#deployment tools:

netlify-tool where you can buy domain names

GitLab-it has rich UI

git and git hub.

fire base

bit bucket

git-hub --> junk in ---> docker ---> cuber-net ---> aws(contains EC2 virtual computer uses ubantu os)

masking is done for ip for customers using s3 buckets

Git & git hub

Git hub :used to store and manage the data.one type of cloud

Git:used to track code in git hub(updation,deletion).version controller

#How to use:

Install software(Gitbash) either run it on vsc or cmd orelse gitbash itself.

Gitbashinstallation:open browser-gitscm download click on downloads git choose your os(windows) ,click on click here to download it starts downloading

git --version to check version of git

When you type code. In cmd it automatically opens vscode go to terminal in that choose new terminal on that in place of powershell choose cmd and type command git --version

Day2 bunk

Day3

**Types of errors in JavaScript**

1. Syntax
2. Reference
3. Range
4. Type

**Operator:**

It is a predefined symbol used to perform specific operation.

Arthematic operator:+,\_,\*,^,/

Assignment operator:+=,-=,\*=,/=,%=

**Increment**:Pre increment,post increment

**Post increment**:first value assigned then 1 will be added

Int a=10;

Console.log(a++);//10

Console.log(a++);//11

**Pre increment**:value added and then assigned

Let b = 20;

Console.log(b);//20

Console.log(++b);//21

**Decrement**:post decrement,pre decrement

**Post decrement:**

Let c=10;

Console.log(c);//10

Console.log(c--);//10

Console.log(c--);//9

**Pre decrement**:

Let d = 10;

Console.log(d);//10

Console.log(--10);//9

**Assignment operators**:used for shot hand

Let m = 10;

Console.log(m=m+10);//20

Console.log(m += 10);//assignment operator

**Logical operators:**

AND(&&)

OR(||)

NOT(!)

**Relational operators:** output format Boolean

<,<=,>,>=,==,===,!=

**Equals to**(==) to check values not data types

Console.log(5 == 5);// true

Console.log(5 == “5”);//true

**Strict equals to**(===)checks both value &data type

Console.log(5 === 5);// true

Console.log(5=== “5”);//false

Whenever we go to validation in JavaScript we go to strict equals to.

**Ternary operator:**

Syntax:

(condition) ? true statement : false statement

**Conditional statements**

**1.Simple If**--> accepts only true statements

If(condition){

-------true statement-----

}

Ex:

If(5<10){

Console.log(“hello world”);

}

Output:hello world

1. **If -Else**

If(condition){

------true------

}else{

-----false----

}

Ex;

If(5<10){

Console.log(“hello”);

}else{

Console.log(“bye”)**;**

}

Output: hello

1. **Else if ladder**

If(condition){

}

else if(condition)

{

}

else if (condition)

{

}else{

}

Ex:

Let a =20;

Let b = 30;

Let c = 40;

If(a>d&&a>c)

{

Console.log(“a is greatest”);

}

Else if (b>a&&b>c)

{

Console.log(“b is greatest”);

}

Else{

Console.log(“c is greatest”);

}

Output: c is greatest

**4.Switch:**works on cases-break,continue,default

Switch(condition)

{

Case:

{

}

Break;

Case:

{

}

Break;

Default{

}

**}**

Ex:

Let n = 1;

Switch(n){

Case0:

{

Console.log(“Sunday”);

}

Break;

Case1:

{

Console.log(“Monday”);

}

Break;

Case2:

{

Console.log(“Tuesday”);

}

Break;

Case3:

{

Console.log(“Wednesday”);

}

Break;

Default{

Console.log(“invalid”);

}

Output: Monday

Ex2:

Let n = 10;

Switch(n){

Case0:

{

Console.log(“Sunday”);

}

Break;

Case1:{

Console.log(“Monday”);

}

Break;

Case2:

{

Console.log(“Tuesday”);

}

Break;

Case3:

{

Console.log(“Wednesday”);

}

Break;

Default{

Console.log(“invalid”);

}

Output: invalid

**Day4:**

**Looping:**used to iterate a statement based on condition

1. **While loop**:is having zero iteration .if it wants to iterate it should satisfy condition.it iterates statement until condition becomes false.

While(condition)

{

------statement-----

}

Ex:

Let n=1;

While(n<=5)

{

Console.log(n)

n++

}

Output:1,2,3,4,5.

1. **Do while loop**:having one default iteration.it iterates set of instruction then it will checks condition

Do{

Statement

}while(statement)

Let m =1

Do{

console.log(m)

m++

}while(m<=5)

1. **For loop:**

Syntax:

For(initialization;condition;increment/decrement)

{

Statement

}

Ex:

For(int i =0;i<=5;i++)

{

Console.log(i)

}

Output:0,1,2,3,4,5.

1. **For each loop:**

**Block scope:**

Console.log---> console is object &.log is a method inside console which is accessed using dot operator

Var is undefined so it goes under global scope.

Ex:

Var a = 10;

Console.log(a);// here a goes to global scope because of var

**FUNCTIONS IN JS**

Functions are first citizens in java script .

Functions are objects in java script

The main purpose of using functions in JavaScript is code reusability.

**Types of functions in java script**:

1. Function declaration statement or pure function or named function
2. Function expression
3. Higher order function,callback function
4. Arrow function[lambda function in java]
5. Nested function
6. IIFE[immediate invoking function expression]
7. **Function declaration statement or pure function or named function:** set of instructions or block of code used to perform specific task.hoisting is possible ,hoisting in the sense calling function before declaration of function.

Syntax:

Function fun-name(parameters,……………)

{

}

Fun-name(arguments,---------------)

1. **Function expression:**

Assigning function as a value to one variable.

It is used to perform specific task.

Object in javaScript:

Used to store the data in the form of key and value pair

Asynchronous function:making a way to execute other function

When one function in running in call stack and all remaining functions are in hold.

2 methods to convert any function to asynchronous:higher order method

1.set timeout(callback fun,delay time)

2.set interval(callback function,delay time)

These methods present inside window object

Asynchronous function will not work in every situation.

Promise :it is an object ,used to look after asynchronous function.

3 phase:

1. pending phase
2. Resolve/fulfill phase
3. Reject phase