q.1 List 5 difference between Browser JS(console) v Nodejs

Both the browser and Node use JavaScript as their programming language.

Building apps that run in the browser is a completely different thing than building a Node.js application.

Despite the fact that it’s always JavaScript, there are some key differences that make the experience radically different.

As a frontend developer who extensively uses Javascript, Node apps brings with it, a huge advantage - the comfort of programming everything, the frontend and the backend, in a single language.

You have a huge opportunity because we know how hard it is to fully, deeply learn a programming language, and by using the same language to perform all your work on the web - both on the client and on the server, you’re in a unique position of advantage.

What changes is the ecosystem.

In the browser, most of the time what you are doing is interacting with the [DOM](https://flaviocopes.com/dom/), or other [Web Platform APIs](https://flaviocopes.com/web-platform/) like Cookies. Those do not exist in Node, of course. You don’t have the document, window and all the other objects that are provided by the browser.

And in the browser, we don’t have all the nice APIs that Node.js provides through its modules, like the filesystem access functionality.

Another big difference is that in Node.js you control the environment. Unless you are building an open source application that anyone can deploy anywhere, you know which version of Node you will run the application on. Compared to the browser environment, where you don’t get the luxury to choose what browser your visitors will use, this is very convenient.

This means that you can write all the modern [ES6-7-8-9](https://flaviocopes.com/ecmascript/) JavaScript that your Node version supports.

Since JavaScript moves so fast, but browsers can be a bit slow and users a bit slow to upgrade, sometimes on the web, you are stuck to use older JavaScript / ECMAScript releases.

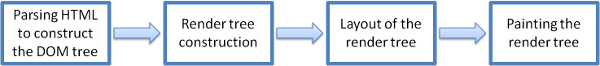
You can use Babel to transform your code to be ES5-compatible before shipping it to the browser, but in Node, you won’t need that.

Another difference is that Node uses the [CommonJS module system](https://flaviocopes.com/commonjs/), while in the browser we are starting to see the [ES Modules](https://flaviocopes.com/es-modules/) standard being implemented.

In practice, this means that for the time being you use require() in Node and import in the browser

q.2 summary of bryan seddon :how a browsers works

1. the browser works in the following way:



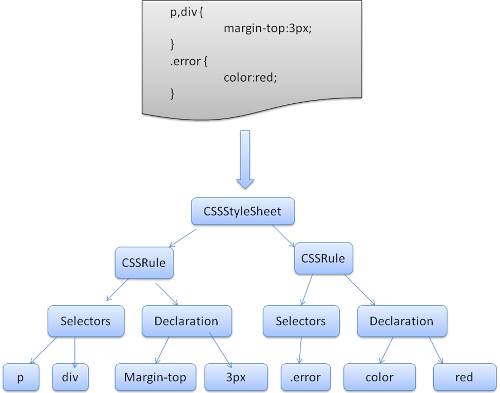
1)parsing html:

* HTML is forgiving by nature
* Parsing isn’t straightforward
* Can be halted
* Will do speculative parsing
* Its reentrant

2)<script/> at bottom

* Parse uninterrupted
* Faster to render
* Defer and async attributes

3)CSS parsing



4)DOM to CSSOM:

* Combines the two object mode, style resolution
* This is the actual representation of what will show on screen
* Not a 1-1 mapping of your html.

5)painting

* Produces bitmap from each layer
* Bitmap is uploaded to gpu as a texture
* Composites the textures into a final image to render to the sheets.

1. Q.3) Execute the below code and write your description in txt file
   1. typeof(1)
   2. typeof(1.1)
   3. typeof('1.1')
   4. typeof(true)
   5. typeof(null)
   6. typeof(undefined)
   7. typeof([])
   8. typeof({})
   9. typeof(NaN)

ans the ans is nil