Javascript

PART 3

Javascript output

- JavaScript can "display" data in different ways:
  - Writing into an alert box, using **window.alert()**
    - This is the one that we have been using so far
  - Writing into an HTML element, using **innerHTML**.
    - For adding text to a paragraph, a label, or any other html element (div, span, …)
  - Writing into the HTML output using **document.write()**.
    - Used for fast testing, we won't use it much
  - Writing into the browser console, using **console.log()**.
    - Interesting for debugging purposes

You can find some examples in the next slides

Javascript output: window.alert()

- You can use an alert box to display data:

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

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<script>
window.alert(5 + 6);
</script>

</body>
</html>
```

Javascript output: InnerHTML

To access an HTML element, JavaScript can use
the ***document.getElementById(id)*** method.

- The **id** attribute defines the HTML element.
- The **innerHTML** property defines the HTML content

This example uses an HTML paragraph, but it also works with a label, a div or a span. Can you try?

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

<h1>My First Web Page</h1>
<p>My First Paragraph</p>

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

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

</body>
</html>
```

Javascript output: document.write()

- For testing purposes, it is convenient to use ***document.write()***:

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

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<script>
document.write(5 + 6);
</script>

</body>
</html>
```

Javascript output: console.log()

- For debugging purposes, you can use the **console.log()** method to display data:

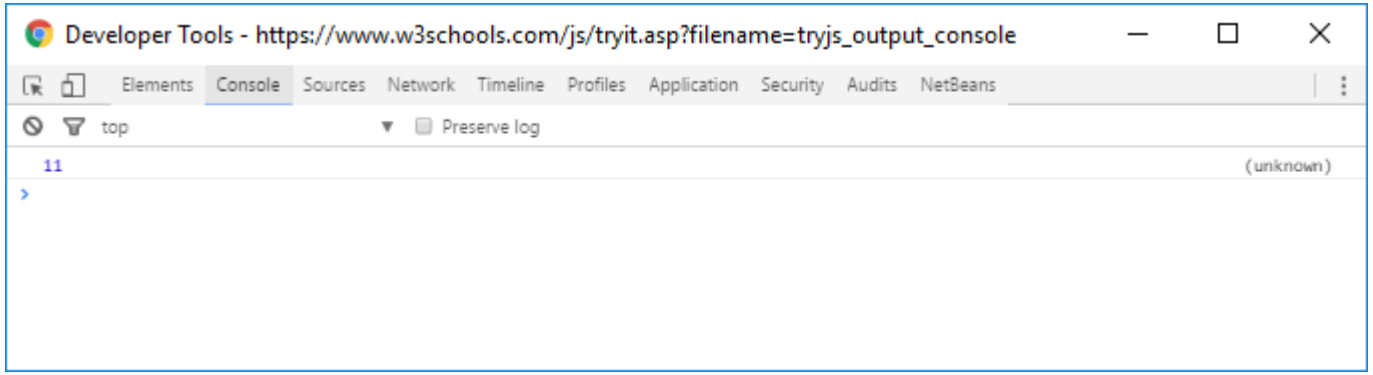
```
<html>
<body>

<h2>Activate debugging with F12</h2>

<p>Select "Console" in the debugger menu. Then click Run again.</p>

<script>
console.log(5 + 6);
</script>

</body>
</html>
```



Javascript input

- To get information from the user, Javascript offers:
  - **prompt**() method:
    - Opens a popup window with a textbox
  - **Confirm()** method:
    - Opens a popup window with buttons
  - Get the information from a textbox, using the **value** property

You can find some examples in the next slides

Javascript input: **prompt() method**

- Displays a dialog box that prompts the user for input:
- Syntax: `prompt(message, default)`
  - **message** is a string of text to display to the user. It can be omitted
  - **default** is a string containing the default value displayed in the text input field. It is also optional.

```
<html>
    <head>
        <title>TODO supply a title</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>
    <body>
        <div id="welcome"></div>
        <script>
            var customerName = prompt("Please enter your name", "<name goes here>");
            document.getElementById("welcome").innerHTML =
            "Hello " + customerName + "! How are you today?";</script>
    </body>
</html>
```

Javascript input: **confirm() method**

- Displays a dialog box with two buttons, OK and Cancel.
- It returns true if the user clicks “OK”, and false otherwise.
- Syntax: `confirm(message)`
  - **message** is an optional string to be displayed display in the dialog box

```
<html>
    <head>
        <title>TODO supply a title</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initi
    </head>
    <body>
        <div id="answerDiv"></div>
        <script>
            var answer = confirm("Are you ready?");
            document.getElementById("answerDiv").innerHTML =
            "The user answered " + answer;</script>
    </body>
</html>
```

Javascript input: Textbox value

- You can use the value property to get the contents of a text input field:
  - Add an input element to your HTML document:

```
<input type="text" placeholder="Type your name" id="nameInput">
```

  - Use the getElementById method to identify the input element and get its value:

```
var nameUser = document.getElementById("nameInput").value;
```

- To show the content of the textbox, you can capture the click event of a button:

```
<button type="button" onclick="getInputValue();"> Click for welcome</button>
```

  - When the button is clicked, the browser will execute the getInputValue() function, that should be declared in the Javascript code:

```
<script>
    function getInputValue(){
        var nameUser = document.getElementById("nameInput").value;
        alert("Welcome " + nameUser);
    }
</script>
```

You can find the complete code in the next slide

Javascript input: Textbox value example

```
<html>
    <head>
        <title>TODO supply a title</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>
    <body>
        <input type="text" placeholder="Type your name" id="nameInput">
        <button type="button" onclick="getInputValue();"> Click for welcome</button>
        <script>
            function getInputValue(){
                var nameUser = document.getElementById("nameInput").value;
                alert("Welcome " + nameUser);
            }
        </script>
    </body>
</html>
```

- Note that event handlers (onClick attribute) are not recommended. We will learn how to substitute them in the next section.
- You could also use any other input types (password, radio, checkbox,…) . Can you try?

Control flow

Control flow (or **flow of control**) is the order in which individual statements, instructions or function calls of an imperative program are executed or evaluated.

Javascript supports the following control flow statements:

- Choice statements:
  - if … else
  - switch
- Loop statements:
  - while
  - do
  - for
- Array.prototype methods: map(), forEach(), …

Choice statements: If … else

```
if (condition) {
    block of code to be executed if the condition is true
}
```

```
if (hour < 18) {
    greeting = "Good day";
}
```

```
if (condition) {
    block of code to be executed if the condition is true
} else {
    block of code to be executed if the condition is false
}
```

```
if (hour < 18) {
    greeting = "Good day";
} else {
    greeting = "Good evening";
}
```

```
if (condition1) {
    block of code to be executed if condition1 is true
} else if (condition2) {
    block of code to be executed if the condition1 is false and condition2 is true
} else {
    block of code to be executed if the condition1 is false and condition2 is false
}
```

```
if (time < 10) {
    greeting = "Good morning";
} else if (time < 20) {
    greeting = "Good day";
} else {
    greeting = "Good evening";
}
```

Choice statements: Switch

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

```
switch (new Date().getDay()) {
    case 4:
    case 5:
        text = "Soon it is Weekend";
        break;
    case 0:
    case 6:
        text = "It is Weekend";
        break;
    default:
        text = "Looking forward to the Weekend";
}
```

- The switch expression is evaluated once.
- The value of the expression is compared with the values of each case.
- If there is a match, the associated block of code is executed.
- A break can save a lot of execution time because it "ignores" the execution of all the rest of the code in the switch block.
- In this example case 4 and 5 share the same code block, and 0 and 6 share another code block

Loop statements: For

```
//For
let text="";
for(i=0;i<5;i++) {text += `The number is ${i}\n`;}
console.log(text);

//For..of
const iterable = [10, 20, 30];
for (let value of iterable) {
  value += 1;
  console.log(value); }

//For..in
let person = {fname:'John', lname: 'Doe', age:25,
              getFName:function(){console.log(this.fname)}}
let text2="";
for (let x in person) { text2 += person[x]+'\n'}
console.log(text2);
```

Loop statements: While

```
while (condition) {
    code block to be executed
}
```

```
do {
    code block to be executed
}
while (condition);
```

```
while (i < 10) {
    text += "The number is " + i;
    i++;
}
```

```
do {
    text += "The number is " + i;
    i++;
}
while (i < 10);
```

More Array.prototype methods

(https://medium.com/@mandeepkaur1/a-list-of-javascript-array-methods-145d09dd19a0)