1 .List difference between Browser JS(console) vs Nodejs

S.No	Javascript	NodeJS
1.	Javascript is a programming language that is used for writing scripts on the website.	NodeJS is a Javascript runtime environment.
2.	Javascript can only be run in the browsers.	NodeJS code can be run outside the browser.
3.	It is basically used on the client-side.	It is mostly used on the server-side.
4.	Javascript is capable enough to add HTML and play with the DOM.	Nodejs does not have capability to add HTML tags.
5.	Javascript can run in any browser engine as like JS core in safari and Spidermonkey in Firefox.	Nodejs can only run in V8 engine of google chrome.
6.	Javascript is used in frontend development.	Nodejs is used in server-side development.
7.	Some of the javascript frameworks are RamdaJS, TypedJS, etc.	Some of the Nodejs modules are Lodash, express etc. These modules are to be imported from npm.
8.	It is the upgraded version of ECMA script that uses Chrome's V8 engine written in C++.	Nodejs is written in C, C++ and Javascript.

2.How does the browser actually render a website

The browser it's probably one of the most complex application that we use our 90% of our interaction with the computer,like text editor or browser open

sort of like the components that make up a browser for the rendering stuff

Binding a lot of operating system stuff so when it talks to the network, it

will use certain APIs depending on the operating system, then the rendering part,

that's about actually constructing the website from the HTML that gets sent to it,

platform stuff that's dependent on OS, there are different things btw OS, then

JavaScript Virtual machine.

Rendering the parsing of the HTML,how it lays out the page and pints to it the

screen and you get the final result of the actual website

High level flow

parse html and parse css combine pass into RenderTree to layout to paint

parsing HTML

- HTML is forgiving by nature
- parsing isn't straight forward and can be halted
- will do speculating parsing
- it's reentrant

will halt the parser as a using <script>,<link>,<style>

Speculating parsing:

- will look ahead
- External images,scripts,css

<script/> at the bottom

- Parse uninterrupted
- Faster to render
- defer adn async attributes
- Trade off

DOM + CSSOM

- Combines the two object models,styles resolution
- This is the actual representation of the what will show on the screen
- Not a one to one mapping of your Html

Multiple trees

- render objects,styles,layers,line boxes

Dom node to renderObject

- visual output
- Geometric info
- can layout and paint
- hold style and computed metrics

Calculating visual properities

- combines all styles

- defaults,external,style elements & inline
- complexity around matching rules for each elements

Paint setup

- will take the layed out render trees
- creates layers
- incremental process
- builds up over 12phases

3 typeof operator

a. typeof(1) - `"number"`
b. typeof(1.1)- `"number"`
c. typeof('1.1')- "string"
d. typeof(true)- "boolean"
e. typeof(null)- `"object"`
f. typeof(undefined)- `"undefined"`
g. typeof([])-"object"
h. typeof({ })-`"object"`
i. typeof(NaN)- `"number"`