CSC-318
Web Technology
(BSc CSIT, TU)

Ganesh Khatri
kh6ganesh@gmail.com

JavaScript Introduction

- HTML and CSS can be used to display information and present it in a specific way
- However, CSS and HTML have limitations: You cannot change the contents of the page after it has been drawn on the screen
- HTML is not very interactive, you can display a page but not control what happens when the user interacts with it

JavaScript

- HTML is a Markup Language this means it describes how data is structured
- When the HTML code is run, it is interpreted by the browser to generate an output and run in order, line by line top to bottom
- Javascript is a programming language. This means you as the developer has control over how the program is executed, it’s not usually executed in linear fashion

JavaScript

- Like CSS, Javascript code should be placed in its own file
- Javascript files have a .js extension
- To run the .js file in a HTML page you must reference the javascript file using a <script> HTML tag

JavaScript

- The script tag has an src attribute (like the <img> tag) that points to the javascript file:
- <script> tags should go inside the page's <head> tag:

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <script src="script.js"></script>
  </head>

  <body>
    <h1>Page heading</h1>
    <p>Page content</p>
  </body>
</html>
```

There has been some debate about whether to place the <script> tag in the head or body tag.

For modern browsers, the <head> tag is preferred

<script> Tag

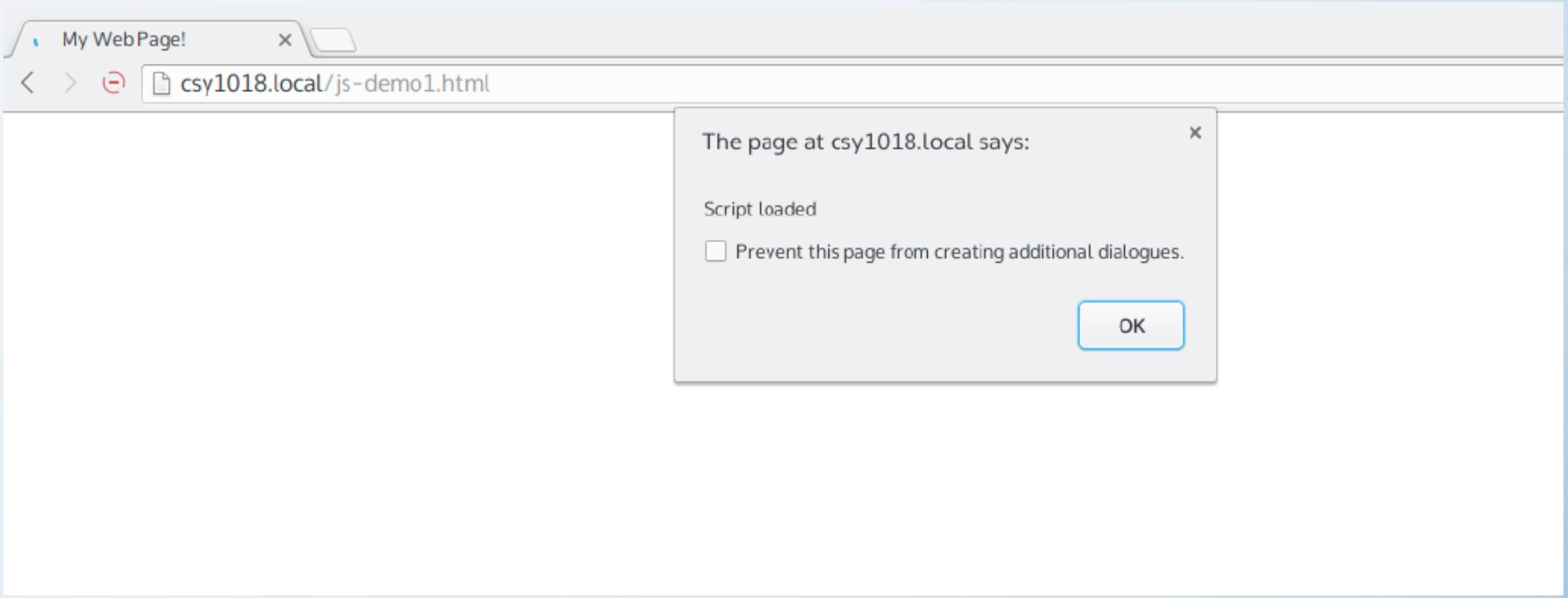
- The script tag does not need anything between <script> and </script> but both start and end tags are required
- <script src="file.js" /> **will not work in all browsers, you must use <script type="text/javascript" src="file.js"></script>**

JavaScript

- To check your script.js is loading correctly you can add some code to its
- Javascript includes the function alert
- The alert function lets you create a pop up alert box to display some text
- Using this code in script.js:

```
alert('Script loaded');
```

Script.js


My Web Page!
csy1018.local/js-demo1.html
The page at csy1018.local says:
Script loaded
Prevent this page from creating additional dialogues.
OK

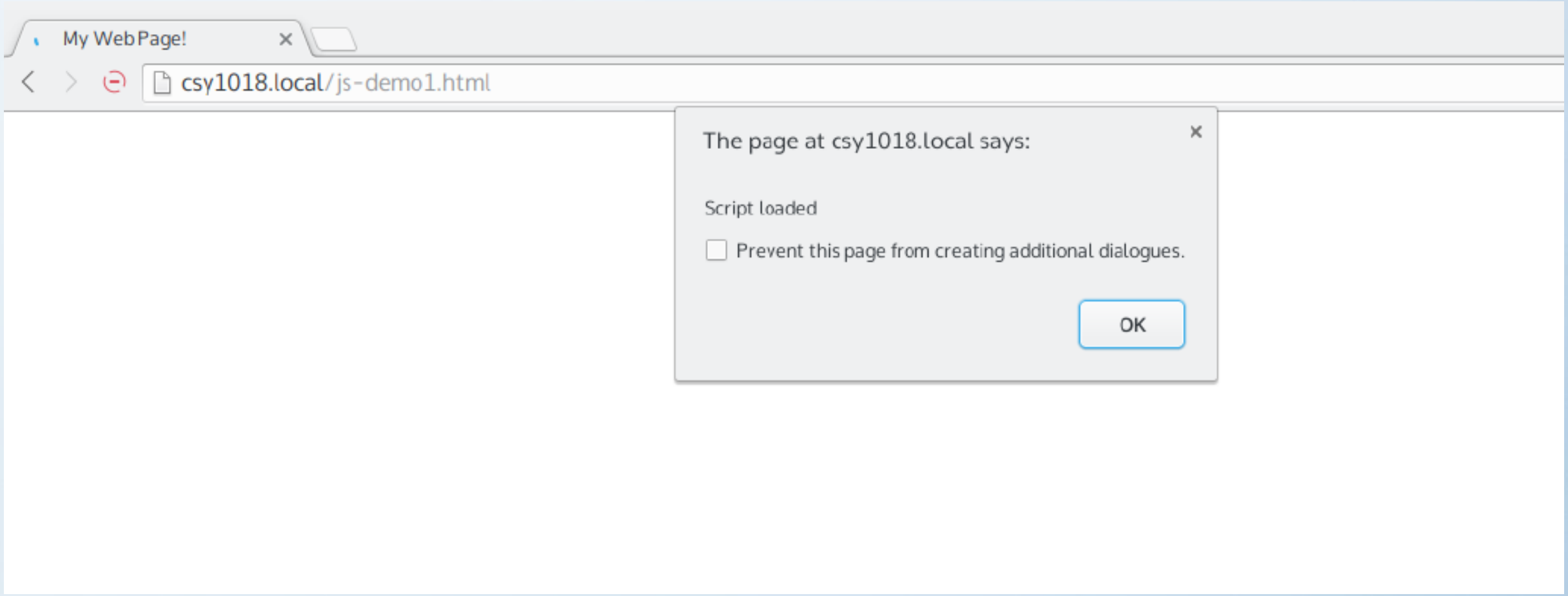
JavaScript Pop Up Boxes

- alert()
- confirm()
- prompt()

alert()

- Used to display the alert information

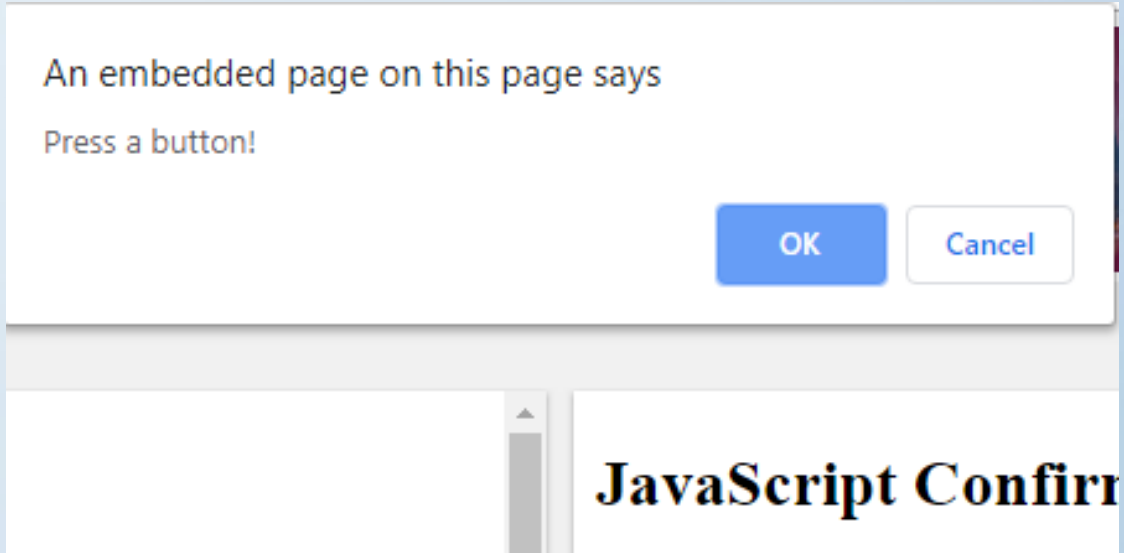
alert("I am an alert box!");



confirm()

- confirm box is often used if you want the user to verify or accept something
- When a confirm box pops up, the user will have to click either "OK" or "Cancel" to proceed
- If the user clicks "OK", the box returns true. If the user clicks "Cancel", the box returns false.

```
if (confirm("Press a button!")) {
  txt = "You pressed OK!";
} else {
  txt = "You pressed Cancel!";
}
```



prompt()

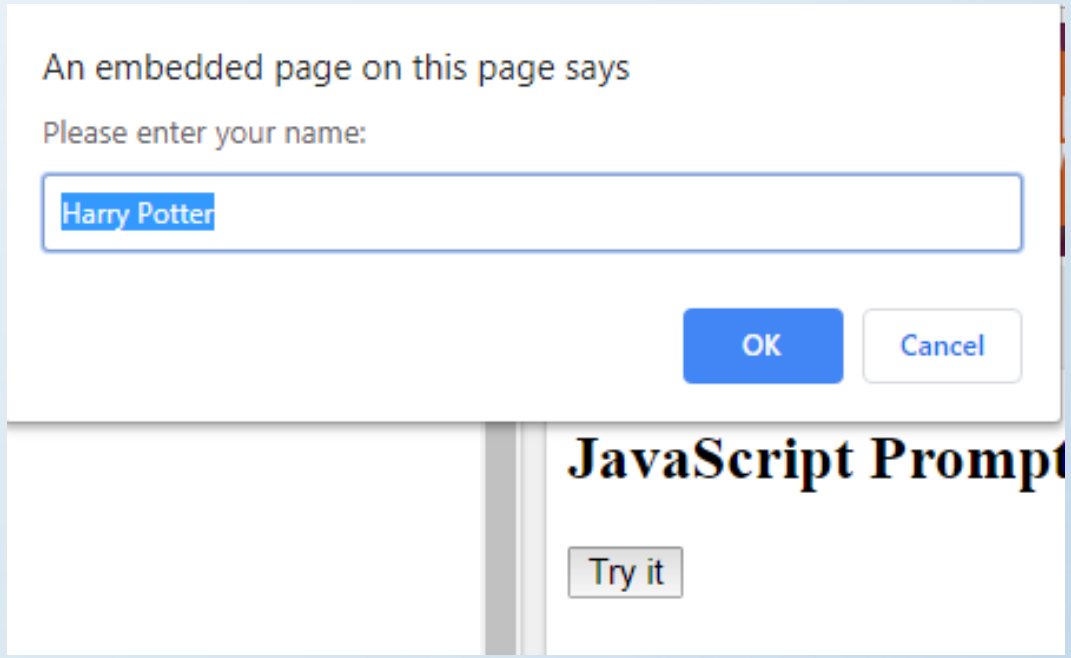
- A prompt box is often used if you want the user to input a value before entering a page
- When a prompt box pops up, the user will have to click either "OK" or "Cancel" to proceed after entering an input value
- If the user clicks "OK" the box returns the input value. If the user clicks "Cancel" the box returns null

```
var person = prompt("Please enter your name", "Harry Potter");

if (person == null || person == "") {
  txt = "User cancelled the prompt.";
} else {
  txt = "Hello " + person + "! How are you today?";
}
```

prompt()

- A prompt box is often used if you want the user to input a value before entering a page
- When a prompt box pops up, the user will have to click either "OK" or "Cancel" to proceed after entering an input value
- If the user clicks "OK" the box returns the input value. If the user clicks "Cancel" the box returns null

```
var person = prompt("Please enter your name", "Harry Potter");

if (person == null || person == "") {
  txt = "User cancelled the prompt.";
} else {
  txt = "Hello " + person + "! How are you today?";
}
```



Exercise 1

- Create a basic web page with a html, body and head tag. Inside the body tag, place a H1 tag and a P tag that contain text of your choice
- Add a script tag that references a file called script.js
- Create a file script.js and practice all types of alert boxes. You may change the text to say whatever you like by changing the text inside the quotes
- Open your html file in a browser and verify that script file is working

Exercise 1 alert()

```html
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <script src="script.js"></script>
  </head>

  <body>
    <h1>Page heading</h1>
    <p>Page content</p>
  </body>
</html>
```

```javascript
alert('Script loaded');
```

JavaScript

- You'll notice that when the alert popup appears that the contents of the page are not visible behind it
- Once you click 'OK' the contents of the page appear
- This is because the Javascript runs before the page has been drawn on the screen

JavaScript

- Javascript can be used to control HTML elements on the page
- Javascript can be used to:
  - Assign CSS to the element
  - Add or remove HTML attributes
  - Read the contents of form elements
  - Detect when an element is interacted with (moused over, clicked, typed into, etc)

JavaScript Variables

- Javascript allows you to give values labels
- A label is called a variable and can store a single value
- To declare a variable use the code

```
var variableName = variableValue;
```

- You can give the variable any name you like, this is chosen by you, not javascript
- The value of the variable is also chosen by you
- The only parts that are defined by the language are the:
  - var keyword – this tells javascript you are creating a variable
  - = sign – this tells javascript you are writing a value to the variable

JavaScript Variables

- There are two main types of variable
  - Numbers
  - Strings (text)
- To assign a number variable you can use

```
var numberVariable1 = 123;
var numberVariable2 = 123.45;
```

- To assign a string variable you must surround the string with quotes:

```
var stringVariable = 'Script loaded';
```

JavaScript

- Note that each statement must be ended with a semicolon
- The semicolon means “end of statement” and can be thought of like a full stop in an English sentence

Functions

- Reduce repeated code
- You can label a block of code using a function
- This will store the code for later use where it can be referenced and run
- This allows you to write code out of sequence

```
function scriptLoaded() {
        alert('Script loaded');
}
function addition() {
        var num1 = 5;
        var num2 = 6;

        var num3 = num1 + num2;

        alert(num3);
}
```

Functions

- Once a function has been defined it has to be called
- A function is called using the name followed by brackets
- Normally code gets run in the order it is written
- Functions allow you to run code in a different order
- To run the code in the addition function, it must be called using the code

```
addition();
```

Selecting elements with Javascript

- Javascript contains inbuilt functions for selecting HTML elements so you can change properties on the (css, attributes, etc)
- The simplest way is to give an element an ID in the HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <script src="script.js"></script>
  </head>

  <body>
    <h1 id="pageheading">Page heading</h1>
    <p>Page content</p>
  </body>
</html>
```

Selecting elements with Javascript

- Once an element on the page has an ID, you can use the javascript function document.getElementById() to select it and store the element in a variable

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <script src="script.js"></script>
  </head>

  <body>
    <h1 id="pageheading">
        Page heading
    </h1>
    <p>Page content</p>
  </body>
</html>
```

```
var element = document.getElementById('pageheading');
```

Selecting Elements

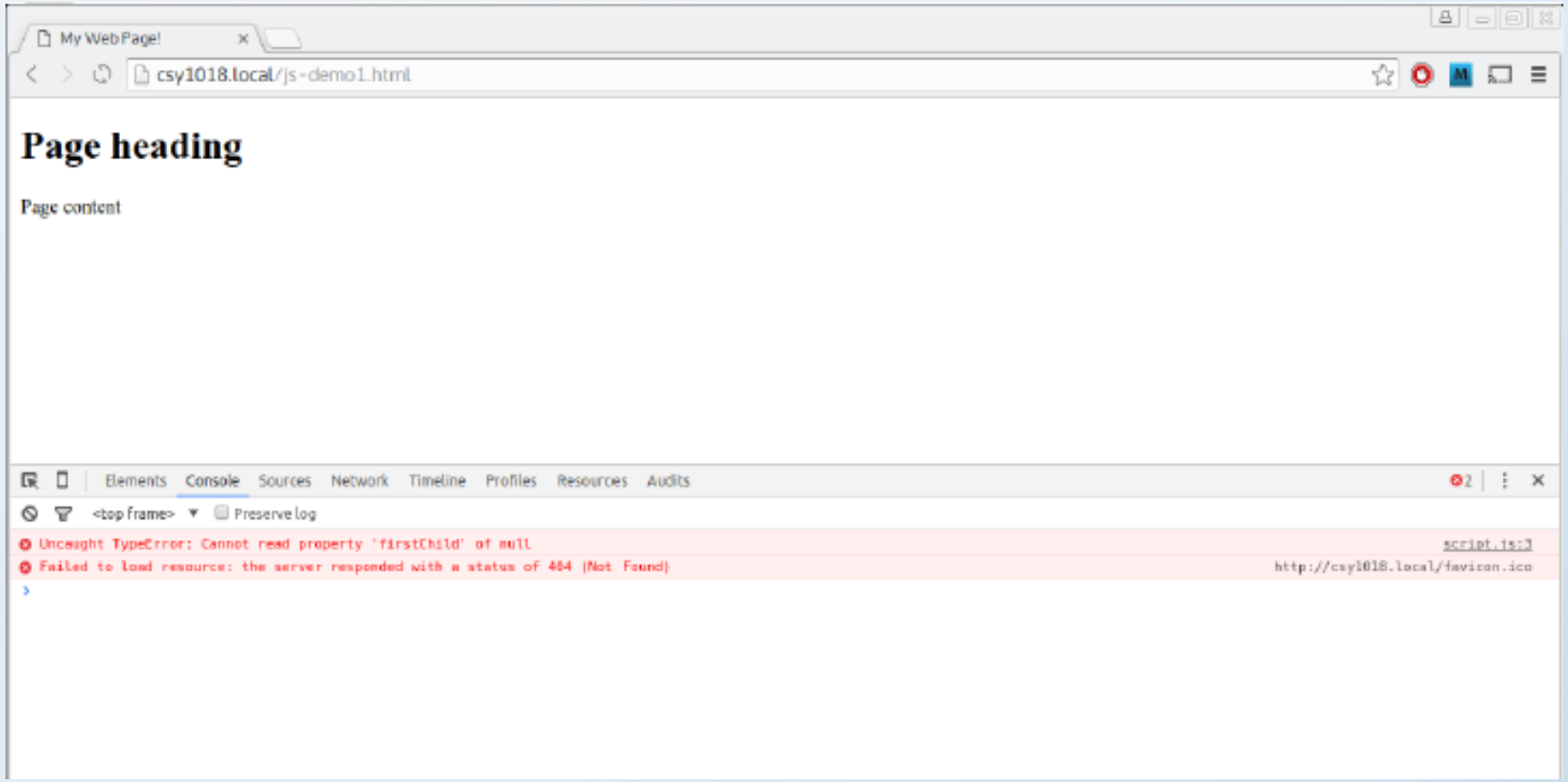
- Once you have an element you can make changes to it
- E.g. to update the content you can use:

```
var element = document.getElementById('pageheading');

element.firstChild.nodeValue = 'New Heading';
```

- Or element.innerHTML = ‘New Heading’

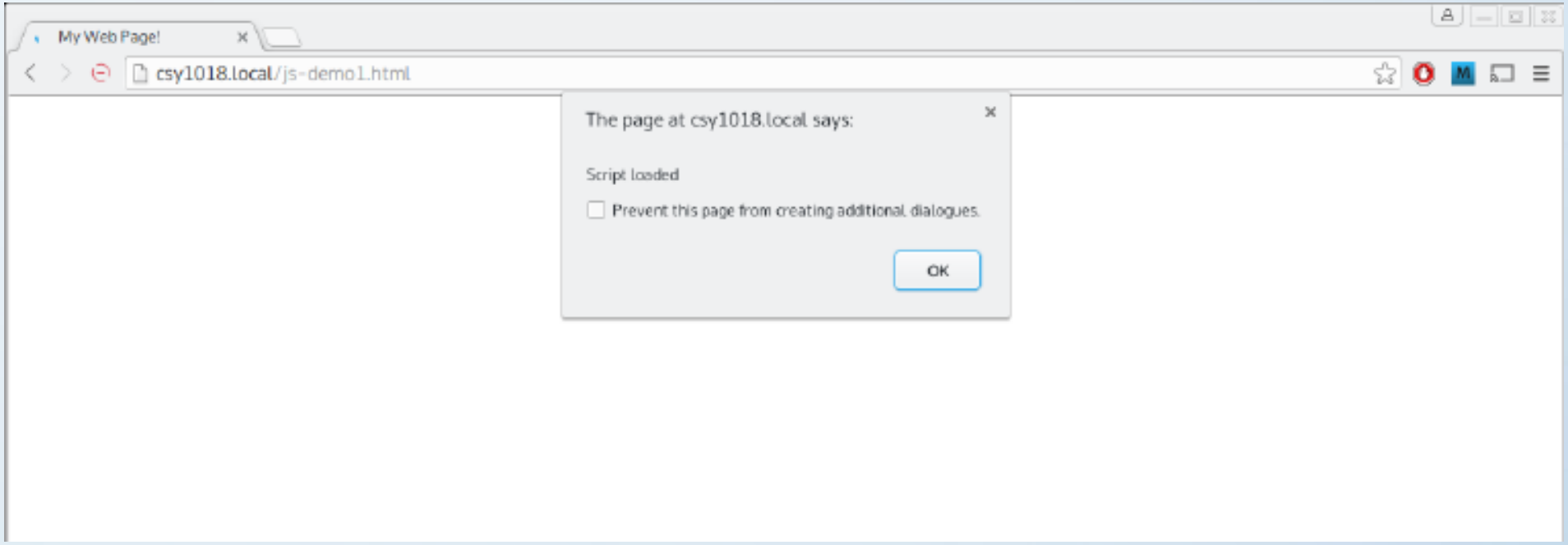
JavaScript

- Running this code won't quite have the desired effect
- Hint: Always keep the console open as it will tell you if there are any errors in your code!



JavaScript

- Remember the first alert box()
- The Javascript code is run before any elements exist on the page, which is why the code is failing



JavaScript

- Rather than having the code run before the page has loaded, it's possible to write a function that is run when the page loads
- This requires 2 steps:
  - 1) Move the code you want to run when the page loads into a function
  - 2) Inform the browser you want to run this function when the page loads

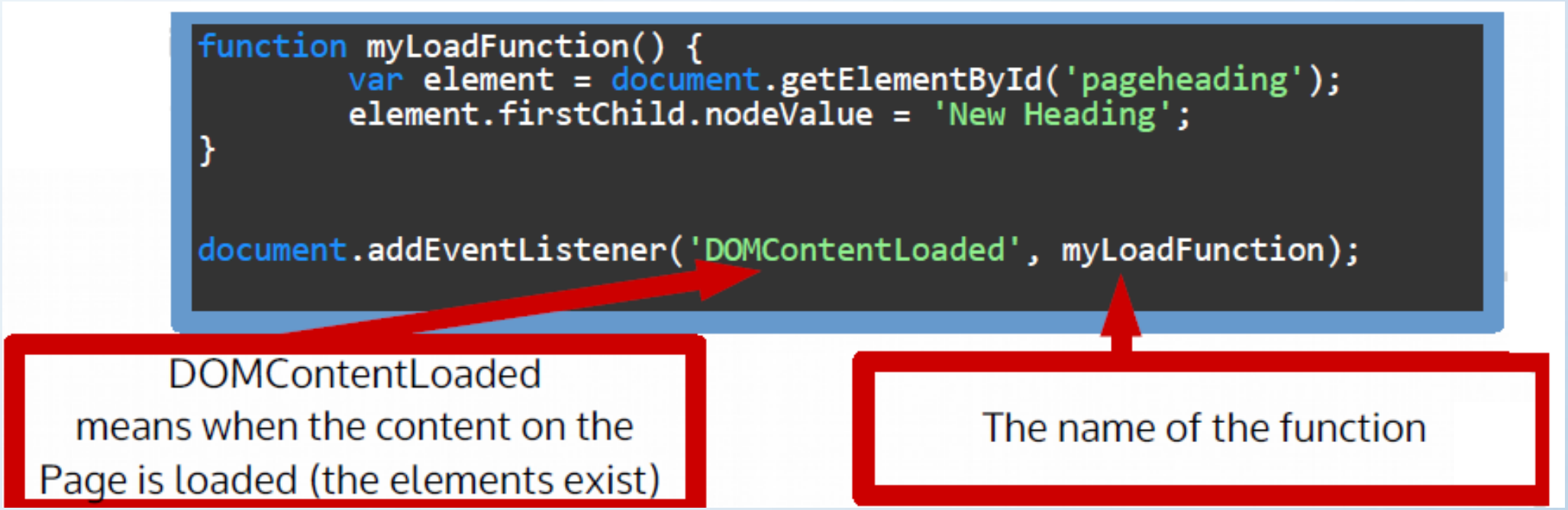
JavaScript

- 1) Move the code you want to run when the page loads into a function

```
function myLoadFunction() {
        var element = document.getElementById('pageheading');

        element.firstChild.nodeValue = 'New Heading';
}
```

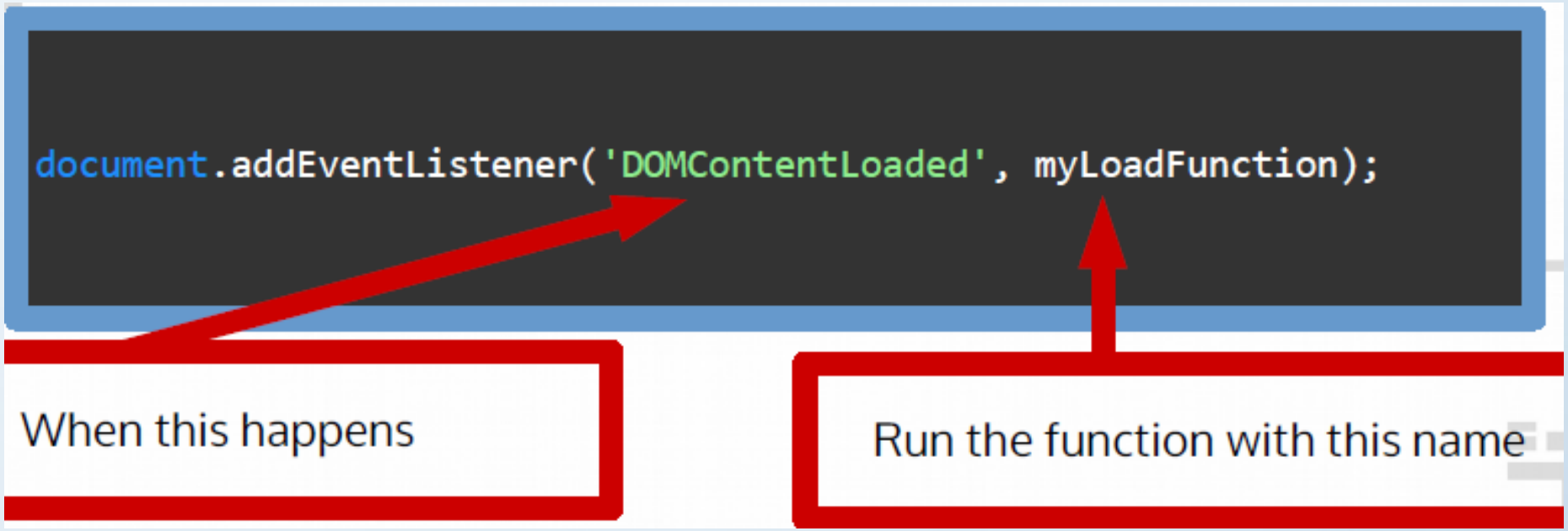
JavaScript

- 2) Inform the browser you want to run this function when the page loads
- This is done using the inbuilt function document.addEventListener() function



JavaScript

- addEventListener is a very useful function
- It allows you to run a function when a specific event occurs



Javascript Events

- click - when user clicks an element
- mouseenter - when cursor is on the element
- mouseleave - when cursor leaves the element
- Form Submit - when form is submitted
- change – when user selects an option from select box
- keyup - when user releases a key
- keydown – when user presses a key
- Events based on timers

Click

- There is also a 'click' event which occurs whenever the element is clicked on

```
document.addEventListener('click', myClickFunction);
```

- This will run `myClickFunction` whenever the document is clicked on (the document is the entire page)
- If you need to do add some functionality when user clicks on an element, then use element in place of document

JavaScript Display Possibilities

- JavaScript can "display" data in different ways:
  - Writing into an HTML element, using innerHTML
  - Writing into the HTML output using document.write()
  - Writing into an alert box, using window.alert() or just alert()
  - Writing into the browser console, using console.log()

Using innerHTML

- Writes content inside an element

```
<!DOCTYPE html>
<html>
<body>
    <p id="demo"></p>
    <script>
        document.getElementById("demo").innerHTML = "Hello";
    </script>
</body>
</html>
```

document.write()

- For testing purposes, it is convenient to use document.write()
  - Writes directly to the html page

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

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

</body>
</html>
```

Using alert or window.alert()

- Used to display popup information

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

Unsing console.log()

- For debugging purposes, you can use the console.log() method to display data
  - Displays data in console of browser

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

JavaScript Keywords

- Keywords are reserved words by JavaScript
- JavaScript statements often start with a keyword to identify the JavaScript action to be performed
- Here is a list of some of the keywords
  - break
  - continue
  - debugger
  - do …. while
  - for
  - function
  - if …. else
  - return
  - switch
  - var etc.

JavaScript Expressions

- An expression is a combination of values, variables, and operators, which computes to a value
- The computation is called an evaluation
- For example, 5 * 10 evaluates to 50
- Expressions can also contain variable values
  - eg. x * 10
- The values can be of various types, such as numbers and strings
- For example, "John" + " " + "Doe", evaluates to "John Doe"

JavaScript Comments

- Not all JavaScript statements are "executed"
- Code after double slashes // or between /* and */ is treated as a comment
- Comments are ignored, and will not be executed

```
var x = 5;   // I will be executed

// var x = 6;   I will NOT be executed
```