**JavaScript Workshop**

**SLIDE 1**

* **Intro on me**

**SLIDE 2**

* **Any devs or know some coding?**

**SLIDE 3**

* **Anyone knows what JS is?**

**SLIDE 4**

* **Intro to JS**
  + Created in 1995 by Netscape to glue the predominant web language (HTML) to the user input and therefore improving experience
  + Not Java (originally called LiveScript and as it was developed taking inspiration from Java was called JavaScript also to leverage growing popularity)
  + **Q: Does anyone know how the web works?**

**SLIDE 5**

* **How the web works** 
  + RE Agent example
  + HTML (markup language) + CSS give structure and styling to a page
  + + JS is a programming language and gives behaviour
  + Runs in browser engine
  + Developer tools and console
* Test execute some code in browser console (console.log, addition)

**SLIDE 6**

* **Client-side rendering**
  + All happening in your browser
  + <https://www.lewagon.com/> (list for cities)
  + <http://lookbook.wedze.com/>
  + <https://www.amazon.co.uk/> (with and without JS)
  + Google Maps
* **Frameworks**
  + React, Angular, Vue, Electron (laptop)
* Netflix, Google Maps
  + PROS:
  + New UX / UI best (single page applications)
  + Components
  + Integrations for native mobile (React) / VR (React)
  + CONS:
  + Load time
  + Heaviness of pages (avg. 2.4 Mb, which in 1995 was the size of one of the biggest videogames)
* Benchmark The Guardian via console

**SLIDE 7**

* **Versatility, adoption rate and job demand charts**
  + Most used programming language (stack overflow)
  + Of NPM (package manager) users do all sorts of things
  + Amount of pull requests by language (Github)
  + Second most searched and paid job in UK (TechWorld)

**SLIDE 8**

* **Standardised**
  + ES6 and JSON

**SLIDE 9**

* **Front-end / back-end**
  + **Q: Any ideas why this is the case?**
  + **Q: Anyone know what it means?**
  + Restaurant example
  + Node is C++ program with Chrome engine
  + Mean stack
  + Brief intro on my workstation (Terminal + code editor)
* Show same code running in node

**ANY QUESTIONS?**

**SLIDE 10  
SLIDE 11**

* **JS Bin** 
  + Development environment VS. Workstation
  + Semi-colons in JS (use and best practice)

**SLIDE 12**

* **Types**
  + Tells the computer what is it that we are working with (and how to interpret it)
  + I.e. You can sum numbers but you can’t sum words

**SLIDE 13**

**SLIDE 14**

* **0 and 1 in computer memory which only has bits**

**SLIDE 15**

* **Non-numerical characters**

**SLIDE 16**

* **Objects**
  + Also called dictionaries or hash in other languages
  + Assigned via the curly braces
  + Set of multiple information organised in pairs of key / value
  + Remember JSON?
* **Arrays**
  + Same as objects but without keys, values can be accessed via their indexes (the position order in the sequence of elements)

**SLIDE 17**

* **We will see these later**

**ANY QUESTIONS?**

**SLIDE 18**

* **Variables**
  + **Q: anyone can explain what they are?**

**SLIDE 19**

* **Store information in locations of the computer memory (memory is like set of boxes with labels)**

**SLIDE 20**

* **Assign values**
  + Like placing something in the memory box with label firstName
  + Do that by using the assignment operator =

**SLIDE 21**

* **Console.log with const**
* **Reassign new name**
  + Let and const (difference in performance and bugs)
  + Dynamically-typed language
* **Console.log second time with lower case**
  + **Q: Any guess what will happen?**
  + Camel case naming convention (JS IS CASE SENSITIVE)

**SLIDE 22**

* **String Concatenation**
  + Remember what I told you about types? They tell computer what to do (i.e. sum numbers, but what with strings)
  + **Q: Any guesses?**
* **Show in JS Bin concatenation of First + Last name (WITHOUT SPACE)**
  + **Q: How do we fix this?**
* **Add space in concatenation**

**ANY QUESTIONS?**

**SLIDE 23**

* **Conditions**
  + Imperative language (default computer runs everything, unless we say so)
  + This is why also called flow control

**SLIDE 24**

* **If statement**
  + Triple === to check equality (have other possibilities)
  + Curly brackets delimiting what code is executed

**SLIDE 25 - 26  
SLIDE 27 – 34**

**SLIDE 35**

* **Show in JS Bin**
* **3 const of names and 1 let of combination**
* **Q: Any guesses?**
* **3 if statements with concatenation**
* **ANY QUESTIONS?**
* **Q: Issues with this code? LONG**

**SLIDE 36**

* **Loops**
  + Perform the same action a number of times without having to repeat the code

**SLIDE 37 – 39**

**SLIDE 40**

* **Show in JS Bin**
* **Q: Any guesses?**
* **Put names in array**
* **Insert looping structure**
* **Q: So remember we have this issue of the space at the beginning? Fix it**

**SLIDE 41**

* **Functions**
  + It’s a bit of code to which we give a name
  + We can also make them dynamic by setting parameters that can change every time we use it
* **JS Bin Square function**

**SLIDE 42 – 43**

**SLIDE 44**

* **Show in JS Bin**
* **Q: Any guesses on how to refactor using functions?**
* **How many things will be dynamic?**
* **2 parameters**

**SLIDE 45**

* **The DOM (Document Object Model)**
  + How JS calls the HTML file once received from server and displayed on the user’s browser
  + JS main purpose to manipulate this DOM in response to user’s behaviour
  + Explain why this is only possible with JS (i.e. the server doesn’t know when it’s sending back the HTML file what the user will do with it)
* Show The Guardian home page with and without JS (in latter case no weather info is displayed, because server doesn’t know where user is located and what weather there is at that precise time)

**SLIDE 46 – 61**

* **Select things on Le Wagon webpage**

**SLIDE 62**

* **Event handling**
  + This allows us to specify behaviour in response to user’s actions
* Code listener on first button on le wagon page to popup alert saying “How cool is JS?!”