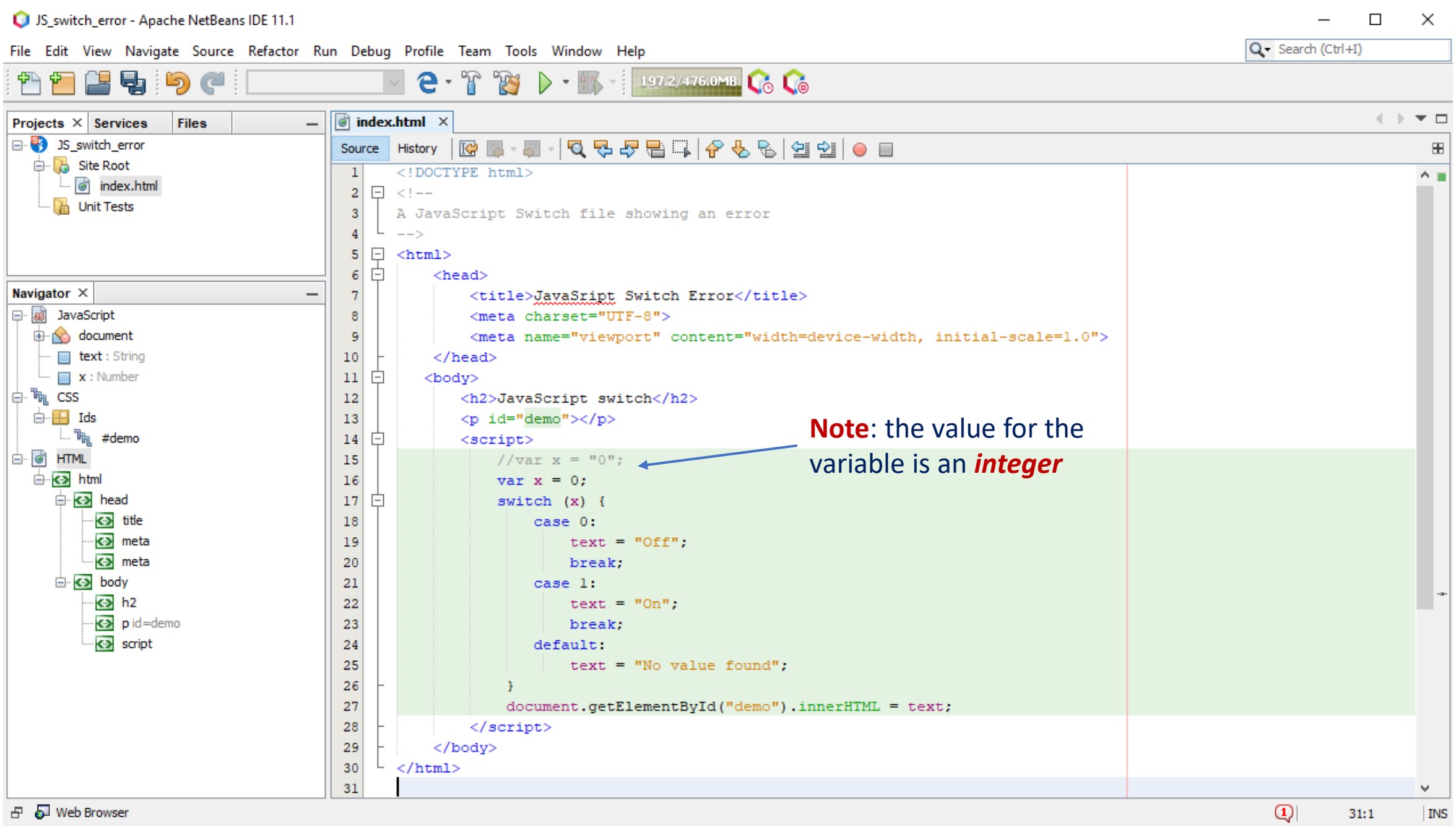

NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
203.0/258.0MB
Projects
Services
Files
JS_Switch_default
Site Root
index.html
Unit Tests
Navigator
Browser DOM
http://localhost:8383/JS_Switch_defa...
html
head
title
meta
meta
body
h2
p#demo
script
index.html
Source
History
1 <!DOCTYPE html>
2 <!--
3 A JavaSCript file showeny the switch...default operators
4 -->
5 <html>
6 <head>
7 <title>JavaScript Switch...default</title>
8 <meta charset="UTF-8">
9 <meta name="viewport" content="width=device-width, initial-scale=1.0">
10 </head>
11 <body>
CSS Styles
JavaScript Switch...defau...
http://localhost:8383/JS_Switch_default/index.html
100%
JavaScript switch...default
default: Looking forward to the Weekend
Variables
Call Stack
Breakpoints
Name
30:1
INS

The Switch `default` Operator

- A switch statement may include the **`default`** operator
- When JavaScript reaches **`default`** the processing of the input is stopped
  - The **`default`** is generally located at the end of the switch block
  - The **`default`** is not always located at the end of the switch block
- The **`default`**
  - Will be used when all other cases have been tested
  - The **`default`** may also be used to catch errors

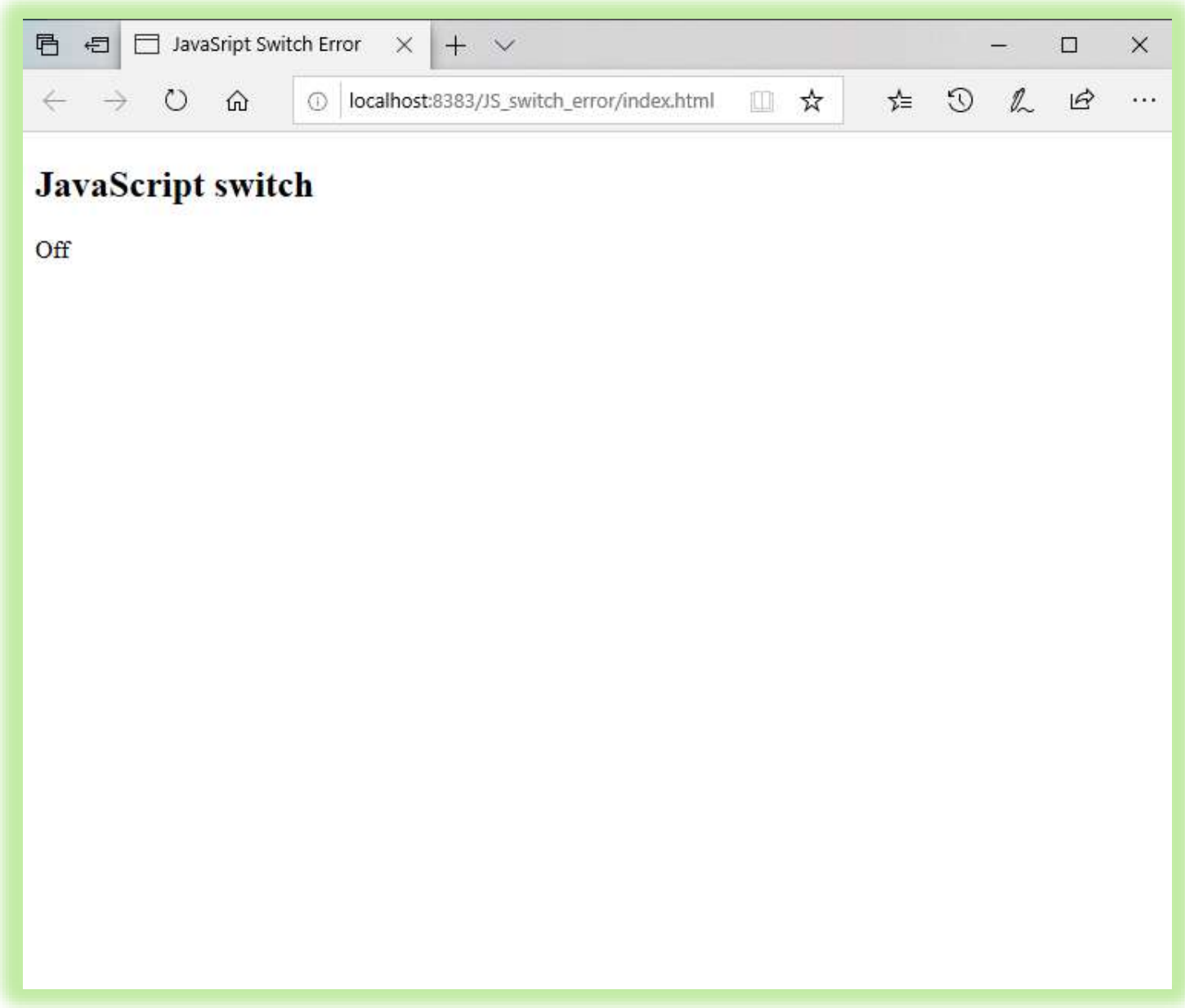
The `Switch` Statement

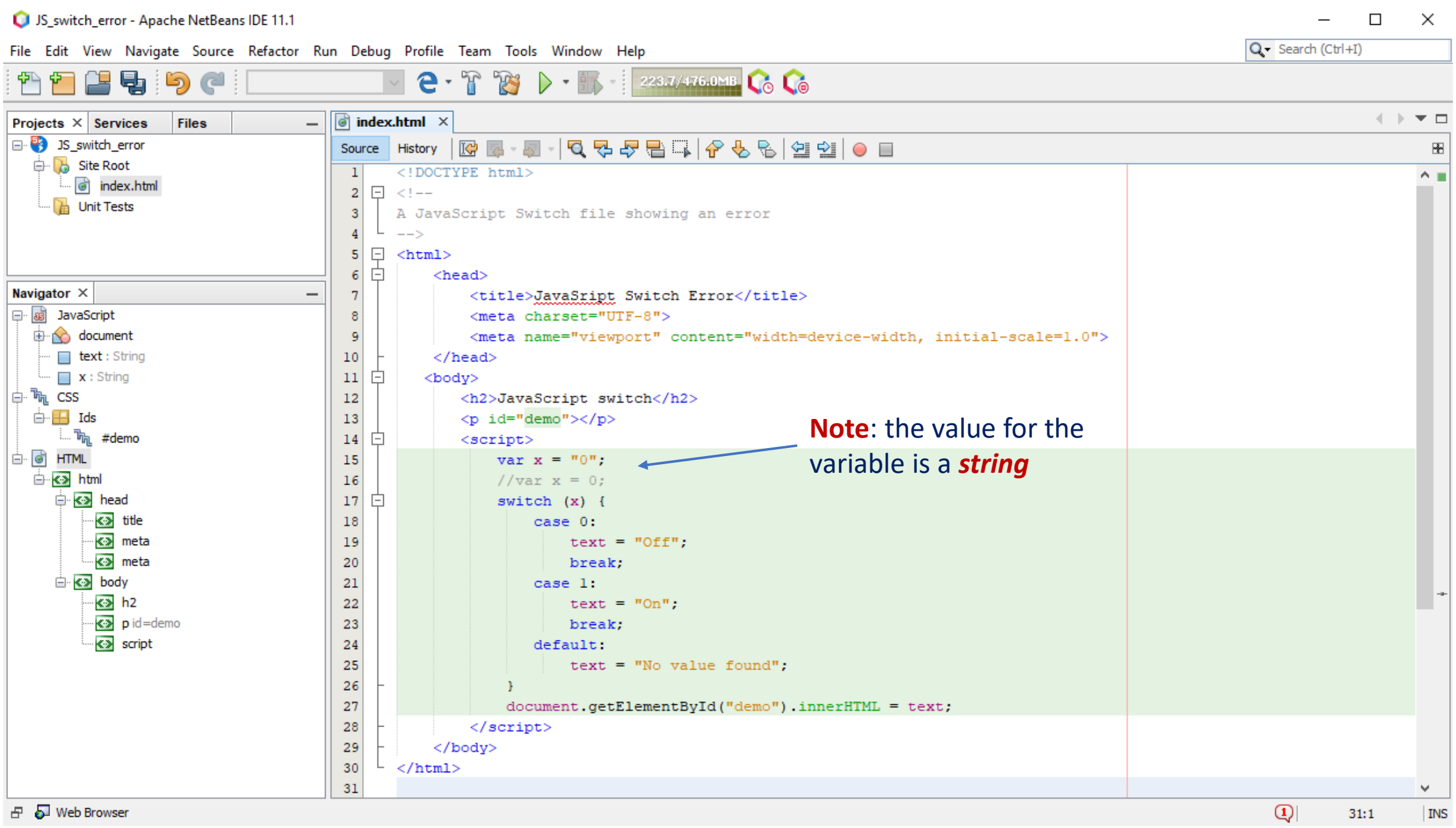
- Switch operates a *strict* Comparison
  - Switch cases the *strict* comparison operator (`===`)
  - The values must be of the same type to match
- A strict comparison can only be true if the operands are of the same type
- In the following examples show the correct value for the variable x and the incorrect value for the variable x
  - The results show the resulting output
  - The output is *Off* and *No value found*


JS_switch_error - Apache NetBeans IDE 11.1
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
197.2/476.0MB
Projects
Services
Files
JS_switch_error
Site Root
index.html
Unit Tests
Navigator
JavaScript
document
text : String
x : Number
CSS
Ids
#demo
HTML
html
head
title
meta
meta
body
h2
p id=demo
script
index.html
Source
History
1 <!DOCTYPE html>
2 <!--
3 A JavaScript Switch file showing an error
4 -->
5 <html>
6 <head>
7 <title>JavaScript Switch Error</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 switch</h2>
13
14 <script>
15 //var x = "0";
16 var x = 0;
17 switch (x) {
18 case 0:
19 text = "Off";
20 break;
21 case 1:
22 text = "On";
23 break;
24 default:
25 text = "No value found";
26 }
27 document.getElementById("demo").innerHTML = text;
28 </script>
29 </body>
30 </html>
31
Note: the value for the variable is an integer
Web Browser
31:1
INS

The Output

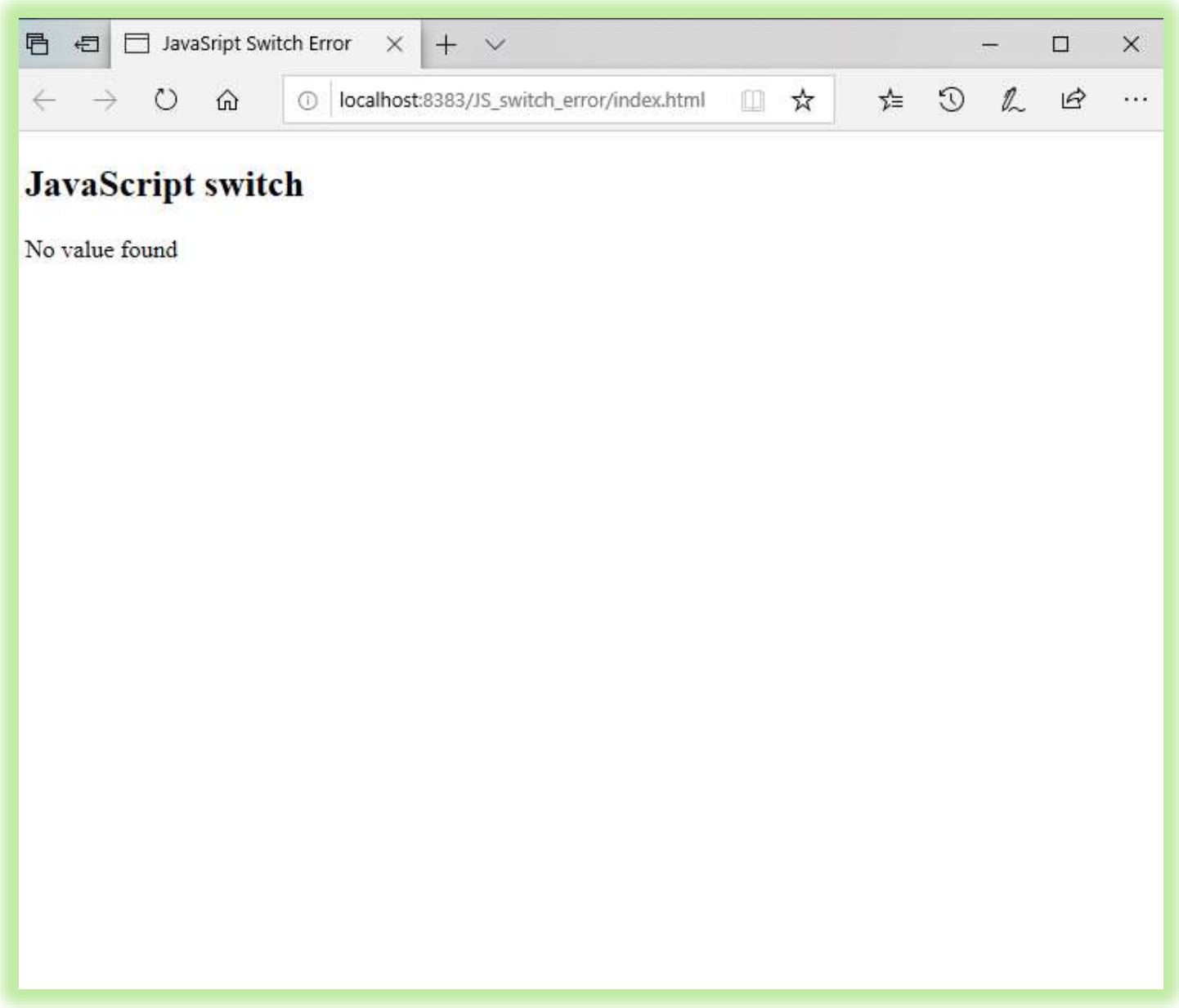
- The output shown in the Microsoft Edge browser
- The result shows:
  - The output using an *integer* as the value for the variable




JS_switch_error - Apache NetBeans IDE 11.1
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
223.7/476.0MB
Projects
Services
Files
JS_switch_error
Site Root
index.html
Unit Tests
Navigator
JavaScript
document
text : String
x : String
CSS
Ids
#demo
HTML
html
head
title
meta
meta
body
h2
p id=demo
script
index.html
Source
History
1 <!DOCTYPE html>
2 <!--
3 A JavaScript Switch file showing an error
4 -->
5 <html>
6 <head>
7 <title>JavaScript Switch Error</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 switch</h2>
13
14 <script>
15 var x = "0";
16 //var x = 0;
17 switch (x) {
18 case 0:
19 text = "Off";
20 break;
21 case 1:
22 text = "On";
23 break;
24 default:
25 text = "No value found";
26 }
27 document.getElementById("demo").innerHTML = text;
28 </script>
29 </body>
30 </html>
31
Note: the value for the variable is a string
Web Browser
31:1
INS

The Output

- The output shown in the Microsoft Edge browser
- The result shows:
  - The output using a string as the value for the variable
  - The result shows the error in the variable value



Iteration (loops)

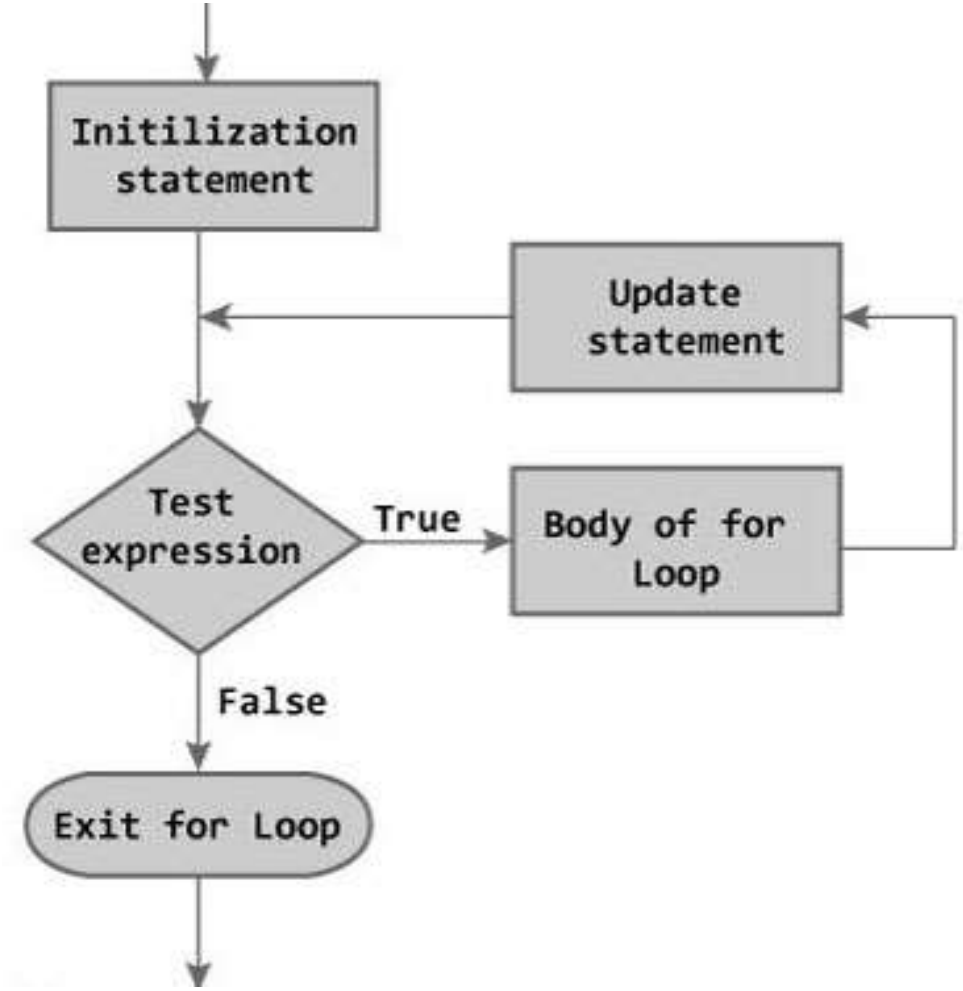
`for` / `for…in` / `while` / `do…while`

Iteration Operations

- Iteration operations fall under four approaches
  - A **`for`** loop
    - combines the *initialisation* and *increment expressions* with the loop *conditional* expression
  - A **`for…in`** loop
    - loops through the properties of a specified object
  - A **`while`** loop
    - statement is a basic loop which repeatedly executes a *statement* while an *expression* is **`true`**
  - A **`do while`** loop
    - Repeatedly executes a statement while an expression is **`true`**
    - It is similar to the **`while`** loop ***except that*** the *loop condition* appears (and is *tested*) at the bottom of the loop
    - This means that the body of the loop is always executed at least once
- The following slides illustrate the basic syntax with worked examples

For

For Loop Logic Model



`for` loop

- The syntax for a **`for`** loop is as follows:

```
for ( initialise ; test ; update){
    statement
}
```

- The **`for`** loop:
  - Repeatedly executes the { *statement* } while the *test* expression is **`true`**
- The **`for`** loop evaluates the *initialise* expression once before starting the loop
- The **`for`** loop then evaluates the *update* expression at the termination of each iteration

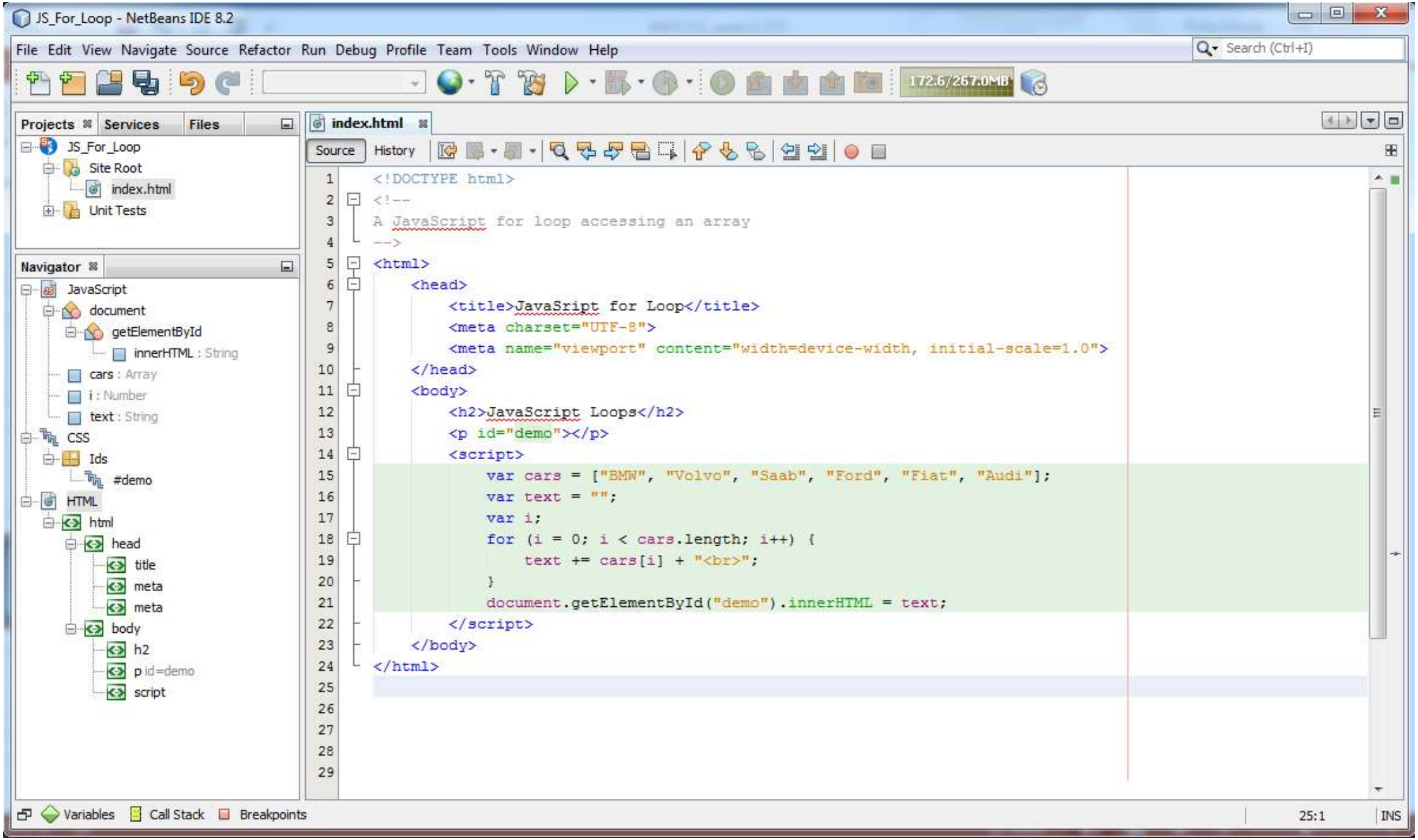
`for` loop

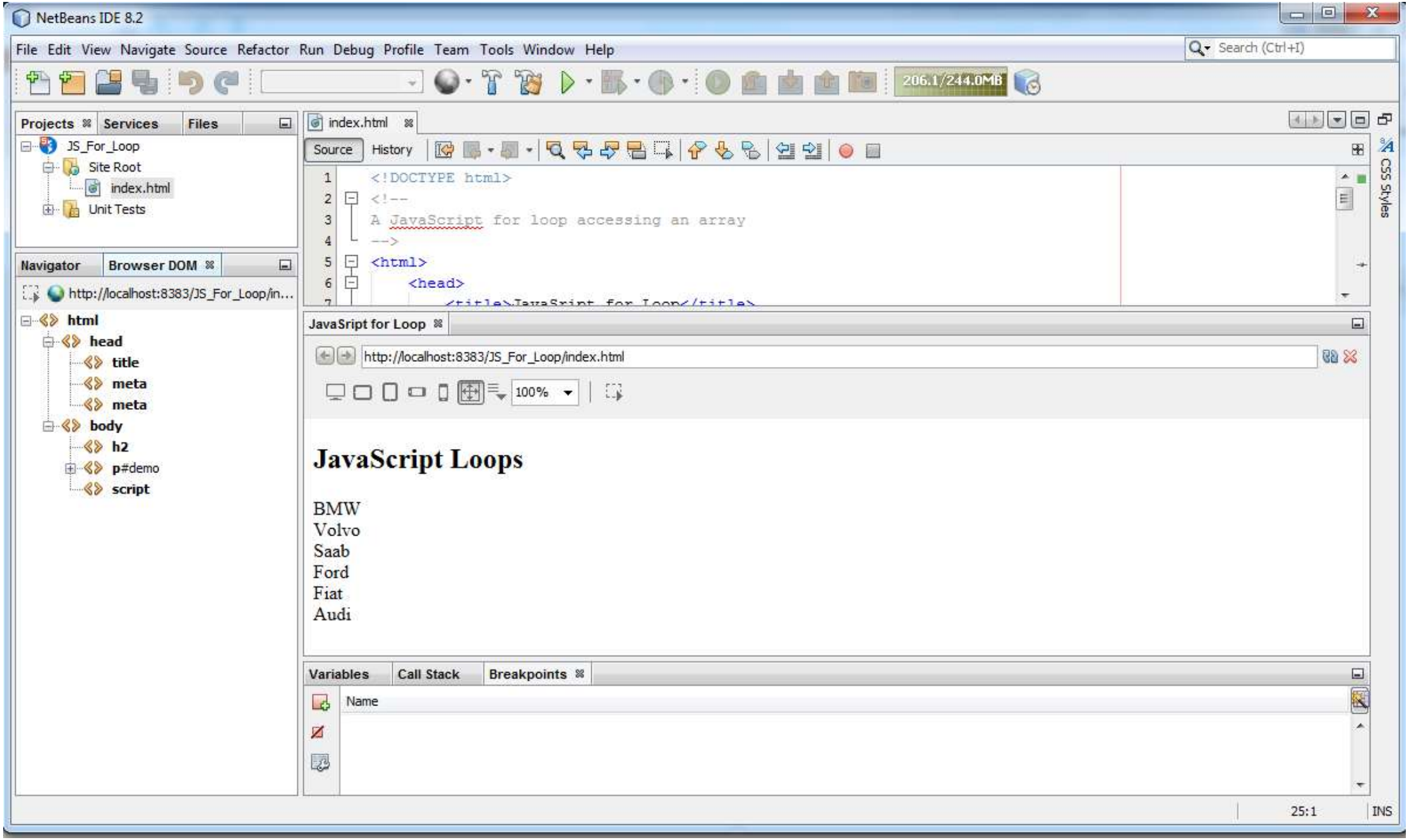
- For example the two code snippets produce the same result
- Individual array element access

```
text += cars[0] + "<br>";
text += cars[1] + "<br>";
text += cars[2] + "<br>";
```

- Accessing array elements using a loop

```
for (i = 0; i < cars.length; i++) {
    text += cars[i] + "<br>";
}
```


JS_For_Loop - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
172.6/267.0MB
Projects
Services
Files
JS_For_Loop
Site Root
index.html
Unit Tests
Navigator
JavaScript
document
getElementById
innerHTML : String
cars : Array
i : Number
text : String
CSS
Ids
#demo
HTML
html
head
title
meta
meta
body
h2
p id=demo
script
index.html
Source
History
1 <!DOCTYPE html>
2 <!--
3 A JavaScript for loop accessing an array
4 -->
5 <html>
6 <head>
7 <title>JavaScript for Loop</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 Loops</h2>
13
14 <script>
15 var cars = ["BMW", "Volvo", "Saab", "Ford", "Fiat", "Audi"];
16 var text = "";
17 var i;
18 for (i = 0; i < cars.length; i++) {
19 text += cars[i] + "
";
20 }
21 document.getElementById("demo").innerHTML = text;
22 </script>
23 </body>
24 </html>
25
26
27
28
29
Variables
Call Stack
Breakpoints
25:1
INS


NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
206.1/244.0MB
Projects
Services
Files
JS_For_Loop
Site Root
index.html
Unit Tests
Navigator
Browser DOM
http://localhost:8383/JS_For_Loop/in...
html
head
title
meta
meta
body
h2
p#demo
script
index.html
Source
History
1 <!DOCTYPE html>
2 <!--
3 A JavaScript for loop accessing an array
4 -->
5 <html>
6 <head>
CSS Styles
JavaSript for Loop
http://localhost:8383/JS_For_Loop/index.html
100%
JavaScript Loops
BMW
Volvo
Saab
Ford
Fiat
Audi
Variables
Call Stack
Breakpoints
Name
25:1
INS

`for…in` Loop

- JavaScript uses a **`for…in`** loop
- The syntax is different from a normal **`for`** loop
- The prototype syntax for a **`for…in`** loop is as follows:

```
for ( variable in object ){
    statement
}
```

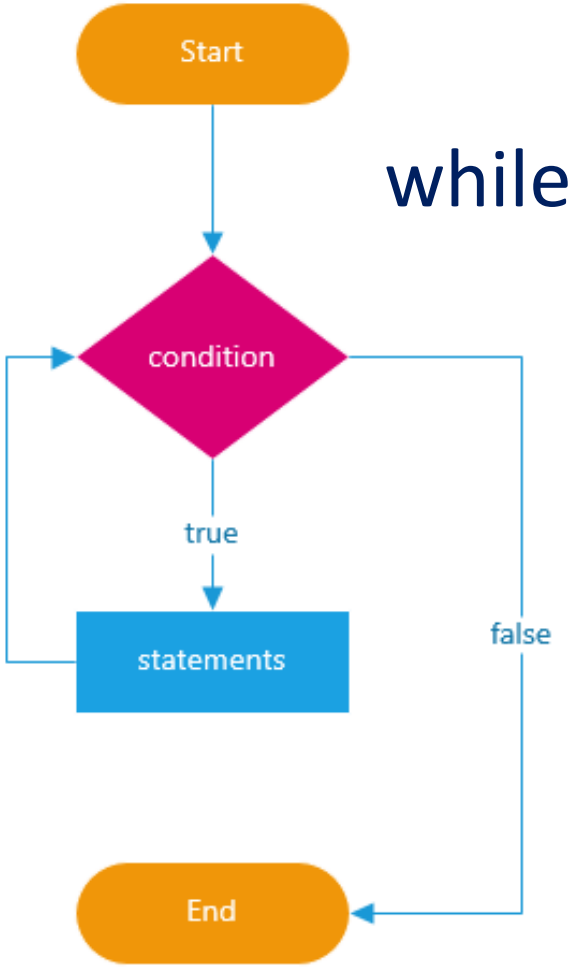
- The following worked example demonstrates the **`for … in`** loop

`For … in` Loop

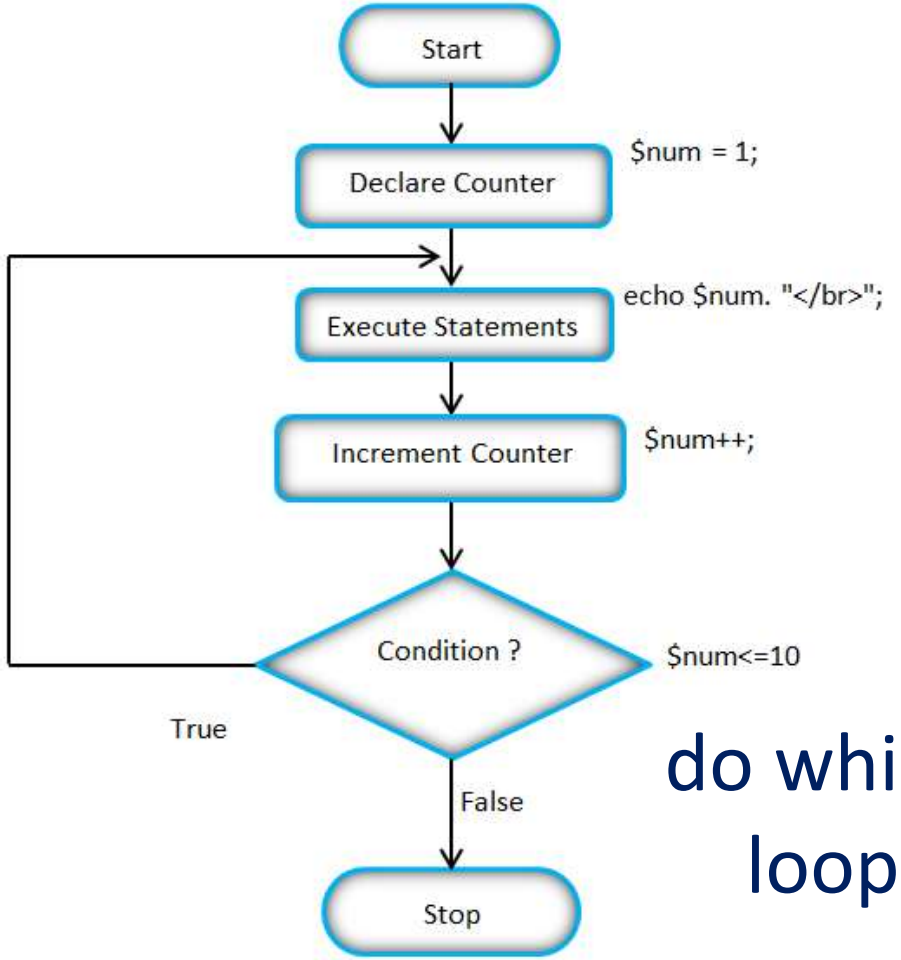
- The `for … in` loop iterates over an object:
  - It executes a statement once for each property in an object
  - Each time through the `for … in` loop is assigns the name of the current property to a specified variable
  - Some properties of pre-defined JavaScript objects are not enumerated by the `for … in` loop
  - User defined properties are always enumerated
- The following example demonstrates the `for … in` loop

While

While Loop Logic Models



while loop



do while loop

`while` / `do…while` Loops

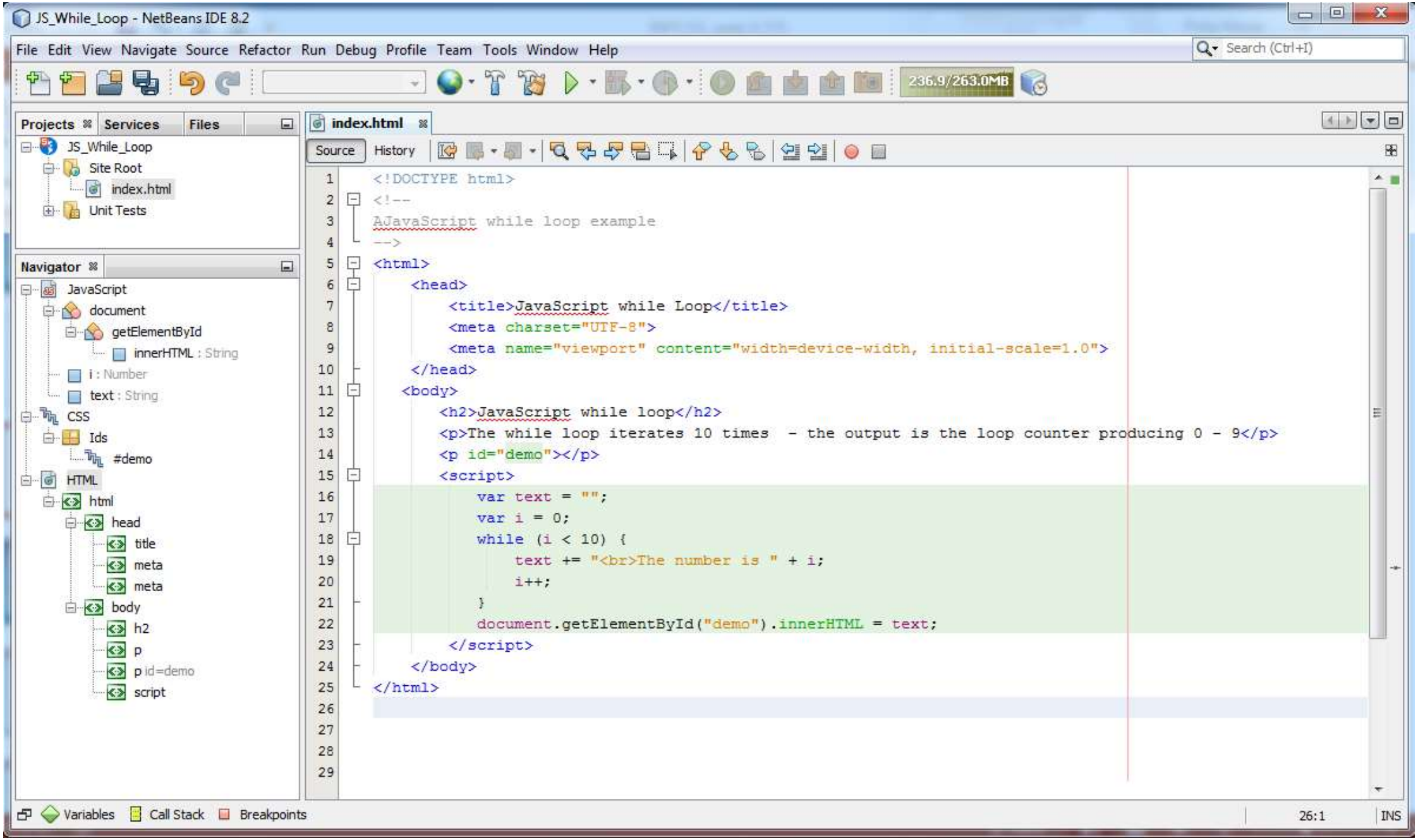
- The syntax for a **`while`** loop is as follows

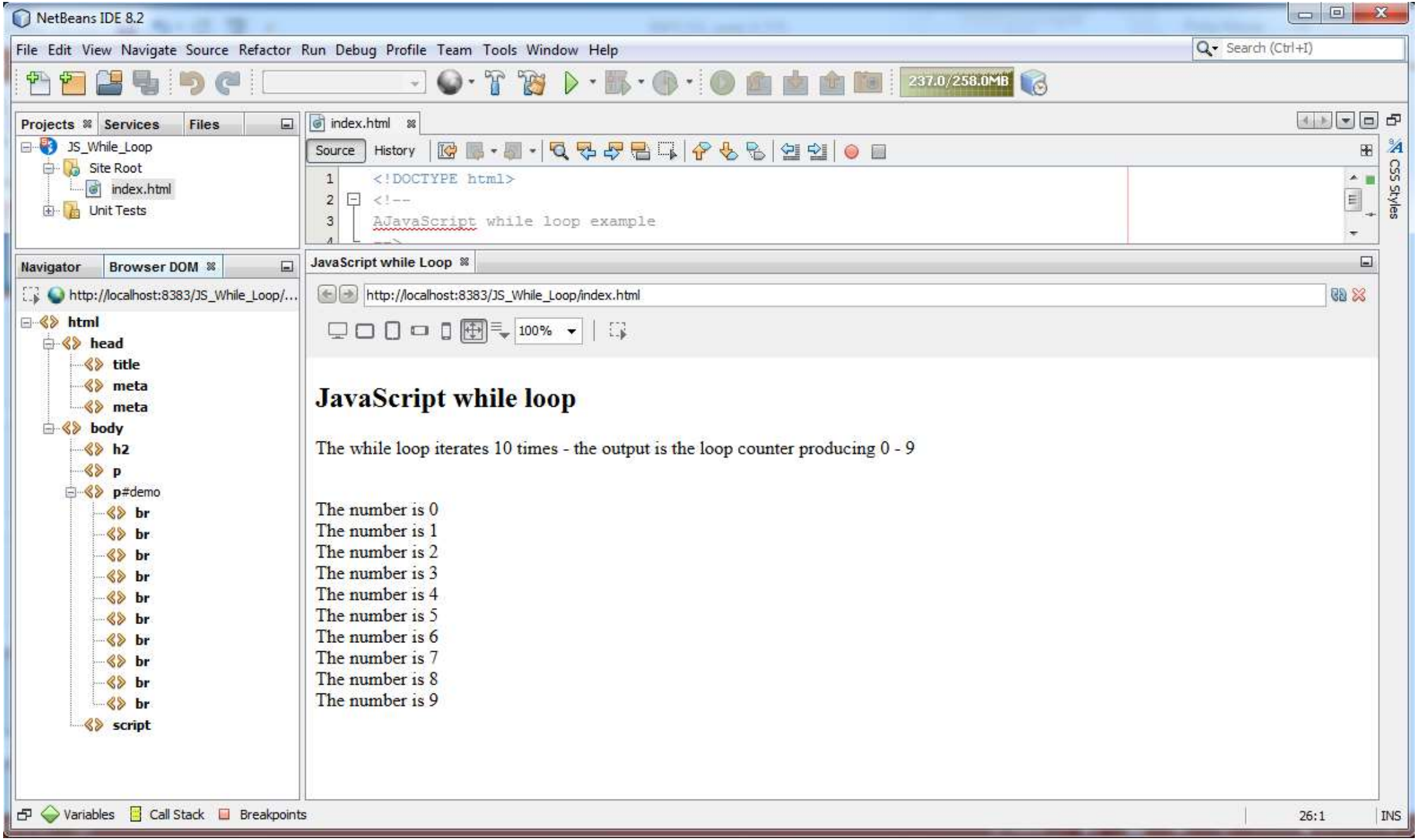
```
while ( expression ){
    statement
}
```

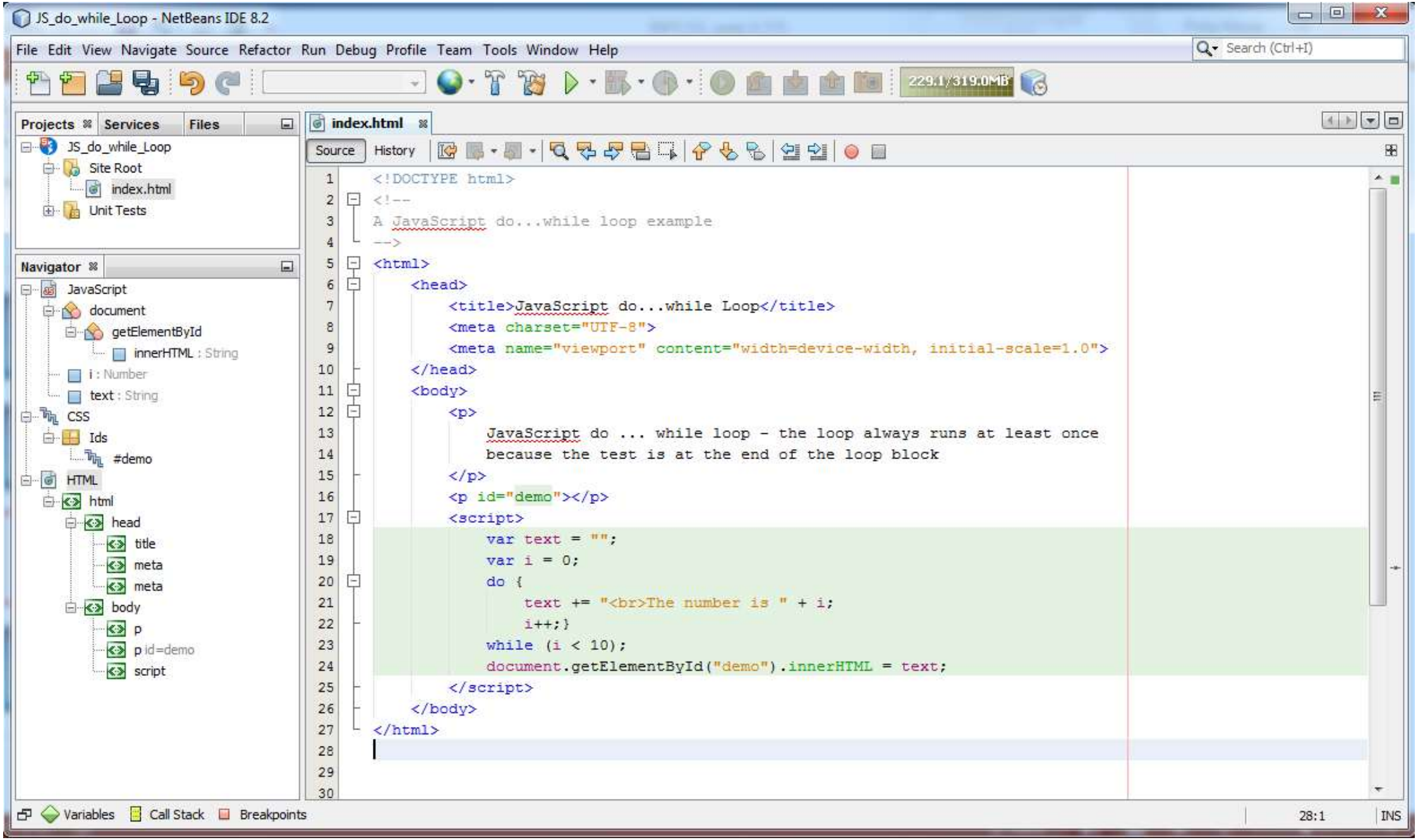
- The syntax for a **`do…while`** loop is as follows

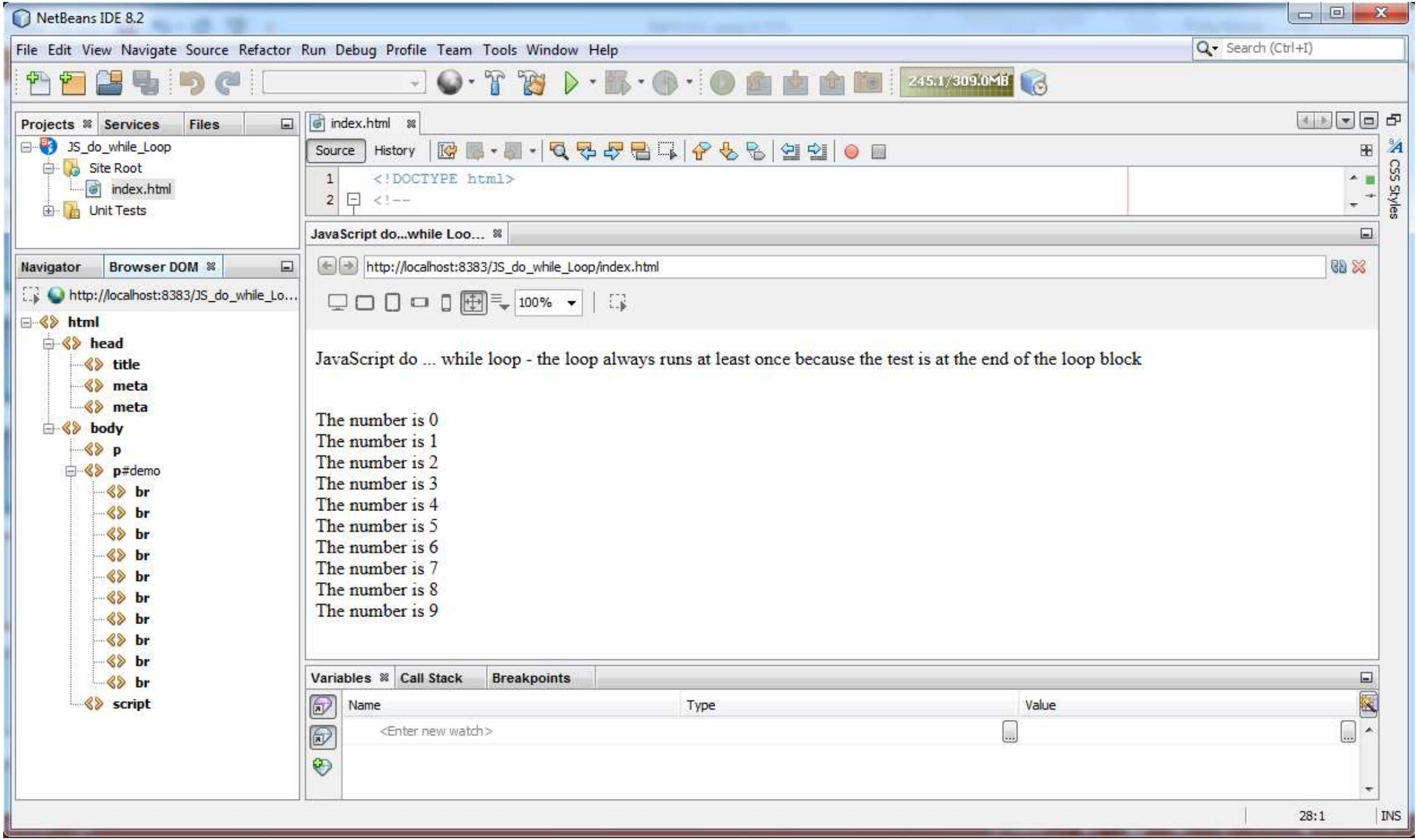
```
do{
    statement
} while ( expression );
```

- The following worked examples show the **`while`** and **`do…while`** loops

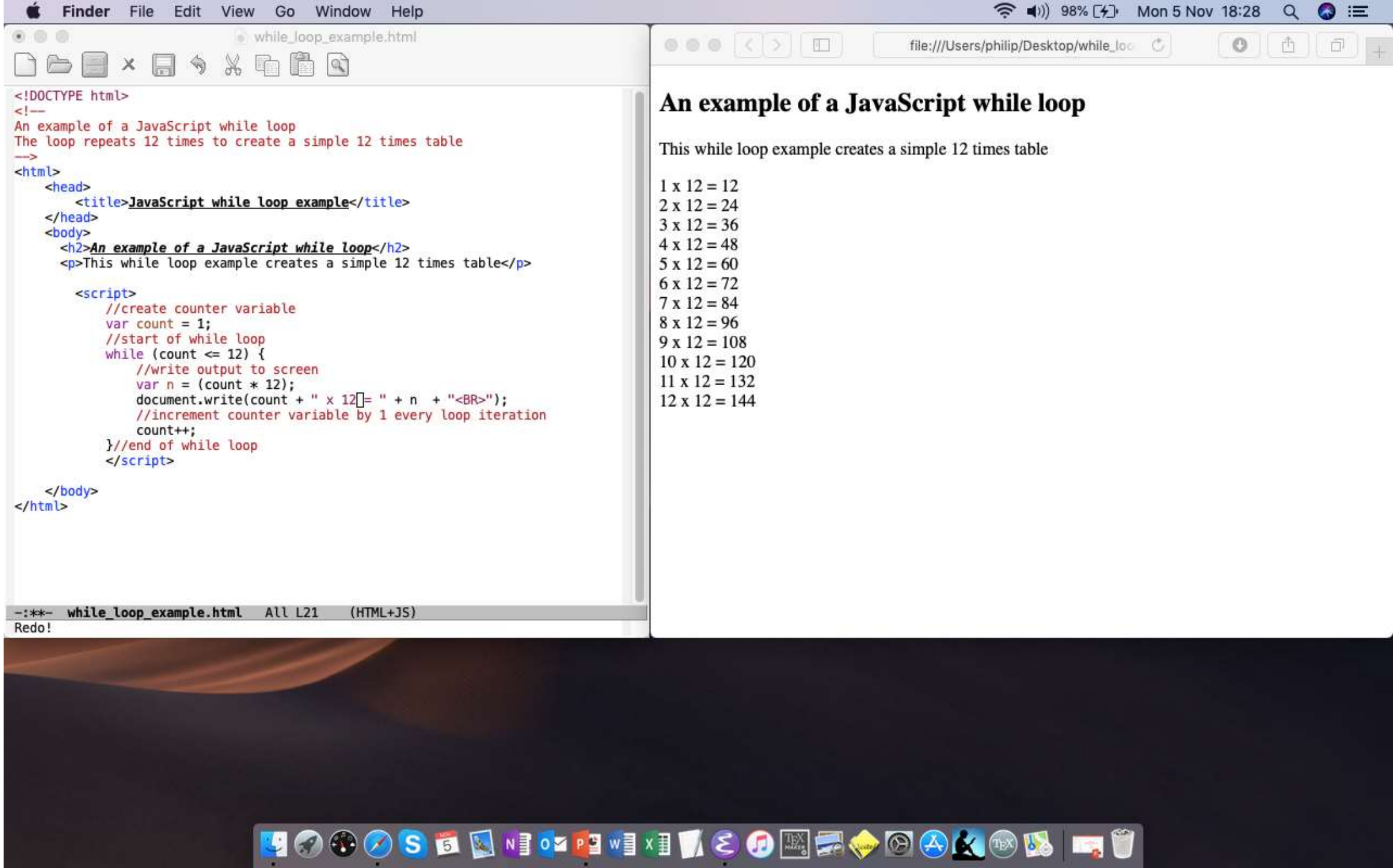

JS_While_Loop - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
236.9/263.0MB
Projects
Services
Files
JS_While_Loop
Site Root
index.html
Unit Tests
Navigator
JavaScript
document
getElementById
innerHTML : String
i : Number
text : String
CSS
Ids
#demo
HTML
html
head
title
meta
meta
body
h2
p
p id=demo
script
index.html
Source
History
1 <!DOCTYPE html>
2 <!--
3 AJavaScript while loop example
4 -->
5 <html>
6 <head>
7 <title>JavaScript while Loop</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 while loop</h2>
13 The while loop iterates 10 times - the output is the loop counter producing 0 - 9
14
15 <script>
16 var text = "";
17 var i = 0;
18 while (i < 10) {
19 text += "
The number is " + i;
20 i++;
21 }
22 document.getElementById("demo").innerHTML = text;
23 </script>
24 </body>
25 </html>
26
27
28
29
Variables
Call Stack
Breakpoints
26:1
INS

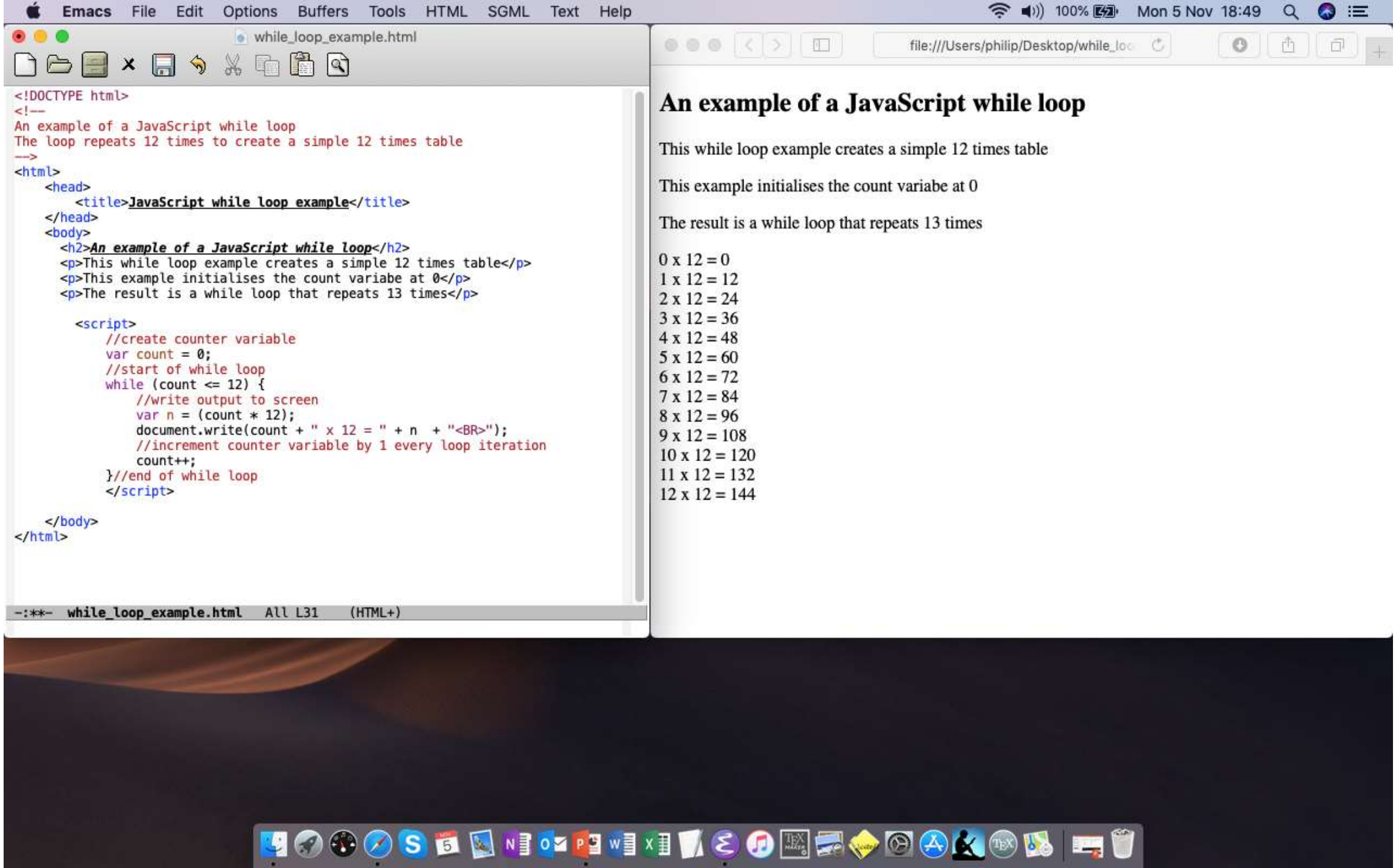

NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
237.0/258.0MB
Projects
Services
Files
JS_While_Loop
Site Root
index.html
Unit Tests
index.html
Source
History
1 <!DOCTYPE html>
2 <!--
3 AJavaScript while loop example
CSS Styles
Navigator
Browser DOM
http://localhost:8383/JS_While_Loop/...
html
head
title
meta
meta
body
h2
p
p#demo
br
br
br
br
br
br
br
br
br
br
script
JavaScript while Loop
http://localhost:8383/JS_While_Loop/index.html
100%
JavaScript while loop
The while loop iterates 10 times - the output is the loop counter producing 0 - 9
The number is 0
The number is 1
The number is 2
The number is 3
The number is 4
The number is 5
The number is 6
The number is 7
The number is 8
The number is 9
Variables
Call Stack
Breakpoints
26:1
INS


JS_do_while_Loop - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
229.1/319.0MB
Projects
Services
Files
JS_do_while_Loop
Site Root
index.html
Unit Tests
Navigator
JavaScript
document
getElementById
innerHTML : String
i : Number
text : String
CSS
Ids
#demo
HTML
html
head
title
meta
meta
body
p
p id=demo
script
index.html
Source
History
1 <!DOCTYPE html>
2 <!--
3 A JavaScript do...while loop example
4 -->
5 <html>
6 <head>
7 <title>JavaScript do...while Loop</title>
8 <meta charset="UTF-8">
9 <meta name="viewport" content="width=device-width, initial-scale=1.0">
10 </head>
11 <body>
12
13 JavaScript do ... while loop - the loop always runs at least once
14 because the test is at the end of the loop block
15
16
17 <script>
18 var text = "";
19 var i = 0;
20 do {
21 text += "
The number is " + i;
22 i++;}
23 while (i < 10);
24 document.getElementById("demo").innerHTML = text;
25 </script>
26 </body>
27 </html>
28
29
30
Variables
Call Stack
Breakpoints
28:1
INS


NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
245.1/309.0MB
Projects
Services
Files
JS_do_while_Loop
Site Root
index.html
Unit Tests
Navigator
Browser DOM
http://localhost:8383/JS_do_while_Lo...
html
head
title
meta
meta
body
p
p#demo
br
br
br
br
br
br
br
br
br
br
script
index.html
Source
History
1 <!DOCTYPE html>
2 <!--
JavaScript do...while Loo...
http://localhost:8383/JS_do_while_Loop/index.html
100%
JavaScript do ... while loop - the loop always runs at least once because the test is at the end of the loop block
The number is 0
The number is 1
The number is 2
The number is 3
The number is 4
The number is 5
The number is 6
The number is 7
The number is 8
The number is 9
Variables
Call Stack
Breakpoints
Name
Type
Value
<Enter new watch>
CSS Styles
28:1
INS

Worked `while` Loop Examples


Finder File Edit View Go Window Help
98% Mon 5 Nov 18:28
while_loop_example.html
<!DOCTYPE html>
<!--
An example of a JavaScript while loop
The loop repeats 12 times to create a simple 12 times table
-->
<html>
<head>
<title>JavaScript while loop example</title>
</head>
<body>
<h2>An example of a JavaScript while loop</h2>
This while loop example creates a simple 12 times table
<script>
//create counter variable
var count = 1;
//start of while loop
while (count <= 12) {
//write output to screen
var n = (count * 12);
document.write(count + " x 12 = " + n + "
");
//increment counter variable by 1 every loop iteration
count++;
}//end of while loop
</script>
</body>
</html>
-:**- while_loop_example.html All L21 (HTML+JS)
Redo!
file:///Users/philip/Desktop/while_loc
An example of a JavaScript while loop
This while loop example creates a simple 12 times table
1 x 12 = 12
2 x 12 = 24
3 x 12 = 36
4 x 12 = 48
5 x 12 = 60
6 x 12 = 72
7 x 12 = 84
8 x 12 = 96
9 x 12 = 108
10 x 12 = 120
11 x 12 = 132
12 x 12 = 144


Emacs File Edit Options Buffers Tools HTML SGML Text Help
100% Mon 5 Nov 18:49
while_loop_example.html
<!DOCTYPE html>
<!--
An example of a JavaScript while loop
The loop repeats 12 times to create a simple 12 times table
-->
<html>
<head>
<title>JavaScript while loop example</title>
</head>
<body>
<h2>An example of a JavaScript while loop</h2>
This while loop example creates a simple 12 times table
This example initialises the count variabe at 0
The result is a while loop that repeats 13 times
<script>
//create counter variable
var count = 0;
//start of while loop
while (count <= 12) {
//write output to screen
var n = (count * 12);
document.write(count + " x 12 = " + n + "
");
//increment counter variable by 1 every loop iteration
count++;
}//end of while loop
</script>
</body>
</html>
-:**- while_loop_example.html All L31 (HTML+)
file:///Users/philip/Desktop/while_loc
An example of a JavaScript while loop
This while loop example creates a simple 12 times table
This example initialises the count variabe at 0
The result is a while loop that repeats 13 times
0 x 12 = 0
1 x 12 = 12
2 x 12 = 24
3 x 12 = 36
4 x 12 = 48
5 x 12 = 60
6 x 12 = 72
7 x 12 = 84
8 x 12 = 96
9 x 12 = 108
10 x 12 = 120
11 x 12 = 132
12 x 12 = 144

Nesting

Nesting

- Simple output can be achieved with a *single* statement (see slides 69 and 70)
- More complex output requires *multiple* statements (see slide 74)
  - The use of multiple statements is termed *nesting*
  - Generally: the number of multiple statements should be limited – for example: two / three **`for`** loops is adequate
- Nesting can use the selection (**`if`**) and iteration (**`for`** / **`while`**) statements
- The following worked examples demonstrate *nested* for loop *JavaScript* program code with the output
  - Note: the *outer* and *inner* for loops in a *nested* structure

Nested `for` loop

- The following code snippet shows a *nested* `for` loop – the syntax prototype is as follows:

```
for ( initialise ; test ; update){
    statement
   for ( initialise ; test ; update){
      statement
   }
}
```

Worked Nested `for` Loop Example

```
<!DOCTYPE html>
<!--
An example of a nested for loops in JavaScript
The script uses nested for loops to perform a calculation (the output is a 12 times table)
The result is sent to the output (a web browser
The output is not formatted
-->
<html>
   <head>
      <title>Nested for loops examplet</title>
   </head>
   <h2>An example of nested for loops in JavaScript</h2>
   <p>The outer for loop repeats 12 times</p>
   <p>The inner for loop repeats 12 times</p>
   <body>
     <p>start of JavaScript script</p>
     <script>
       //loop variables (outer loop counter: var outer)
       //loop variables (inner loop counter: var inner)
       //conditional variable: var total
       var outer, inner, total;

       //outer for loop
       for (outer = 1; outer <= 12; outer++) {
         //inner for loop
         for (inner = 1; inner <= 12; inner++) {
            total = outer * inner;
               //if conditional operator
               if (total < 12) { //start of if statement
                  total = " " + total;
                }//end of if statement
                else {//start of else statement
                    total  = " " + total;
                }//end of else statement
                //write output
                document.write(total);
            } //end of inner loop
            //create new line using "<br">
            document.write("<BR>");
        }//end of outer loop
        </script>
    <p>end of JavaScript script</p>

   </body>
</html>
```

-:--- **nested_loop_example.html** All L20 (HTML+JS)

Wrote /Users/philip/Documents/Academic/LZU_Teaching_2018/151/Session_Slides/Week_5/html/nested_loop_example.html

An example of nested for loops in JavaScript

The outer for loop repeats 12 times

The inner for loop repeats 12 times

start of JavaScript script

1 2 3 4 5 6 7 8 9 10 11 12
2 4 6 8 10 12 14 16 18 20 22 24
3 6 9 12 15 18 21 24 27 30 33 36
4 8 12 16 20 24 28 32 36 40 44 48
5 10 15 20 25 30 35 40 45 50 55 60
6 12 18 24 30 36 42 48 54 60 66 72
7 14 21 28 35 42 49 56 63 70 77 84
8 16 24 32 40 48 56 64 72 80 88 96
9 18 27 36 45 54 63 72 81 90 99 108
10 20 30 40 50 60 70 80 90 100 110 120
11 22 33 44 55 66 77 88 99 110 121 132
12 24 36 48 60 72 84 96 108 120 132 144

end of JavaScript script

Review

- In this session we have provided:
  - An overview of:
    - The basic principles of Objects and Object-Oriented Programming
    - Inheritance in Object-Oriented Programming
  - Selection:
    - Using conditional operators to select from a range of alternatives with worked examples
  - Iteration:
    - Using loops to repeat a program run until a condition is satisfied or a termination criteria is reached with worked examples
  - Using multiple statements in a nested structure