Hi there, my name is Ju Gonçalves and I do computer stuff.
github twitter

How to start with JavaScript Programming

January 24, 2015

For Short

To have an environment running JavaScript code, you should preferably install Mozilla Firefox. Then you'll need to hit `Cmmd + Opt + k` or `Ctlr + Shift + k` , to open up the console.

You are ready to start programming in JavaScript.

Here is some details you should know before start typing:

JavaScript...

1. has C-like syntax.
2. has Functional features, like First-Class and High-Order Functions. But not immutable state.
3. has Object-oriented features. Everything is an Object and it's Prototype-based rather Class-based, as the most well-known OOP languages.

4. is dynamically typed and uses duck typing.
5. uses call-by-sharing as evaluation strategy, as well as Ruby and Python.
6. has single threaded engines. You're basically going to handle async computations using events or callbacks (or promises).
7. has a pretty good package manager called npm. You can use it by installing Node.js or io.js.

Good luck !

Not-so-short version

If you've reached to this point, you'd want someone to show you a safe path. Here it is.

`Warning` This guide is obviously biased by my experience and background, even though you should check out the resources by any means.

###Instalation

Node.js (or io.js) instalation is pretty straightforward. You can download it from their website or grab it from repositories out there.

As many other modern programming language, you'd be able to use its package manager as well after instalation. It's called npm and has a bunch of nice features and a lot of modules. You should definitely use it.

###First Steps

Initially, you must understand the language itself and there are 2 good (and free) resources for that:

- Speaking JavaScript, by Dr. Rauschmayer
- Eloquent JavaScript, by Marijn Haverbeke

Those books are specially good if you don't have any background in (imperative) programming or low-to-none experience at it.

Once you're done learning JavaScript basics, you can dive into Effective JavaScript, by David Herman. Herman is one of the authors of the language, it's a good idea to check out what he has to say.

###Mastering

To master JavaScript, I believe you must successfully get along with 3 specific topics:

1. Functional Programming
2. Prototype-based Object-Oriented Programming
3. Async/Concurrent Programming

Differently from Eric Elliot, who believes JavaScript has 2 pillars (the first two listed), I think it's important to have a good understanding on how to handle Async computations and to correct synchronize them. I've seen a lot of good programmers failing on this job and triggering race conditions, etc. It's a subject you really should put some efforts on.

###Functional Programming

I don't think you should dive into other programming languages (as I did) to understand some Functional Programming Principles and *apply* them into your everyday code in JavaScript.

The minimum knowledge:

- What is and how to get along with First-Class and High-Order Functions;
- How to use `map`, `filter` and `reduce`;
- How to build curry-ed functions and use partial

application;
- How to compose functions and pipeline function calls.

There are 2 books that may help you in your journey:

- Functional JavaScript, by Michael Fogus
- JavaScript Allongé, by Reg “Raganwald” Braithwaite

###Prototype-based Object-Oriented Programming

That’s a tough subject for the most experienced programmers out there, specially because we’re all *too much* used to Class-based OOP. In addition, there is that perpetuated idead that classes are must-have concept for OO programs. But it’s not, like JavaScript and some others have proved us.

Prototype-based style is more expressive in comparison to Class-based style, since you can write the second in terms of the first (the reverse is not possible). The sad part is that we don’t have too much resources regarding the topic. Instead, there’s a plenty of content teaching you how to magically mutate it into Class-based and moving on.

The minimum knowledge:

- How to work with Prototypes
- How to properly use `Object.create`, `Object.defineProperty`, etc
- Understand correctly the Prototype chain

I think you should read Elliot's chapter regarding Objects and Object-Oriented Programming from his book and his blogpost on the subject. Sorella also wrote a good article on JavaScript OOP.

###Async/Concurrent Programming

It's not that common a environment running async by default. However, it is JavaScript's scenario. Most of the time, you'd need to handle async computations and sync them back. Async programming is not optional or avoidable here.

The minimum knowledge:

- Correctly understand the Event Loop and how to manage native and artificial events
- Master promises

For this, there's an interesting book called Async

JavaScript, by Trevor Burnham, that covers the basics of the single threaded nature of JavaScript engines, events, callbacks and promises. Also, there's a good article and talk about Event Loop you can follow.

Nevertheless, events and promises aren't all you got. You should be aware of `async/await`, which is going to be available on JavaScript engines *soonish*, but also CSP, FRP and maybe Transducers. How to correctly do async computations in JavaScript (and in many other languages) is still under discussion and you're a probably going back to the subject many times during your work life.

Closing Thoughts

Of course there's a lot more of JavaScript knowledge you should have for building application. I think most of them you can derive while you're practicing or working, when you master the above-mentionend topics. However, if you'd like to have a guided walk-through, I suggest you to have a look into Human JavaScript book, by Henrik Joreteg. It covers modern JavaScript development of client-side applications, by going through the fundamentals of models, views, templates and routing.

I don't think you should put too much efforts on learning a library or a framework. They will be replaced soon or later.

You should always seek a strong undestanding of the concepts and how to apply them in your everyday work.

Wanna read more?

Found anything wrong?