# Lab 5a: JavaScript intro

**IDE: You need to work with a local IDE like VScode for this lab.**

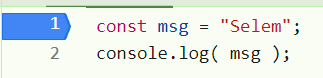
|  |
| --- |
| Summary of JavaScript google style guide (see annex at the end of this document for more details):   1. File names must be all lowercase and may include underscores (\_) or dashes (-) 2. Braces are required for all control structures (i.e. if, else, for, do, while, as well as any others), even if the body contains only a single statement. The first statement of a non-empty block must begin on its own line. 3. For nonempty blocks and block-like constructs:  * No line break before the opening brace. * Line break after the opening brace. * Line break before the closing brace.   Line break after the closing brace if that brace terminates a statement or the body of a function.   1. Each time a new block or block-like construct is opened, the indent increases by two spaces.   One statement per line. |

# Instructions:

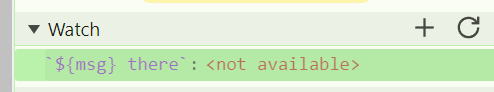
1. Download the provided starter code from Moodle.
2. Inspect the code that is provided. Notice the data that is in the data.js file.

In the following, never modify the data.js file.

1. Open the page with live server. Click on ctrl+shift+i to open the developer tools of chrome.
2. On the browser developer tools, open the “Sources” tab. Select the “lab05a.js” file then add a breakpoint on the first line and refresh the page. The debugger stops at the selected breakpoint.



1. In the debugger panel, expand the “Watch” element. Add two new expression in the watch list: document, and `${msg} there`.



Refresh the page to check the watched expressions and move to the next instructions by using the right arrow of the debugger.

A screenshot of a computer

Description automatically generated

1. In the file lab05a.js, write code to create an array of objects by parsing the given data using JSON.parse(). Check that your data is correctly parsed by using the debugger or a console.log().
2. Create a constructor function (***Customer***) to create new objects with the same structure. Create two new objects using this constructor function. Add these two new objects to the array containing the objects loaded from the data file.

*Note: Arrow functions cannot be used as constructors*

1. Add an output function property that outputs a string representation for objects created using the Customer constructor.
2. Write a function that returns an array containing all the customers from a given city. Write some test code using console.log()
3. Add the same output function to the objects that were parsed from the data file.

# ANNEX: JavaScript Simplified Google Style Guide

### File names

File names must be all lowercase and may include underscores (\_) or dashes (-), but no additional punctuation. Follow the convention that your project uses. Filenames’ extension must be .js

### Braces are used for all control structures

Braces are required for all control structures (i.e. if, else, for, do, while, as well as any others), even if the body contains only a single statement. The first statement of a non-empty block must begin on its own line.

**Disallowed:**

if (someVeryLongCondition())

doSomething();

for (let i = 0; i < foo.length; i++) bar(foo[i]);

**Exception:** A simple if statement that can fit entirely on a single line with no wrapping (and that doesn’t have an else) may be kept on a single line with no braces when it improves readability.

if (shortCondition()) foo();

### Nonempty blocks:

For nonempty blocks and block-like constructs:

* No line break before the opening brace.
* Line break after the opening brace.
* Line break before the closing brace.
* Line break after the closing brace if that brace terminates a statement or the body of a function. Specifically, there is no line break after the brace if it is followed by else, catch, while, or a comma, semicolon, or right-parenthesis.

**Example:**

class InnerClass {

constructor() {}

/\*\* @param {number} foo \*/

method(foo) {

if (condition(foo)) {

try {

// Note: this might fail.

something();

} catch (err) {

recover();

}

}

}

}

### Block indentation: +2 spaces

Each time a new block or block-like construct is opened, the indent increases by two spaces. When the block ends, the indent returns to the previous indent level. The indent level applies to both code and comments throughout the block.

### One statement per line

Each statement is followed by a line-break.

#### Reference:

JavaScript Google style guide: <https://google.github.io/styleguide/jsguide.html>