1. **Differences Between Browser JS vs Console JS:-**

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | Description | Browser JS (Console) | Node JS |
| 1 | Execution | The code only gets executed in the browser. | The code gets executed outside the browser as its both environment and interpreter for running an JavaScript. |
| 2 | Usage | Browser JS is mostly used for front-end and client-side applications. | Node.JS is mostly used for backend applications. And server-side applications. |
| 3 | Access to System | Browser JS access is limited to browser. | Node.JS has full access to system to read and write directly to the file. |
| 4 | User Interface | Browser JS has GUI. | Node JS doesn’t have any GUI |
| 5 | Modularity | It is not mandatory to put everything in a module. | It is mandatory in Node JS to put everything in a module. |
| 6 | Running Engine | Browser JS runs any engine like Firefox, safari, Chrome. | Node JS only runs in V8 engine. |

1. **Browser and its Functionality :-**

Components of a browser:-

* Bindings
* Rendering :- Parsing, layout, painting
* Platform
* JavaScript VM

Parsing:- Parsing isn’t straight forward.

Parsing stages:-

* Tokenizer:- It helps in breaking the code into smaller chunks, Tokens in HTML are start tag, end tag, self-closing tag and plain text content. Output of this stage is series of tokens.
* Tree construction: - series of tokens from tokenization is the input to this state, after the inputs are taken for tree construction and once the parser is created, tree construction is associated with DOM.
* DOM:- It treats everything as an object and DOM is constructed as a tree of objects. As JavaScript cannot understand tags easily, It interprets DOM easily.
* Like Document Object Model, we have CSSOM (CSS object Model) :- It’s a map of all CSS selectors and relevant properties of each selector.
* Once the HTML and CSS are parsed the next stage is Rendering. It has multiple trees like render objects, render styles, render layers, line boxes.
* Layout is a recursive process where it traverses the render tee and has to layout children layouts. It computes where the element should be placed on the screen.
* Painting takes all the input from the render tree and gives the visual of the output that is expected.

1. **Execute the below code and write your description in txt file**

* typeof(1) :- ‘ number'
* typeof(1.1) :- 'number'
* typeof('1.1') :- 'string'
* typeof(true) :- 'boolean'
* typeof(null) :- 'object'
* typeof(undefined) :- 'undefined'
* typeof([]) :- 'object'
* typeof({}):- 'object'
* typeof(NaN) :- 'number'