Web Programming

JavaScript 1

Prof. Josué Obregón

Department of Industrial Engineering- ITM

Seoul National University of Science and Technology

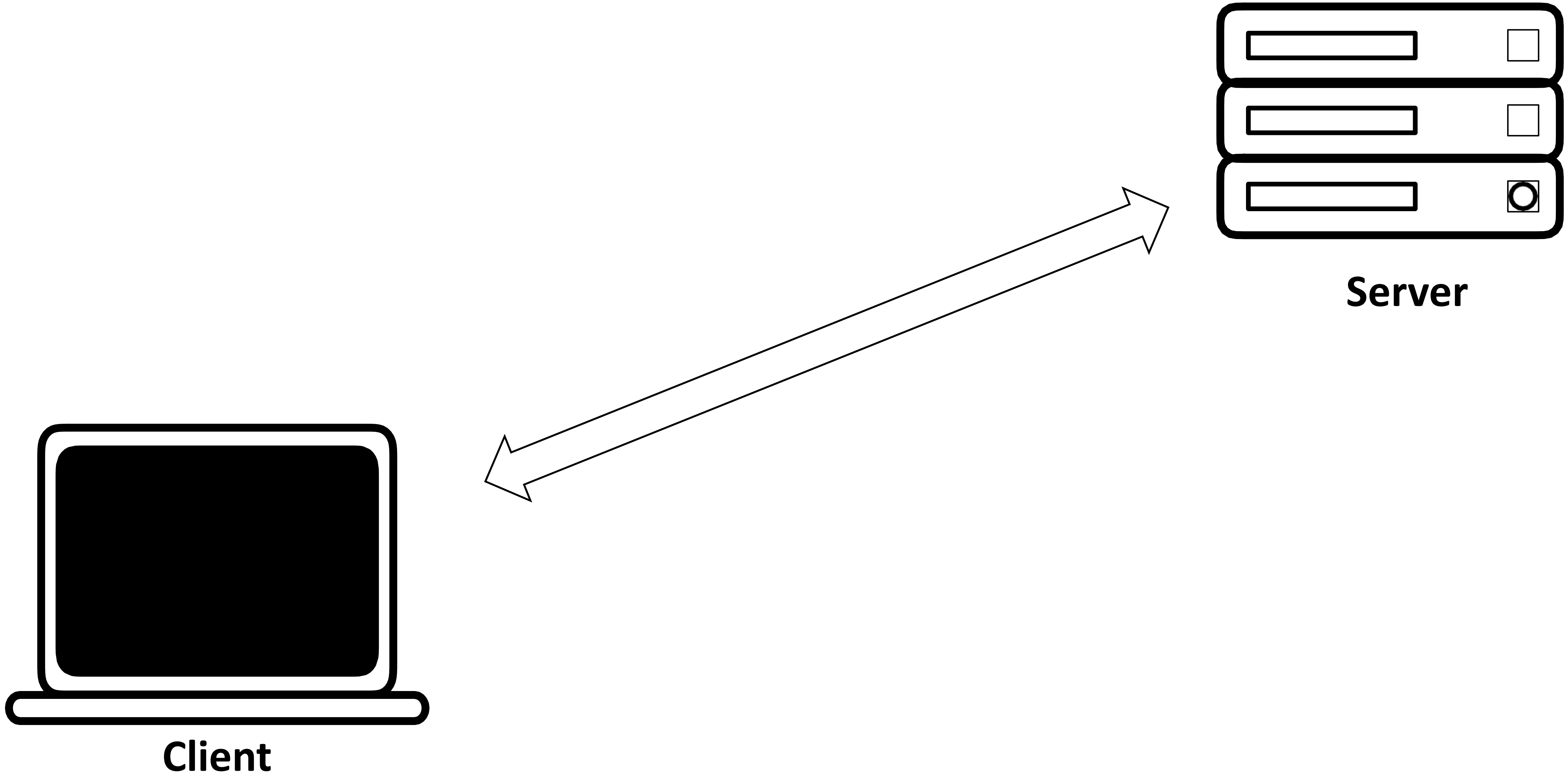
Objectives

- Identify and describe the basic JavaScript statements that are used for manipulating the DOM and handling events on web pages.
- Apply the acquired knowledge of JavaScript to construct dynamic web pages.

Agenda

- JavaScript basics
- Events
- Variables
- querySelector
- DOM Manipulation

JavaScript


Server
Client

Why use client-side programming?

Django already allows us to create dynamic web pages. Why also use client-side scripting?

- client-side scripting (JavaScript) benefits:
  - **usability**: can modify a page without having to post back to the server (faster UI)
  - **efficiency**: can make small, quick changes to page without waiting for server
  - **event-driven**: can respond to user actions like clicks and key presses

Why use client-side programming?

- server-side programming (Django) benefits:
  - **security**: has access to server's private data; client can't see source code
  - **compatibility**: not subject to browser compatibility issues
  - **power**: can write files, open connections to servers, connect to databases, ...

What is JavaScript?

- a lightweight programming language ("scripting language")
  - used to make web pages interactive
  - manipulate the DOM dynamically (ex: add elements or change styling)
  - **react to events** (ex: page load user click)
  - get information about a user's computer (ex: browser type)
  - perform calculations on user's computer (ex: form validation)

What is JavaScript?

- a web standard (but not supported identically by all browsers)
- NOT related to Java other than by name and some syntactic similarities

JavaScript vs Java

- interpreted, not compiled
- more relaxed syntax and rules
  - fewer and "looser" data types
  - variables don't need to be declared
  - errors often silent (few exceptions)
- key construct is the function rather than the class
  - "first-class" functions are used in many situations
- contained within a web page and integrates with its HTML/CSS content

Linking to a JavaScript file: `script`

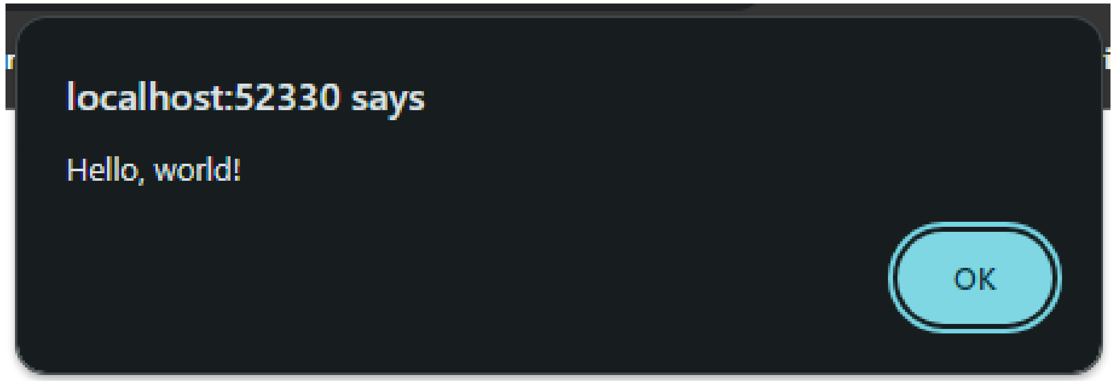
```
<script>
  alert('Hello, world!');
</script>                                           HTML
```

```
<script src="filename" type="text/javascript"></script>
                                                    HTML
```

- script tag should be placed in HTML page's head
- script code is stored in a separate .js file
- JS code can be placed directly in the HTML file's body or head (like CSS)
  - but this is bad style (should separate content, presentation, and behavior

A JavaScript statement: `alert`

```
    <script>
        alert('Hello, world!');
    </script>
```

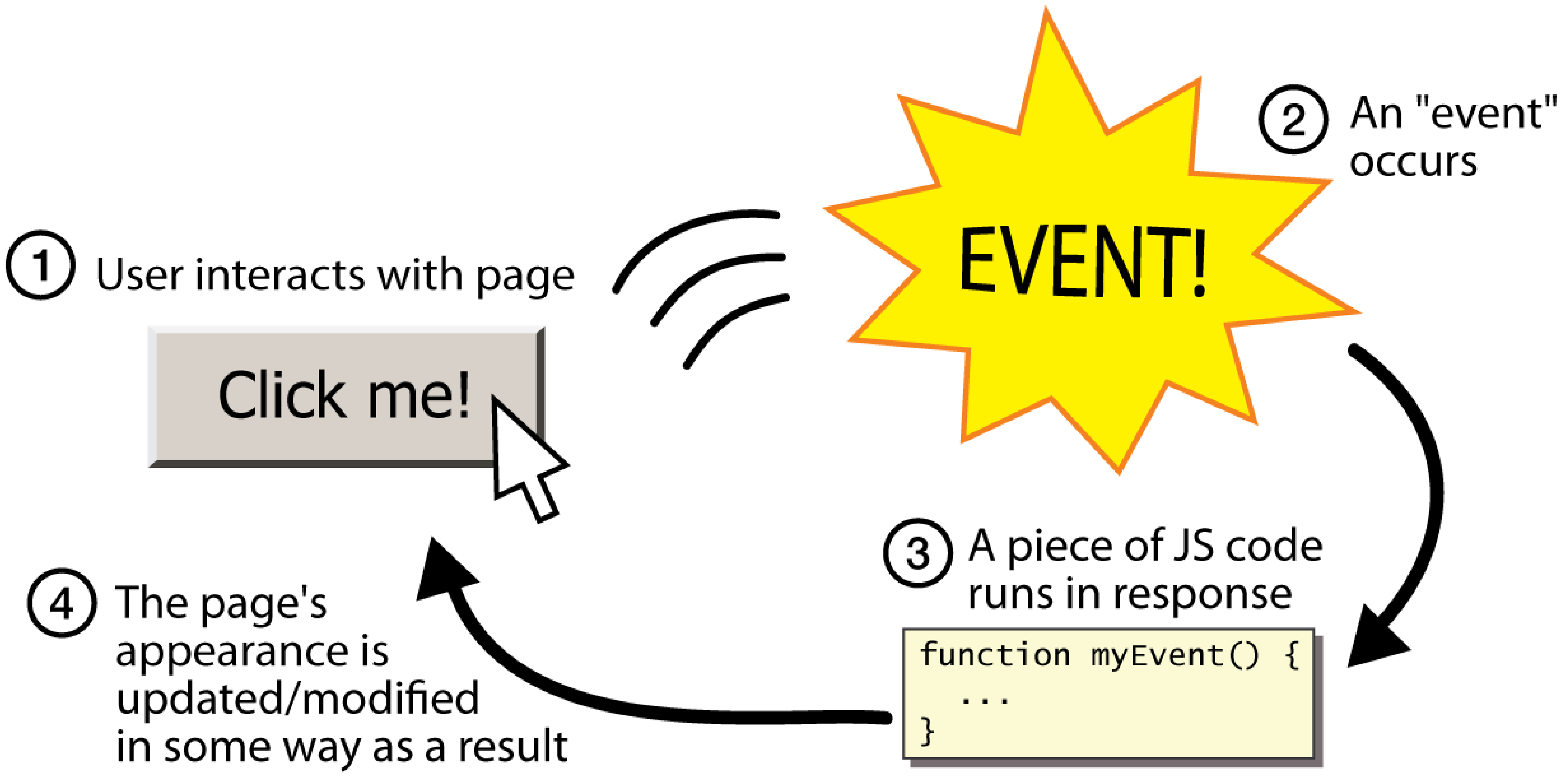

localhost:52330 says

Hello, world!

- a JS command that pops up a dialog box with a message

```
alert("message"); // message
confirm("message"); // returns true or false
prompt("message"); // returns user input string
                                                        JS
```

Event-driven programming


1 User interacts with page
Click me!
EVENT!
2 An "event" occurs
3 A piece of JS code runs in response
4 The page's appearance is updated/modified in some way as a result

```
function myEvent() {
  ...
}
```

Event-driven programming

- you are used to programs start with a main method or directly execute statements
- JavaScript programs instead wait for user actions called *events* and respond to them
- event-driven programming: writing programs driven by user events
- Let's write a page with a clickable button that pops up a "Hello, World" window...

JavaScript functions

```
function name() {
    statement ;
    statement ;
    ...
    statement ;
}                                                        JS
```

```
function hello() {
   alert('Hello, world!');
}                                                        JS
```

- the above could be the contents of example.js linked to our HTML page
- statements placed into functions can be evaluated in response to user events

Buttons and Events

```
<button onclick="hello()">Click Here</button>                    HTML
```

- button's text appears inside tag; can also contain images
- To make a responsive button or other UI control:
  - choose the control (e.g. button) and event (e.g. onclick) of interest
  - write a JavaScript function to run when the event occurs
  - attach the function to the event on the control

Variables

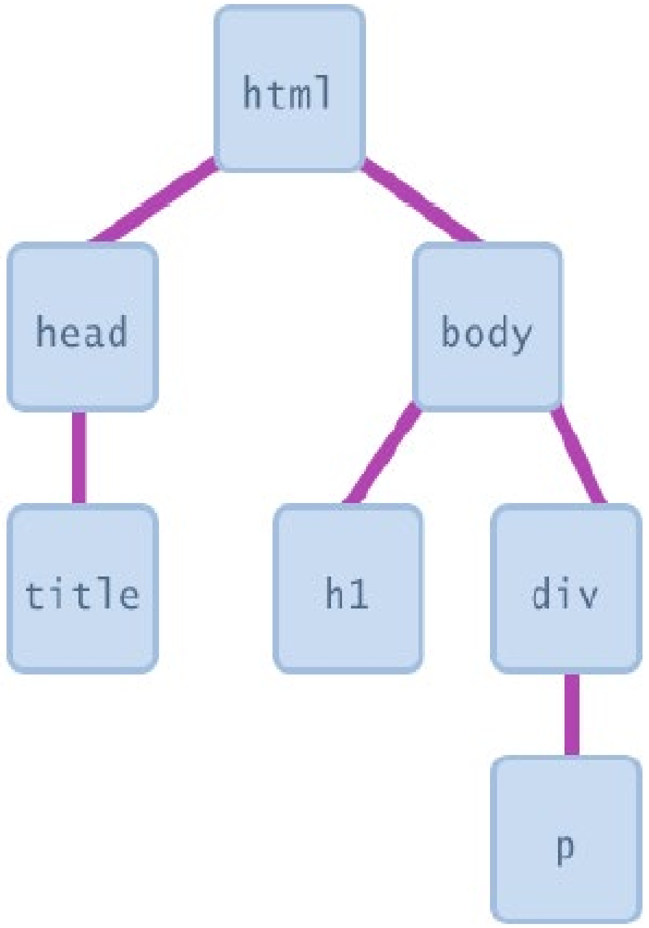
```
var name = expression; // define a variable globally
let name = expression; // define a variable with limited scope
const name = expression; // define a constant (cannot change)     JS
```

```
var age = 20;
let counter = 1;
const PI = 3.14;                                          JS
```

- variables are declared with the `let` keyword (case sensitive)
- types are not specified, but JS does have types ("loosely typed")
  - `Number, Boolean, String, Array, Object, Function, Null, Undefined`
  - can find out a variable's type by calling `typeof`

Document Object Model (DOM)

- most JS code manipulates elements on an HTML page
- we can examine elements' state
  - e.g. see whether a box is checked
- we can change state
  - e.g. insert some new text into a div
- we can change styles
  - e.g. make a paragraph red



Accessing elements: `document.querySelector`

```
<h1>Hello!</h1>
<button onclick="hello()">Click Here</button>                HTML
```

```
function hello() {
   document.querySelector('h1').innerHTML ='Goodbye!'
}
                                                             JS
```

- document.querySelector returns the DOM object for an element with a given CSS selector (“#” for id, “.” for class, or only using the element tag)
- can change the text inside most elements by setting the innerHTML property
- can change the text in form controls by setting the value property

if/else statement (same as Java)

```
if (condition) {
        statements;
} else if (condition) {
        statements;
} else {
        statements;
}
                                                                    JS
```

- identical structure to Java's if/else statement

Logical operators

- > < >= <= && || ! == != === !==
- most logical operators automatically convert types:
  - 5 < "7" is true
  - 42 == 42.0 is true
  - "5.0" == 5 is true //equality operator attempt to convert types
- === and !== are strict equality tests; checks both type and value
  - "5.0" === 5 is false

Accessing elements: `document.addEventListener`

```
document.addEventListener('DOMContentLoaded', function(){
                //some code here
          })
                                                                                      JS
```

- The addEventListener() method attaches an event handler to a document
- his function takes in two arguments:
  - An event to listen for (eg: 'click’, ‘DOMContentLoaded’)
  - A function to run when the event is detected (eg: hello from above or anonymous function)

https://www.w3schools.com/jsref/dom_obj_event.asp

Changing element style: `element.style`

Attribute	**Property or style object**
color	color
padding	padding
background-color	backgroundColor
border-top-width	borderTopWidth
Font size	fontSize
Font famiy	fontFamily

Using data attributes: `data-attribute`

```html
<button data-color="red">Red</button>                         HTML
```

```js
const button = document.querySelector("button");

button.dataset.color; // 'red’                                   JS
```

- Allow us to store extra information on standard HTML elements
- Any attribute on any element whose attribute name starts with `data-` is a data attribute
- In JS To get a data attribute through the dataset object, get the property by the part of the attribute name after data-

Arrays

```
let name = []; // empty array
let name = [value, value, ..., value]; // pre-filled
name[index] = value; // store element
                                                    JS
```

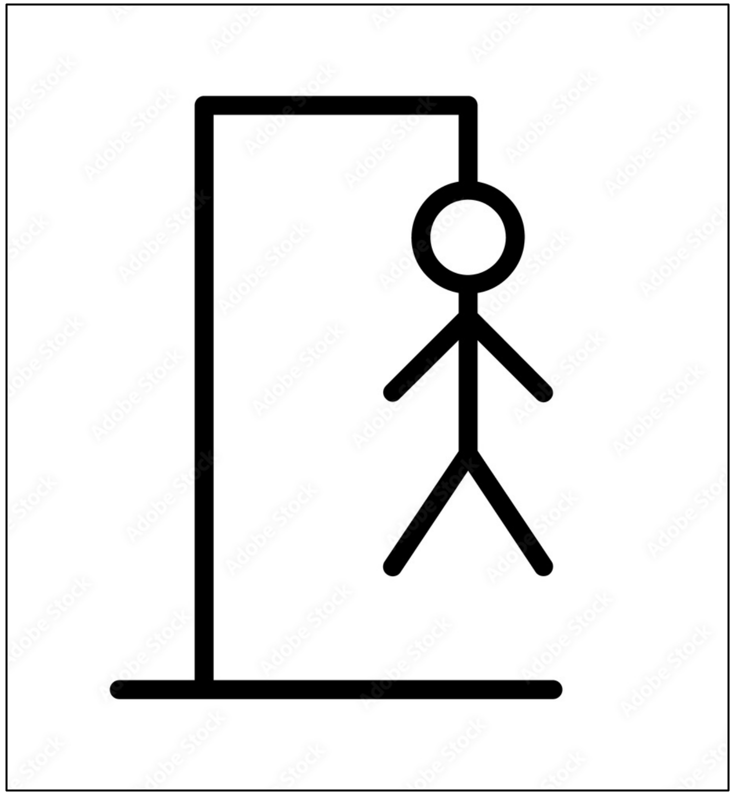
```
var ducks = ["Huey", "Dewey", "Louie"];
var stooges = []; // stooges.length is 0
stooges[0] = "Larry"; // stooges.length is 1
stooges[1] = "Moe"; // stooges.length is 2
stooges[4] = "Curly"; // stooges.length is 5
stooges[4] = "Shemp"; // stooges.length is 5
                                                    JS
```

Arrays functions: `array.forEach`

```
document.querySelectorAll('button').forEach(function(button){
        button.style.color = button.dataset.color;
})                                                             JS
```

- Calls a function for each element in an array
- The `forEach()` method calls a function for each element in an array.
- The `forEach()` method is not executed for empty elements.

Lab Assignment # 3 – JavaScript



GitHub Classroom link

https://classroom.github.com/a/5qydsLDj

Due date: Due May 12, 2024, 23:59 UTC

Web Programming