**What is a network?**

-> Interconnection between two or more computers.

**What is Internet?**

-> Interconnectivity between networks. Small networks which are connected through some mechanism to exchange data.

**What is World Wide Web?**

-> An application/ mechanism which is widely used by people. It is applied on Internet. Applications used on the internet is www.

**What are the following protocols? Protocols are needed so that the communication can be done properly.**

1) TCP/IP: Transmission Control Protocol/ Internet Protocol

2) FTP/S: File Transfer Protocol/ Secure

3) HTTP/S: Hypertext Transfer Protocol/ Secure

4) IMAP: for mailing

5) POP3: Post Office Protocol (for mail)

6) SMTP: Simple Mail Transfer Protocol (for mail)

**Client and Server Architecture**

**What are clients?**

-> Machine from which requests are generated is known as client-side machine. Machines used by users. Clients are not connected to each other; they are end nodes/points. Ending device on network.

**What are servers?**

-> Machine that responds to the client request is server-side machine. Any device can be used for server. Servers are interconnected. They are only responding to the request. Specialized hardware.

Server farm- multiple servers are kept together.

Examples-

Mail Servers

File Servers

Web Servers

DB Servers

**Languages for the Web**

1. HTML
2. XML - language with which you can define rules and regulations for other markup languages.
3. CSS
4. JavaScript
5. PHP
6. Perl
7. Python
8. Java

**LAMP Stack**

What is LAMP Stack?

Linux Apache MySQL/MariaDB PHP (Perl/Python)

XAMPP/WAMP

Localhost: points to an IP Address 127.0.0.1 (reserved, which points to your own machine.)

**Web UI and Web Applications-**

Web UI and UX

* Over a period of years web has emerged from serving simple static website
* To interactive sites
* Providing content based on

-Search

-Geo location

-Video streaming

-Live chat

-Online training and more

-Directly to your browser.

**Web UI**

***User Interface Design***

The practice of designing the front end of/for a software, application, or web (site) interface.

The look, design, and details of the interface.

**User Experience (UX) Design-**

The overall process or the experience the user gets while working with interface.

Primary idea behind UX is each user should have a positive experience while working or dealing with a site or product.

**Web Application-**

An application designed to be accessed over the internet via a web browser file like FireFox or similar.  
A web application may not be exactly designed like traditional application. Usually designed with server-side scripting languages like PHP, Node JS, etc.

*Advantages-*

One s/w for all access: *only a web browser is required*

Anywhere anytime access: *with net and any computer you can access the data and app.*

Update to application: *As applications are maintained on the server-side, user need not to update anything usually. Rare case, you might need to update your web browser.*

*Disadvantages-*

Offline accessibility

Privacy and Security

Limited functionality

**HTML**

Q. What is HTML?

Hypertext markup language.

Q. Which version do you use?

HTML 5.

Q. Why do we need HTML version?

?

HTML is maintained by Web Hypertext Application Technology Working Group or the WHATWG.

A consortium of the major browser vendors (Apple, Google, Mozilla, Microsoft).

Authentic info about HTML is found on https://html.spec.whatwg.org/multipage/

**Attributes-**

* Attributes are used in tags to further define the tag.
* Attributes are a name-value pair:

<ol start=’5’>

* Attribute name, equal sign, opening quotes, attribute value, closing quotes
* Exception to the name-value pair is:

Boolean attribute

* A tag can have multiple attributes.

*Id and class attribute-*

* Id is used for giving unique identity to a tag
* Class is used to assign similar/same values to tags, usually css.
* Attributes can be global
* Global attributes can be applied to all tags
* e.g.: id and class and style
* Non-Global Attribute are attributes applied to a specific instance of a tag.

Style Vs. Content

<b> Vs. <strong>

Bold is a style that makes letters thicker so it stands out among other text but it also has no semantic meaning.

Strong is an indication of how something should be. It looks like bold in a browser, but it could mean speak with urgency or seriousness when reading text aloud.

<p> Hey! <strong>You</strong> are amazing </p>

<i> Vs. <em>

Italics slants text.

Emphasis is used to stress emphasis of its contents.

Semantic elements

1: of or relating to meaning in language

2: of or relating to semantics

Semantic HTML is HTML that concentrates on the meaning of information in web pages instead of its presentation or look.

To add a paragraph, use <p> tag

To add a heading, use header tags (<h1>...<h6>)

To add an image, use <img> tag.

These tags along with their id and class attributes are semantic because they suggest the purpose of the content within the page.

*Why semantic tags are important?*

* Beneficial to both the developer and browser
* Convey more info about HTML doc’s content and structure.
* Additional communication about structure is useful for a developer who can understand the markup structure better.
* For the browser it can differentiate different types of data.
* Results in better display of content in different devices.
* Assistive technology, such as screen reader
* Read content and convey info about the content depending on the semantic meaning. For example, identifying headers and reading them in a different tone.

**Better tags in HTML 5-**

Tags such as

*<article>*

*<section>*

*<header>*

*<nav>*

*<footer>*

*<aside> :* side content other than main content, like sidebar. These are not considered as part of the main page outline.

*<summary>:* Used within the <details> tags. Specifies the visible content. The reast of the content in details is shown/hidden by user.

*<details>:* A way to provide additional info that the user can show or hide. Other content is hidden and can be expanded to view.

*<figure>:* Contains an image and can be used to group with an image’s caption.

*<figcaption>:* Provides a caption (explanation) of an image. To be used with <figure>.

*<mark>:* Defines a part of a text that is to be highlighted.

Were specifically used in HTML 5 to define the web page structure.

**Selectors and Declarations**

CSS is just a list of rules

Each rule consists of a selector and a declaration

*p{font-size : 12px;}*

*Selector*

In the above, p is the selector

When a selector appears non-prefixed by any punctuation, then it is assumed to match to an HTML tag.

*Declaration*

The declaration part of a CSS rule opens and closes with curly braces

**Common CSS Properties**

There are hundreds of CSS properties.

List of the most useful and common CSS properties-

1. font-size- Used to size the text of a tag. *p{font-size: 18px;}*
2. line-height- drives the height of the space a font will fit. *E.g. p{font-size:16px; line-height: 20px} #line height should be more than the font size or the texts will be mixed up.*

1. text-align- Used to align the text *E.g. p{text-align : left;}*
2. text-decoration-
3. font-weight
4. font-style
5. font-family

**What is JavaScript?**

It is not Java. It is a scripting language. Used primarily with HTML.

With JavaScript you can basically manipulate HTML page i.e. the DOM

*Self-learn and describe: What is DOM?*

*Get more details about document.write(). Where to use and where not use, and why?*

*Ternary operators*

**What is jQuery?**

jQuery is JavaScript library i.e. predefined set of JavaScript code.

**Why this code?**

JavaScript is used on lot of webpages. Many of these tasks are common. jQuery is full of this kind of code. We use this for our requirements reducing a lot of work and efforts.

jQuery is an open-source software under permissive MIT license.

Designed to make it easier to work with (HTML) DOM- select and manipulate DOM elements, create animations, handle events, develop Ajax Application.

Originally created in January 2006

BarCamp NYC by John Resig.

As of Dec 2023, jQuery is used by around 77.3% of websites.

Latest release of jQuery (core) is 3.7.1 (Jan 2023)

**How to use jQuery?**

To use jQuery you need its file. This file contains the primary functionality. All code is written in JavaScript. To use this functionality, you need to have access to the code. You can download this file from

**AJAX**

**What is AJAX?**

Stands for Asynchronous JavaScript and XML.

It is not exactly a programming language

It primarily uses the *XMLHttpRequest* Object.

Send and receive data in the background. How we do it and what advantages it has?

**What is XMLHttpRequest or XHR?**

It is a built-in browser object.

All moder browsers like Firefox, chrome, edge, etc have this.

Allows to make HTTP requests in JavaScript

**What can XMLHttpRequest do?**

It is used to exchange data with a server behind the scenes.

That means, update a part of web page, request data from a server after the page has loaded, receive data from a server after the page has loaded. Send data to a server in the background.

All this without reloading the complete page from the server.

**AJAX working-**

Request is sent to the server

Page is loaded on client

Event occurs in a web page(client)

Page is loaded, a button clicked etc. Then XMLHttpRequest object is created which sends a request to a web server.

The server processes the request and sends a response back to the webpage.

Response is read by JavaScript, action (like page update) is performed.

**Ajax V/s jQuery AJAX:-**

Ajax example

101\_ajax\_test.html

101\_ajax\_test.php

**Node.js**

JavaScript is a client-side scripting language.

Anything associated with website is saved on web-server.

The execution of JavaScript on the server is not done by Node. It’s done with a virtual machine like V8 or Chakra.

Node.js is an open source, cross platform runtime environment.

Used for developing server-side and networking applications.

Node.js applications are written in JavaScript.

Node.js = Runtime environment + JavaScript Library

**Benefits of Node.js:**

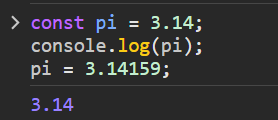
It is asynchronous as compared to PHP or ASP which waits for the system to return info like open a file, return contents to client and be ready for next request.

Node.js eliminates the waiting time by simply continuing with the next request.

Memory efficient.

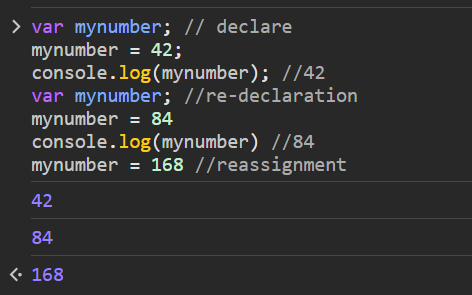
**JavaScript and Variable**

Var, const, let



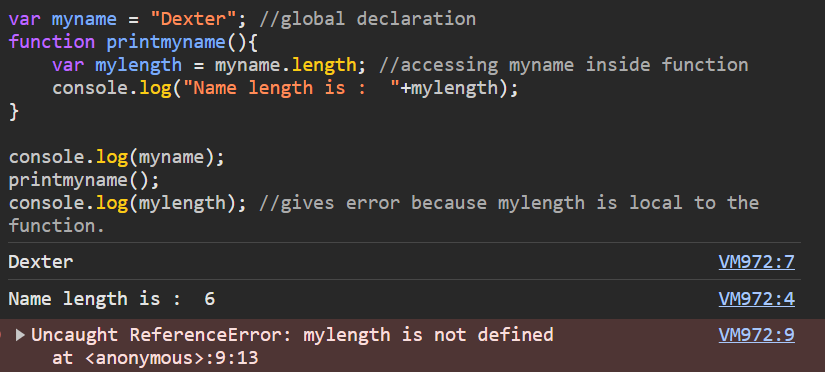
We defined pi using const, it can’t be changed later.

*console.log prints the variable value*

**

*We can reassign the value of a variable if it is a* var *variable*

*We can assign integers using var as well. Var is not solely for strings.*

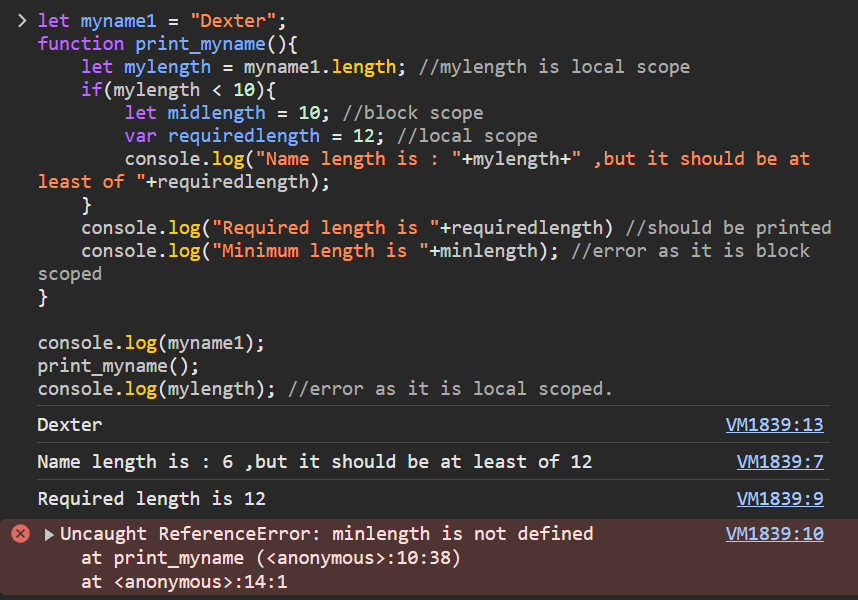


We cannot access local variables which are inside the function.

**let:**

**re-declaration is not allowed in let, reassignment is.**

let can have global, local and block scope.



Scope of const and let is same.

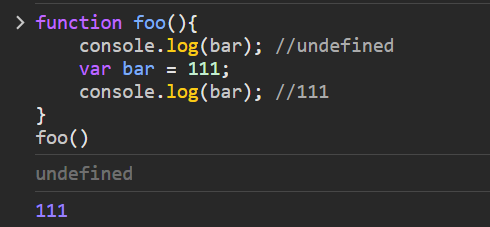
Hoisting: Variable declarations are processed before any code within the script is executed irrespective of where they occur in a script.

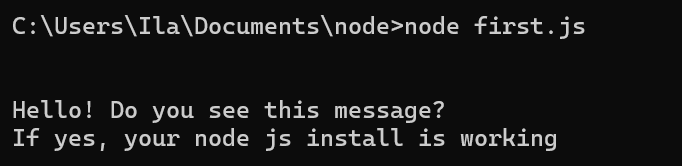
Declaring a variable anywhere in the code is equivalent to declaring it at the top.

a = 2 var a

==

var a a = 2





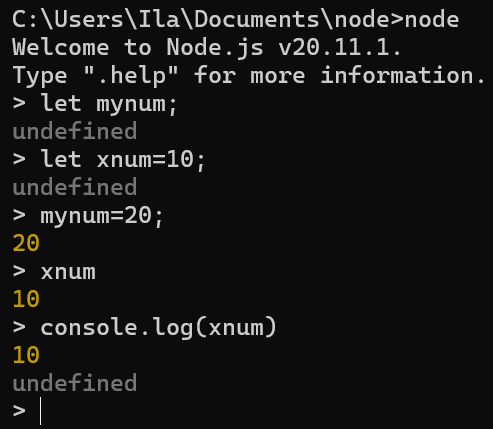
Node programming

localhost –

if we all are connected to same network, we can access others server.

Type node on cmd, you get REPL (read, evaluate, print, loop)

Undefined is one of a primitive data type

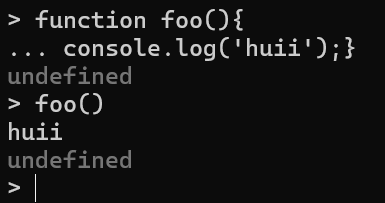


Let mynum and let xnum = 10 are not evaluating. They are declaration

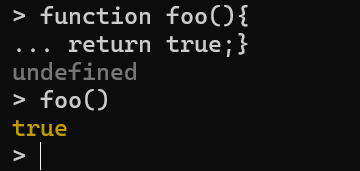
Mynum=20 is an (assignment) operation so it is evaluating

Mynum is evaluating the value of mynum and printing it.

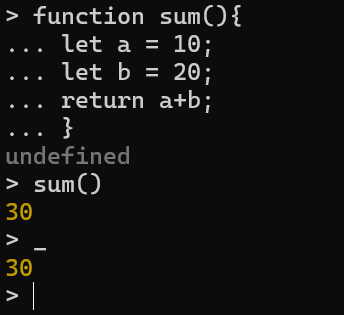
Console.log(xnum). Console.log is function and functions are supposed to written a returned value.



The undefined is not because of console.log. It is because of function as function is not evaluating anything or returning anything.



Now the function is evaluating and returning some value.



\_ is a default variable. Returns value of previous called function.

REPL Commands-

Ctrl + c : terminate the current command

Ctrl + c c : terminate the Node REPL

Ctrl + d : terminate the Node REPL

Up/down Keys : see command history and modify previous commands.

Tab keys : list of current commands

.help : list of commands

.break : exit from multiline expression

.clear : exit from multiline expression

.save filename : save the current Node REPL session to a file

.load filename : load file content in current Node REPL session.

**Module** is a functionality which is available in single or multiple JavaScript files.

This can be reused throughout the application

Modules can be simple or complex.

http is the example of module in myserver.js

**Modules are of three types:**

1. Core Modules : are built-in modules. These come with Node.js installation. These modules can be loaded into the program by using the require function.

To use core modules, we need to import it using require() function

Var module = require(‘module\_name’);

1. Local Modules: are created locally in your Node.js application. Modules include different functionalities of your application in separate files and folders.

Basically, code written by you for your specific needs and is reusable.

1. Third-party Modules: are available online. These can be used with the help of Node Package Manager (NPM).

Modules can be installed in the project folder or globally. Some of the popular third-party modules are mongoose, express, angular, and react.

**Synchronous**

Happening or existing at the same time

**Asynchronous**

Not happening or done at the same time or speed.

Node.js is asynchronous. JavaScript is synchronous

**Synchronous code**

It is executed sequentially from top to bottom. Each statement is completed before the next one begins.

Synchronous code is also called “blocking” as it halts the program until all the resources are available.

Sync = Synchronous = Blocking I/O model

**Asynchronous code (execution)**

Allows multiple processes to run independently w/o blocking each other or previous one.

Statements do not have to wait for other processes to finish before they can run.

Asynchronous code maximizes utilization of system resources.

Async = Asynchronous = non-blocking I/O model

One can get better output in asynchronous.

Reading file in Sync Vs Async mode

Node provides with

1. fs.readFileSync() //Sync method
2. fs.readFile() //Async method

What is the file ‘package.json’?

All npm packages contain a package.json file usually in the project root. This file holds various metadata relevant to the project.

**File System**

var fs = require(‘fs’);

fs.readFile(‘page.html’, function(err, data){

if(err) throw err;

console.log(data.toString(‘utf8’))

});

writeFile() is used to write content in the file.

**Node js and database**

Node Js can connect to variety of DB servers

* MySQL/Maria
* MongoDB
* Postgres
* And More