**Lab 2**

**Learning outcomes**

Become more familiar with JavaScript fundamentals.

Required reading: [Lab 2 notes](https://docs.google.com/document/d/13K3QlmmNco78kAfTOJ1sLuH5PFlg5J8lGZLotEGTjic/edit)

**Important note**

Directly after each step you need to commit your work with a message

Examples

git commit -am “Lab 2-1 Intro done”

git commit -am “Lab 2-4 Send back done”

(You can have more messages if you want but you need at least these)

Reminder: the directory structure is the root directory of your git repo contains a folder for each lab: Lab1, Lab2, Lab3, etc.

## **1 Intro**

This lab will use codewars - one of the several sites used for becoming a master coder and for assessing candidates for job interviews.

As you do these, do as much by yourself as you can (the point is to get familiar with JavaScript). If you get stuck, two great resources are:

[MDN](https://developer.mozilla.org/en-US/docs/Web/JavaScript)

and

[stackoverflow](http://stackoverflow.com/)

Instructions for codewars solutions:

1. Add your function to the Lab2.js

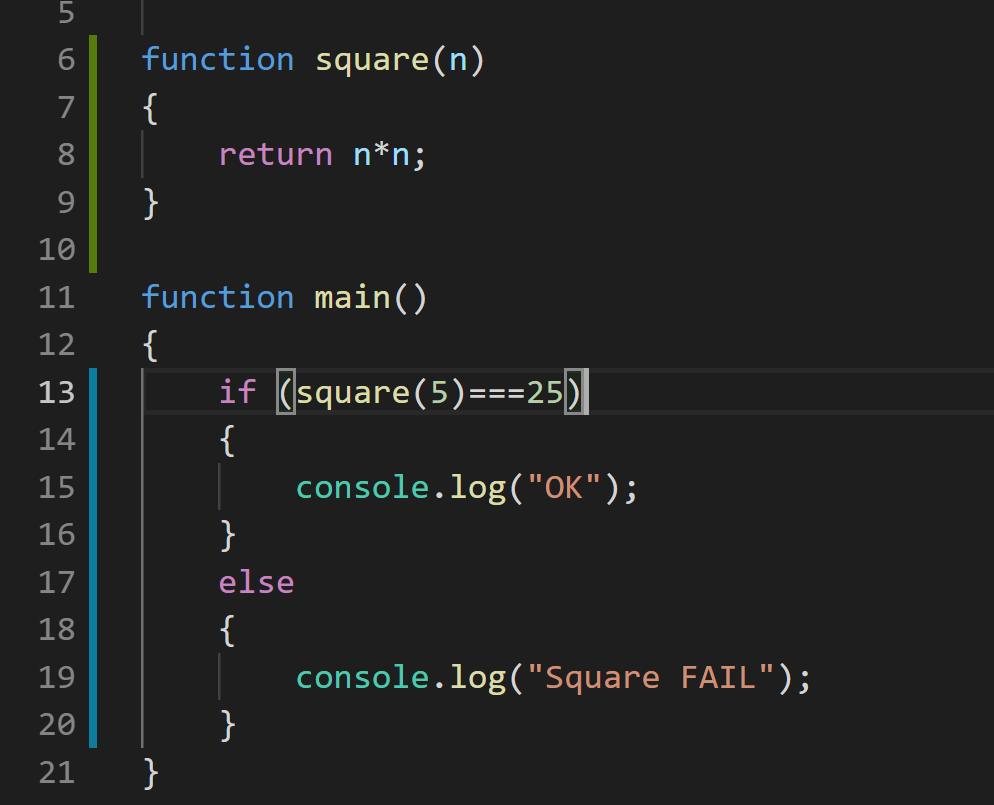
2. In main() in Lab2.js call each of the functions in turn, and check the output against the expected output. Use console.log to output OK or FAIL for each case.

Example

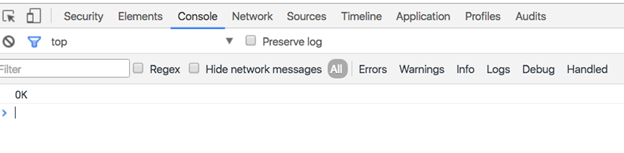
Square - a simple function that returns the square of the number given.

Download [Lab 2 Starter](https://drive.google.com/open?id=0B67fsQcmeL5bOHNDeE5BUVExQ0k)

Which contains index.html and Lab2.js



When you open the index.html and open the console you see:



..that shows that the square function is correct.

Reminder: the step is now finished so

Add the files to your repo

git add [filename]

Check they are all added using git status

git commit -m “Lab 2-1 Intro done”

git push origin master

## 

The exercises below are hosted on codewars.com - there is no need to sign up to codewars to complete these exercises. Instead, write your code using your favourite Javascript editor (e.g VS Code) and upload your solutions to your games devices github repository. Don’t forget to provide test data to test all the control paths through your code (printing ‘fail’ or ‘pass’ for each test as appropriate - see the function square example above).

## **2 How many lightsabers**

<https://www.codewars.com/kata/how-many-lightsabers-do-you-own/javascript>

## **3 Average marks**

<https://www.codewars.com/kata/get-the-mean-of-an-array/javascript>

* Iterate through an array

## **4 Pirates**

<https://www.codewars.com/kata/pirates-are-the-cannons-ready/train/javascript>

* Iterate through a dictionary
* String comparison

Note: you can just rewrite the first line

const cannonsReady = (gunners) => {

as

function cannonsReady (gunners) {

## **5 Crash override**

<https://www.codewars.com/kata/crash-override/javascript>

Note: Letters from ‘A’ to ‘D’ (instead of ‘A’ to ‘Z’) will suffice for testing.

## **6 OO Piracy**

<https://www.codewars.com/kata/object-oriented-piracy/train/javascript>

* Note: you can write a method of an object like this:

Ship.prototype.isWorthIt = function()

{

}

(If necessary, refer back to the lab 2 notes to see what this means).

Also in case you find the word “draft” in the problem confusing, in this case it just means an estimate of the total weight of the ship based on how low the boat is in the water.



(source wikipedia)

## **7 Refactor Squares**

Using what you’ve just learned about creating objects and methods, refactor the

code you wrote for Lab 1 (drawing 100 coloured squares around the screen).

* Create a copy of your old JavaScript file (originally called Empty.js) and call it RefactoredSquares.js and a copy of the html file and call it RefactoredSquares.html
* Also copy empty.css, otherwise your html page will generate an error.
* Create a new JavaScript file Square.js
* In this file, create a Square constructor which takes the x,y position values, width, height values and a colour, i.e.

function Square(x, y, width, height, colour)

* Write a draw method of square that draws the square.
* Next add Squares.js to your html file so you can access that code just like this:



* Next update your code in RefactoredSquares.js so it uses your new code in Square.js
* Test! It should work exactly as it did before… (just better organised code - this is very very very important for when your programs start getting big as they will quite soon)

## **8 Final git commit**

The final git commit for the lab should include the total time it took to complete the lab in the format of a smart commit

Example

git commit -m “Lab 2 complete #time 2h 30m”