(CC & IoT) Nokia lab 3

Javascript introduction

|  |  |  |  |
| --- | --- | --- | --- |
|  | | |  |
| Author | Cosmin Ivan, Daniel Iercan, Cristian Cosariu |
| Owner | Nokia Timisoara, Politehnica University of Timisoara |
| Organization | Nokia Romania |
| Approver | Florin Ciocan |
| Document ID |  |
| Document location | public |
|  | |  |
|  | | |  |

Contents

[1 New project in GitLab 3](#_Toc463967257)

[2 Hello World in JavaScript 4](#_Toc463967258)

[3 Web browser debuging toools 6](#_Toc463967259)

[4 JavaScript objects 7](#_Toc463967260)

[4.1 Object Literal definition: 7](#_Toc463967261)

[4.2 JavaScript key word new 7](#_Toc463967262)

[4.3 JavaScript constructor 8](#_Toc463967263)

[5 JavaScript Scopes 9](#_Toc463967264)

[6 JQuery 10](#_Toc463967265)

[6.1 Including JQuery (to do) 10](#_Toc463967266)

[7 JavaScript events 11](#_Toc463967267)

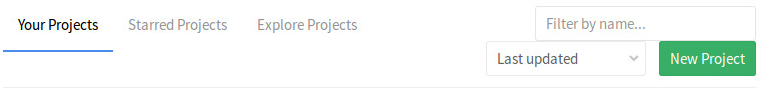
[8 Exercise 12](#_Toc463967268)

[8.1 My first JavaScript objects (to do) 12](#_Toc463967269)

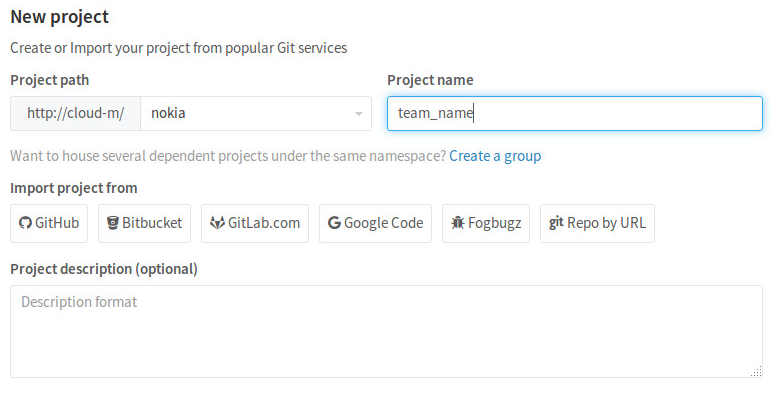
[8.2 First JavaScript application (to do) 12](#_Toc463967270)

# New project in GitLab

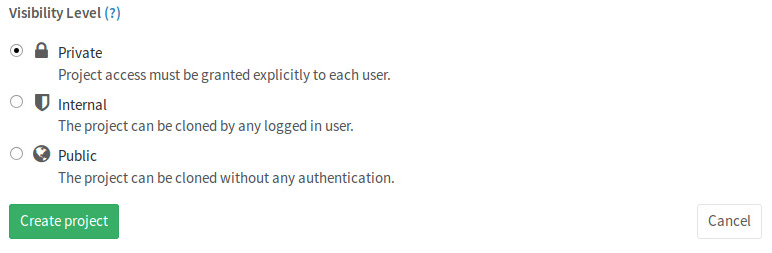
1. Open gitLab in a browser: <http://10.11.8.14>
2. Click on the button to create a new project



1. Add your **teams** name as the project name



1. Set the project visibility to private and click create project



# Hello World in JavaScript

1. Connect to your virtual machine using MobaXterm (use the ip address, username and password from the paper your received). Each team has its own user. More info can be found in lab 1.
2. Generate a new ssh key and add it to gitLab. (see lab 2 for details). Notice: you will have to generate a new key.
3. Go to your working directory. (notice in case it does not exist create it using **mkdir ~/ws**):

**cd ~/ws**

1. Pull the git project that you created from the server. Each team has its own project:

**git clone [url can be found in the project you have created]**

1. Pull the demo git project from the server.

**git clone git@10.11.8.14:demo-js/cloud\_demo.git**

1. You should now have two projects in the ws folder (uising **ls -al** you should see this folders).
2. Copy the required files from the demo project to your team’s project

**cp -R cloud\_demo/\* [project name]**

1. Open atom and add the project folder (the folder where you copied the files)

**atom**

1. Open the file index.html and add the following code:

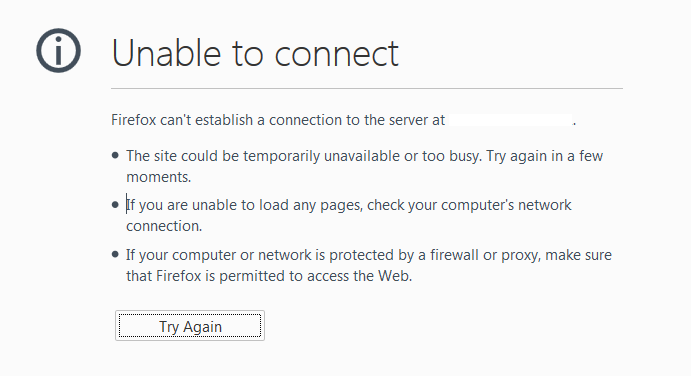


1. Now let’s test it. Open a browser on your machine and access the url:

http://[10.11.8.xxx]:8081/index.html

Replace **[10.6.99.xxx]** with the ip you have received on the paper

1. You will receive the following error



1. Why? In order to access a remote web page, you need to start a web server on your virtual machine. To start it open a new connection to your virtual machine and run the following commands:

**cd ~/ws/[project name] /**

**nodejs server.js**

You will receive a message like this that means that your serve is started and is listening to your localhost on the port 8081:

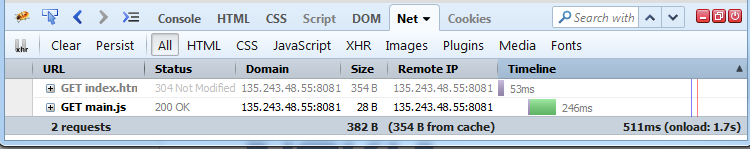
****

1. Now try to reload the page. You will see an alert with the message Hello World.

Info: If you received an error please check if you changed the directory to ~ws/[project\_name] before starting the server.

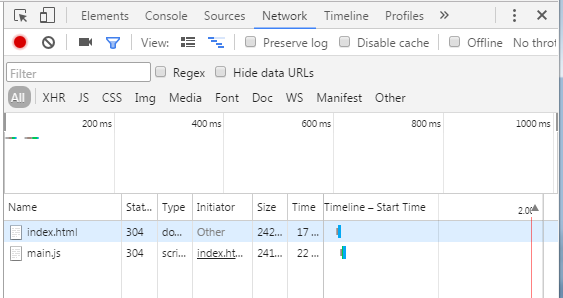
# Web browser debuging toools

Firebug for Firefox:



<http://getfirebug.com/>

Chrome developer tools

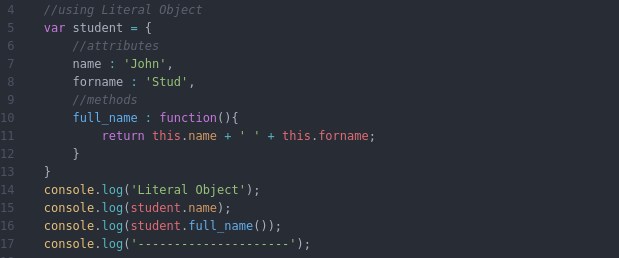


Hot key: **Ctrl+Shift+I or F12**

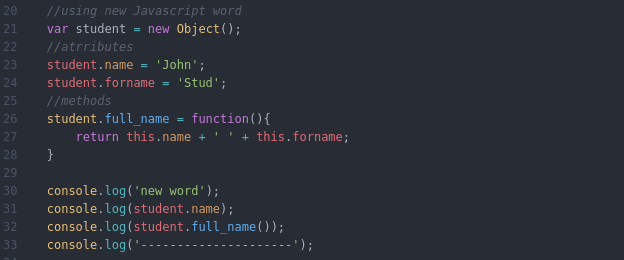
# JavaScript objects

There are three methods to define an object in JavaScript: Object Literal, JavaScript Key Word new, using a JavaScript constructor:

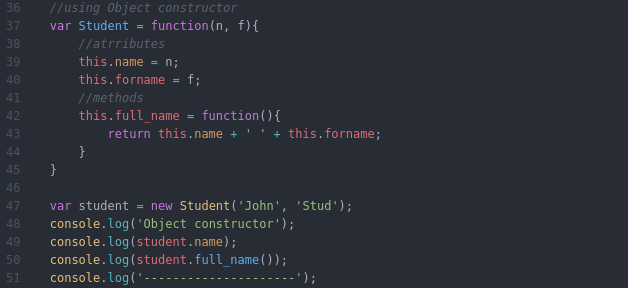
## Object Literal definition:



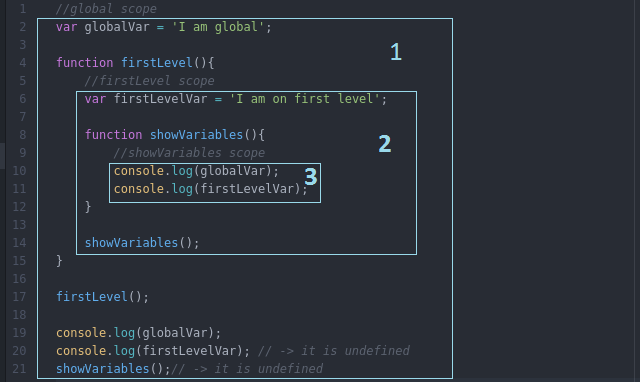
## JavaScript key word new



## JavaScript constructor



# JavaScript Scopes



1. The global scope of the application
2. The internal scope of the function firstLevel. All variables and function from global are available inside
3. The internal scope of the function ShowVariables. Because this function is declared inside firstLevelVar all variables and functions will be available in this scope including the ones defined in an upper scope.

In the example above line 20 and 21 will trigger an error because firstLevelVar and showVariables are present only in the firstLevel scope (the box with number 2)

# JQuery

Jquery is a library to make your life easy. By using css selectors you will be able to manipulate the html elements present in the web page.

## Including JQuery (to do)

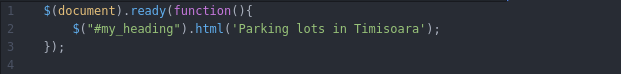
* 1. In index.html file just add the following line of code:



or

<script src=”https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js”></script>

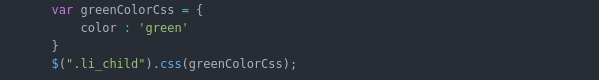
* 1. Now open the file core.js and add the following code:



What happened? First in order to manipulate elements in a web page we must wait for them to be present in the web page. $(document.ready()) will do this trick. Once the page has loaded we want to add a functionality to change the header in the page. For this we use a anonymous function (in our example it is the function(){} ). Using a css selector “#my\_heading” we change the html content of the element with the id my\_heading from the webpage.

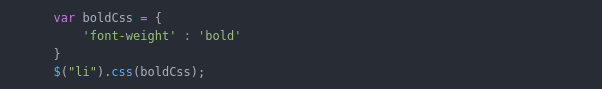


* 1. Include the file in index.html and reload the page
  2. Now let’s try to change the text colour for all elements with the class “.li\_child”



Notice that the last li element has still the color black. This is because it does not have the li\_child class.

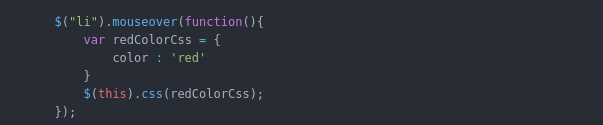
* 1. It is also possible to change the style for all the elements of a type in a web page. Add the following code:



Notice that all li elements in the page have the bold style.

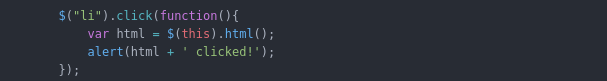
# JavaScript events

1. The next step is to interact with the user. In order to do this we can use the JavaScript events.
2. Add this code in the application. Don’t forget to add it in the $(document).ready function



Notice that when you put the mouse over a li element the color will change to red. $(this) is used to get the element for witch the event is triggered.

1. Let’s add also a click event:



When a li element is clicked an alert will open and will notice us which element has been clicked.

# Exercise

## My first JavaScript objects (to do)

Using the JavaScript constructor method create two objects: one for the parking lot and one for the parking spot. After you finished the task push the changes to your project. (instructions can be found in lab 2 on how to push a commit).

Parking lot: idParking, name, totalSpots, latitude, parking longitude, city, hasFreePlaces, updateParkingInfo

Parking spot: idSpot, name, xPosition, yPosition, isFree

## First JavaScript application (to do)

1. Create one instance for the parking lot and two instances for the parking spots
2. Using javascript (JQuery) create a div (<div></div>) using the parking lot object you created (the id of the div should be the id of the parking lot).
3. Inside the div add a header (<h1></h1>) with the name of the lot, two divs for two parking spots (id of the divs should be the id of the objects you created) and one div to show the totalSpots
4. The divs with parking spots should have a dedicated class attribute (parking\_spot) and inside it should contain the parking spot name and show a message if it is free or not
5. When we click a parking spot it should change its value from free to occupied or the reverse.