Tutorial-3

1We can assign variables to functions.

For example🡺var navtesh=function(){

Console.log(‘hi navtesh’);

};

So if I call, navtesh() the given function will print out, it is not nessasary that everytime we need to name the function like function navtesh(),we can do as in the example, such functions are called **anonamous functions.**

**setTimeout(navtesh,5000); 🡺 it will run after 5sec of the last statement**

**one important point to be seen here is that, as we can assign functions to a var to functions, we can also pass functions to other functions, as we pass variables to functions**

**in ” setTimeout(navtesh,5000); “ navtesh is a function passed to the function setTimeout**

**TUTORIAL-4**

**Soooooooooooo?**

**setTimeout(navtesh,5000); 🡺 what is this function??????what this function does is that when**

**it just schedules the execution to be done after 5sec and goes on to the next statement immediately.**

**Tutorial-5**

**Everything is a reference.**

**Difference between == and ===**

**== 🡺 compares only the value**

**===🡺compares values and data type**

**Tutorial-6**

**var navtesh={**

**fav:function(){console.log('my name is navtesh');**

**console.log(this===navtesh) //true**

**}**

**};**

**navtesh.fav()**

**this refers to which object is calling that function**

**so in navtesh.fav() which object is calling the fav function??? navtesh , so this ===navtesh.**

**But……..if it isnot called by any of the objects then what is this ===?**

**function fav(){console.log('my name is navtesh');**

**console.log(this===global); //true**

**};**

**fav();**

**in this the function fav is not called by any object so, this ===global.**

**Tutorial-7**

**What is prototyping????**

**player.prototype.punch=function punch(opponent)**

**it is like list.append in python, it adds the method punch in player class.**

**Tutorial-8**

**Moduels , import/export**

**Breaking the code in different files called modules**

**Export🡺 module.exports.name=functionName;**

**Import🡺 var varName=require(‘./ModuleName’);**

**HOW to use it in the imported file🡺VarName.name()**

**Tutorial-9**

**One more way of exporting:**

**What module.exports.name=functionName; does is that it adds the name:functionName to the module.exports={} object.**

**But we can directly add the name and functionName to module.exports={}**

**object. module.exports={**

**name1:function(){}**

**name2=””;**

**.**

**}**

**In the file where we import the other file, how do we access the functions, variable .etc,??????????????**

**var varName=require(‘./ModuleName’);**

**now the varName contains all the functions, variables of the imported file.**

**Name2 is a variable defined in the imported file, varName contains all the functions, variables of the imported file.**

**So varName.Name2=”sometext”; is how we access that.**

**Tutorial-10**

**Every file that imports a module shares it, they don’t get a copy of their own.**

**Tutorial-11(object factory)**

**But if you want to have a copy of your own then just make an object in the module that makes other objects.**

**Tutorial-12(core modules)**

**Some modules come built-in with node.js🡺core**

**In core we don’t include the path, the varName equals to the module name(just a practice)**

**Some coremodules🡺**

1. **var fs=require('fs');**

**fs.writeFileSync("path","data");**

**this statement creates a file**

**path=file.type**

**data=contents of the file**

**console.log(fs.writeread(‘file name’).tostring)**

**function to read a file as a string**

**2. var path=require('path');**

**var websiteHome='desktop//navtesh/yo.html';**

**console.log(path.normalize(websiteHome));**

**console.log(path.dirname(websiteHome));**

**console.log(path.basename(websiteHome));**

**console.log(path.extname(websiteHome));**