Javascript

INF1802

Profa. Melissa Lemos

TECGRAF
PUC-RIO

Outline

- Module 1 – The Language Javascript

Module 1 – The Language

Introduction

Motivation

JavaScript is one of the 3 languages all web developers must learn:

1. HTML to define the content of web pages
2. CSS to specify the layout of web pages
3. JavaScript to program the behavior of web pages

Javascritp – The language/ History

- JavaScript was invented by Brendan Eich in 1995, and became an ECMA standard in 1997

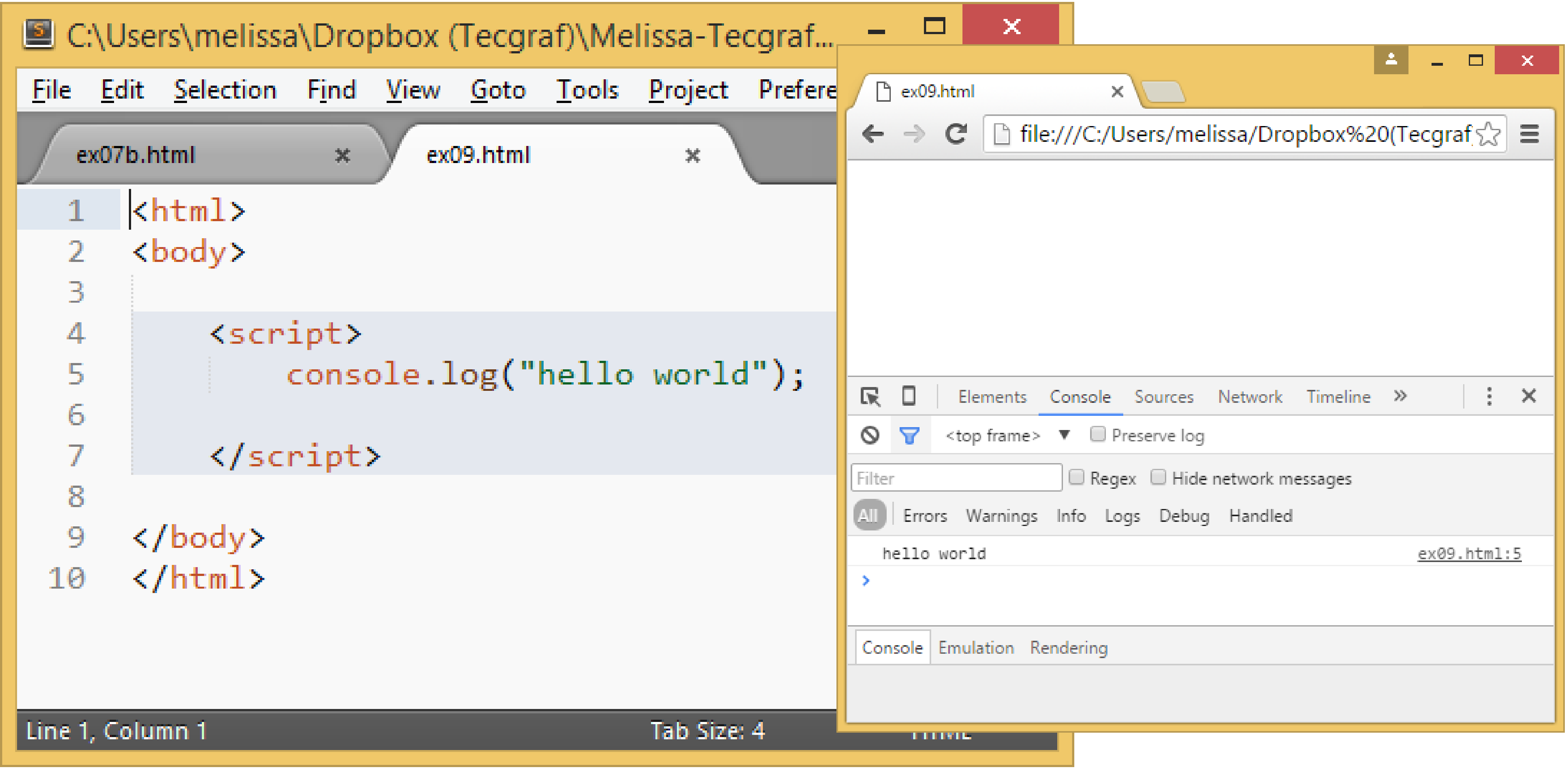
Javascript – Main Tasks

- JavaScript can change HTML content
- JavaScript can change HTML attributes
- JavaScript can change HTML styles (CSS)
- JavaScript can validate data

Javascript and Java

- JavaScript and Java are completely different languages, both in concept and design.

Hello World



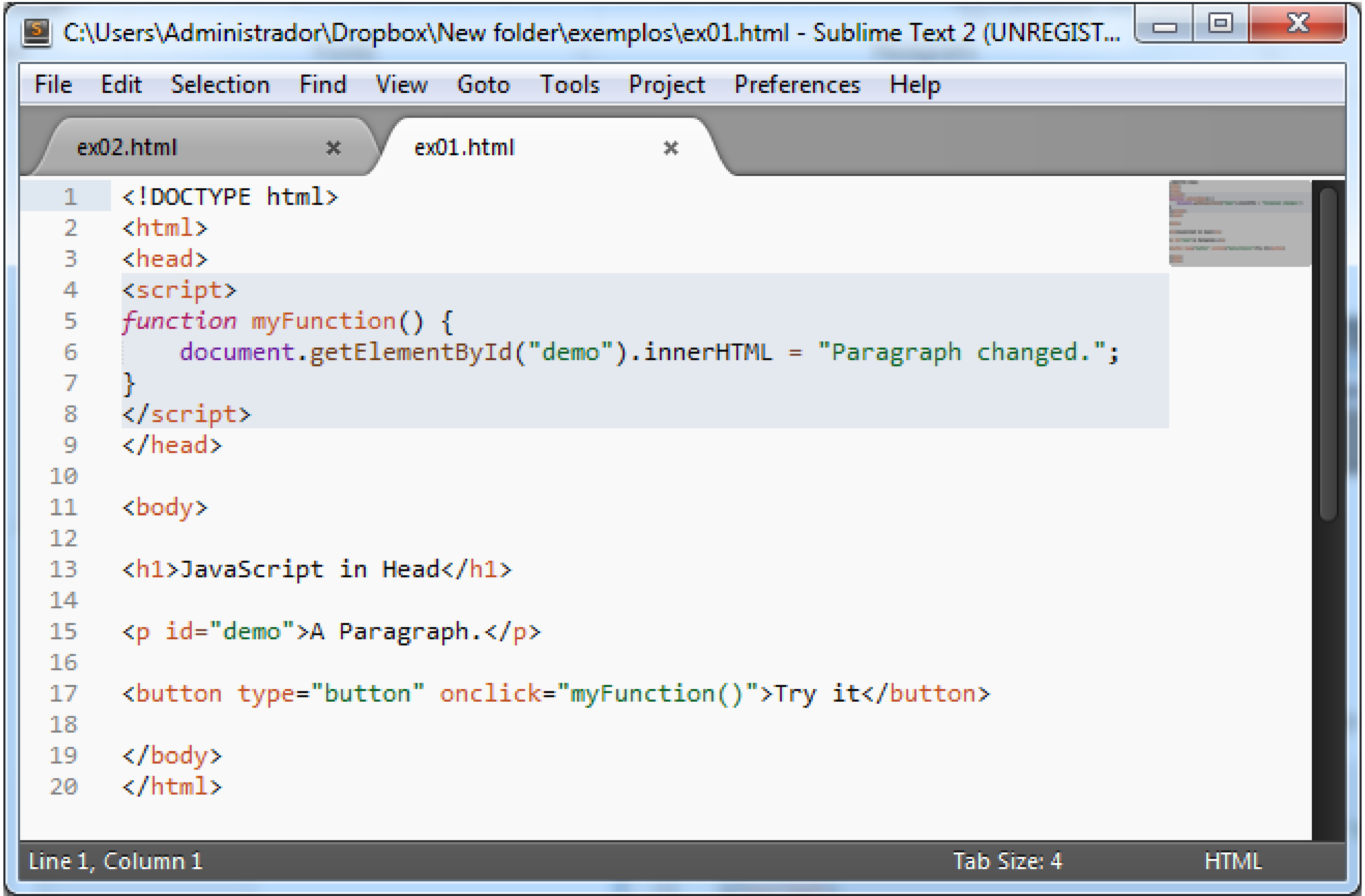
```
<html>
<body>

    <script>
        console.log("hello world");

    </script>

</body>
</html>
```

```
hello world                                                    ex09.html:5
>
```

Where

- In HTML Page
  - JavaScript can be placed in the <body> and the <head> sections of an HTML page.
  - The code must be inserted between <script> and </script> tags.
- In external files

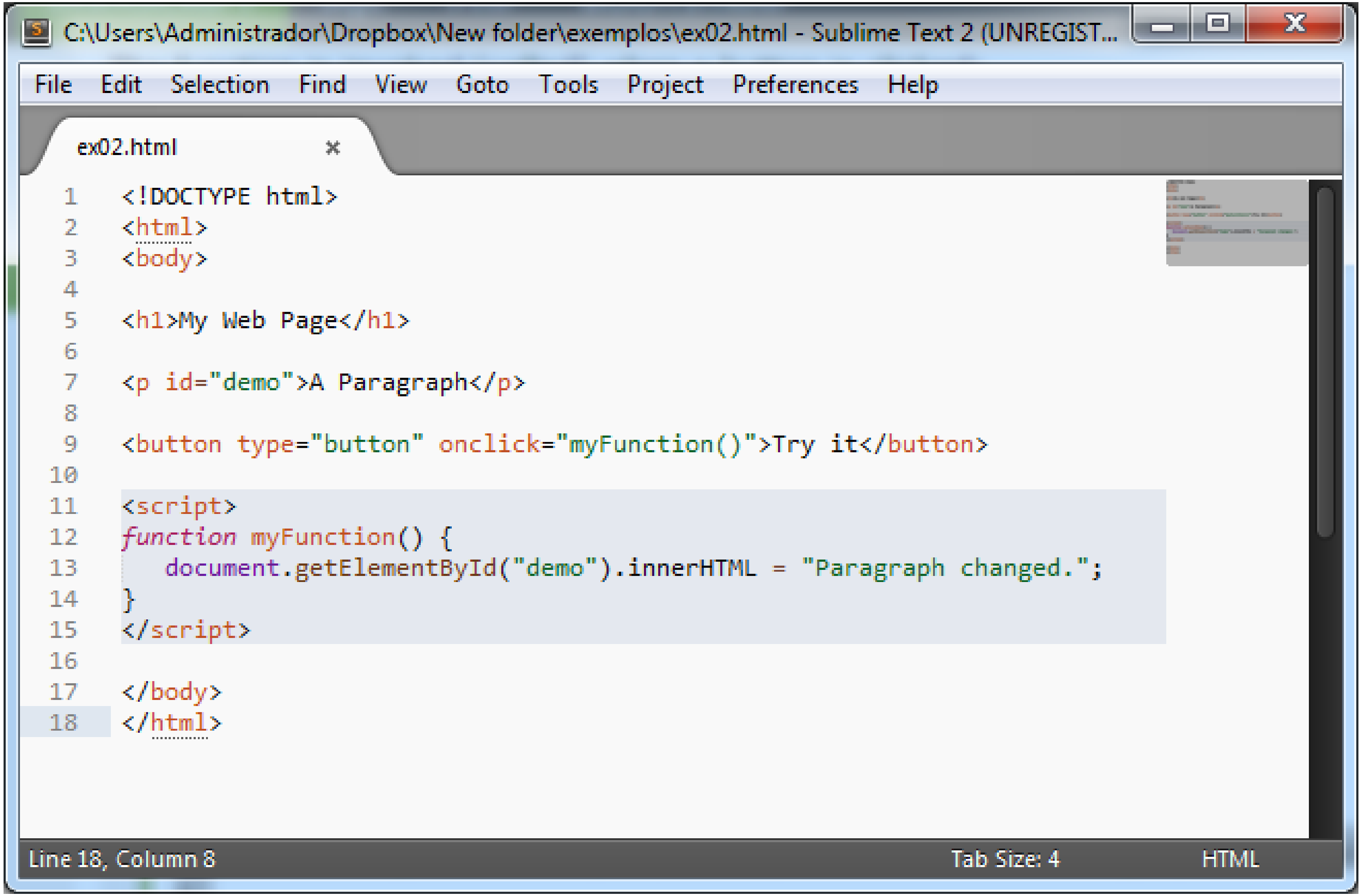
JavaScript in <head>



C:\Users\Administrador\Dropbox\New folder\exemplos\ex01.html - Sublime Text 2 (UNREGIST...
File Edit Selection Find View Goto Tools Project Preferences Help
ex02.html
ex01.html
Line 1, Column 1
Tab Size: 4
HTML

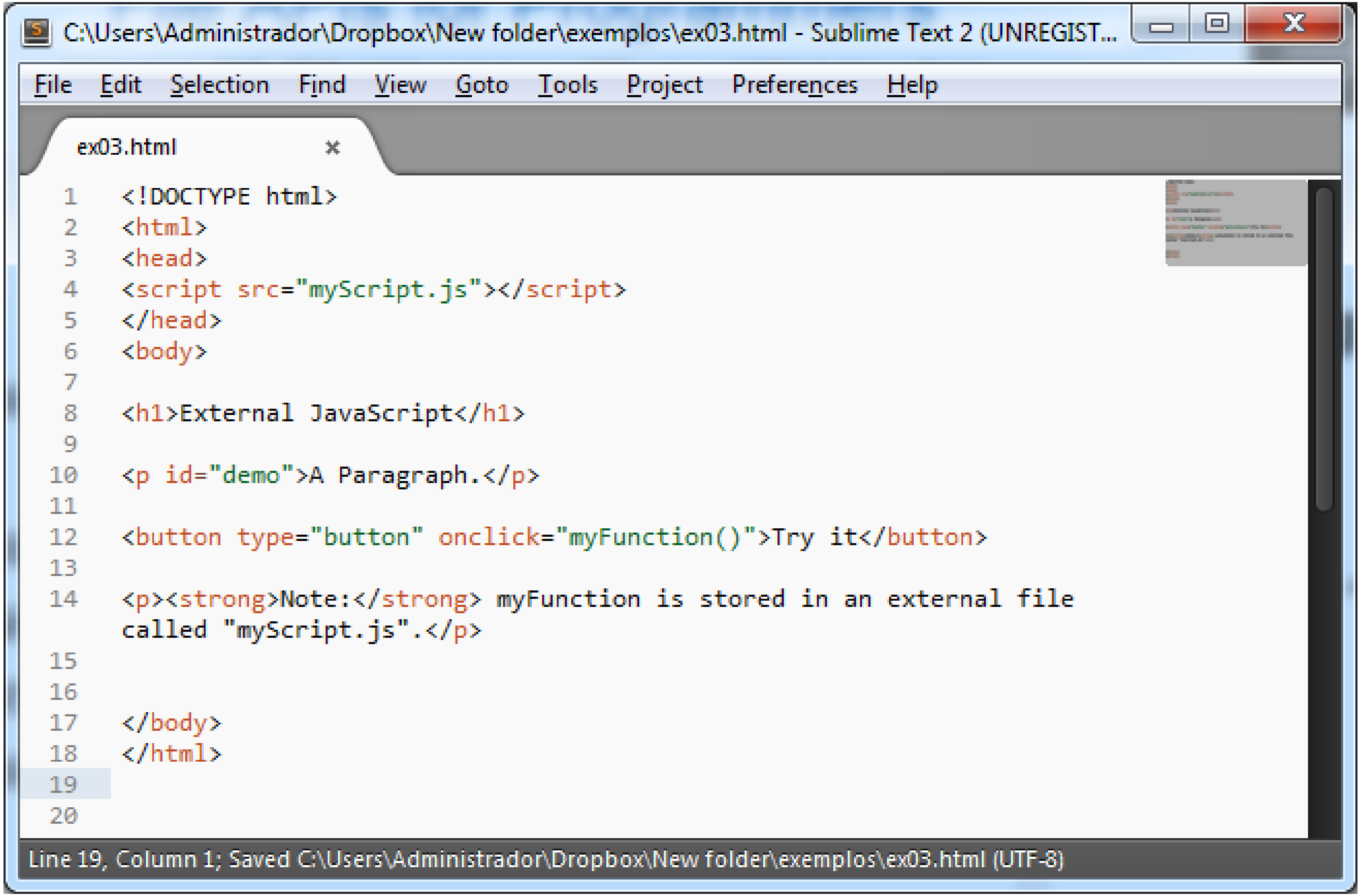
```html
<!DOCTYPE html>
<html>
<head>
<script>
function myFunction() {
    document.getElementById("demo").innerHTML = "Paragraph changed.";
}
</script>
</head>

<body>

<h1>JavaScript in Head</h1>

<p id="demo">A Paragraph.</p>

<button type="button" onclick="myFunction()">Try it</button>

</body>
</html>
```

JavaScript in <body>



```html
<!DOCTYPE html>
<html>
<body>

<h1>My Web Page</h1>

<p id="demo">A Paragraph</p>

<button type="button" onclick="myFunction()">Try it</button>

<script>
function myFunction() {
    document.getElementById("demo").innerHTML = "Paragraph changed.";
}
</script>

</body>
</html>
```

Javascript in external files


C:\Users\Administrador\Dropbox\New folder\exemplos\ex03.html - Sublime Text 2 (UNREGIST...
File Edit Selection Find View Goto Tools Project Preferences Help
ex03.html

```html
<!DOCTYPE html>
<html>
<head>
<script src="myScript.js"></script>
</head>
<body>

<h1>External JavaScript</h1>

<p id="demo">A Paragraph.</p>

<button type="button" onclick="myFunction()">Try it</button>

<p><strong>Note:</strong> myFunction is stored in an external file called "myScript.js".</p>


</body>
</html>


```

Line 19, Column 1; Saved C:\Users\Administrador\Dropbox\New folder\exemplos\ex03.html (UTF-8)

Values, Types and Operators

Values

“Imagine a sea of bits. An ocean of them. A typical modern computer has more than 30 billion bits in its volatile data storage. Nonvolatile storage (the hard disk or equivalent) tends to have yet a few orders of magnitude more.

To be able to work with such quantities of bits without getting lost, you can separate them into chunks that represent pieces of information.

In a JavaScript environment, those chunks are called values. Though all values are made of bits, they play different roles.

Every value has a type that determines its role. There are 5 basic types of values in JavaScript: numbers, strings, booleans, objects and undefined values.”

* http://eloquentjavascript.net/01_values.html

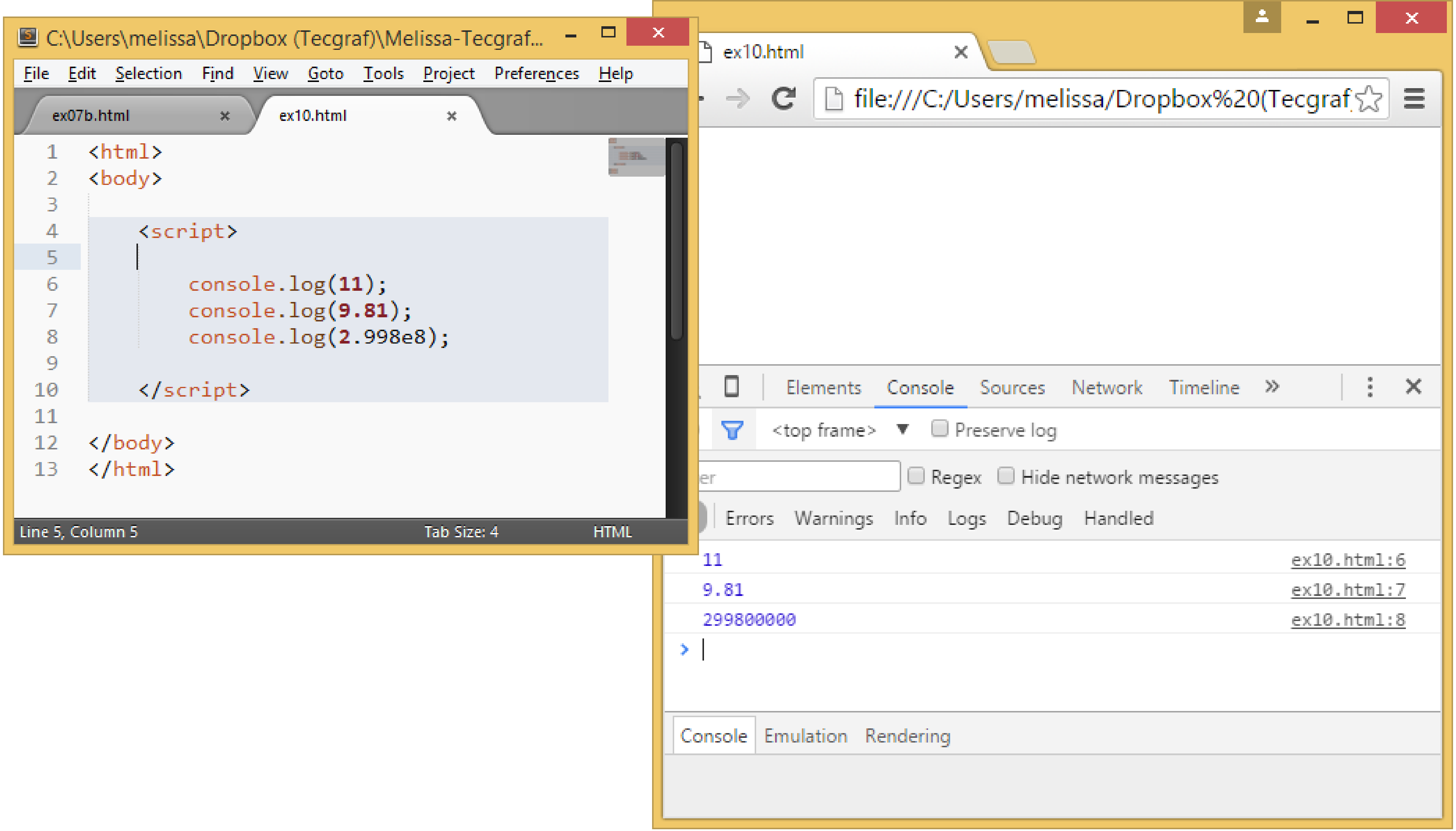
The simple value types and the operators that can act on such values

- Numbers
- Strings
- Booleans
- Undefined

Numbers

- JavaScript uses a fixed number of bits, namely 64 of them, to store a single number value. Given 64 binary digits, you can represent 264 different numbers, which is about 18 quintillion (an 18 with 18 zeros after it). This is a lot.

  - Number: 11
  - Fractional numbers are written by using a dot: 9.81
  - You can also use scientific notation by adding an “e” (for “exponent”), followed by the exponent of the number: 2.998e8. That is 2.998 × 108 = 299,800,000.

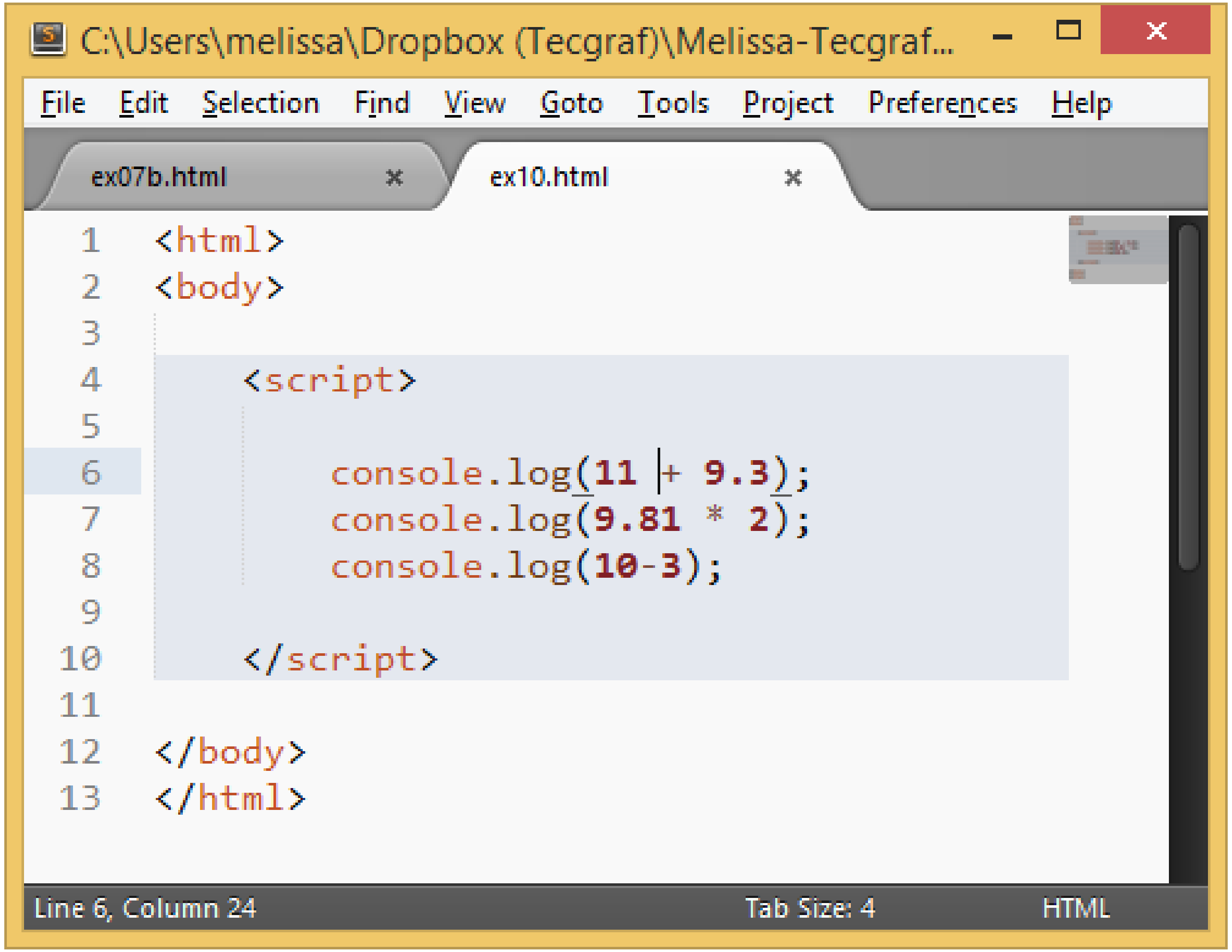

C:\Users\melissa\Dropbox (Tecgraf)\Melissa-Tecgraf...
File Edit Selection Find View Goto Tools Project Preferences Help
ex07b.html
ex10.html
Line 5, Column 5
Tab Size: 4
HTML
ex10.html
file:///C:/Users/melissa/Dropbox%20(Tecgraf
Elements Console Sources Network Timeline
<top frame> ▼
□ Preserve log
□ Regex □ Hide network messages
Errors Warnings Info Logs Debug Handled
Console Emulation Rendering

```
<html>
<body>

    <script>


        console.log(11);
        console.log(9.81);
        console.log(2.998e8);

    </script>

</body>
</html>
```

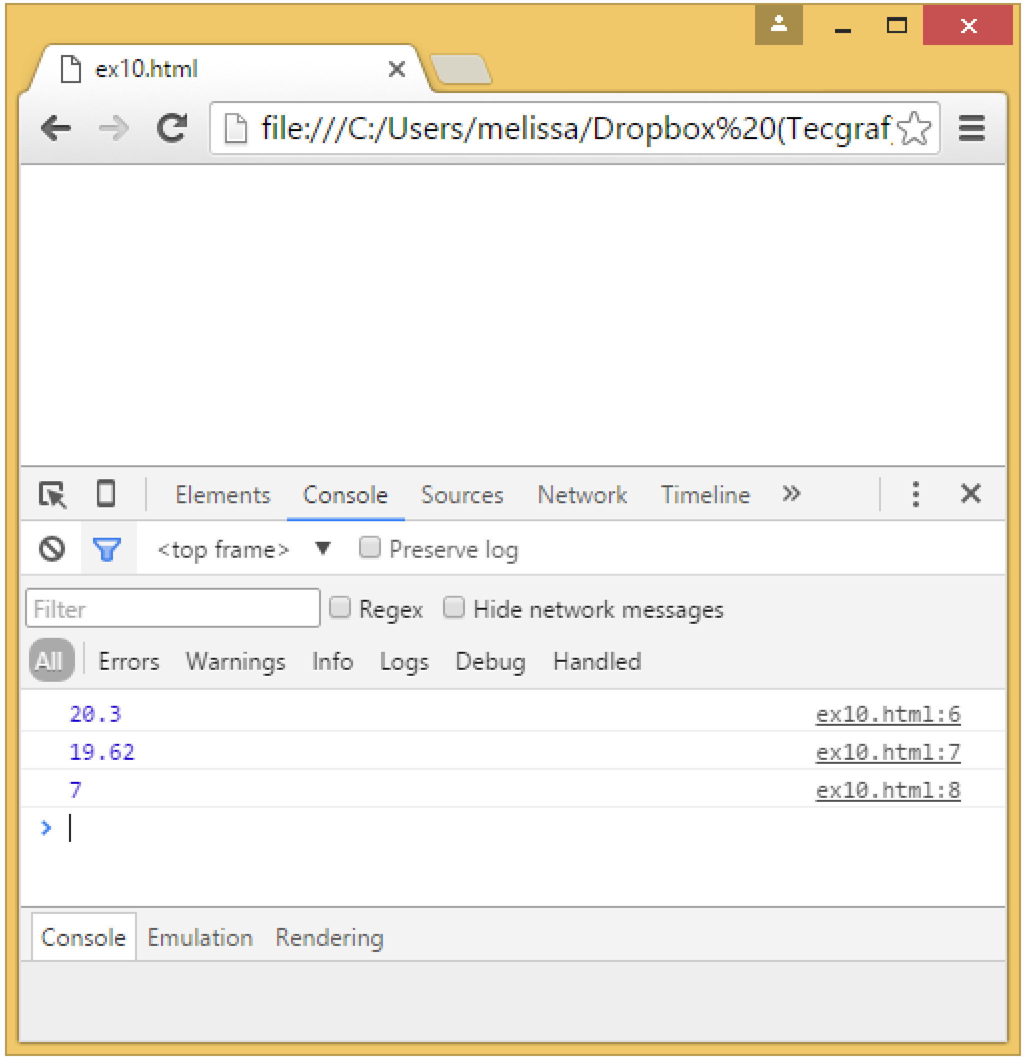
```
11                                                    ex10.html:6
9.81                                                  ex10.html:7
299800000                                             ex10.html:8
>
```

Arithmetic Operators

+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus
++	Increment
--	Decrement


C:\Users\melissa\Dropbox (Tecgraf)\Melissa-Tecgraf...
File Edit Selection Find View Goto Tools Project Preferences Help
ex07b.html
ex10.html
Line 6, Column 24
Tab Size: 4
HTML

```
<html>
<body>

    <script>

        console.log(11 + 9.3);
        console.log(9.81 * 2);
        console.log(10-3);

    </script>

</body>
</html>
```


ex10.html
file:///C:/Users/melissa/Dropbox%20(Tecgraf
Elements Console Sources Network Timeline »
<top frame> ▼ □ Preserve log
Filter □ Regex □ Hide network messages
All Errors Warnings Info Logs Debug Handled
Console Emulation Rendering

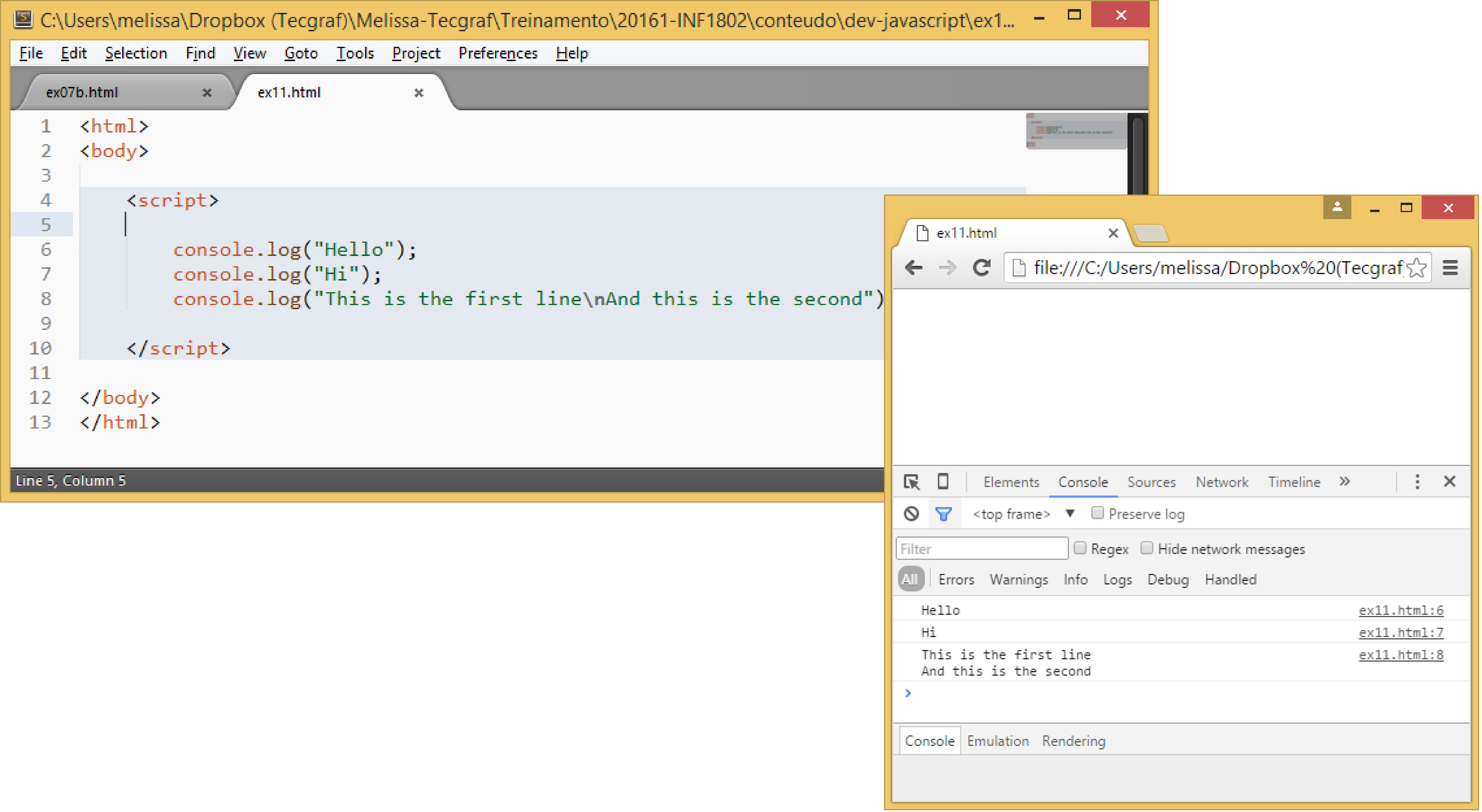
```
20.3                                   ex10.html:6
19.62                                  ex10.html:7
7                                      ex10.html:8
> |
```

Strings

- Strings are used to represent text. They are written by enclosing their content in quotes.

  - "Patch my boat with chewing gum"
  - 'Monkeys wave goodbye‘

- A backslash (\) indicates that the character after it has a special meaning.
  - \n = new line
  - "This is the first line\nAnd this is the second"


C:\Users\melissa\Dropbox (Tecgraf)\Melissa-Tecgraf\Treinamento\20161-INF1802\conteudo\dev-javascript\ex1...
File Edit Selection Find View Goto Tools Project Preferences Help
ex07b.html
ex11.html
Line 5, Column 5
ex11.html
file:///C:/Users/melissa/Dropbox%20(Tecgraf
Elements Console Sources Network Timeline
<top frame>
Preserve log
Filter
Regex
Hide network messages
All Errors Warnings Info Logs Debug Handled
Console Emulation Rendering

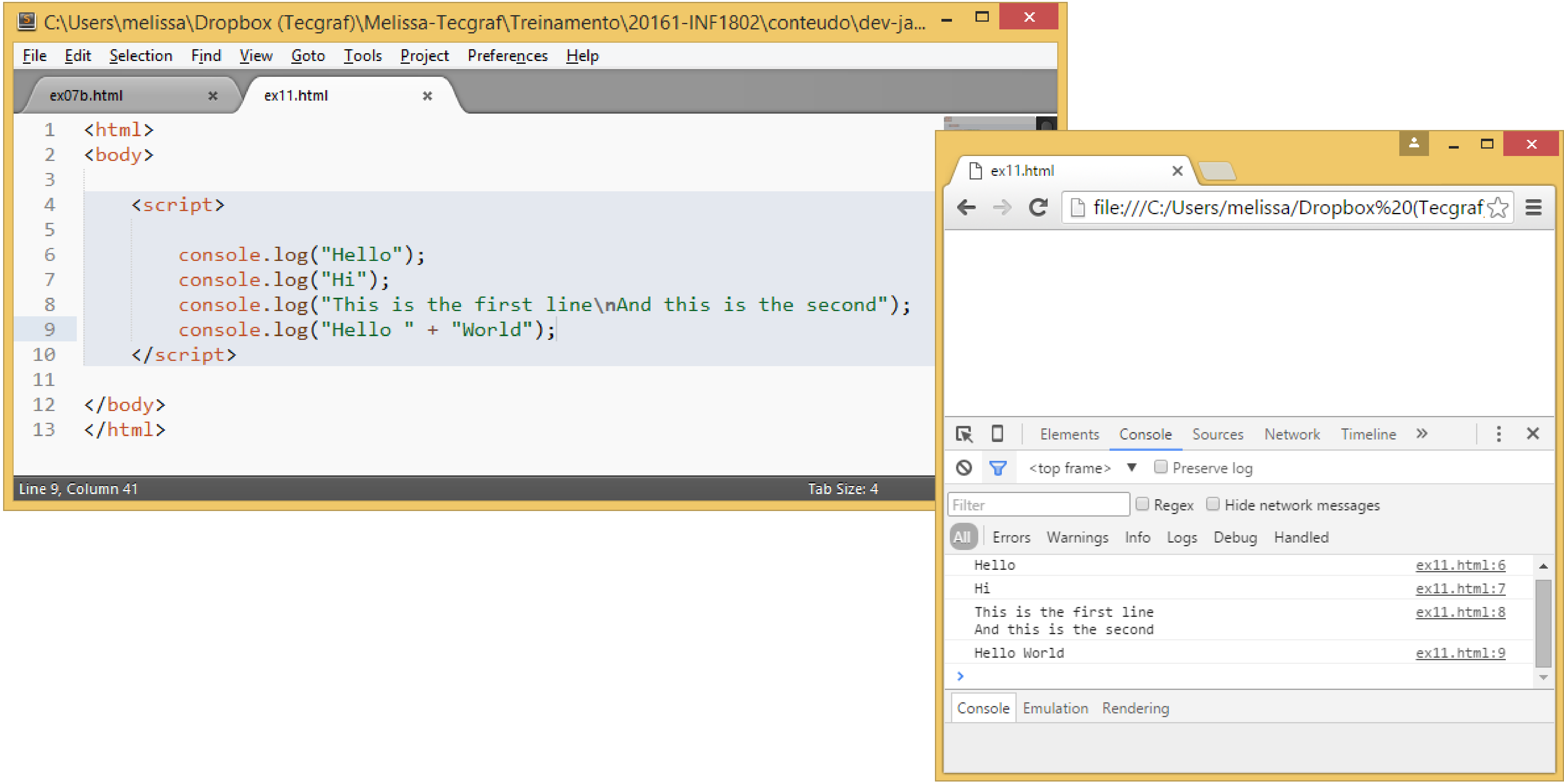
```
<html>
<body>

    <script>
    
        console.log("Hello");
        console.log("Hi");
        console.log("This is the first line\nAnd this is the second")

    </script>

</body>
</html>
```

```
Hello                                                    ex11.html:6
Hi                                                       ex11.html:7
This is the first line                                   ex11.html:8
And this is the second
>
```

String Operator - Concatenation

- The + operator can also be used to add (concatenate) strings

-
  ```
  txt1 = "John";
  txt2 = "Doe";
  txt3 = txt1 + " " + txt2;
  ```


C:\Users\melissa\Dropbox (Tecgraf)\Melissa-Tecgraf\Treinamento\20161-INF1802\conteudo\dev-ja...
File Edit Selection Find View Goto Tools Project Preferences Help
ex07b.html
ex11.html

```
<html>
<body>

    <script>

        console.log("Hello");
        console.log("Hi");
        console.log("This is the first line\nAnd this is the second");
        console.log("Hello " + "World");
    </script>

</body>
</html>
```

Line 9, Column 41
Tab Size: 4
ex11.html
file:///C:/Users/melissa/Dropbox%20(Tecgraf
Elements Console Sources Network Timeline »
<top frame> ▼ □ Preserve log
Filter □ Regex □ Hide network messages
All Errors Warnings Info Logs Debug Handled

```
Hello                                  ex11.html:6
Hi                                     ex11.html:7
This is the first line                 ex11.html:8
And this is the second
Hello World                            ex11.html:9
>
```

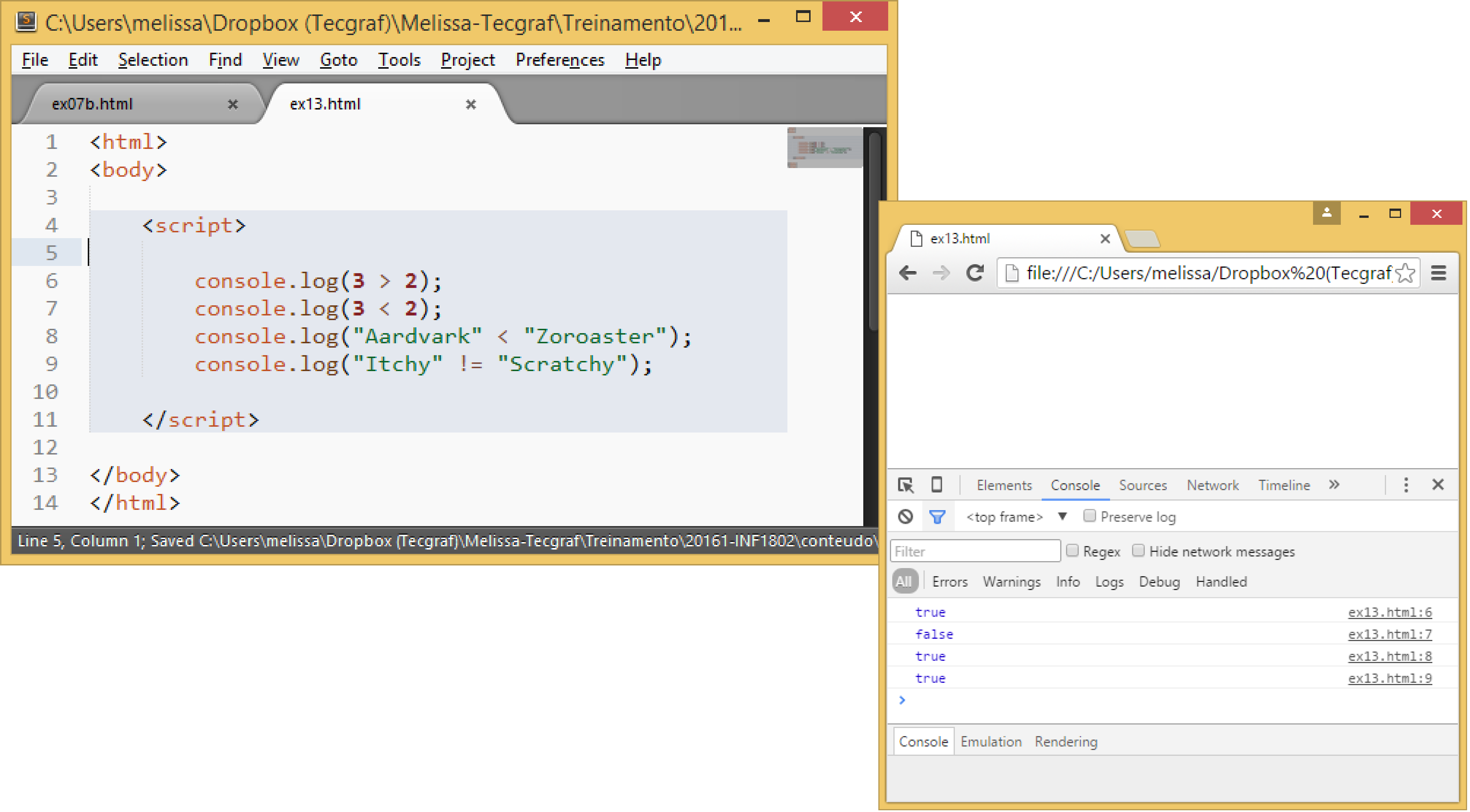
Console Emulation Rendering

Booleans

- If you need a value that simply distinguishes between two possibilities, like “yes” and “no” or “on” and “off”. For this, JavaScript has a *Boolean* type, which has just two values: true and false.

Comparison Operators

==	equal to
!=	not equal
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to



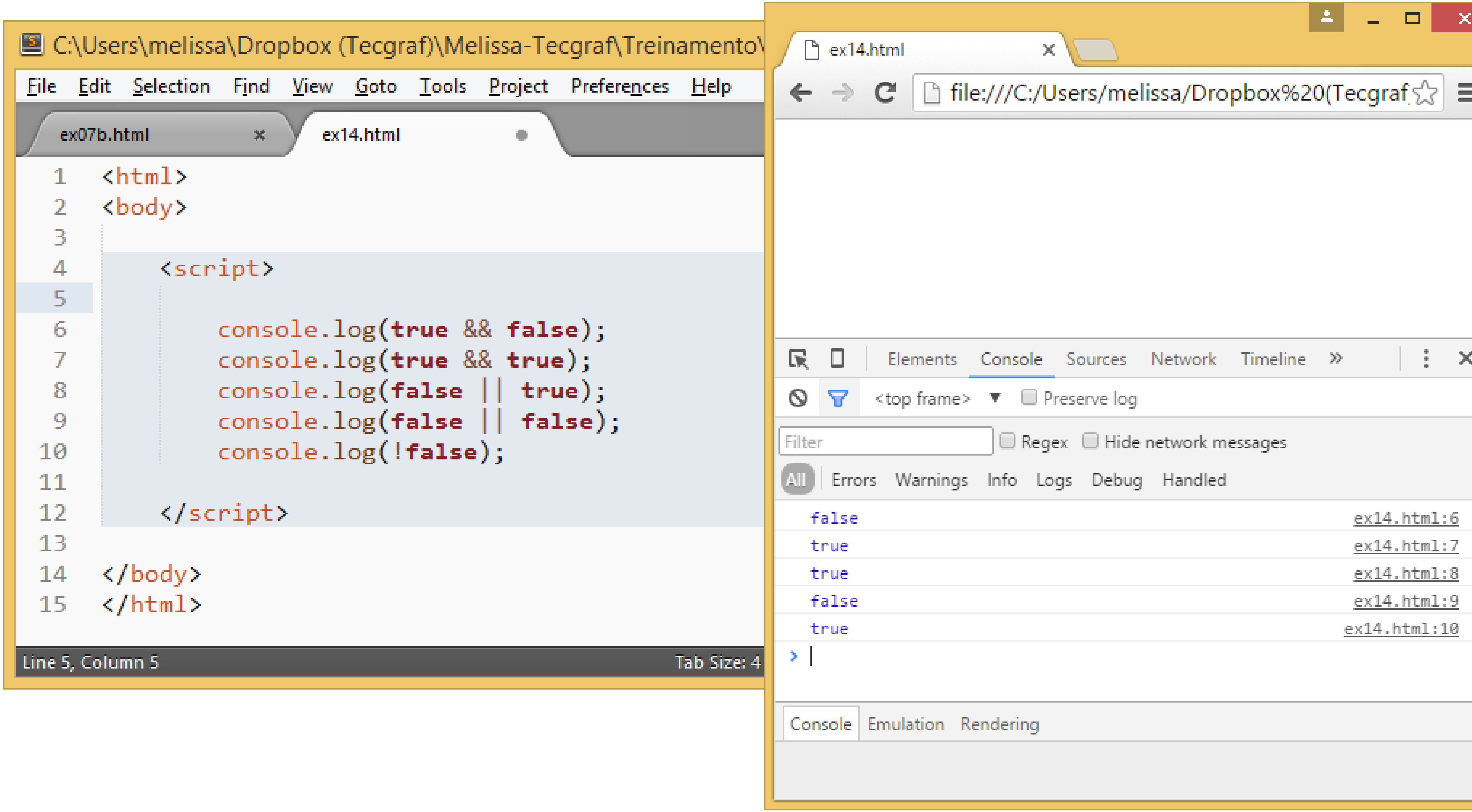
```
<html>
<body>

    <script>

        console.log(3 > 2);
        console.log(3 < 2);
        console.log("Aardvark" < "Zoroaster");
        console.log("Itchy" != "Scratchy");

    </script>

</body>
</html>
```

Line 5, Column 1; Saved C:\Users\melissa\Dropbox (Tecgraf)\Melissa-Tecgraf\Treinamento\20161-INF1802\conteudo\

```
true                                                ex13.html:6
false                                               ex13.html:7
true                                                ex13.html:8
true                                                ex13.html:9
>
```

Logical Operators

&&	AND
||	OR
!	NOT


C:\Users\melissa\Dropbox (Tecgraf)\Melissa-Tecgraf\Treinamento\
File Edit Selection Find View Goto Tools Project Preferences Help
ex07b.html
ex14.html
Line 5, Column 5
Tab Size: 4
ex14.html
file:///C:/Users/melissa/Dropbox%20(Tecgraf
Elements Console Sources Network Timeline
<top frame>
Preserve log
Filter
Regex
Hide network messages
All Errors Warnings Info Logs Debug Handled
Console Emulation Rendering

```
<html>
<body>

    <script>

        console.log(true && false);
        console.log(true && true);
        console.log(false || true);
        console.log(false || false);
        console.log(!false);

    </script>

</body>
</html>
```

```
false                                  ex14.html:6
true                                   ex14.html:7
true                                   ex14.html:8
false                                  ex14.html:9
true                                  ex14.html:10
>
```

Undefined Values

- There are two special values, written null and undefined, that are used to denote the absence of a meaningful value. They are themselves values, but they carry no information.

- The difference in meaning between undefined and null is an accident of JavaScript’s design, and it doesn’t matter most of the time

Undefined, Empty and Null

- In JavaScript, a variable without a value, has the value **undefined**. The typeof is also **undefined**.

```
var person;        // Value is undefined, type is undefined
```

- An **empty** value has nothing to do with undefined. An empty string variable has both a value and a type.

```
var car = "";      // The value is "", the typeof is string
```

- In JavaScript **null** is "nothing". It is supposed to be something that doesn't exist.

```
var person = null;     // Value is null, but type is still an object
```

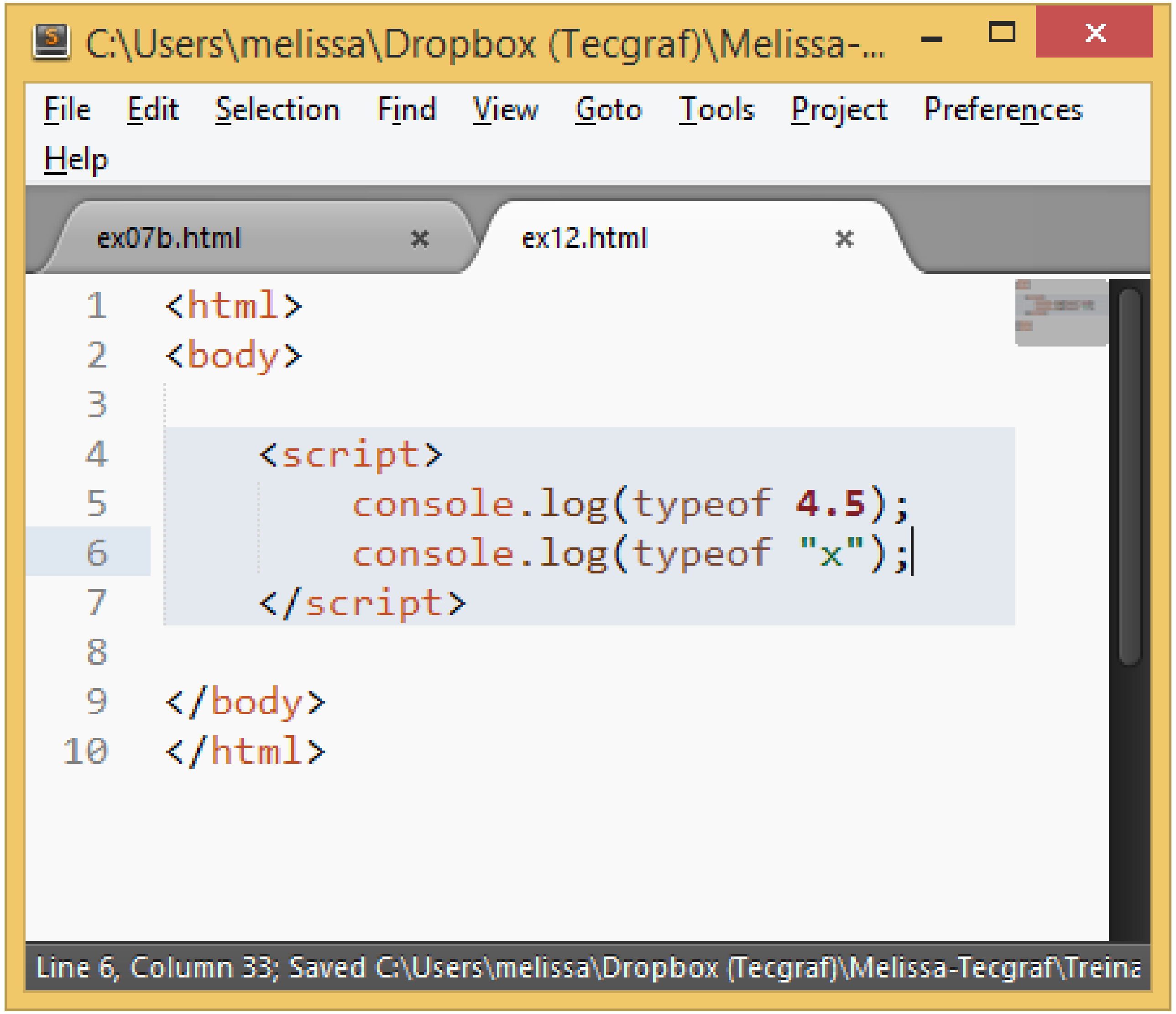
Operator typeof

- Not all operators are symbols. Some are written as words. One example is the typeof operator, which produces a string value naming the type of the value you give it.

Operator typeof

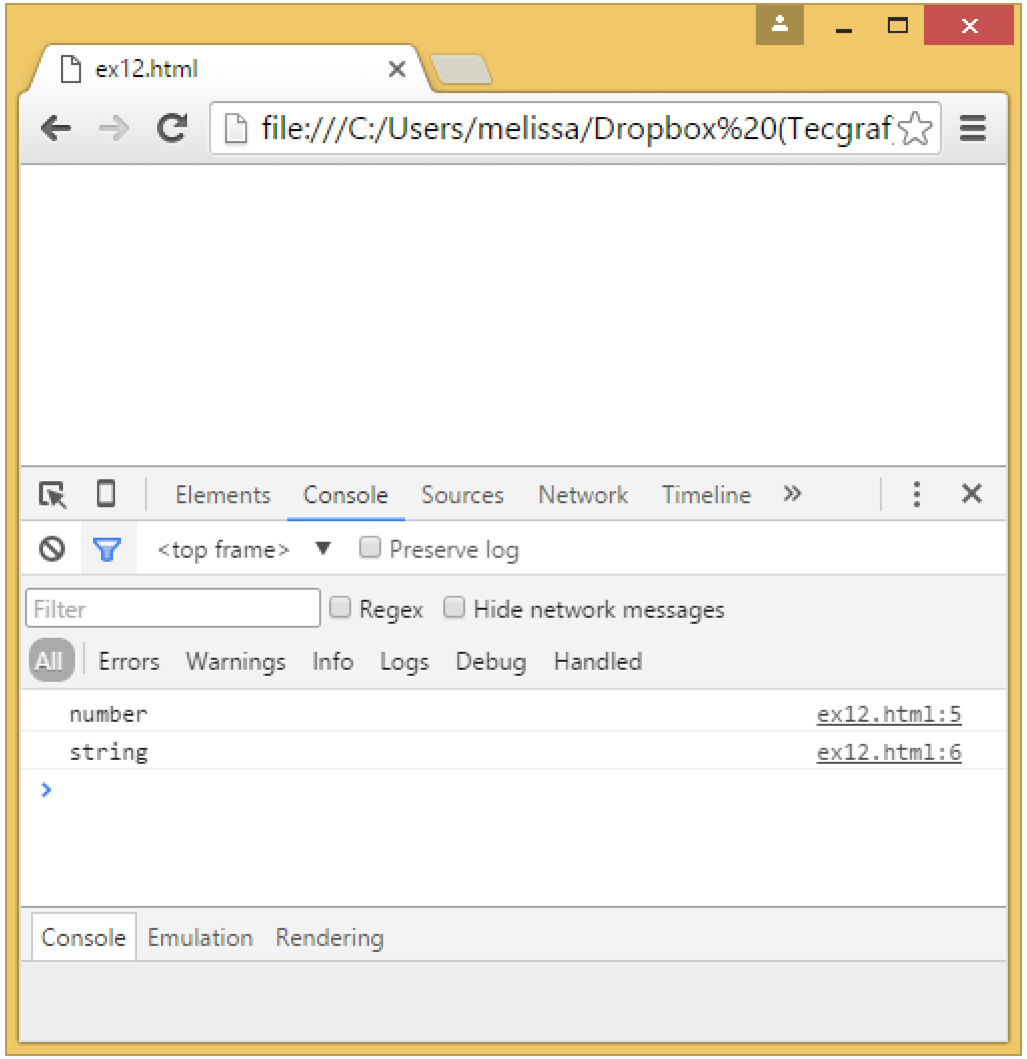
typeof	Returns the type of a variable
instanceof	Returns true if an object is an instance of an object type

```
typeof "John"                 // Returns string
typeof 3.14                   // Returns number
typeof false                  // Returns boolean
typeof [1,2,3,4]              // Returns object
typeof {name:'John', age:34} // Returns object
```


C:\Users\melissa\Dropbox (Tecgraf)\Melissa-...
File Edit Selection Find View Goto Tools Project Preferences Help
ex07b.html
ex12.html

```
<html>
<body>

    <script>
        console.log(typeof 4.5);
        console.log(typeof "x");
    </script>

</body>
</html>
```

Line 6, Column 33; Saved C:\Users\melissa\Dropbox (Tecgraf)\Melissa-Tecgraf\Treina

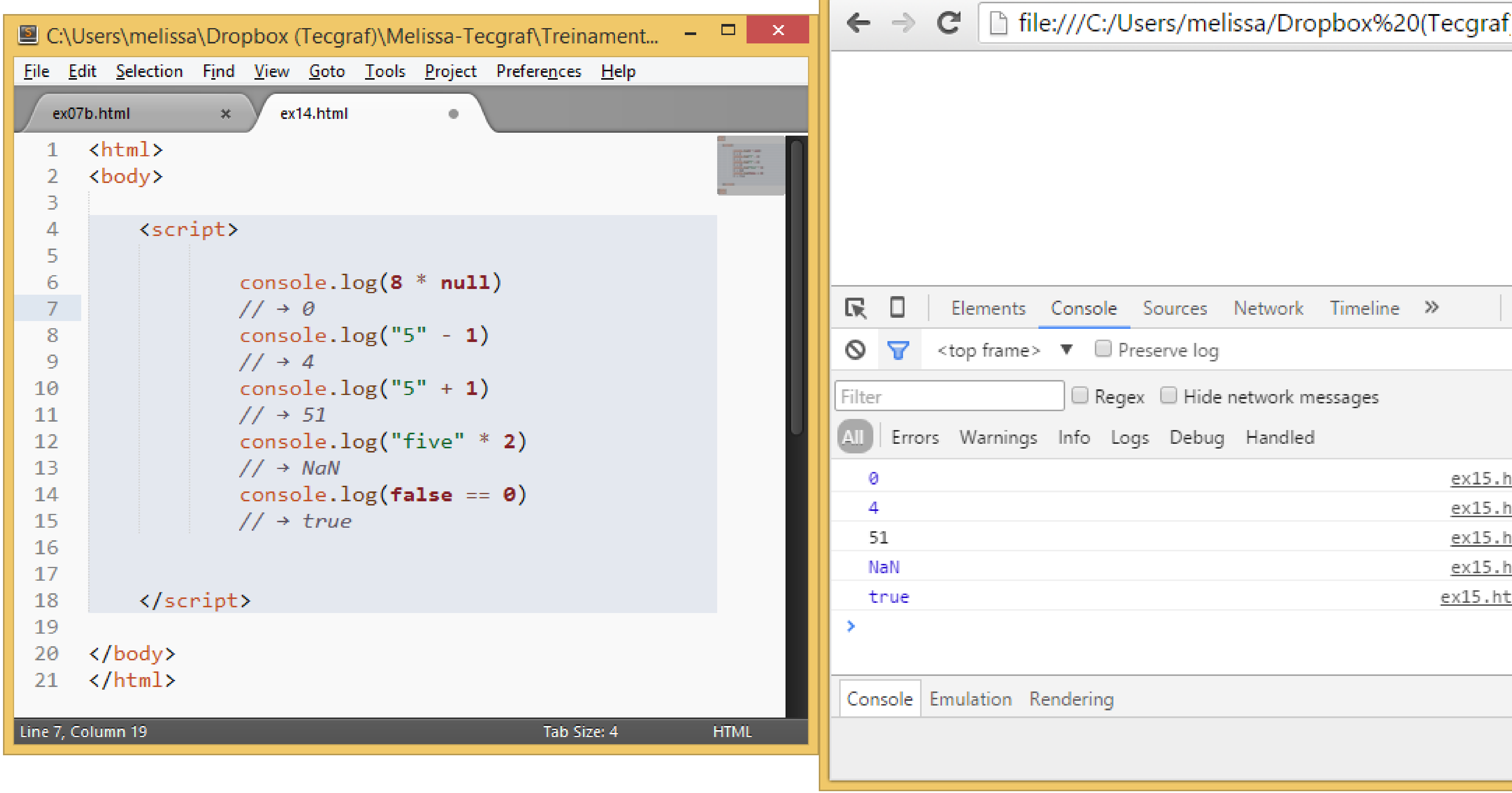

ex12.html
file:///C:/Users/melissa/Dropbox%20(Tecgraf
Elements Console Sources Network Timeline
<top frame>
□ Preserve log
Filter
□ Regex
□ Hide network messages
All Errors Warnings Info Logs Debug Handled
Console Emulation Rendering

```
number                                                    ex12.html:5
string                                                    ex12.html:6
>
```

Be careful

- JavaScript goes out of its way to accept almost any program you give it, even programs that do odd things.

- When an operator is applied to the “wrong” type of value, JavaScript will quietly convert that value to the type it wants, using a set of rules that often aren’t what you want or expect.


C:\Users\melissa\Dropbox (Tecgraf)\Melissa-Tecgraf\Treinament...
File Edit Selection Find View Goto Tools Project Preferences Help
ex07b.html
ex14.html
Line 7, Column 19
Tab Size: 4
HTML
file:///C:/Users/melissa/Dropbox%20(Tecgraf
Elements Console Sources Network Timeline
<top frame>
Preserve log
Filter
Regex
Hide network messages
All Errors Warnings Info Logs Debug Handled
Console Emulation Rendering

```
<html>
<body>

    <script>

            console.log(8 * null)
            // → 0
            console.log("5" - 1)
            // → 4
            console.log("5" + 1)
            // → 51
            console.log("five" * 2)
            // → NaN
            console.log(false == 0)
            // → true


    </script>

</body>
</html>
```

```
0                                                         ex15.h
4                                                         ex15.h
51                                                        ex15.h
NaN                                                       ex15.h
true                                                     ex15.ht
>
```

Program Structure

Expressions and statements

Variables

Keywords and reserved words

Functions

Conditions

Loops

Variables

- To catch and hold values, JavaScript provides variables.
    - var caught = 5 * 5;

- The special word (keyword) *var* indicates that this sentence is going to define a variable. It is followed by the name of the variable and, if we want to immediately give it a value, by an = operator and an expression.

Variables - Names

- Names can contain letters, digits, underscores.
- Names must begin with a letter
- Names can also begin with $ and _ (but we will not use it in this course)
- Names are case sensitive (y and Y are different variables)
- Reserved words (like JavaScript keywords) cannot be used as names

Assignment Operator

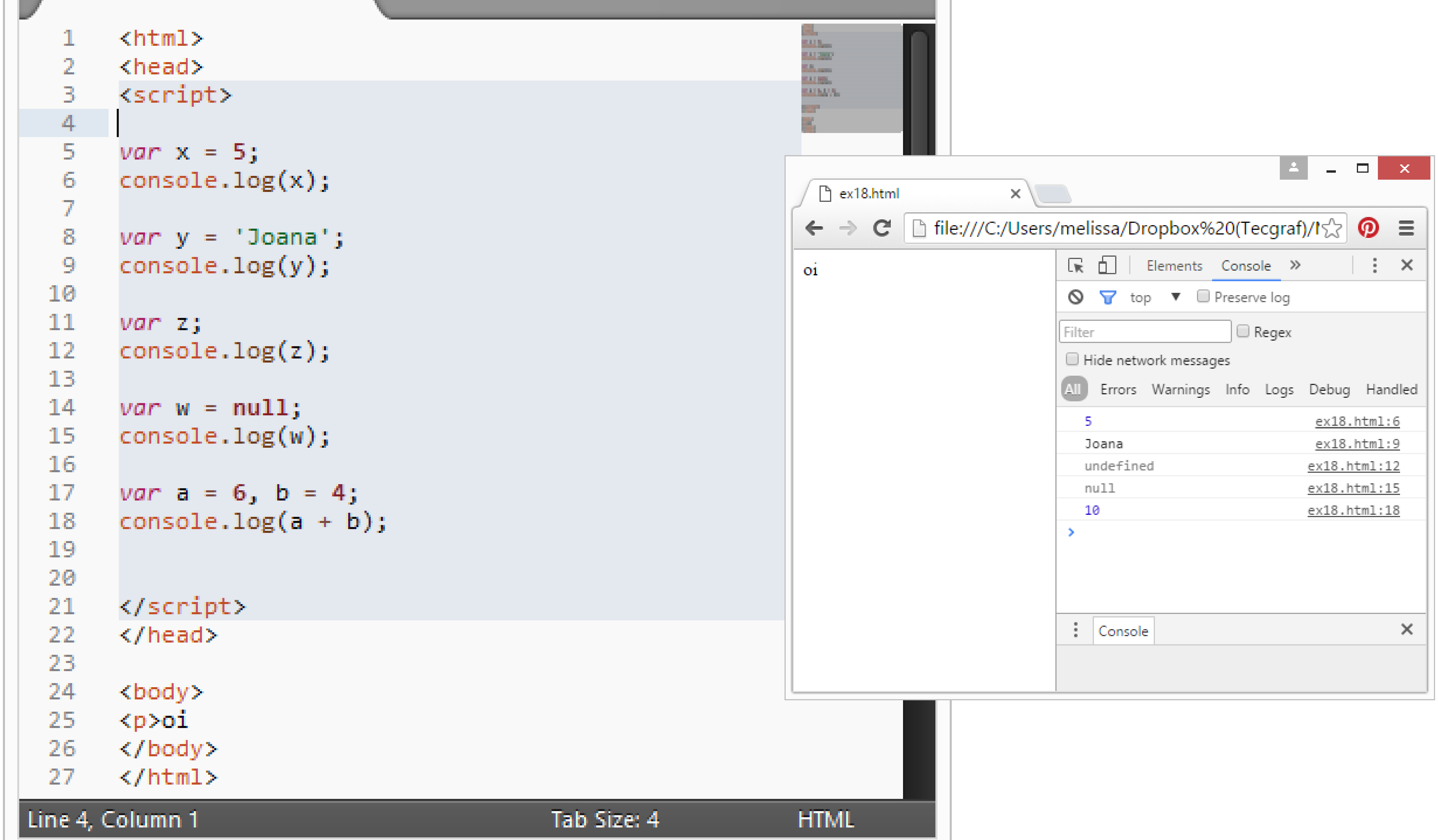
An **equal sign** is used to **assign values** to variables.

```
x = 10;
```

Variables

- If you ask for the value of an empty variable, you'll get the value undefined.

- A single var statement may define multiple variables. The definitions must be separated by commas.



```html
<html>
<head>
<script>

var x = 5;
console.log(x);

var y = 'Joana';
console.log(y);

var z;
console.log(z);

var w = null;
console.log(w);

var a = 6, b = 4;
console.log(a + b);


</script>
</head>

<body>
<p>oi
</body>
</html>
```

oi

```
5                 ex18.html:6
Joana             ex18.html:9
undefined         ex18.html:12
null              ex18.html:15
10                ex18.html:18
>
```

Variables – Dynamic Types

- JavaScript has dynamic types. This means that the same variable can be used as different types.

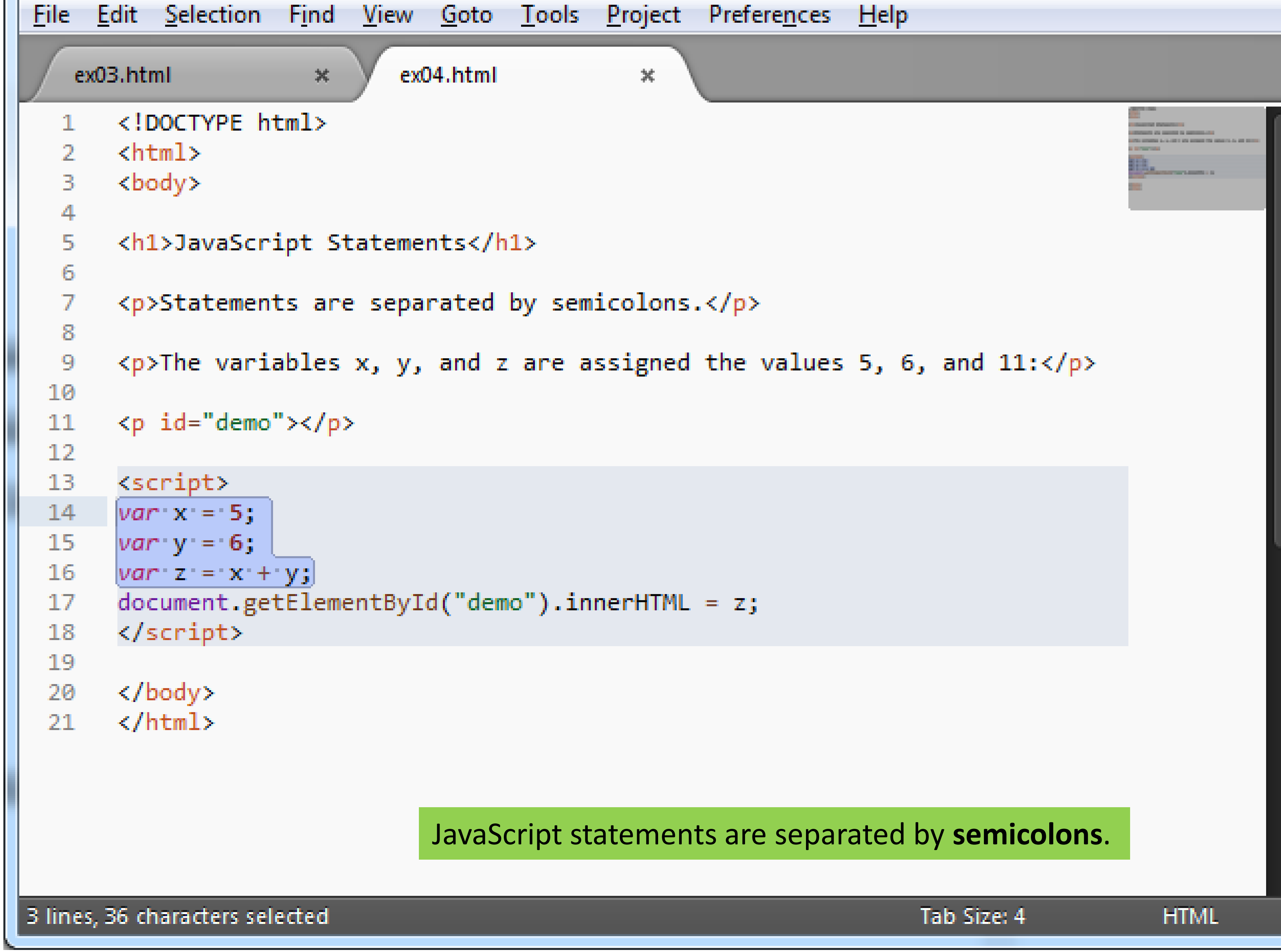
```
var x;                  // Now x is undefined
var x = 5;              // Now x is a Number
var x = "John";         // Now x is a String
```

Reserved Words

- Words with a special meaning, such as var, are keywords, and they may not be used as variable names.

```
break case catch class const continue debugger
default delete do else enum export extends false
finally for function if implements import in
instanceof interface let new null package private
protected public return static super switch this
throw true try typeof var void while with yield
```

Statements



File Edit Selection Find View Goto Tools Project Preferences Help
ex03.html
ex04.html

```
<!DOCTYPE html>
<html>
<body>

<h1>JavaScript Statements</h1>

<p>Statements are separated by semicolons.</p>

<p>The variables x, y, and z are assigned the values 5, 6, and 11:</p>

<p id="demo"></p>

<script>
var x = 5;
var y = 6;
var z = x + y;
document.getElementById("demo").innerHTML = z;
</script>

</body>
</html>
```

3 lines, 36 characters selected

JavaScript statements are separated by **semicolons**.

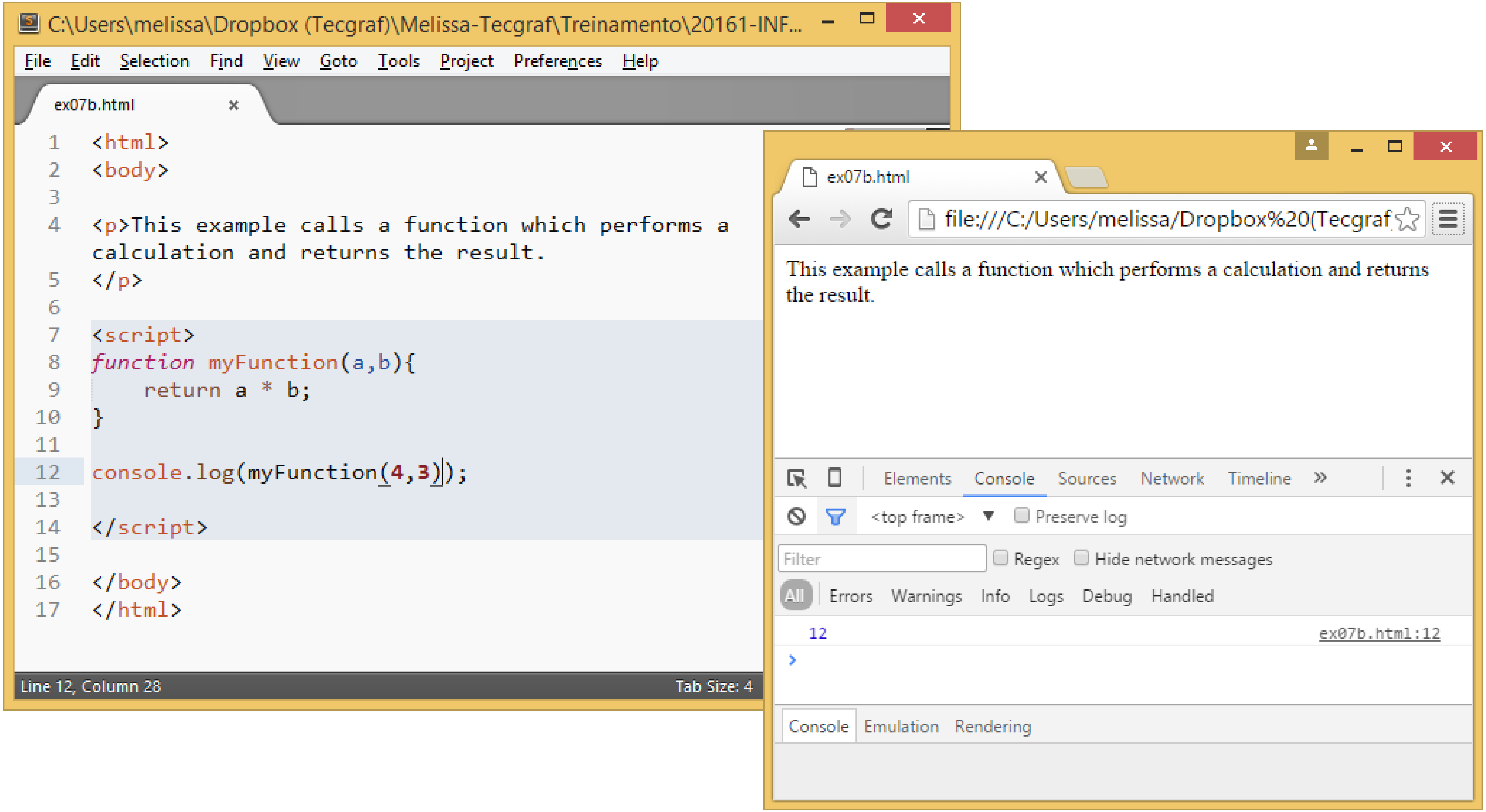
Comments

```
var x = 5;       // Declare x, give it the value of 5


/*
This is a multi-line
  comment.
*/
```

JavaScript comments: use // and /* e */.

Functions

```
function name(parameter1, parameter2, parameter3) {
    code to be executed
}
```


C:\Users\melissa\Dropbox (Tecgraf)\Melissa-Tecgraf\Treinamento\20161-INF...
File Edit Selection Find View Goto Tools Project Preferences Help
ex07b.html

```
<html>
<body>

<p>This example calls a function which performs a
calculation and returns the result.
</p>

<script>
function myFunction(a,b){
    return a * b;
}

console.log(myFunction(4,3));

</script>

</body>
</html>
```

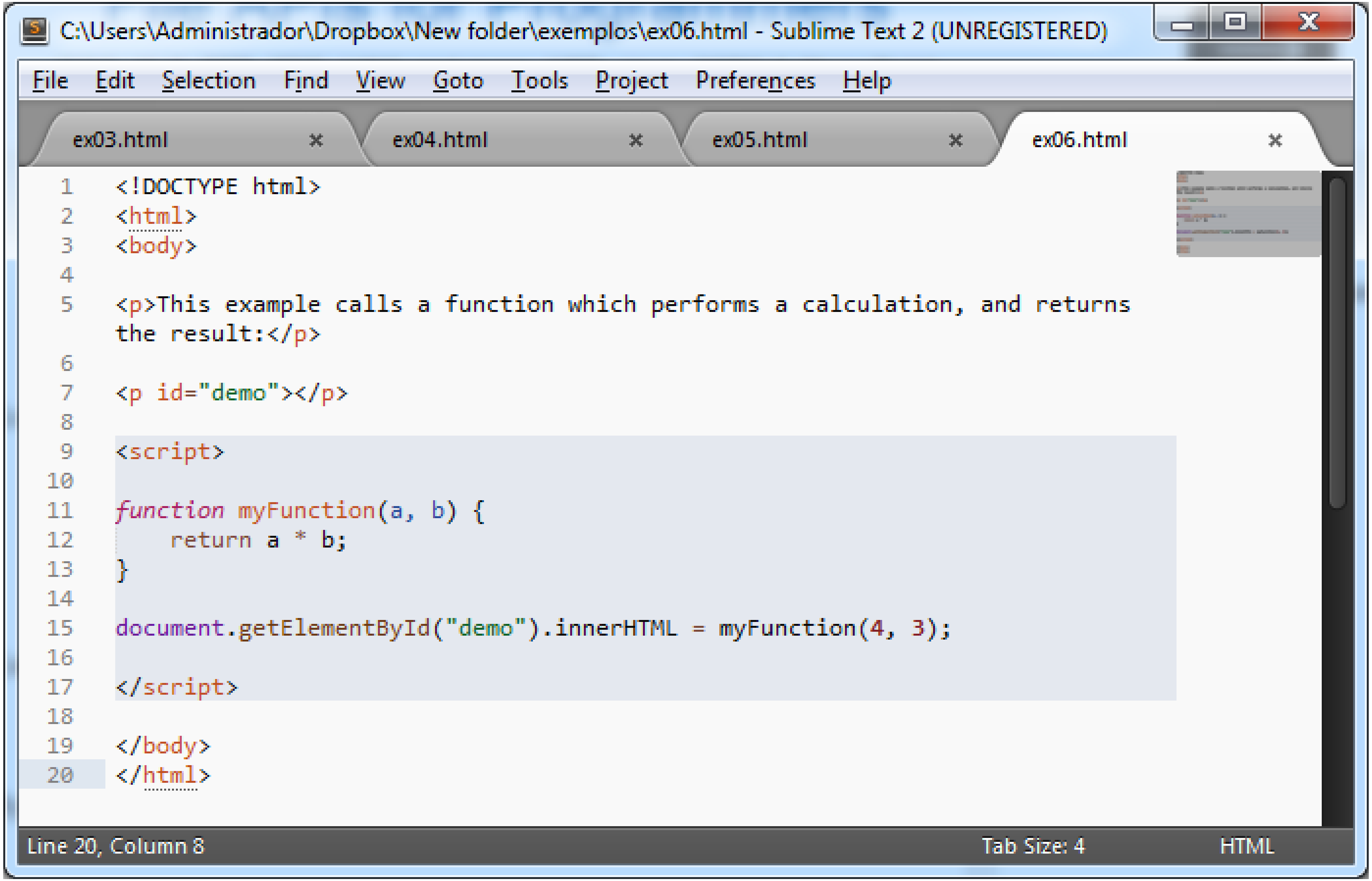
Line 12, Column 28
Tab Size: 4
ex07b.html
file:///C:/Users/melissa/Dropbox%20(Tecgraf

This example calls a function which performs a calculation and returns the result.

Elements Console Sources Network Timeline »
<top frame> ▼ □ Preserve log
Filter □ Regex □ Hide network messages
All Errors Warnings Info Logs Debug Handled

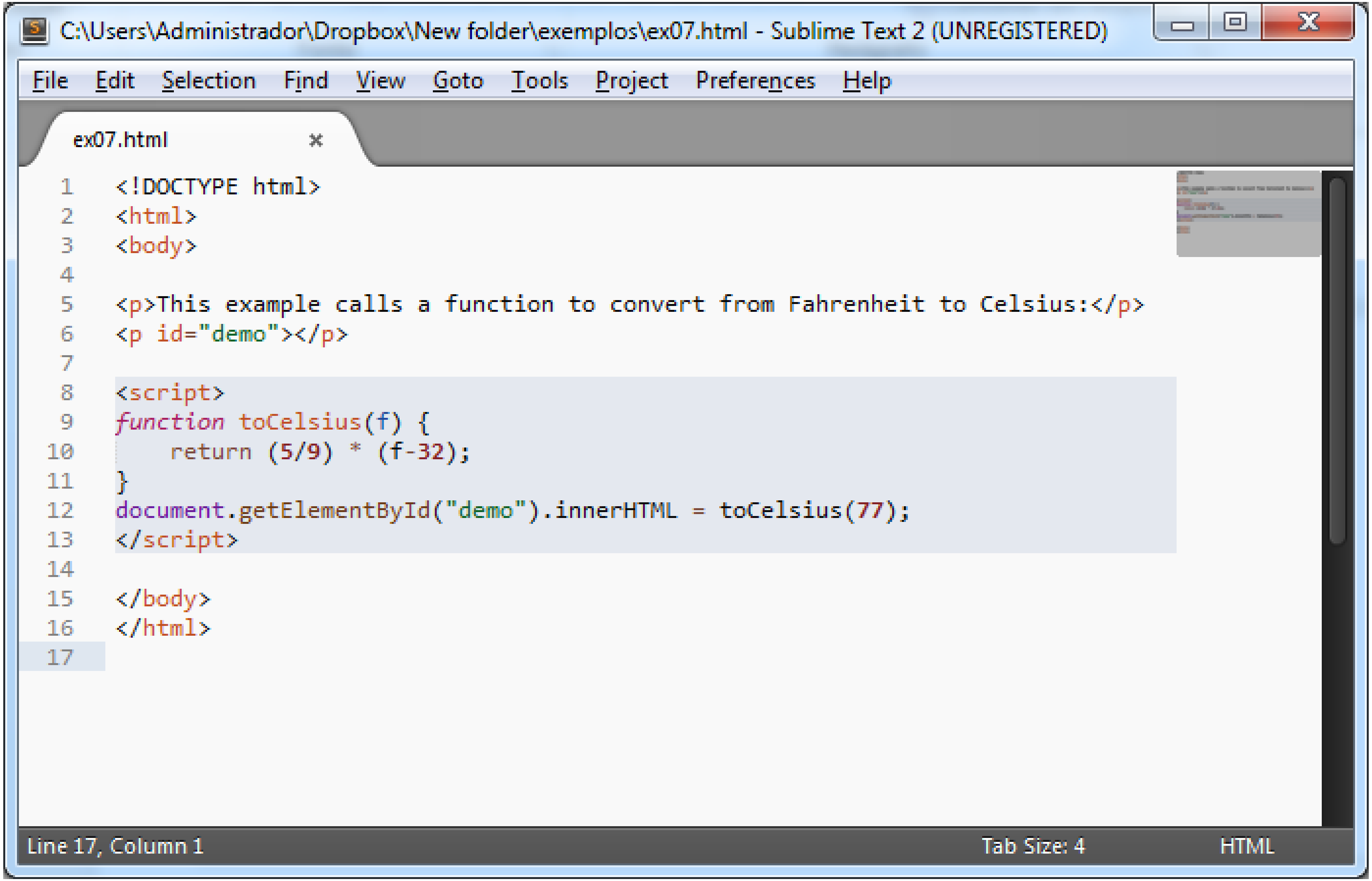
```
12                                                    ex07b.html:12
>
```

Console Emulation Rendering



C:\Users\Administrador\Dropbox\New folder\exemplos\ex06.html - Sublime Text 2 (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
ex03.html
ex04.html
ex05.html
ex06.html
Line 20, Column 8
Tab Size: 4
HTML

```
<!DOCTYPE html>
<html>
<body>

<p>This example calls a function which performs a calculation, and returns the result:</p>

<p id="demo"></p>

<script>

function myFunction(a, b) {
    return a * b;
}

document.getElementById("demo").innerHTML = myFunction(4, 3);

</script>

</body>
</html>
```



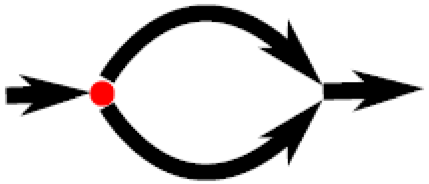
```html
<!DOCTYPE html>
<html>
<body>

<p>This example calls a function to convert from Fahrenheit to Celsius:</p>
<p id="demo"></p>

<script>
function toCelsius(f) {
    return (5/9) * (f-32);
}
document.getElementById("demo").innerHTML = toCelsius(77);
</script>

</body>
</html>

```

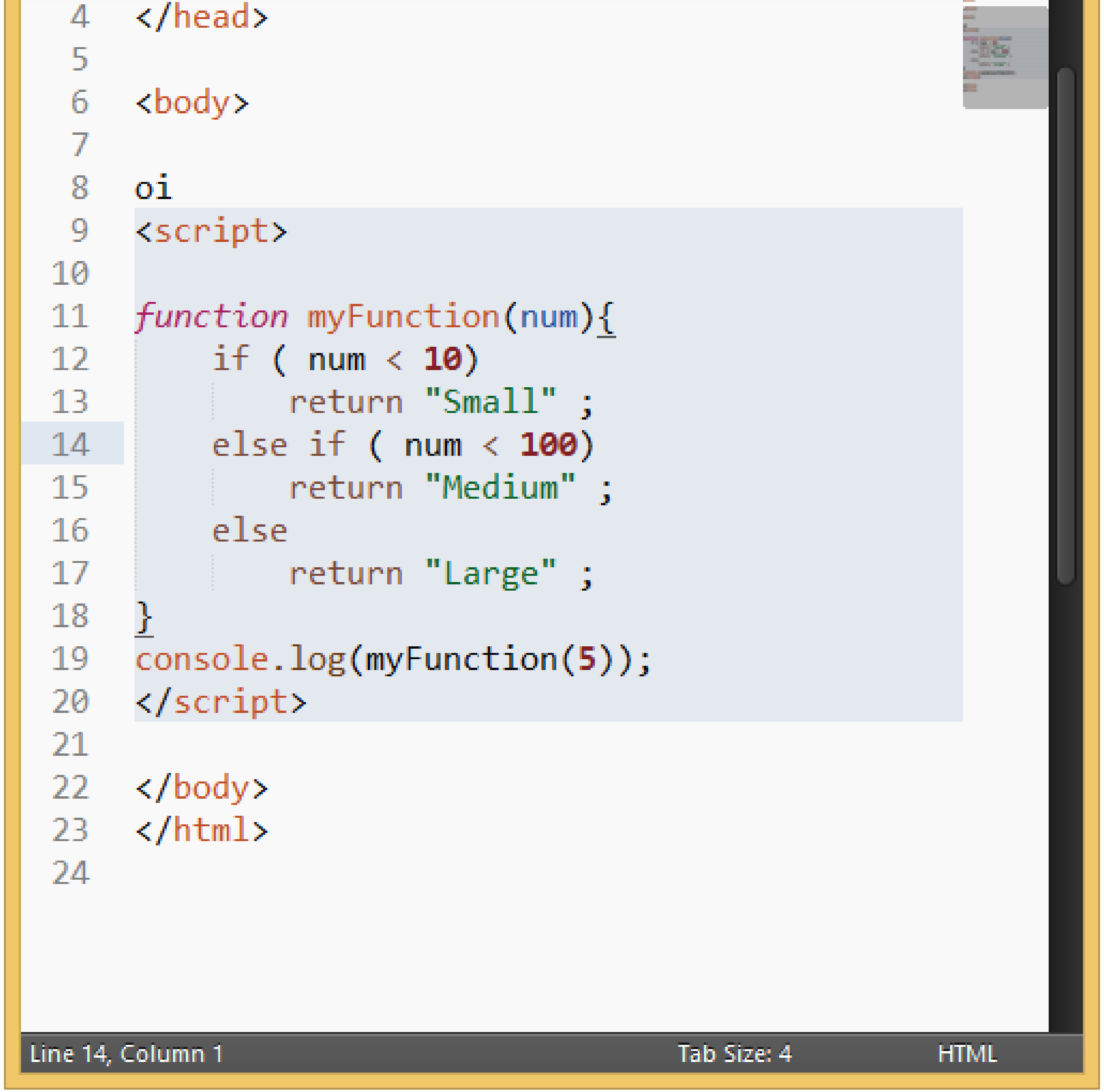
Scope

- In JavaScript, scope is the set of variables, objects, and functions you have access to.
- Local
  - Variables declared within a JavaScript function, become **LOCAL** to the function. They can only be accessed within the function.
  - Local variables are created when a function starts, and deleted when the function is completed.
  - Function arguments (parameters) work as local variables inside functions.
- Global
  - A variable declared outside a function, becomes **GLOBAL.** All scripts and functions on a web page can access it.
  - If you assign a value to a variable that has not been declared, it will automatically become a **GLOBAL** variable.

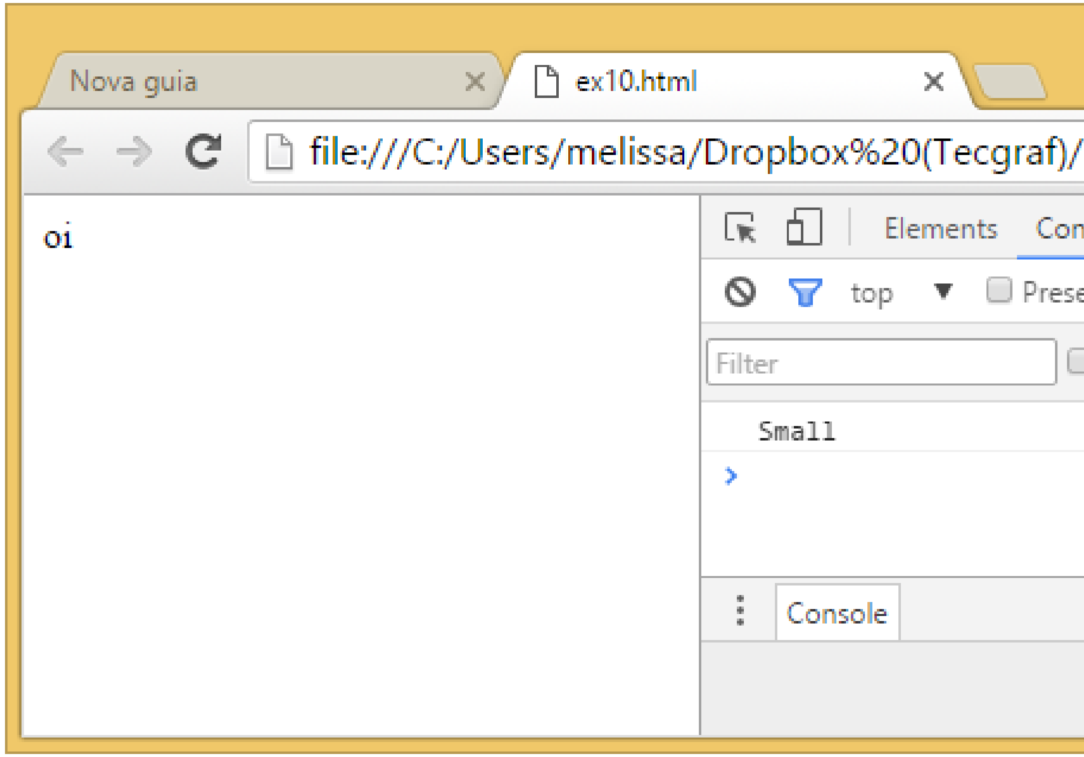
Conditions

- *Conditional execution*: where we choose between two different routes based on a Boolean value.





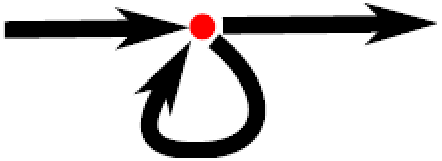
```
</head>

<body>

oi
<script>

function myFunction(num){
    if ( num < 10)
        return "Small" ;
    else if ( num < 100)
        return "Medium" ;
    else
        return "Large" ;
}
console.log(myFunction(5));
</script>

</body>
</html>

```



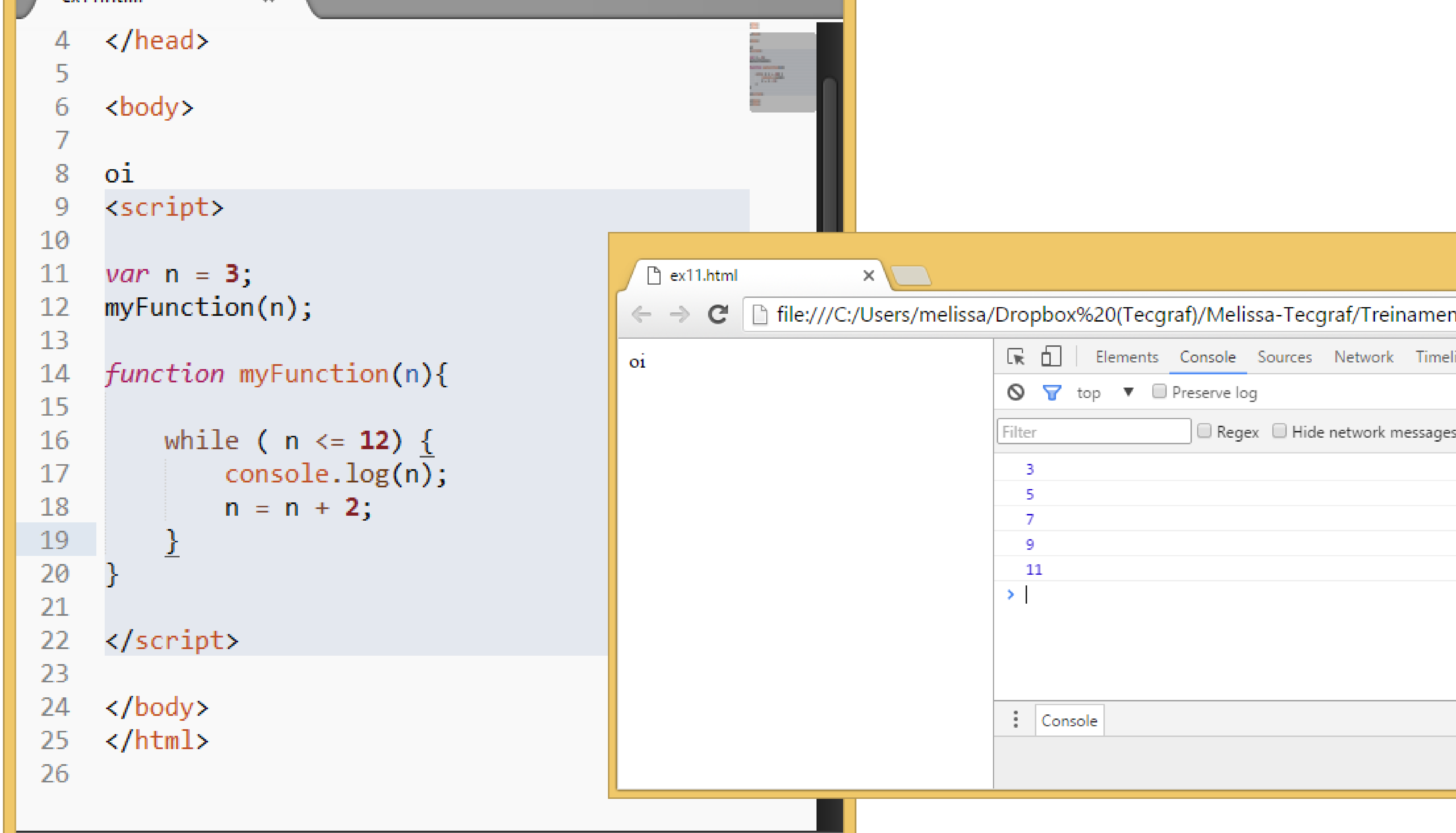
oi

```
Small
>
```

Loops



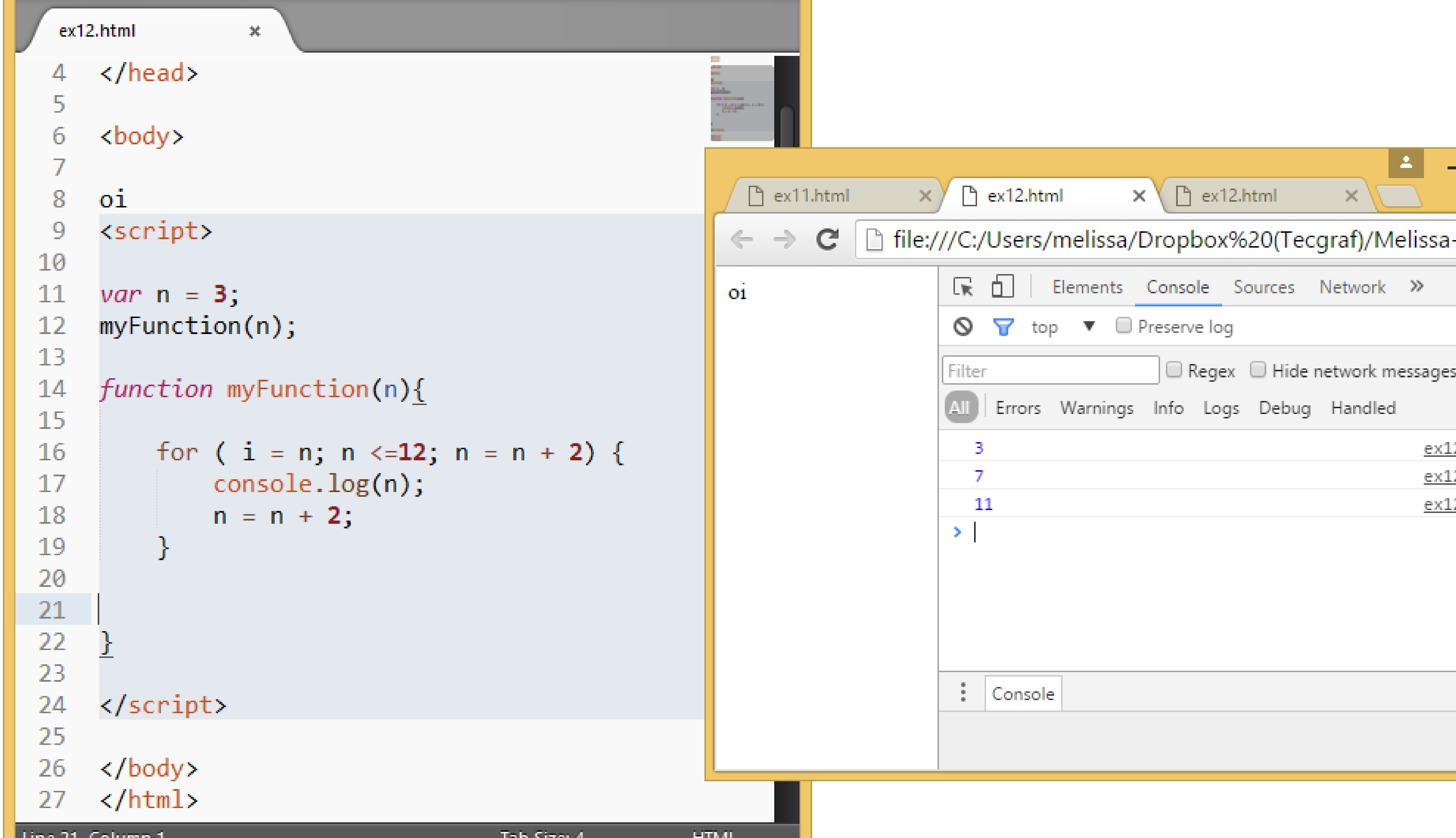
- *To repeat some code*



```html
</head>

<body>

oi
<script>

var n = 3;
myFunction(n);

function myFunction(n){

    while ( n <= 12) {
        console.log(n);
        n = n + 2;
    }
}

</script>

</body>
</html>

```

oi

```
3
5
7
9
11
>
```



```
</head>

<body>

oi
<script>

var n = 3;
myFunction(n);

function myFunction(n){

    for ( i = n; n <=12; n = n + 2) {
        console.log(n);
        n = n + 2;
    }


}

</script>

</body>
</html>
```

file:///C:/Users/melissa/Dropbox%20(Tecgraf)/Melissa-

oi

```
3
7
11
>
```

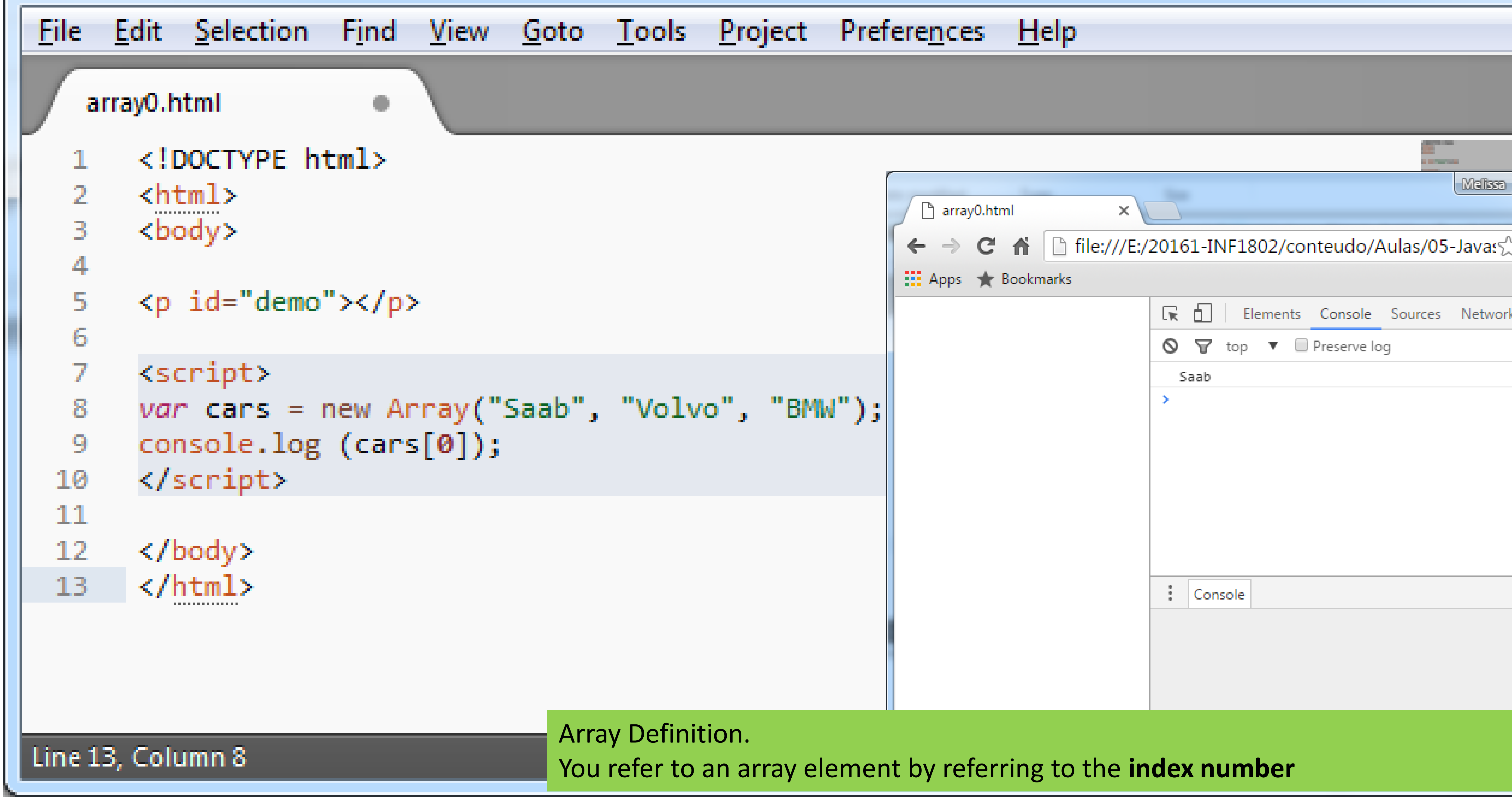
Exercises

- MIN(number1, number2)
  - Write a function min that takes two arguments and returns their minimum.
- CALCULATOR(operation, number1, number2)
  - Write a calculator function. It takes three arguments (two numbers and an operation) and makes addition, subtraction, multiplication, division.
- ODDNUMBERS(number1, number2)
  - Write a function that displays odd numbers from number1 to number2, received as arguments.

Data Structures: Arrays, Objects

Arrays

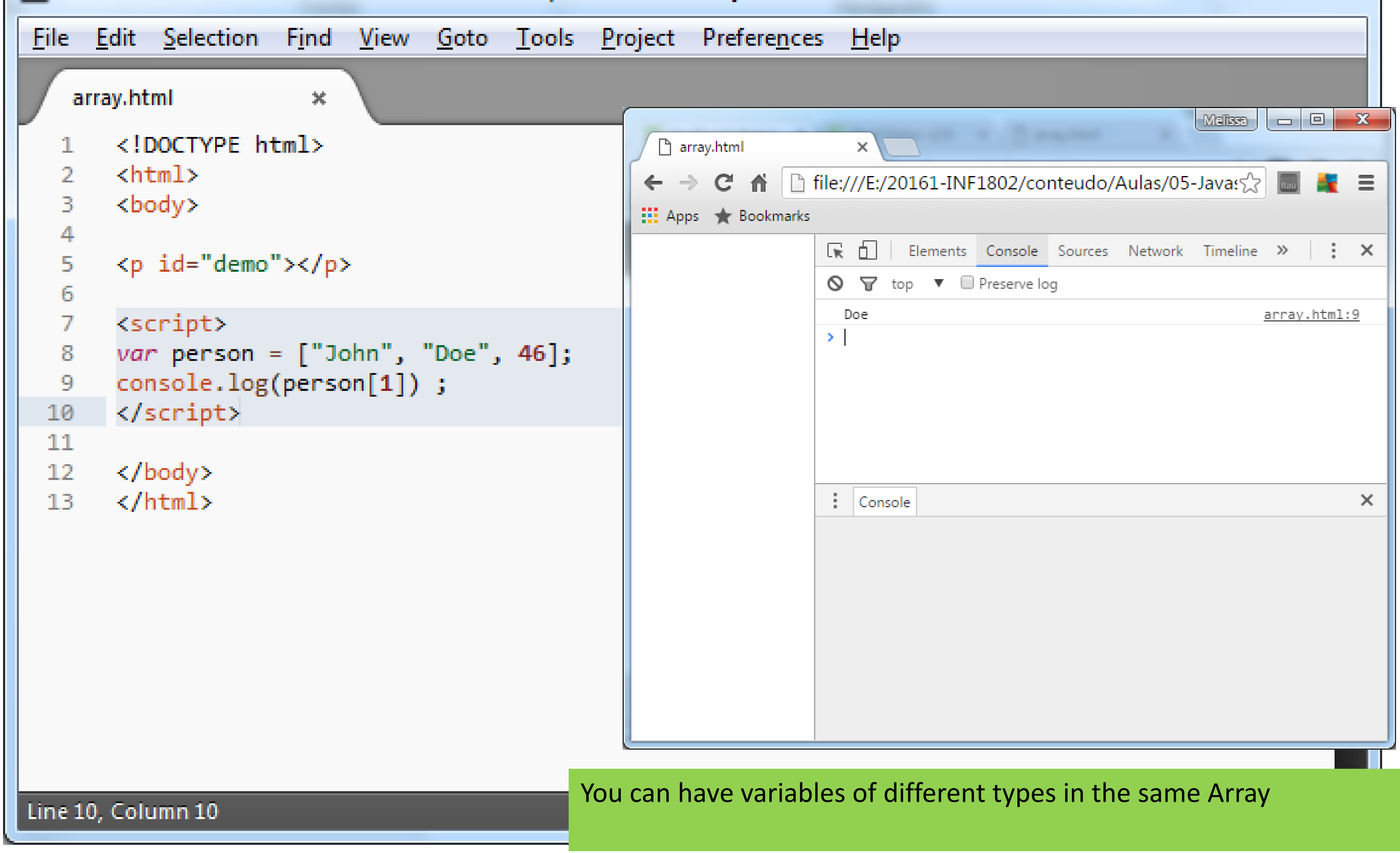
- An array is a special variable, which can hold more than one value at a time.
- An array can hold many values under a single name, and you can access the values by referring to an index number.
- Arrays are a special type of objects. The **typeof** operator in JavaScript returns "object" for arrays.


File Edit Selection Find View Goto Tools Project Preferences Help
array0.html
Line 13, Column 8
array0.html
file:///E:/20161-INF1802/conteudo/Aulas/05-Javas
Apps
Bookmarks
Melissa
Elements
Console
Sources
Networ
top
Preserve log
Console

```
<!DOCTYPE html>
<html>
<body>

<p id="demo"></p>

<script>
var cars = new Array("Saab", "Volvo", "BMW");
console.log (cars[0]);
</script>

</body>
</html>
```

```
Saab
>
```

Array Definition.
You refer to an array element by referring to the **index number**

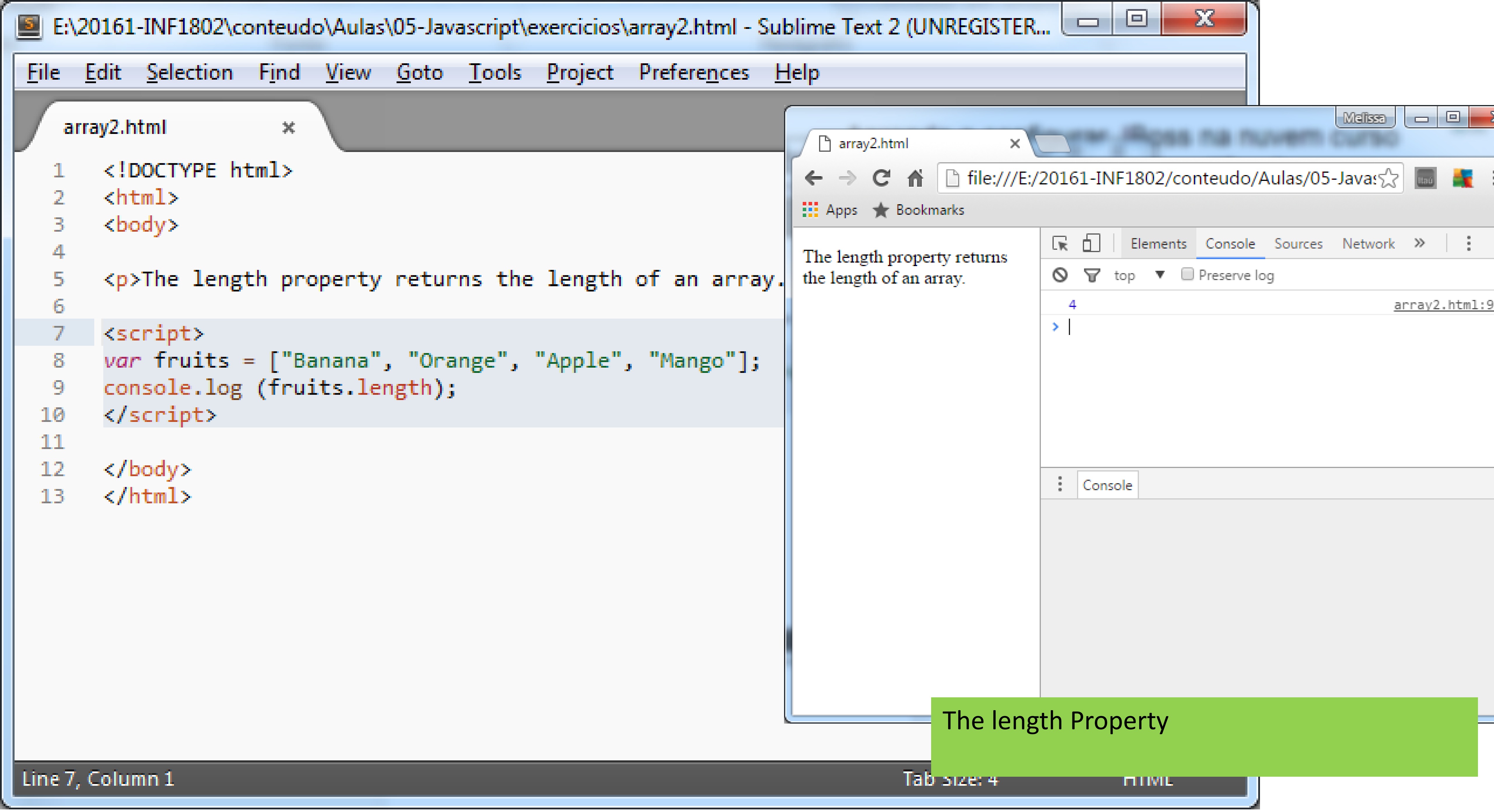

File Edit Selection Find View Goto Tools Project Preferences Help
array.html
Line 10, Column 10
array.html
file:///E:/20161-INF1802/conteudo/Aulas/05-Javas
Apps
Bookmarks
Elements Console Sources Network Timeline
top
Preserve log
Console

```
<!DOCTYPE html>
<html>
<body>

<p id="demo"></p>

<script>
var person = ["John", "Doe", 46];
console.log(person[1]) ;
</script>

</body>
</html>
```

```
Doe                                         array.html:9
>
```

You can have variables of different types in the same Array

The length Property

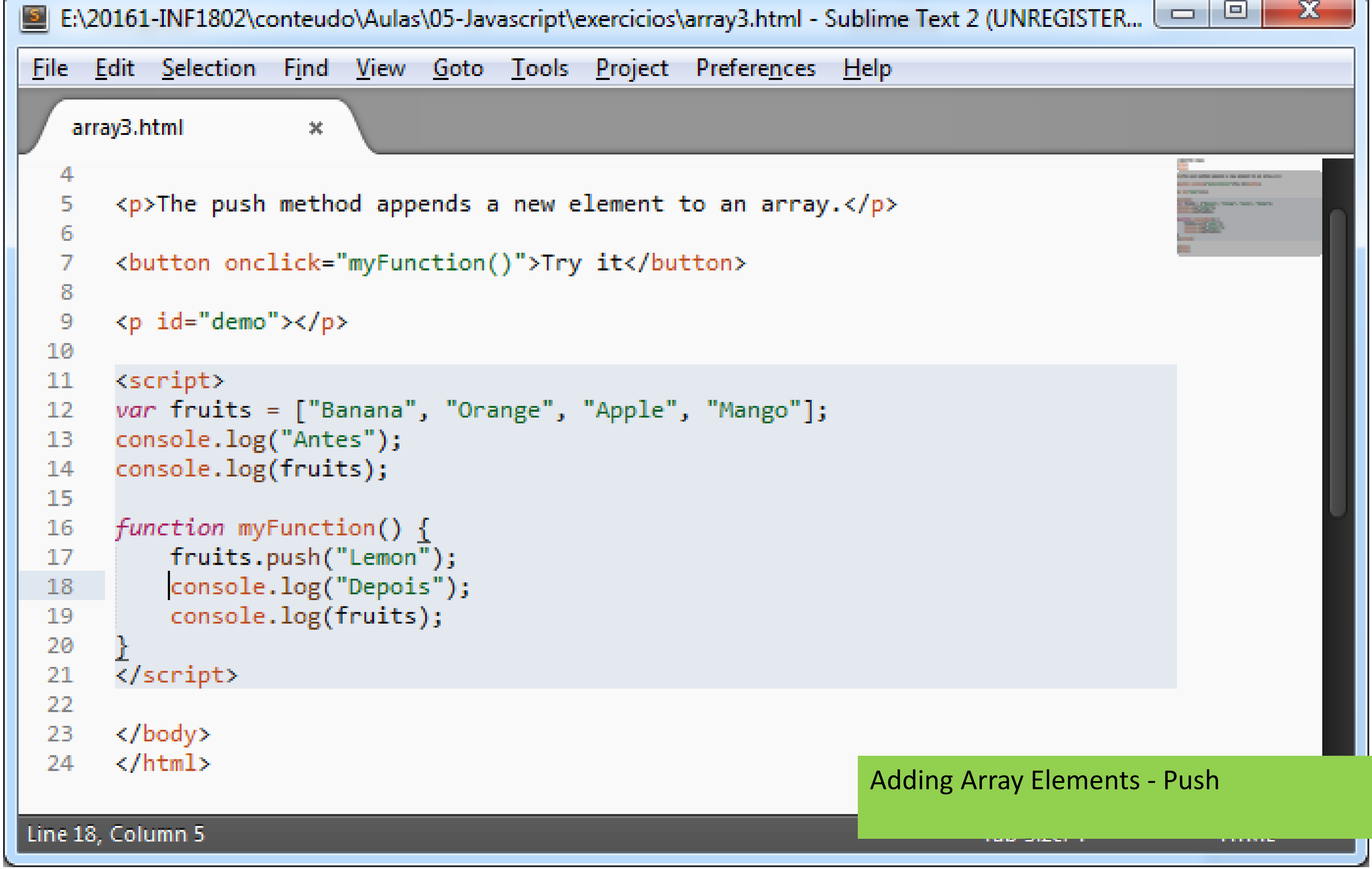

E:\20161-INF1802\conteudo\Aulas\05-Javascript\exercicios\array2.html - Sublime Text 2 (UNREGISTER...
File Edit Selection Find View Goto Tools Project Preferences Help
array2.html
Line 7, Column 1
Tab Size: 4
HTML
array2.html
file:///E:/20161-INF1802/conteudo/Aulas/05-Javas
Apps
Bookmarks
Elements Console Sources Network
top
Preserve log
Console

```
<!DOCTYPE html>
<html>
<body>

<p>The length property returns the length of an array.

<script>
var fruits = ["Banana", "Orange", "Apple", "Mango"];
console.log (fruits.length);
</script>

</body>
</html>
```

The length property returns the length of an array.

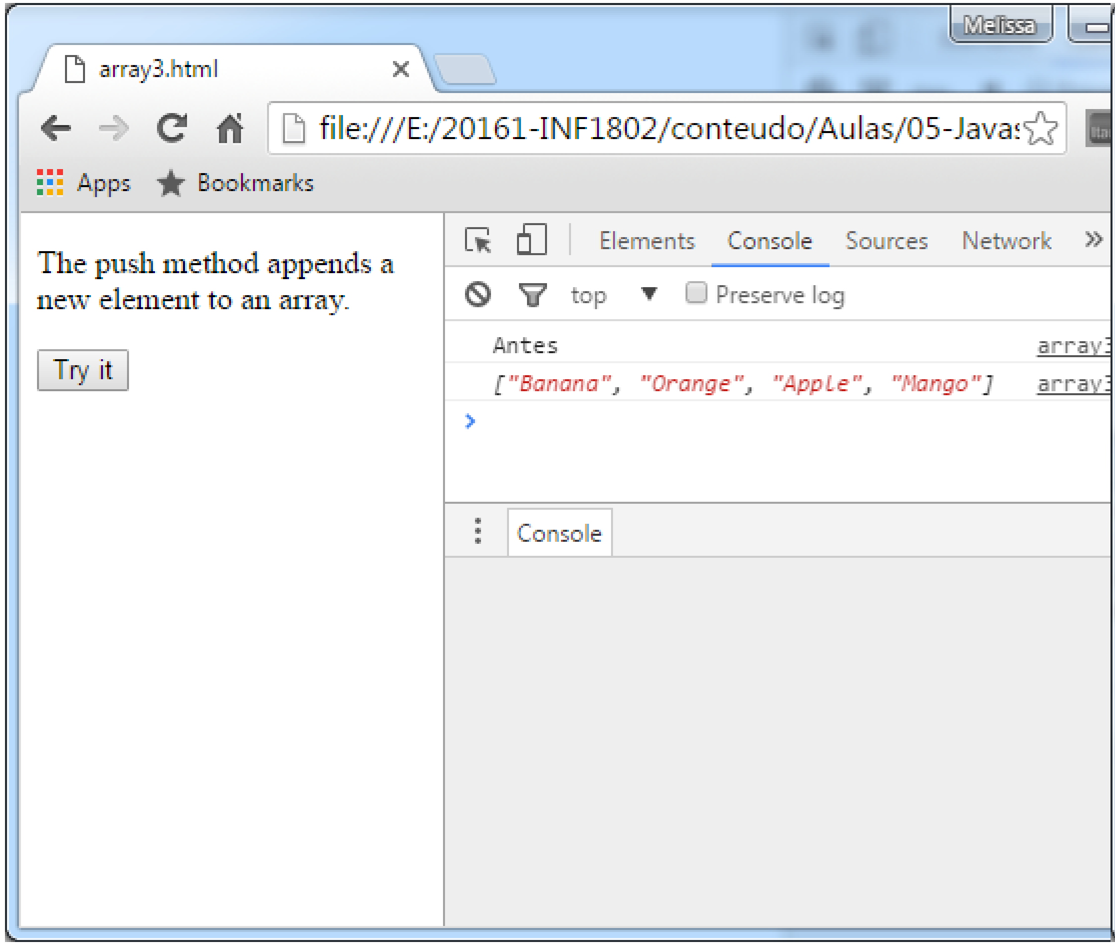
```
4                                                    array2.html:9
>
```

Adding Array Elements - Push


E:\20161-INF1802\conteudo\Aulas\05-Javascript\exercicios\array3.html - Sublime Text 2 (UNREGISTER...
File Edit Selection Find View Goto Tools Project Preferences Help
array3.html

```

<p>The push method appends a new element to an array.</p>

<button onclick="myFunction()">Try it</button>

<p id="demo"></p>

<script>
var fruits = ["Banana", "Orange", "Apple", "Mango"];
console.log("Antes");
console.log(fruits);

function myFunction() {
    fruits.push("Lemon");
    console.log("Depois");
    console.log(fruits);
}
</script>

</body>
</html>
```

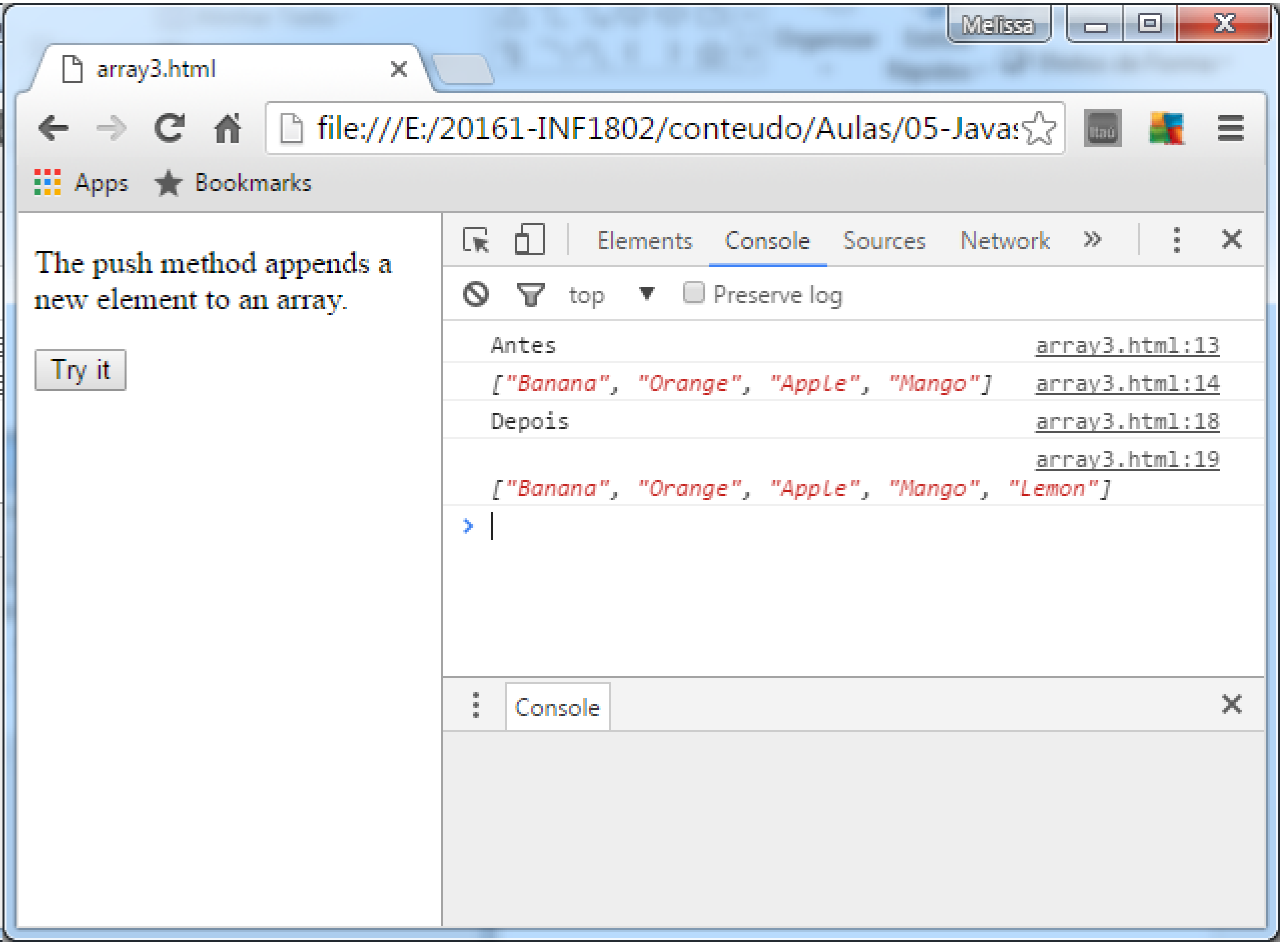
Line 18, Column 5



The push method appends a new element to an array.

Try it

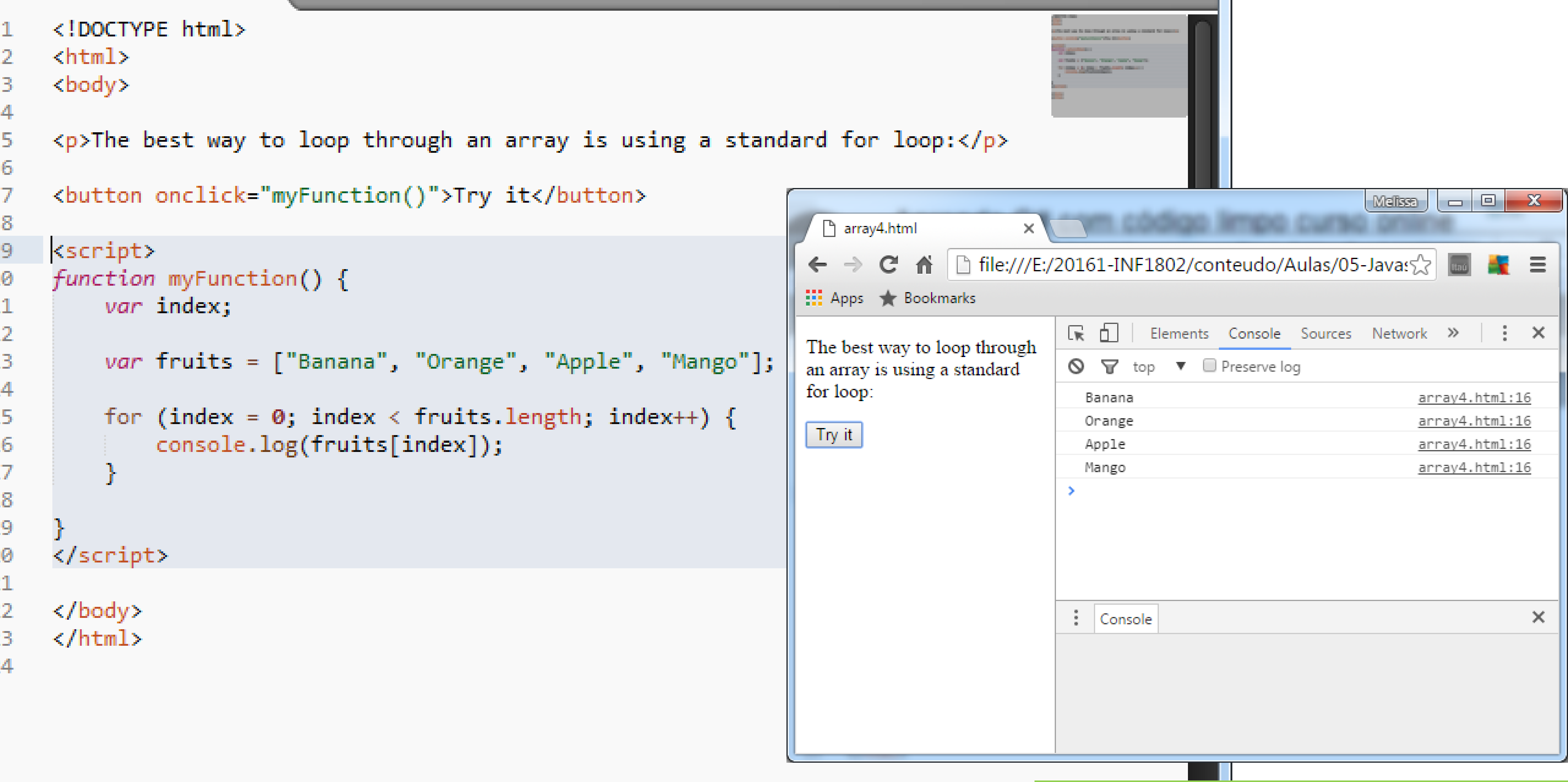
```
Antes                                                array3
["Banana", "Orange", "Apple", "Mango"]               array3
>
```



The push method appends a new element to an array.

Try it

```
Antes                                                array3.html:13
["Banana", "Orange", "Apple", "Mango"]               array3.html:14
Depois                                               array3.html:18
                                                     array3.html:19
["Banana", "Orange", "Apple", "Mango", "Lemon"]
>
```



```
<!DOCTYPE html>
<html>
<body>

<p>The best way to loop through an array is using a standard for loop:</p>

<button onclick="myFunction()">Try it</button>

<script>
function myFunction() {
    var index;

    var fruits = ["Banana", "Orange", "Apple", "Mango"];

    for (index = 0; index < fruits.length; index++) {
        console.log(fruits[index]);
    }

}
</script>

</body>
</html>
```

The best way to loop through an array is using a standard for loop:

```
Banana                                   array4.html:16
Orange                                   array4.html:16
Apple                                    array4.html:16
Mango                                    array4.html:16
>
```

Looping Array Elements

Arrays – more …

- http://www.w3schools.com/js/js_arrays.asp
    - Test yourself with Exercises 2 - 5

Exercises

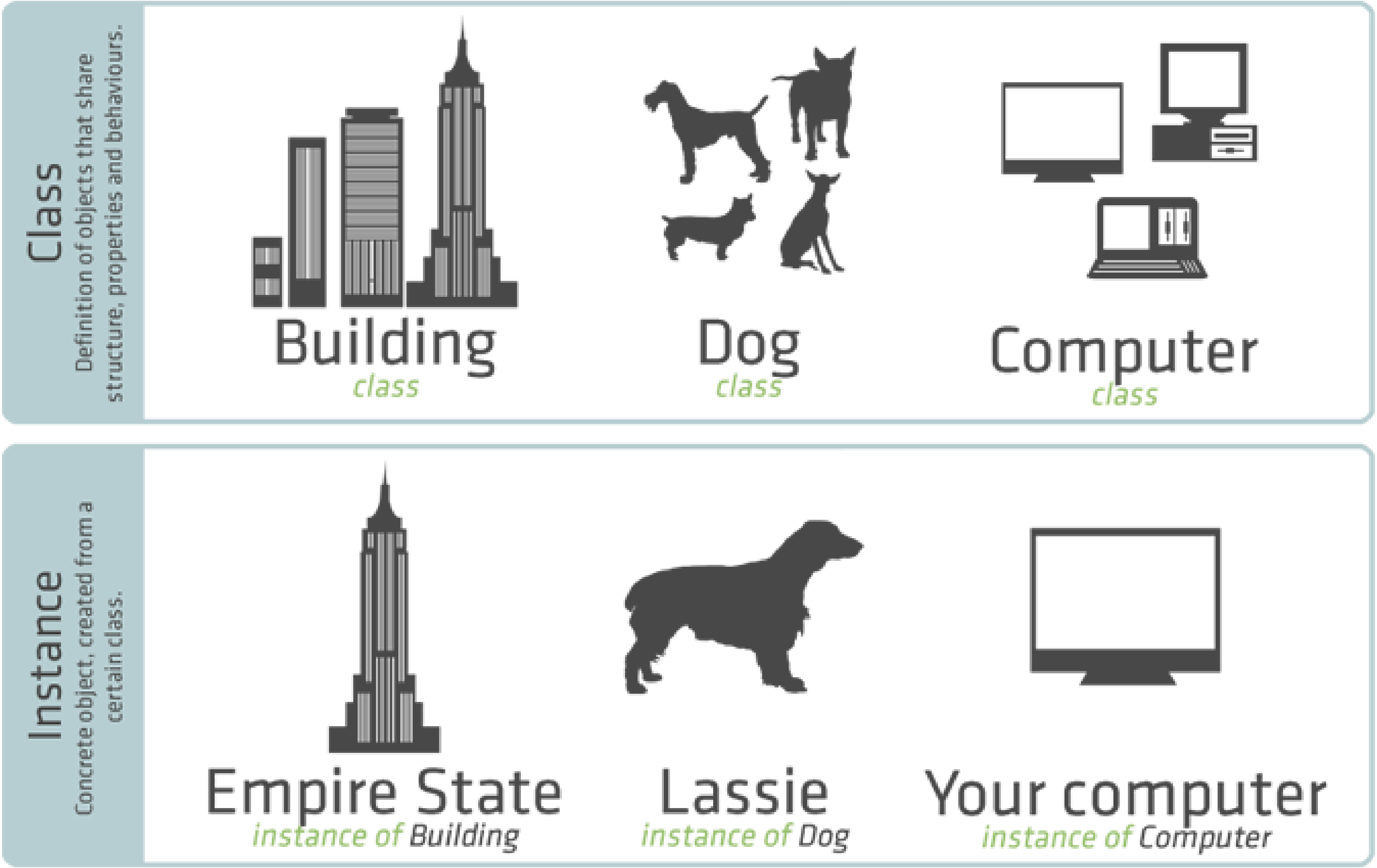
- WHOIS(array, index)
  - Write a function that tells who is the content of a value in a position x of an array.
- RANGE(start, end)
  - Write a range function that takes two arguments, start and end, and returns an array containing all the numbers from start up to (and including) end.
- SUM(array)
  - Write a sum function that takes an array of numbers and returns the sum of these numbers.
- REVERSEARRAY(array)
  - Write a function that takes an array as argument and produces a *new* array that has the same elements in the inverse order.

Objects

- Software objects are often used to model the real-world objects that you find in everyday life. *
- An object is a software bundle of related state and behavior. *
- An object can be considered a "*thing*" that can perform a set of related activities. The set of activities that the object performs defines the object's behavior. **

* https://docs.oracle.com/javase/tutorial/java/concepts/

** http://www.codeproject.com/Articles/22769/Introduction-to-Object-Oriented-Programming-Concep#Object



From https://docs.sencha.com/extjs/6.0/other_resources/oop_concepts.html

Example - Object Car

Properties

- name
- model
- weight
- color

Behavior

- car.start()
- car.drive()
- car.brake ()
- car.stop()

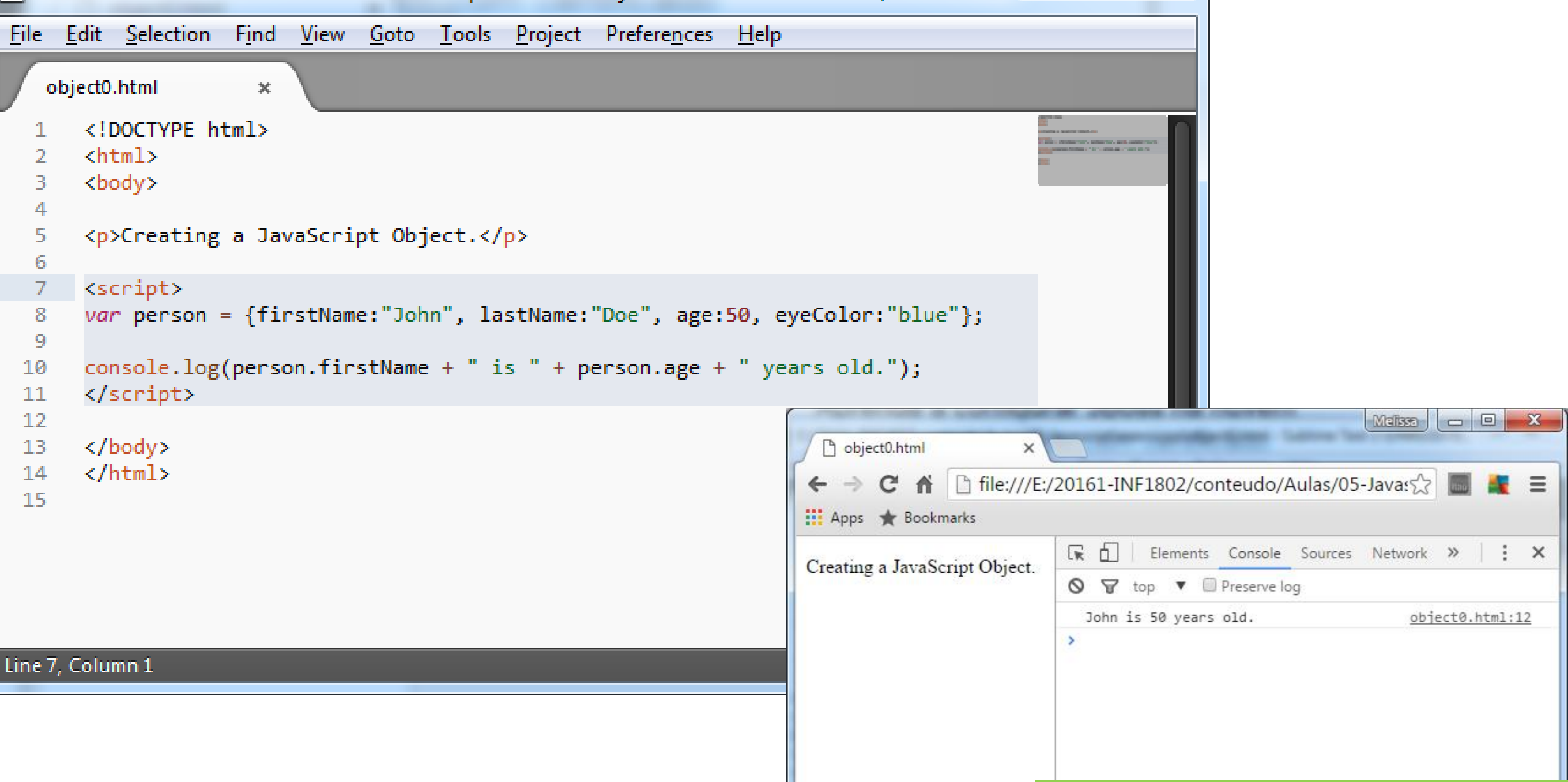
Example - Object Person

Properties

- name
- age
- e-mail

Behavior

- person.walk()
- person.sleep()
- person.eat ()

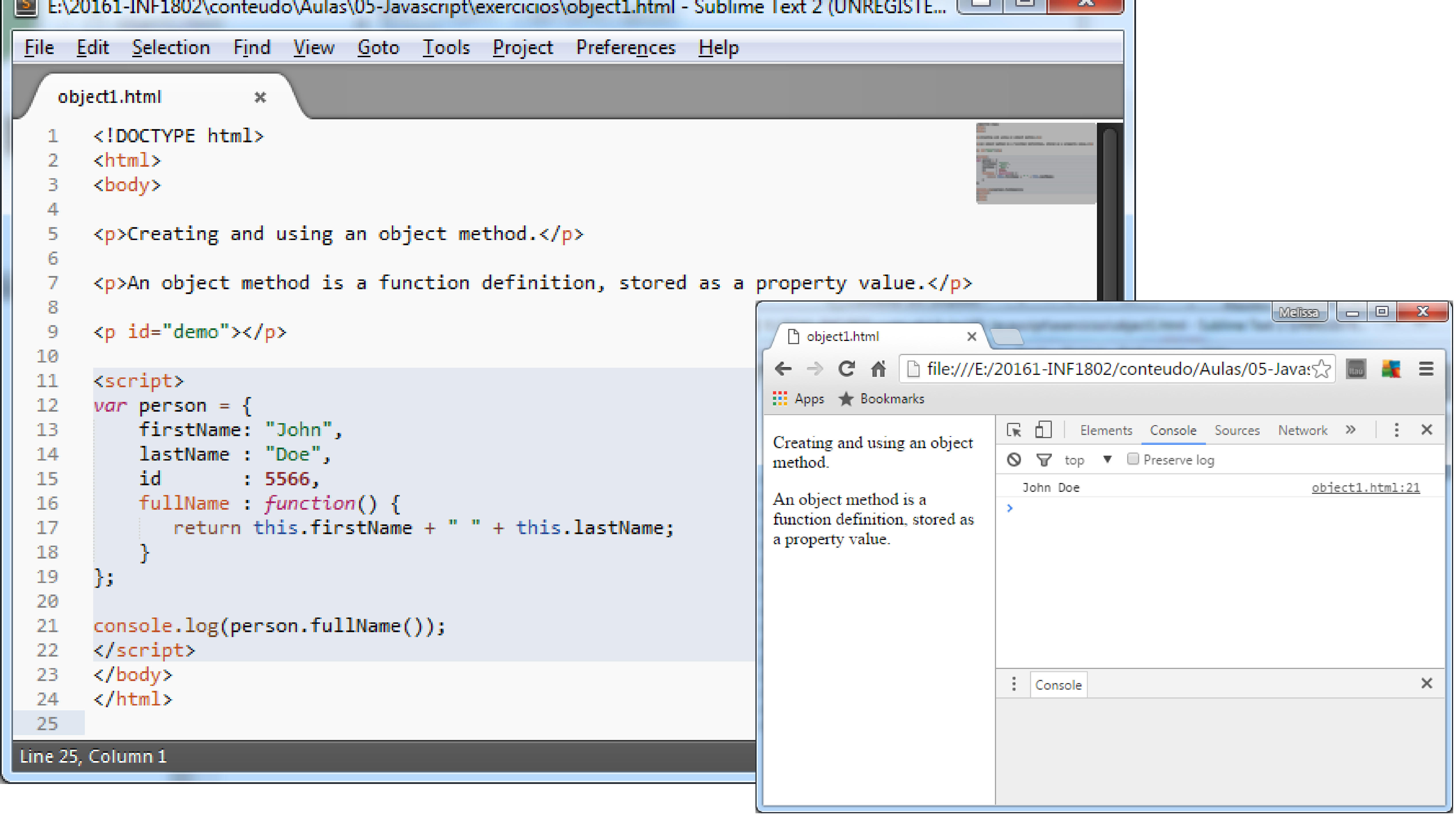

File Edit Selection Find View Goto Tools Project Preferences Help
object0.html
Line 7, Column 1
object0.html
file:///E:/20161-INF1802/conteudo/Aulas/05-Javas
Apps
Bookmarks
Elements Console Sources Network
top
Preserve log

```html
<!DOCTYPE html>
<html>
<body>

<p>Creating a JavaScript Object.</p>

<script>
var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};

console.log(person.firstName + " is " + person.age + " years old.");
</script>

</body>
</html>

```

Creating a JavaScript Object.

```
John is 50 years old.                                    object0.html:12
>
```

Creating a javascript object


E:\20161-INF1802\conteudo\Aulas\05-Javascript\exercicios\object1.html - Sublime Text 2 (UNREGISTE...
File Edit Selection Find View Goto Tools Project Preferences Help
object1.html

```
<!DOCTYPE html>
<html>
<body>

<p>Creating and using an object method.</p>

<p>An object method is a function definition, stored as a property value.</p>

<p id="demo"></p>

<script>
var person = {
    firstName: "John",
    lastName : "Doe",
    id       : 5566,
    fullName : function() {
       return this.firstName + " " + this.lastName;
    }
};

console.log(person.fullName());
</script>
</body>
</html>

```

Line 25, Column 1
Melissa
object1.html
file:///E:/20161-INF1802/conteudo/Aulas/05-Javas
Apps
Bookmarks
Elements
Console
Sources
Network
top
Preserve log

Creating and using an object method.

An object method is a function definition, stored as a property value.

```
John Doe                                          object1.html:21
>
```

Objects

Property

- Definition
- *Access*
  - *`objectName.propertyName`*
  - `person.lastName;`
- *Declaration*

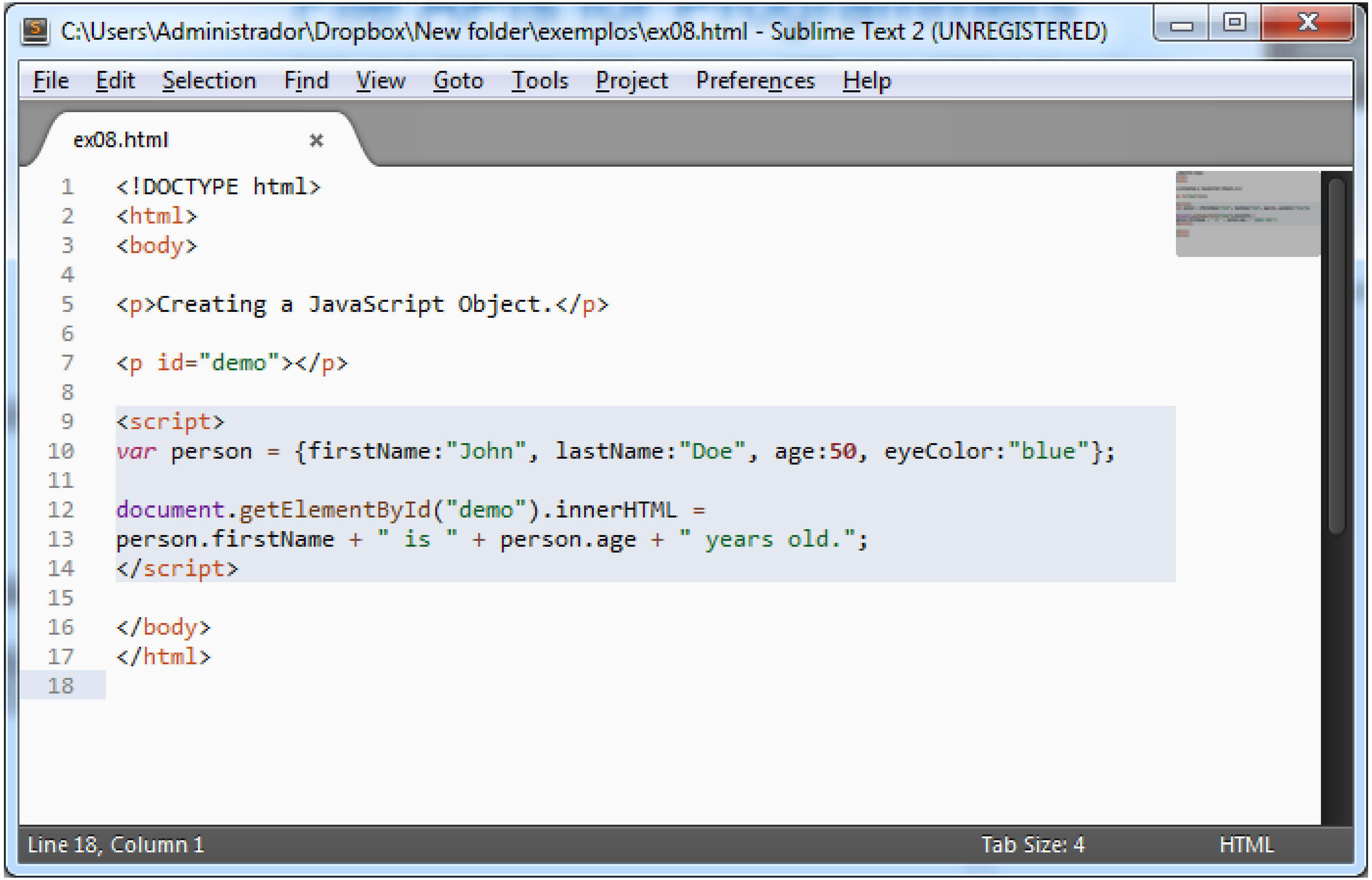
```
var person = {
    firstName:"John",
    lastName:"Doe",
    age:50,
    eyeColor:"blue“
    };
```

Methods

- Actions that can be performed on objects.
- *Access*
  - *`objectName.methodName()`*
  - `name = person.fullName();`
- *Declaration*

```
var person = {
        firstName: "John",
        lastName : "Doe",
        id       : 5566,
        fullName : function() {
           return this.firstName + " " +
                        this.lastName;
        }
```

Objects



C:\Users\Administrador\Dropbox\New folder\exemplos\ex08.html - Sublime Text 2 (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
ex08.html

```
<!DOCTYPE html>
<html>
<body>

<p>Creating a JavaScript Object.</p>

<p id="demo"></p>

<script>
var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};

document.getElementById("demo").innerHTML =
person.firstName + " is " + person.age + " years old.";
</script>

</body>
</html>

```

Line 18, Column 1
Tab Size: 4
HTML

Objects – more …

- http://www.w3schools.com/js/js_objects.asp
    - Test yourself with Exercises 2 and 3