SeniorCircle TechStack Preparation

## Git and Github

## HTML/ CSS

* MERN:

## Javascript for Frontend(React)

It depends on how much you want to regret the code you written in your first React project after a year.

I compiled a list of Awesome JavaScript Learning (https://github.com/micromata/awesome-javascript-learning) resources which might be able to help you.

I don’t think you need to work through the whole list and study language and Web APIs for weeks, but you need to know a few things to make React work for you.

I personally would recommend the following to get a solid understanding of the following JavaScript language concepts and APIs:

1. Basics including scopes, hoisting, functions as first class citizens, closures
   1. [A re-introduction to JavaScript (JS tutorial)](https://developer.mozilla.org/en-US/docs/Web/JavaScript/A_re-introduction_to_JavaScript)
   2. Understanding Hoisting in JavaScript (https://scotch.io/tutorials/understanding-hoisting-in-javascript)
2. How to manipulate data in a functional way
   1. [Array methods](https://javascript.info/array-methods)
   2. JavaScript's Map, Reduce, and Filter • Dan Martensen (https://danmartensen.svbtle.com/javascripts-map-reduce-and-filter)
   3. [Object.keys, values, entries](http://javascript.info/keys-values-entries#object-keys-values-entries)
3. Pure functions
   1. Master the JavaScript Interview: What is a Pure Function? (https://medium.com/javascript-scene/master-the-javascript-interview-what-is-a-pure-function-d1c076bec976)
4. Handling asynchronous code
   1. [An incremental tutorial on promises in javascript](https://www.sohamkamani.com/blog/2016/08/28/incremenal-tutorial-to-promises/)
   2. Why JavaScript’s Async/Await Blows Promises Away (Tutorial) (https://hackernoon.com/6-reasons-why-javascripts-async-await-blows-promises-away-tutorial-c7ec10518dd9)
5. Asynchronously sending and receiving data
   1. [Using Fetch](https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch)
   2. [Why I won’t be using Fetch API in my apps](https://medium.com%2F@medium.com/@shahata/why-i-wont-be-using-fetch-api-in-my-apps-6900e6c6fe78)
6. ES6+
   1. [ES6 Modules](https://ponyfoo.com/articles/es6#modules)
   2. [Let and Const](https://ponyfoo.com/articles/es6#let-and-const)
   3. [Arrow Functions](https://ponyfoo.com/articles/es6#arrow-functions)
   4. [Classes](https://ponyfoo.com/articles/es6#classes)
   5. [Template Literals](https://ponyfoo.com/articles/es6#template-literals)
   6. [Spread Operator and Rest Parameters](https://ponyfoo.com/articles/es6#spread-operator-and-rest-parameters)
   7. [Destructuring](https://ponyfoo.com/articles/es6#assignment-destructuring)

The last three ES6 points are not really needed as they really are syntactic sugar. But they let you write less code to get things done. Which brings me back to beginning of my answer:

»It depends on how much you want to regret the code you written in your first React project after a year.«

## Javascript for Backend(Nodejs)

The most important thing is to get started. Good luck on your journey.

* + Class
  + Modules (including import and export)
  + Inheritance
  + Data types and type casting
  + Constructor, member variable, member function
  + Null and undefined
  + Array.forEach(), Array.map(), Function.call(), Function.apply(), Function.bind() etc function
  + Block scoping and hoistation
  + Working with keyword **this**
  + Arrow function
  + Exception handling
  + String manipulation
  + Callbacks
  + Asynchronous programming
  + Associative array, for-in loop, Map, WeakMap, Set, WeakSet
  + Promise
  + Working with events, event propagation & delegation, bubbling, capturing etc