**Front-End Web Developer Interview**

**Soft Skills**

About Yourself?

Why our company?

How do you plan a Web Dev project?

Likes and Dislikes about being a Web Dev?

What will you bring to our company?

**HTML & CSS**

Box model

Every element is represented as a Box.

* Element has CONTENT
* Content has a BORDER
* PADDING: Space between Content-Border
* MARGIN: Distance between Element-Screen, Sides.
* OUTLINE: Outside of the Border.

Border-Box: Prevents the boxmodel to change the given size of an element.(Default is Content-box: Adding borders, paddings, margins, etc... will modify the elemnt size).

Specificity

Rules to define who will take precedence.

1. !important
2. Id
3. Class, pseudo-class, attributes
4. Elements

Last Rule: Same selectors, dirreferent values. The last one written will be in priority.

Universal Selector (\*) has the LEAST amount of specificity.

Position

* FLOAT: Elements float around it's parent. Their siblings will wrap around, we prevent this using Clear: Both (L-R)
* STATIC: Default, each element will be displayed one after the other, as we code them into the HTML
* RELATIVE: Relative to it's current position, we can use T-B-L-R. Broweser **doesn't** fill the space where the element used to be.
* ABSOLUTE: Relative to ithe Body of Document, unless it's parent is Positioned Relative. Browser **fills** the space where the element used to be.
* FIXED: Relative to the Viewport. Fixes the element to the screen, keeping it in place even if we scroll.
* STICKY: Toggles between Relative and Fixed once the position is met in the Viewport. Then it's sticks. Must have one of the values T-B-L-R.

Visibitily: hidden Vs Display: none

Hidden: Not visible but it occupies it's original space.

None: Its not visible and does NOT take space.

Flexbox Vs Grid

* FLEXBOX: Efficient way to layout(**one-dimensional**), align and distribute space among items in a container, even when their size is unknown and/or dynamic. Give the container the ability to alter its item's width/height (and order) to best fill the available space A flex container expands items to fill available free space or shrinks them to prevent overflow.
* GRID: **Two-dimensional** grid-based layout system. Define a Container element to be display:grid, and then we set Columns and Rows sizes.

The Flexbox layout is best suited to application components and small-scale layouts, while the Grid layout is designed for larger-scale layouts that are not linear in design.

Preprocessors (Less, Sass)

A CSS preprocessor is a program that lets you generate CSS from the preprocessor's own unique syntax. The can have some Features that don't exist in CSS, Mixins, Nesting selectors, Inheritance selectors, etc...

The goal is to make CSS structure more readable and easier to mantain.

Mobile-First Approach

Generally speaking, a mobile-first approach means building your website with your mobile users in mind, with the main goal of improving these mobile users experience on your site. The development team makes all these design decisions first for mobile size, then move their way up for desktop sizes.

Why? Because the internet usage for mobile devices has increased a lot over the last years.

**JAVASCRIPT**

What is Javascript

JavaScript is a scripting or programming language that allows you to implement complex features on web pages.

* Inline: Inside our element we add the JS functions

<button onclick="alert("Hello")">Click </button>

* Internal: Inside our HTML we use Script tag

<script> document.querySelectorAll(".class")</script>

* External: Create a new JS file and import it

<script src="./app.js"></script>

What is ECMAScript

Const / Let / Var

* VAR: 1995-2015(ES6), Then was replaced with Let and Const. It has a Function Scope, can be re-asigned.
* LET: Can be re-asigned, no need to be initialized when called.
* CONST: Cannot be re-asigned, must be initialized when called.

Array Properties and Methods

* Length: Returns the quantity of items inside a given array

console.log(array1**.length**)

* Concat: Joins two arrays together.

const allArrays = array1**.concat(**array2**)**

* Reverse: Flips the order of the items

console.log(allArrays.**reverse()**)

* Unshift / Push: Add items into the array (first place / last place)
* Shift / Pop: Remove items from the array (first place / last place)
* Splice: Mutates the original array, takes Index and Amount of items to be deleted

allArrays.**splice(**2, 1**)** -> .splice(index,amount)

Array Iterators

* ForEach:
* Map:
* Filter:
* Find:
* Reduce:

== // ===

Promise \*\*\*

null // undefined \*\*\*

AJAX \*\*\*

sync // async \*\*\*

DOM\*\*\*

**REACT**

**GENERAL**

Add Basic Responsive Styles