Make your web page alive

Write in html page of the application called scripts runs automatically when you application loads

Live script

Java was so popular on those days the makers of this language decided to make it as younger brother of java

[ECMAScript](http://en.wikipedia.org/wiki/ECMAScript) is standard for the JavaScript

**String and variable:**

Let: reserved word in js

Define a variable

Any thing you add it to string and result will be string

Compare “” with null returns false

Append any text to Null, undefined, Nan we get this special fields as value + our text

**Js comments:**

We use // indicating the start of comment anything follows it will be ignore in the code.

**Number:**

Add number with undefine, NaN you end up getting NaN as answer.

Append number to text you get text

More on variable:

1. Can’t define same variable name multiple times
2. Variable name: variable should always stats with letter, $, \_ no other special character are allowed
3. Cannot use language reserved keyword.

**Boolean & logical operator:**

=== equal

!== not equal

< less than

>Greater than

<= less than equal to

>= greater than or equal to

As well we can use if statement for condition check:

Advanced if statement:

If()

{

}

Else if()

{}

else

**Logical operators:**

&& logical and

|| logical or

**Variable scope in javascript:**

Lexical scoping (static scoping)

Variable shadowing:

Basically define a variable in local scope/ container whose name is same as variable defined in global scope or its containing container with different value

Such variables are called as shadow variable.

In container tree structure when any variable is referenced. JavaScript engine traverse up the hierarchy to check if there any variable found with same name this process continues until it reaches the global scope.

Leaked global:

When the code is referring to variable which was never declared. when the traversing reached the root and it doesn’t find the referred variable it makes a declaration for that variable in root global level with assigned value.

Functions:

Let nameOfcunction = function()

{

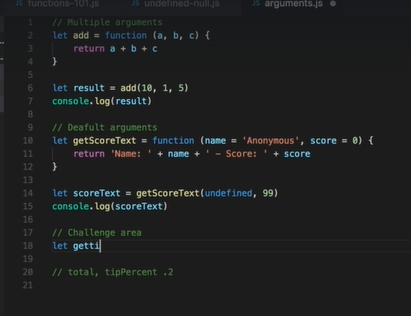
}

Undefined & null;

Undefined: means value is not available for the variable. It was never assigned with the value

Also explicitly loose the value of the variable by assigning it with undefined.

**Multiple argument and default arguments for the function:**



Similar to c# assign default value with variable in function definition

Template string: make string template so that part of sting is static and part is dynamic

To use this my string will have the format something like this

`String ${variable1},${variable2},${variable3}`

Objects in JS:

Used for modelling real-world entity

For example Note consist of body and title

Simple object in JS is represented as json object with variable assigned.

let person = {

    name:'Jhon',

    age:27,

    city:'Philadelphia'

};

Objects to and from function:

let tempratureConverter = function(fahrenheit)

{

  return {

      celsius: (fahrenheit - 32) \* (5 / 9),

      kelvin: (fahrenheit + 459.67) \* 5 / 9,

      fahrenheit: fahrenheit

  };

}

let temp1 = tempratureConverter(33);

console.log(`In fahrenheiht=${temp1.fahrenheit}, In celsius=${temp1.celsius}, In kelvin=${temp1.kelvin}`)

let temp2 = tempratureConverter(34);

console.log(`In fahrenheiht=${temp2.fahrenheit}, In celsius=${temp2.celsius}, In kelvin=${temp2.kelvin}`)

let temp3 = tempratureConverter(35);

console.log(`In fahrenheiht=${temp3.fahrenheit}, In celsius=${temp3.celsius}, In kelvin=${temp3.kelvin}`)

Object reference:

When you passing object to a functional basically you are making a reference to the same object

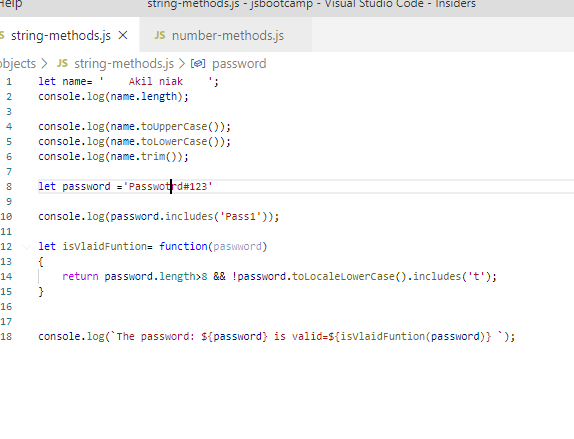
So modifying the object property value reflected in all the places where the reference is used

Say If I change the entire object then this changes will reflect only in the scope or the reference object created for it will not affect the main object in use.

**Object methods / function properties to the objects:**

Object method is nothing but a property inside the object whose value is a method.

**String in js and methods associated with String:**



Number methods:

We have extension methods on number which an be used or

Use Meth library to use function on numbers.

**Different ways of defining variable in JS:**

Let based

Let isRaining=true;

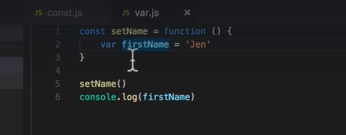
Constant:

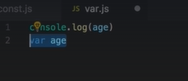
Const isRaining=true;

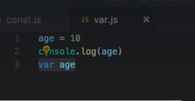
We can also define an object as constant. After defining we cannot changes the reference =, But we can manipulate the object property value.

Var based:

1. Possible to declare same variable multiple times with same name
2. Function scoped not block scoped.
3. Use variable defined after words will work fine because this will hoist the variable declaration at the top.







**Arrays in js:**

Store list of values

In js array represented as []

Let list=[]

Individual item is accessed with index value starts with 0;

Push method used to insert new item t the end of the array.

Pop will remove the last element from the array

Push pop operate at the end of the array

Shift unshift does the same operation as above but at the beginning of the array this operation will takes place.

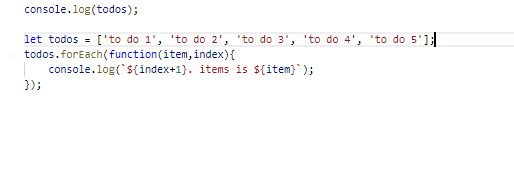
Splice will modify array from the between

Splice(index where to start, how many items to remove, new item to insert)

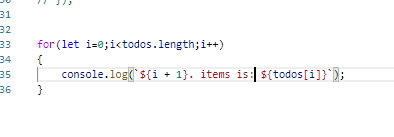
For insert remove item count is zero

**Iterating over array using for each loop:**

So to iterate here we need to put a call back function to foreach loop as below.



For loop similar to loop in other programming language where we exactly know how many times the loop has to execute.



Array of object & searching:

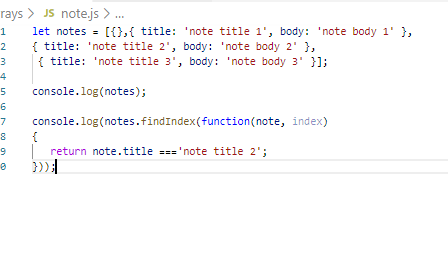
Indexof method

notes.indexOf('note33')

this will perform the equality check on the entire object to be searched so this will work only with primitive value.

Another extension method called

finIndex on array make a call back to a function to get the [position of the object we are looking at with custom rule inside the call back method.



We also have another method called find with similar call back method that will directly return the math object

Arrays always passed by reference so any changes done to it will reflect both local and passed in array.

let findObejct= function (notes, titleTofind) {

  return  notes.find(function (item) {

        return item.title === titleTofind;

    });

}

**Filtering in array:**

Based on some criteria we filter the data from the array and return the subset of the array from the main list of array items.

Method used is filter that iterates over the arrays and call a call back function to put custom logic.

const filteredNote =  notes.filter(function(note,index){

    const isBodyMatch = note.body.toLowerCase().includes('rr');

    const istitleMatch = note.title.toLowerCase().includes('rr');

    return isBodyMatch ||istitleMatch;

});

**Sorting in array:**

For the primitive data types the data will be sorted by default in alphabetical order. Also we can have a custom call back function to have custom sort rule

Sort takes 2 parameter sting1, string 2for comparison

Values:

-1 : first string is less

0: both string are same

1: 2nd string is small

const sortTodo= function(lsttodo)

{

   return todos.sort(function (a, b) {

        if (a.text.toLocaleLowerCase() < b.text.toLocaleLowerCase())

            return -1;

        else if (a.text.toLocaleLowerCase() > b.text.toLocaleLowerCase())

            return 1;

        else

            return 0;

    });

}

**JS in browser:**

While installing node js npm will be installed by default

$npm -v gets the version

Live server: command used to install this is

npm install -g live-server

once the installation is done check the version of the server

live-serevr –version

now creating the web site and generating url to access it

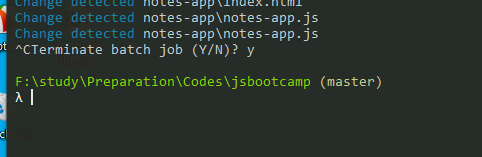
$ live-server app folder name

$ live-server notes-app

When the application launched the live serve will be on

We can shit down the live server by clicking on Ctrl+c

Then hit yes.



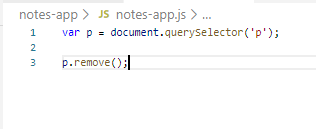
Js in browser

DOM: document object model

For html document Is DOM

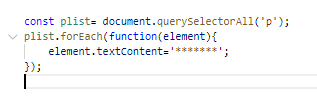
In js Document is a object mapped to the HTML document where it runs

Javascript object using which we can manipulate the html document.

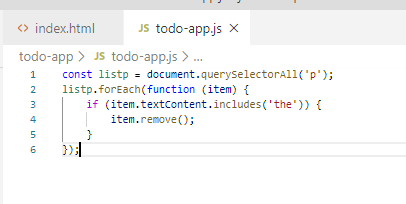


Select first occurrence of the p tag and removes it from the html document.

Suppose if you want to get all element.



Js to remove all p with the text



Add new element using DOM

First create element

Then append it to parent element

**Handling user interaction:**

Registering the event for the action

Document.querySelector(‘’).addEventListner(‘click’, function(){});

**Advanced query selector for selecting the element.**

document.querySelector('#btnCreate')

document.querySelectorAll('.note').

**Input and live data filtering:**

document.querySelector('#txtNote').addEventListener('input', function (e) {

   console.log(e.target.value);

});

On even we can have ‘change’ event

Forms with javascript:

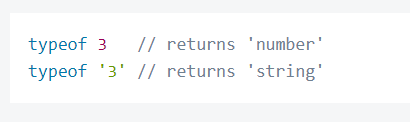
String in Js is immutable once you assign a value you cannot change the character value. Full text value should be changed

**Operator used to check the type of the variable:**

Typeof ‘a’

Typeof 1

typeOf true



Object property can be acces using . or [] operator

Adding deleting property from and object:

Mydog{

Name:’’,

Legs:4

}

Dynamically add property

Mydog.tail=1;

Mydog[tail]=1

Removing the property:

Delete Mydog.tail

Also can check if the object has some property in it

.hasOwnProperty(prop name)

**Random function to generate random number:**

Math.Random() generates random fraction between 0-1

parseInt(‘’) parses string value and generated the integer value

2nd argument to this number is radix which basically indicates the base

Local storage:

To access the local storage we need to use the global javascript object

localStorage:

Methods supported for crud operation

Insert:

localStorage.setItem(key, value);

Select:

localStorage.getItem(key)

Delete:

localStorage.removeItem(key);

to clear everything from the local storage we need to run a command called

LocalStorage supports only string based data. So there Is need to convert from other type to string and from string back to other type.

localStorage.Clear();

So objects can be converted into string format using Jason parser

Jason.stringify(pass object)

Getting object back from the storage and convert back to object

Use JSON.parse(pass string to get object)

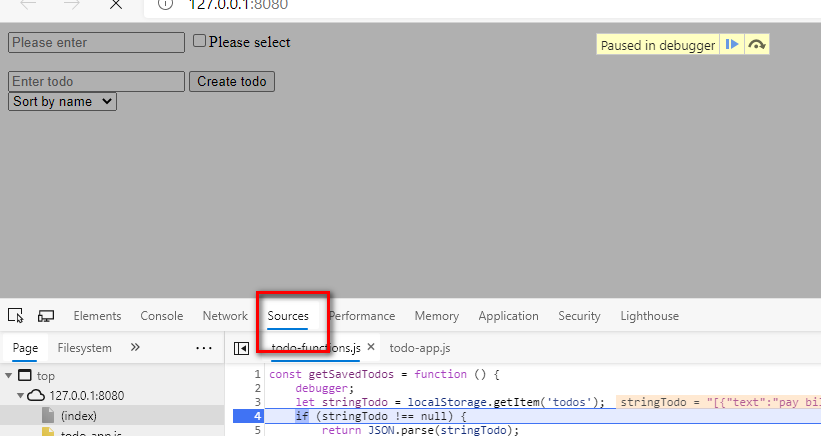
**Debugging js:**

Console.log display data and try

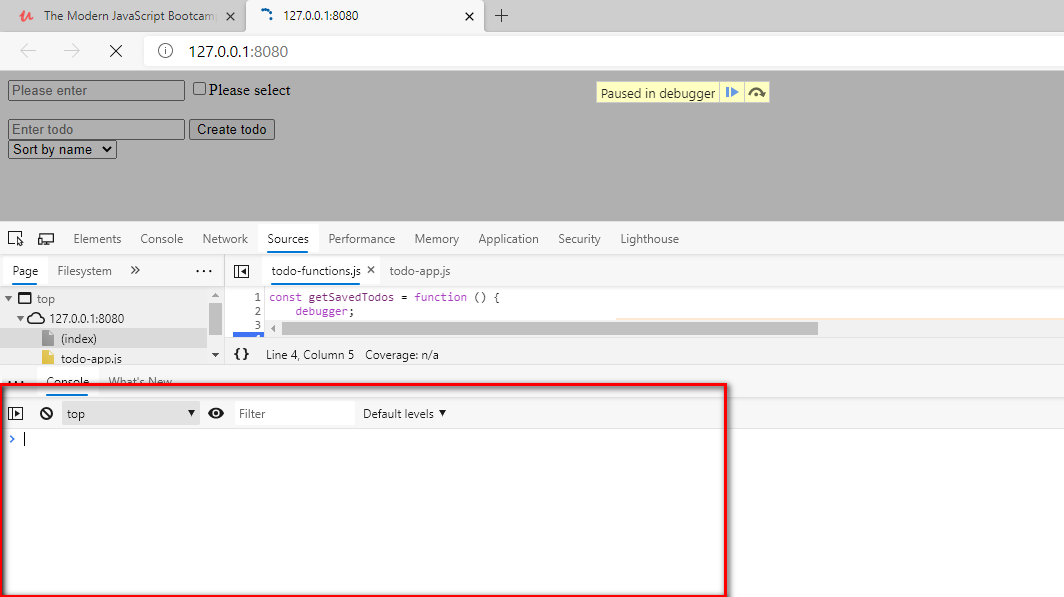
2. Debugger statement

We need to open developer tool and load the application

Got to source tab



To get he console as split screen here press escape button



Dynamically add checkbox to the screen:

 let chk = document.createElement('input');

    chk.setAttribute('type','checkbox');

Setting up third party library:

node UUid: universaly unique identifier 128 bit value .

called node because initially when developed it was only supported in node based script

Either load from CDN

Or use a local copy of the the script from

links.mead.io/uuidv4

There is a single method that we need to call to get unique id

Uuidv4();

Another object used to navigate between the pages

Is location object similar to our document object

Location object has all the information about the url that we are dealing with.

Add a special methods using which we can redirect to another page.

Location.assign(url)

When the redirection happened fort he edit basically we should be knowing

Which record is being edited for this we need to send a record id as a part of the url

In js its separated by #id

Using location.hash we can access this value.

Es6 concepts:

ECMA script is a standardized version of the javascript.

Goal is to unify the language features and specifications.

Most recent version is ES6 add powerful features to programming language.

1. Class
2. Let, const
3. Arrow functions
4. Promises
5. Modules
6. Generators

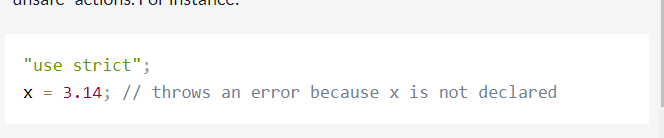
Var and let:

1. Var is not strict declaration we can redefine is multiple times in same scope. This will not throw any error . So in large application you might end up overwriting something unintentionally.
2. Scope: local and global cope are same, but id the variable is define inside the block of code scope ends there in le++t but not in var.

Strict mode:

“use strict”

With this statement we are making our code to work in strict mode. That means we cannot use a data without being declared.



Const has all the feature of let but with one extra thing I.e value assigned to it is constant we cannot assign value multiple times once its being assigned.

Most of the time const is used in place of let

Eg: functions and array where we might change the definition or insert new value to array but not redefining the object reference

Prevent JavaScript object from mutation:

This can be done with JS freez method. With that you cannot add, remove property to object and modify value from the object

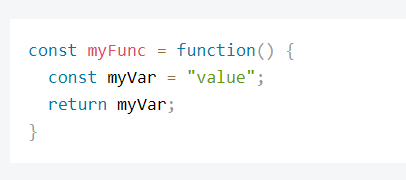


Arrow functions ():

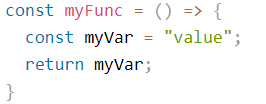
Write anonymous function with that

It not compulsory to name a function always specially when you pass a function as parameter to another function. Instead create inline function

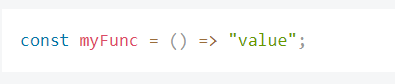
Arrow function is a syntactic sugar over inline functions. Where we define function without using function keyword.



With inline this can be written as

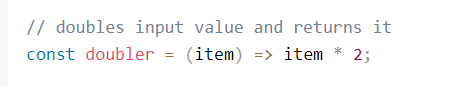


When there is not multiple statement inside the arrow function and has only return statement then we can even omit the brackets and returns statement.

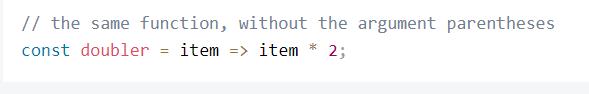


Basically this can be used as accessor to return some value and for simple inline function with inline operation that returns a simple value.

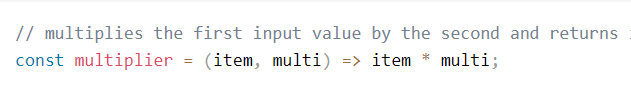
Arrow functions with parameters:



If arrow functions has single argument we ca omit the parenthesis as well.



We can have multiple parameter as well.

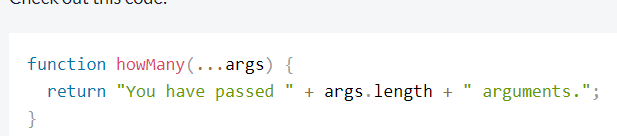


Rest parameter to function:

With this we can pass as many parameter to the function. That is then saved inside the arrays and used in function.

It eliminate us to check for array and allow us to apply operation like,

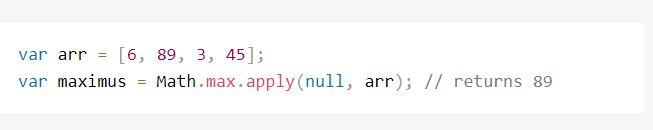
Map, filter, reduce.





Spread operator for the array:

Es5:

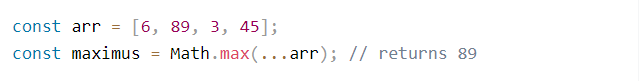


Es6 we use:

Math.max(arr) but this will ail returns NAN because it expects string as parameter to the function.

So try to use spread method to achieve this

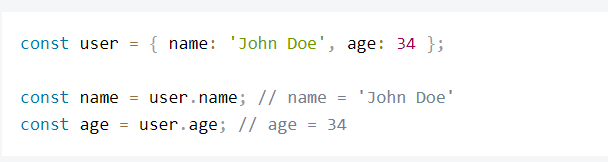
This can be used only with inline function we cannot use it as stand alone method.



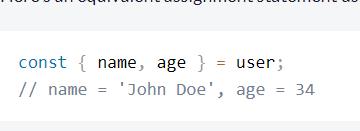
DEstructuring assignment:

Used to extract the value from object and assign it to some other variable

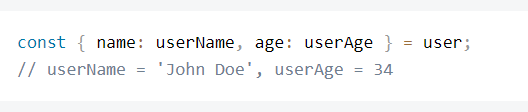
Basically read value at once from object property and assign it to multiple variable,

ES5 

ES6:

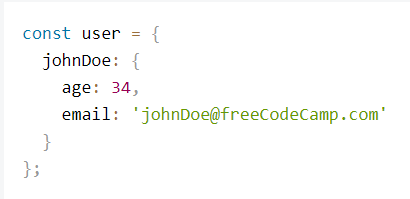


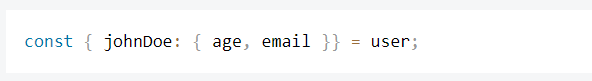
Give custom name to after Destructuring the value red from the object:



Nested object destructuring:

Object looks like this.

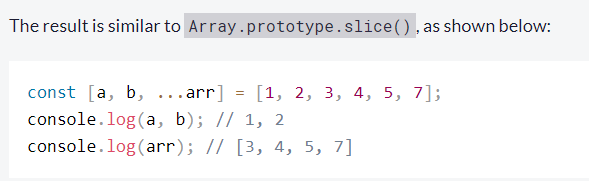




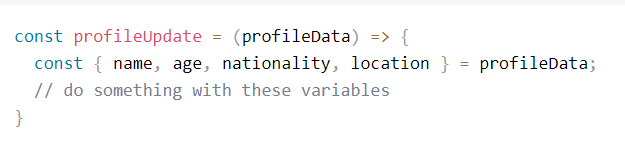
Custom name:

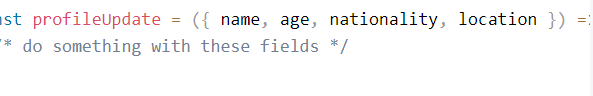


Use rest variable to assign elements.

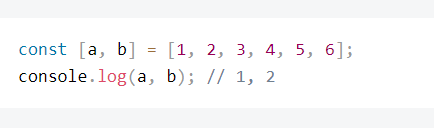


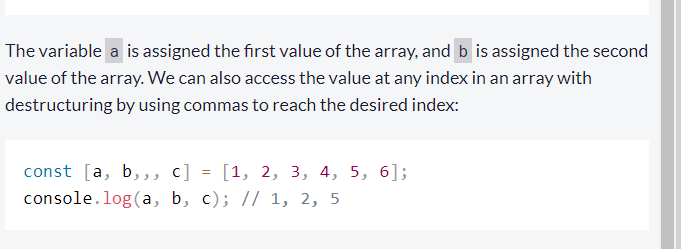
Destructuring data by passing object s parameter to the function:





Destructuring assignment of the array variables.



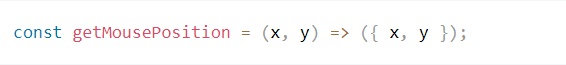


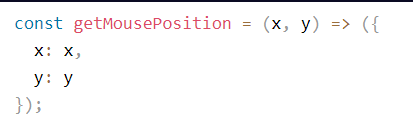
String with templated litters:

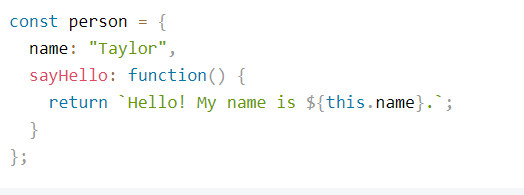
With this we can have string with place holders

` string text ${placehilder1} ${place holder 2}

Inline function that return object





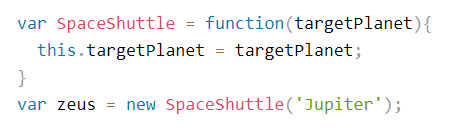


ES6:

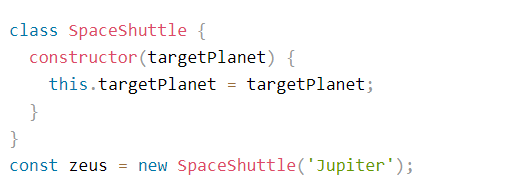


Constructor functions:

ES5:



ES6:



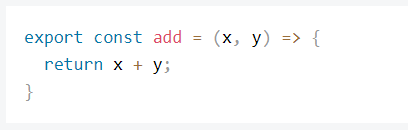
Getter & setter:

Using this we can get and set values to an object properties.



Exporting methods from js

ES5



ES6



So this will export only such methods from the js file

Importing will import the specific methods from the JS file.

Import {method1, method2} from ‘./ js file location’

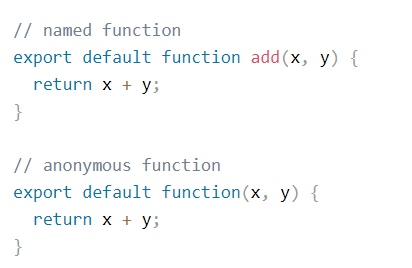
Export import is used to share the methods from certain file to another file so that w can have code sharing. This ill help in code centralization

Suppose if want to import all methods from the JS we need to use function like

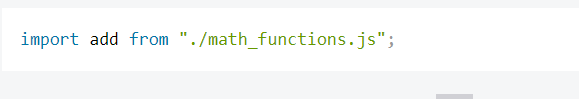
Import \* as someObject from ‘./jsfilename.js’

So with this we can use the .js file name methods with this reference object

export default



Used as default call back



Promise objects:

As a name suggest it promises to do something

Constructor function that takes function as parameter that does the things asynchronously

Constructor takes two arguments basically indicates what has to be perform to fulfil the promise

Resolve: do something when completed

Reject: do something when not able to process promise

Every promise has 3 state

Fulfil: completed

Reject: not able to process

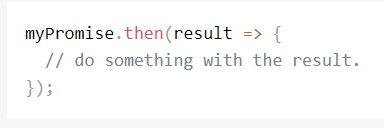
Pending: still running

Promise is useful when a process take lot of time for execution.

Basically we can do asynchronous operation with the promise object

Define promise with promise constructor which takes function as parameter.

Once define object then we have to use then operation on that object



Result always comes from the resolve parameter passed to resolve method.

Also we have method to handle error form reject function



Regular expression:

Special string represent the search pattern.

Object used id Regex or Regexp.

Used to search text and replace characters.

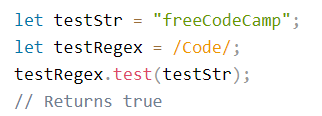
It’s a cryptic text contains few special character for helping the match.

Suppose if you want to match ‘the‘ in string regular expression could be /the/

Regex can be used in may ways

One way is .test() method

Returns true or false based on the match



Method is type sensitive

Suppose you want to match the multiple word then use | for each separated word.

Patter matching with ignore case

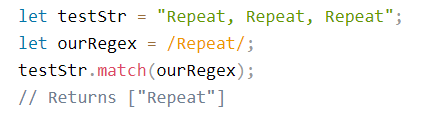
/text1|text2/i

With test we can check if the matching string is there in the text

Suppose you want to return the match out of the matching operation then we should use

.match() operation

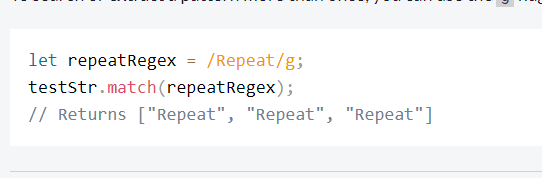
Syntax of the match operation is opposite of test



Returns first occurrence of the matching text

Suppose if you want to return the multiple

/text/g



/text/gi multiple words with case insensitive

Pattern to be matched will always to be put between //

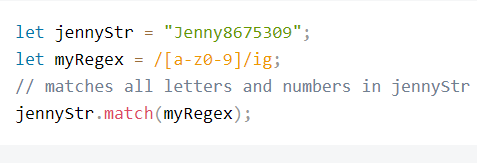
Wild cards for patter match

. (dot period) used to match single character (match anything)

[] place characters you want to match inside this bracket this indicates the character set you want to basically match.

Range to match inside character set

[a-e] this will matches character between a to e



Suppose you want to negate the operation. Ie is you want to ignore set of characters from matching

User ^ carrot:

Match character that appears one or more number of times in a row

/a+/ig match one or more occurrence of a

Suppose you want to match the string that begin with some text you can do matching like this.

Use carrot in the beginning

/^text is good/

Match for so thing that ends with the text:

$ (dollar sign)

Alpha numeral pattern match \w

\w [0-9a-zA-Z\_]

\W [^0-9a-zA-Z\_]

For digit pattern match we use the \d [0-9]

Negation of this is \D [^0-9]

Matching space \s \S negation

Carriage return \r

Tab \t

Form feed \f

new line \n

Specifying the occurrence of the character certain number of time.

/a{3,5}h/.

In above patter it will match the word where a repeats 3 or 5 times between a and h

Where 3 is lower range 5 is higher range

{3,0}

Minimum of 3 occurrence and maximum could be anything

Exact match {3}

Check if the patter may or ma not exists. For this may be we need to use the symbol ?

A?a

Match zero or one occurrence of the A

Search and replace with pattern match

String.replace(regex, new text);

Check the type of the element used

Typeof “”

Typeof ‘a’

Typeof 12

Typeof []

Typeof {}

Data structures in JS:

Arrays: collection of the data

Array accessed using single index is called one dimensional array.

Insert element to array from end use push()

From beginning use unshift()

In js array is mutable

Array.push() and Array.unshift().

Remove element from front unshift()

From back pop()

Access element from any position we have to use splice method

numbers.splice(startIndex, amountToDelete, 13, 14);

startIndex from which index you want to start

amountToDelete number of element to be removed

preceding value tells the items to be inserted in same position.

Copy array elements

Slice(start index, end index)

It just gets and copy the data to target will not modify existing array

Copy with rest operator of es6

Arra=[]

Var newarr= […Arra];

With this we can copy a content of one array to another array and at any given position.

Finding the position of the element.

indexOf(‘text’);

If element found returns position if not -1

Iterating over the method

every(),

forEach(),

 map()

for loop

remove property from object

let food ={a:a,b:b}

delete food.a this will remove the a property from object

check if the object has mentioned property.

1. hasOwnProperty
2. using in ‘property’ in object

looping over the property of the object

done with the help of for in

user= {a:’asds’, b:’dsfdsf’}

for(let a in user)

{

Console.log(a);

}

Prints a,b all key values.

let users = {

  Alan: {

    age: 27,

    online: false

  },

  Jeff: {

    age: 32,

    online: true

  },

  Sarah: {

    age: 48,

    online: false

  },

  Ryan: {

    age: 19,

    online: true

  }

};

You want to return all the keys from this object then use a object method

Object.keys(users);

Constructor:

A special function that issued to create a new object

Makes object property and the behaviours associated with it

And returns an object.

So we can treat it as a template/blue print for object creation.

Conventions used here:

Uses capital letter for function name to differentiate itself from other function definition

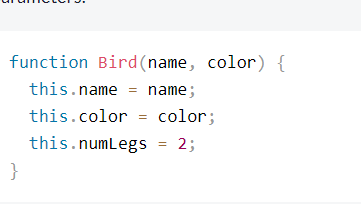
All the property inside the constructor will uses this key word that indicates the new object being created.

Function created new object rather than returning the value as other normal functions do.

Creating a class with function as follows:



Constructor with argument:



Verifying if the object created from a particular constructor

Use: instanceof

A instanceof type



Own properties:

Properties defined inside the function constructor are called as Own properties.

Because each object instance has its own copy.

Looping through each object property

For(let prop in object)

{

}

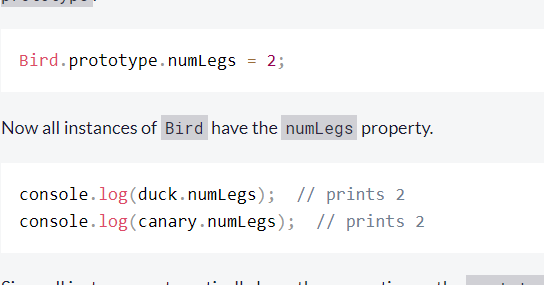


In the example of car object number of wheel value and property is same

Birds: number of legs is same and common value so when you create thousands of object same copy is maintained multiple time in a memory with separate objects.

So to eliminate this and maintain the single copy of this we uses something called proto types.

That maintains a single copy shared by all the objects.



So it’s a -art of object constructor and every object in JS has this property value.

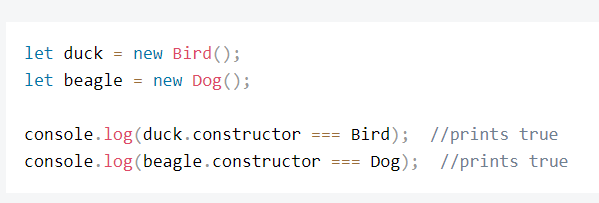
Proto type basically act as recipe for creating an object

Own: defined directly on the object instance level

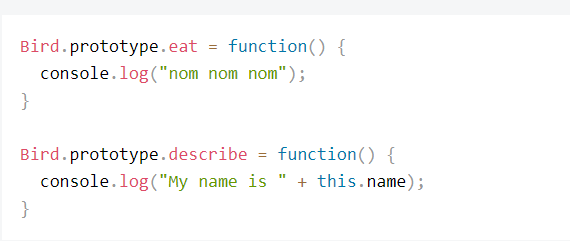
Prototype: defined at the prototype level

There is a special property on the created object

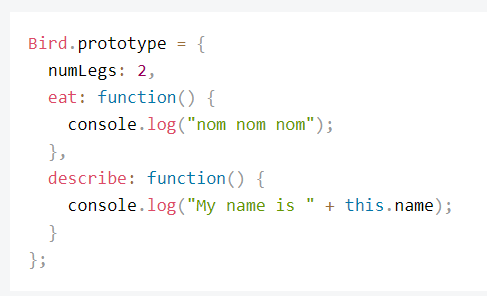
Constructor property is a reference to the constructor function being created.



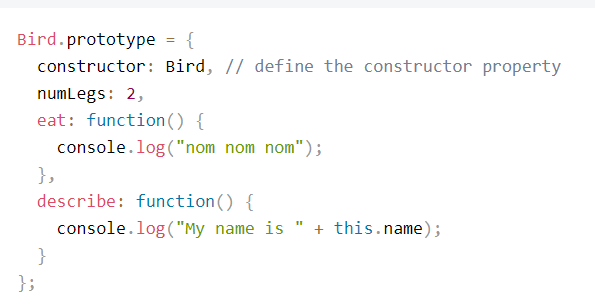
We can as well add new function to prototype



Adding all property and methods at once:

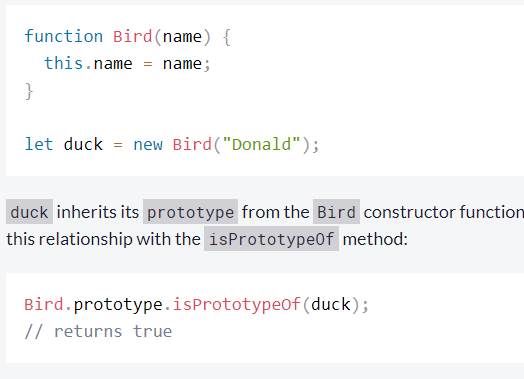


There is one side effect of manually setting prototype to the object. It erases the constructor so we need to manually reset it back from the code.



Like person inherits genes from their parents similarly. Objects inherits its prototype from constructor function that creates it.

Check can be doe with function isProtoTypeOf



Every object in js has its own prototype. In root level Object.prototype is prototype of all prototype.

hasOwnProperty  defined in Object.prototype,

which is accessed by Bird.prototype which then be called by duck

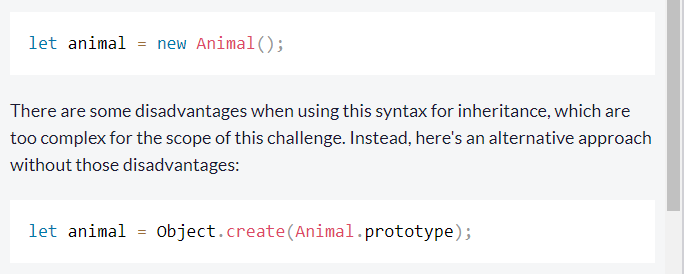
Code sharing with prototype:

Create a super type and move all the common code into that object.

When assigning the prototype object it erases the Constructor.

Inheritance to reuse the code from another master object defined

Another way of creating the object instance is

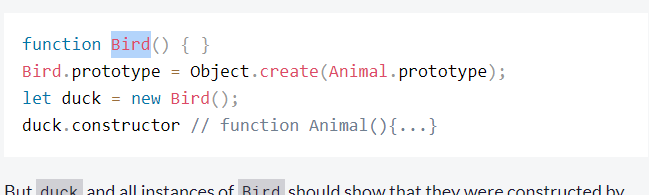


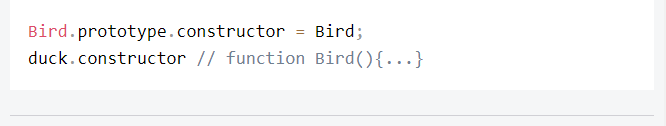
For inheritance set child object prototype as a instance of parent object



Prototype is ingredient now the bird recipe has all ingredients from animal

While assigning the prototype of parent to child prototype it will also reset the constructer of child which need to be reset to the child manual





Own methods inside apart from the inherited methods:

For this assign the prototype of the child class with new method after the inheritance.

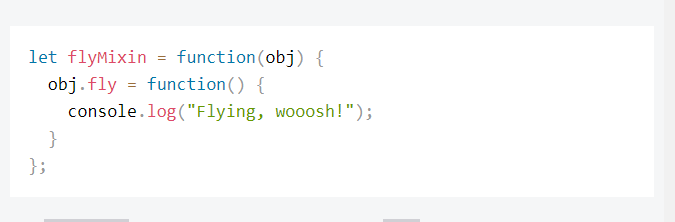
With that all the object created with this constructor will get this method

Even it possible to override the method from the parent class.

TO the child object prototype you assign a method with same name it will override the methods from the parent.

Mixin to add common behaviour to object

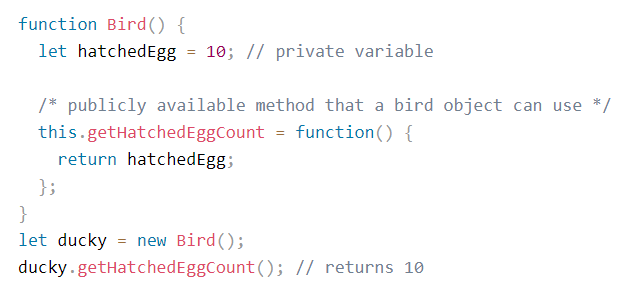
We use principles of dependency injection here to achieve this.





Protect object variable from external access so nobody modifies it this can be done with closures.

Define local variable inside the constructor function and method to return the value.



JS IIFE

The function that is called on load of the JavaScript is called immediately invoked function expression.

(function(){body of the function};)();

Core principles:

Independent of object and the state of the global variable

Minimal side effect in the program

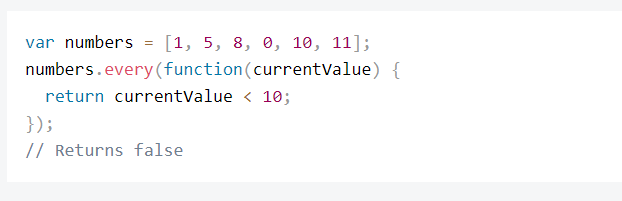
Limit the changes to the state of the program object and global variable.

Functional programming is style of programming where entire application is divided into small testable functions without ant side effects.

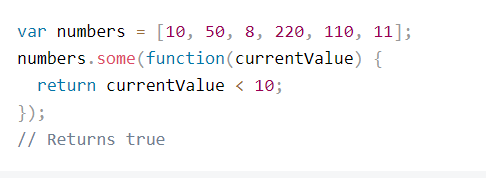
1. Each function is independent of other function and sate and value of the global variable.
2. Pure function where it same input gives same output
3. Limited side effect

Array method to check if every or some method meets a specific criteria.

Every



Some



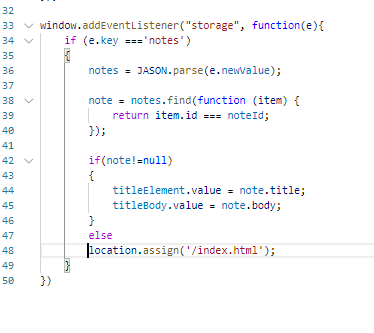
Say a same list screen opened in multiple tabs in same session. Suppose a tab modifies a data in one tab that need to be reflected in all tab what can we doe for this

We need to sync the data in all the tabs.

So we need to listen to the changes done to local storage

Here we are using the window object

Storage is a event triggered on the window object



Javascript date:

Use Date object

**Let now = new Date();**

Making a custom date can be done with constructor function as below

‘Moth day year time’

**Let date = new Date(‘January 21 2018 6:24:01’);**

Instead of representing the date like this we can represent it as a number using the time stamp property.

**Let now= new Date().getTime()**

It generated some number

Getting date value back from this time stamp

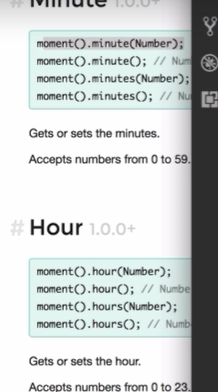
**Let newDate = new Date(now);**

Time stamp for old date will have lower numeric value

Time stamp for latest date will have large value

**Moment:**

JS also has library called moment to deal with date time



Property inside the object are methods ones with no parameter are called the getters

One with the parameter value are called setters.

Creating object:

**Let now =moment();**

**now.toString()**

Generate time stamp()

**Let timestamp = now.valueOf()**

Getting value back from time stamp

**Let date= moment(timestamp);**

Formatting the value:

**Moment().format(‘DD MM YYYY’);**

We have different value associated with this

Also we can add or subtract value from the date

Arrow function:

Function usually used as an single line function or shorthand representation of the function

Arrow function

const square = (num)=>{return num\*num};

Short hand representation:

const square = (num)=> num\*num;

here we don’t include the body inside the flower bracket so whatever expression goes into arrow function is the return expression

Arguments object of the function

We can call methods with list of all arguments to the functions. Need not add formal parameter to the function

Access to this object available only in old way of defining the function

**Conditional operator**

a>b?true:false

**Truthy and falsy values:**

Expression involved the Boolean context where it result into true or false value.

Null, 0, undefined, ‘’ all result into false value.

**Type Coercion:**

Converting the type based on the situation

‘5’+5= 55

‘5’-5=0

So based on the available best option the conversion happens.

Coercion happens a string, a number, a Boolean

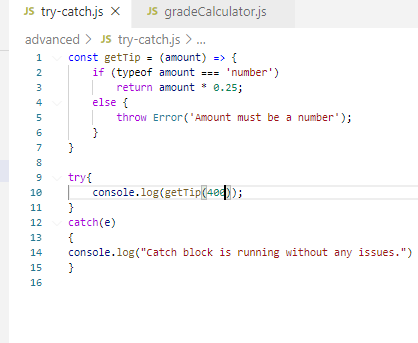
**Exception handling:**

To throw any error occurred in the application we use a expression

Throw ‘text value to indicate the error message’

Or

Throw Error(‘error message’);



Handling application error:

Strict mode in JS:

Eliminates all the unexpected behaviour and eliminates error from the user code

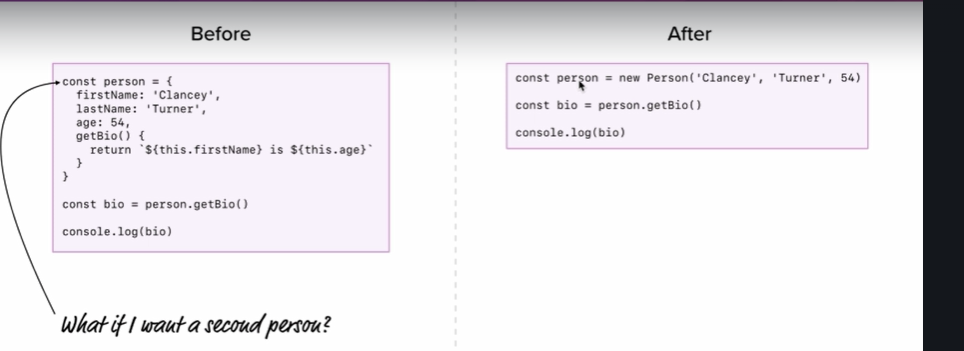
‘use strict’

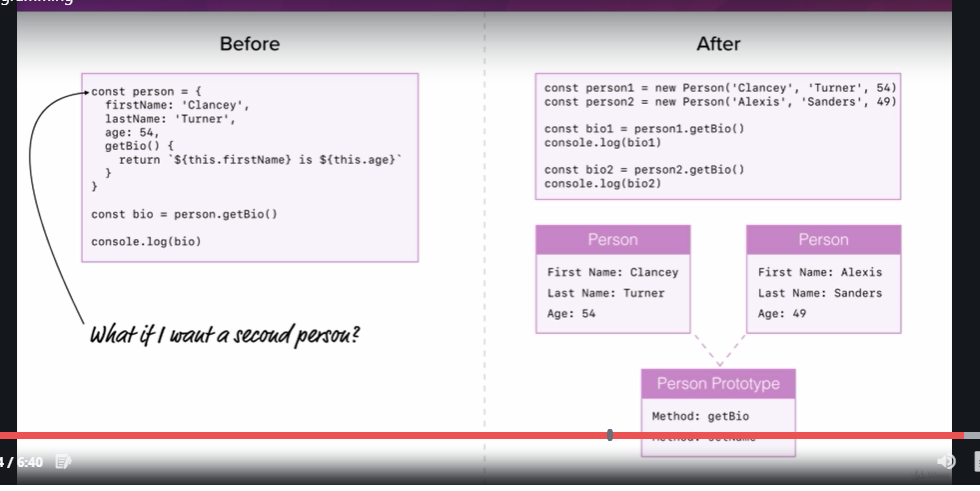
Add this line in the beginning of the js file this will by default load the file in strict mode.

It disables old feature which will be removed soon from JS.\

Prevents developer from using such codes.

Object js:





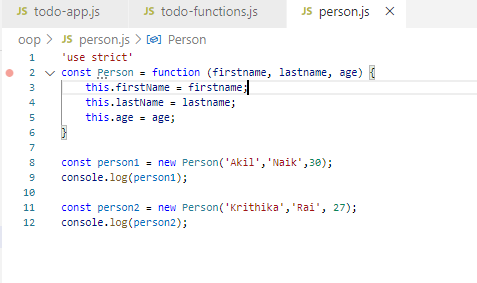
**Constructor function:**

Function used with new keyword is called constructor function.

Function name should begin with capital letter

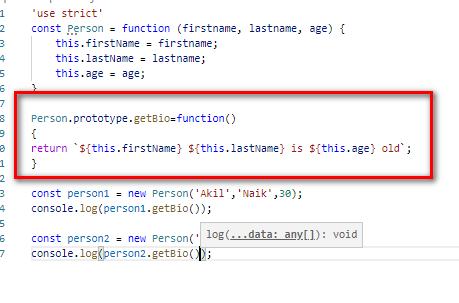
And this act as template or blue print for crating object of that type

All property inside it would have this keyword attached to it.



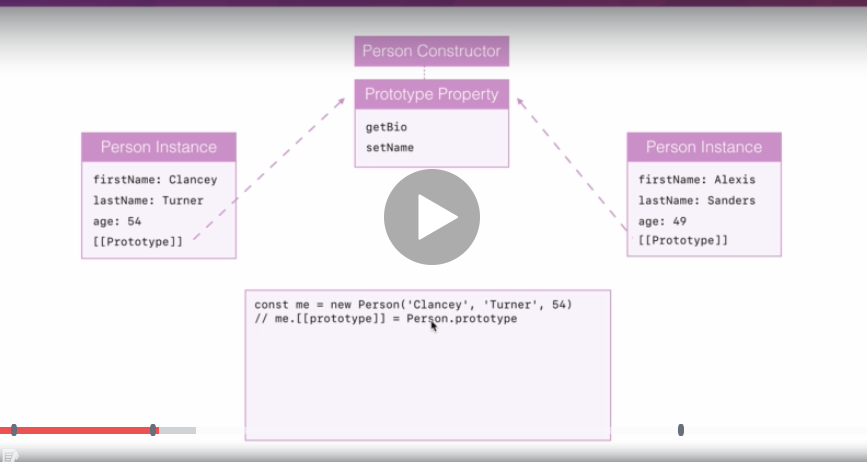
**Prototypal inheritance:**

Suppose the instances of the constructor function want to share a common code then we put all such thing inside the prototype of the constructer object



Also we can have static data shared across all object inside the property added to the prototype.

All the property associated with the methods are accessible inside the



When we create new object instance using the constructor function prototype of the object instance will be assigned with prototype of the constructor function.

When function is called from the object it first check is there a behaviour define in its scope if not then look for the parent prototype.

We can as well change the behaviour of the prototype function in the individual object level.

**Primitives and objects:**

Objects that are created using primitive syntax are called primitive objects.

Eg:

Let person={‘a’:’value’, ‘b’:’value’, ‘c’:value}

Here it inherits the prototype of the parent object called

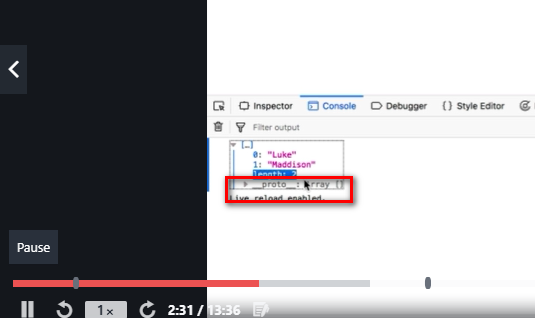
Object.prototype

Primitive values:

Values that do not have property like an object

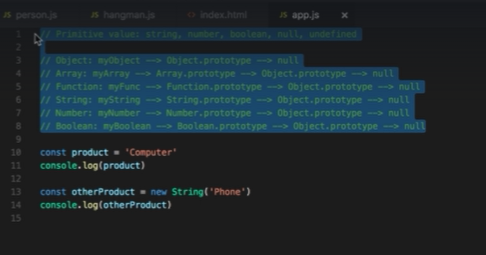
String, number, Boolean, null, undefined

\_\_Proto\_\_ property for the prototype mapping



Non of the primitive types has the prototype

Internally it converts primitive type to the respective object type.



Even primitive types has its own constructor functions using which we can make new primitive value

Using which even we can access the prototype.

**Class syntax:**

New syntax for creating constructer function and other things.

It’s a syntactic sugar on top of old way of creating the object

As below

Class Person

{

Constructor(parameter list)

{

This.a =a;

This.b=b;

}

Function1()

{

}

Fucntion2()

{

}

…

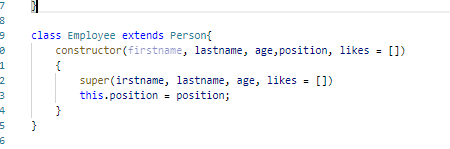
}

**Class inheritance:**

Extends key word

And user class level constructor function for the assignment

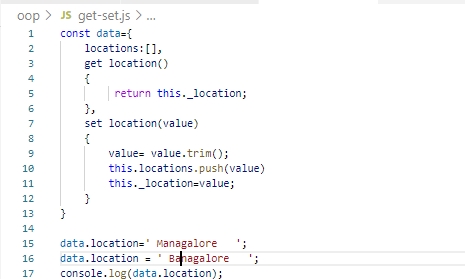
Super() to call the parent constructorx`



**Getter setter mutator/accessors:**

Both are function

Uses key word get, set



Asynchronous programming and ajax

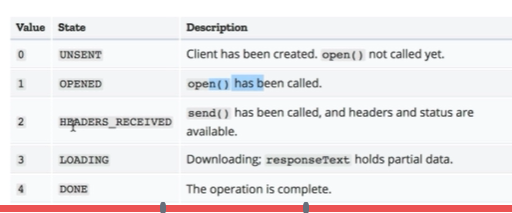
HTTP protocol

Request repose protocal

HTTP request from JS:

With this we can call a server side Api using the

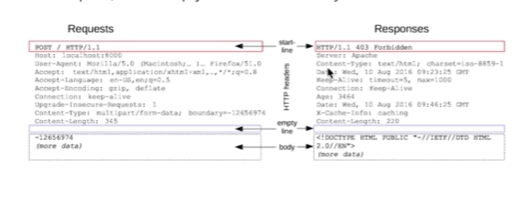
XMLHttpRequest objectd





Every http request will respond back with the unique code called status code

HTTp request has



Request

Headers

Body

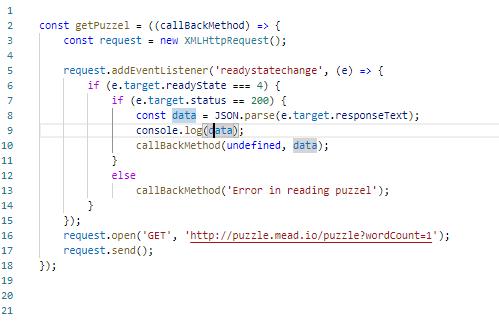
Running request asynchronously:

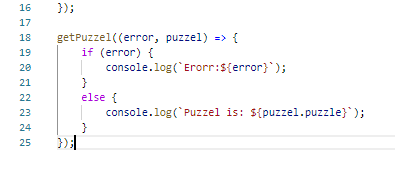
Use call back approach:

Where in you have to send the function to be called when some action is completed

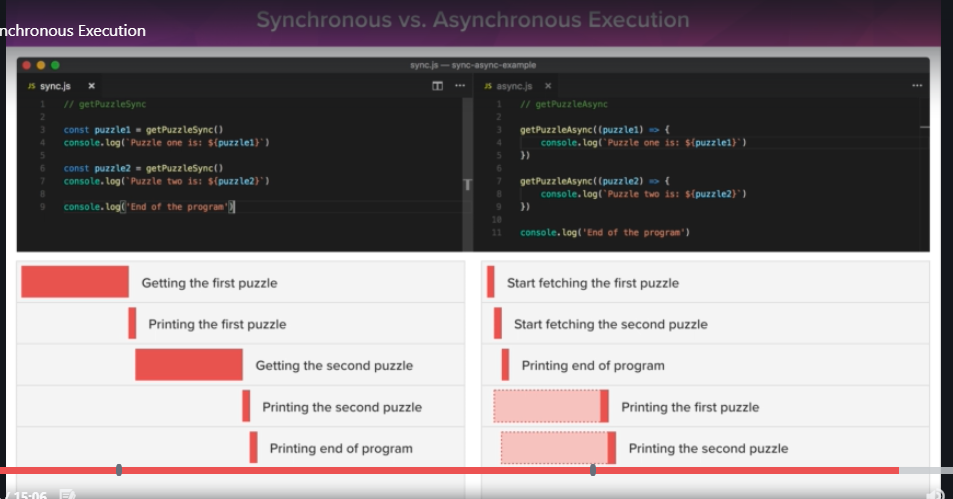
Because we never know when the request from the server is processed successfully and response is sent back to the client

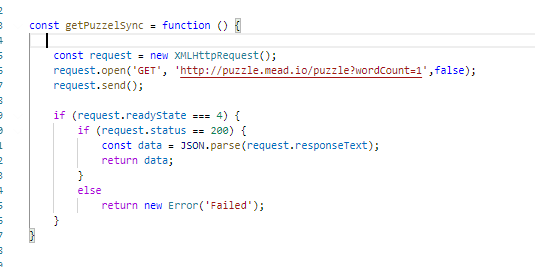
Where in to another function call you will be sending a function reference as parameter and call that method for it





Synchronous VS Asynchronous execution

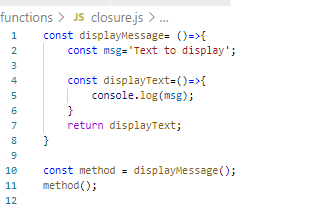


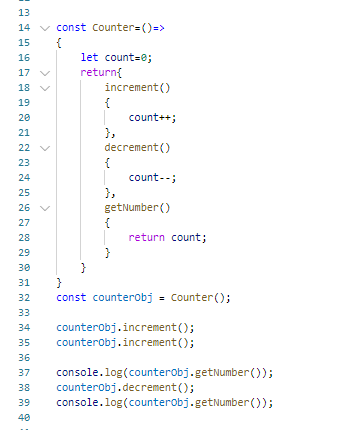


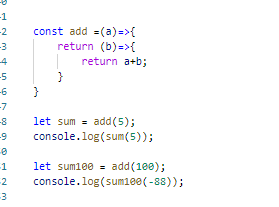
Closure:

Related to function and function scope and lexica scope in which the function is defined.

And data that it can access from it scope.







Promise Api:

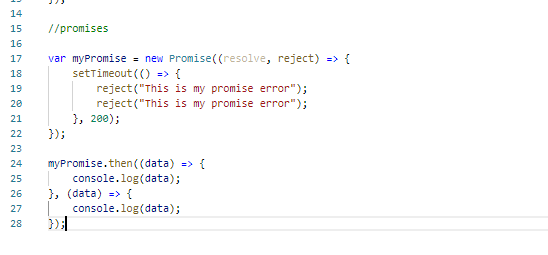
API used to run long running process inside the method supported by the promise called resolve and reject

Promise constructor parameter

Point to the functions. So long running process we need to put it inside promise method

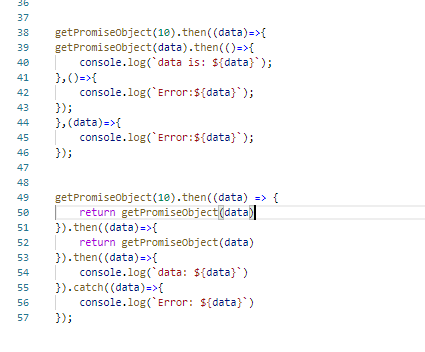
On successful execution call resolve method

On failure call reject method

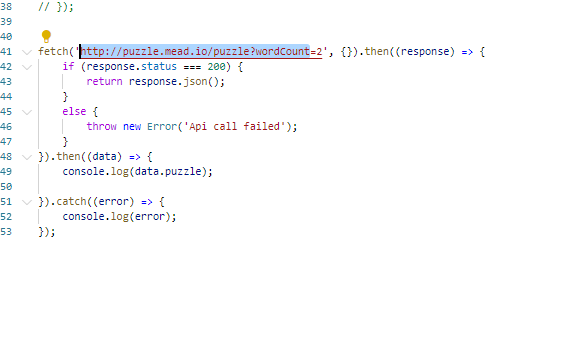


Promise chaining:

Calling one promise inside other



**Fetch Api another way of making hTTP request from client side**

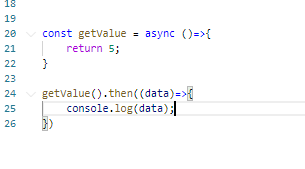


Async await:

Long running process called inside the method marked as async and then we mark a function call inside await

So the code written after the await will not execute until the previous code is completed.

Always method marked as async will return the promise object and await can be used only with async object



With await return data as normal and print it. And await can be used only inside the async.

**App theming:**

Adding CSS to the application

**Babel:**

Used to run and support JS feature in almost all browsers including the outdated browser

**WebPack:** used to structure the js as module based js for structure the js.

Babel:

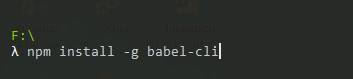
For cross browser compatibility

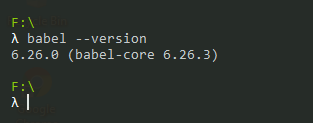
<https://babeljs.io/>

to use this inside the application we need to install the babel first from the NPM

command is

npm install -g babel-cli





Before doing this we need to initiate babel in our application.

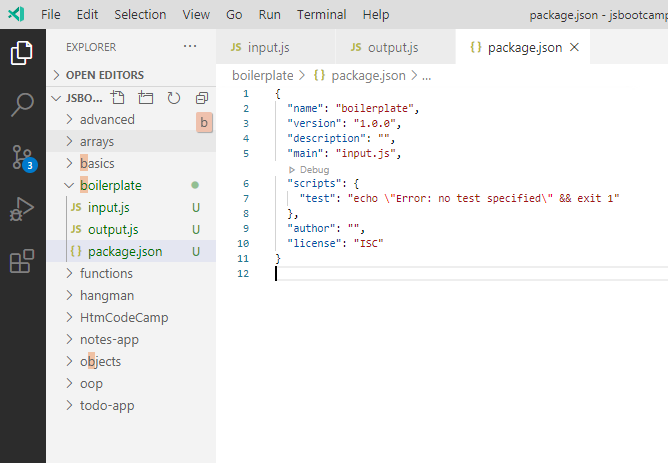
This is done with command

Go to working directory

**$Babel init**

Answer all questions asked in the screen. This will generate the

Package.json file inside our project.

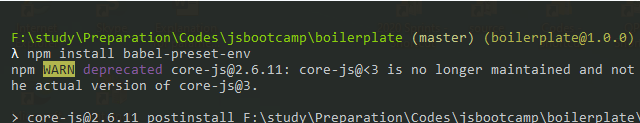


With npm command if I use -g this will install the things globally that means inside the system

Without that it installs inside the project locally

So I the above code in order to generate the compatible code we need a new module called preset

λ npm install babel-preset-env



Final query used is

**Babel input.js -o output.js –preset env**



Simplest way of installing all dependency inside the project is configure everything in package .js

Then run install command

$npm install

Frame work used to generate the JS file that is compatibility to almost all the browser versions and also automate the process of the script generation

Steps involved:

1. Install babel

Npm install -g babel -cli

1. Initiate the babel for the project

Babel init

This will add package.json file. Where in you can configure all the packages used by the project

1. Install the pre-sets used to generate the compatible scripts

Npm install –-preset env

1. Noe the setup is ready

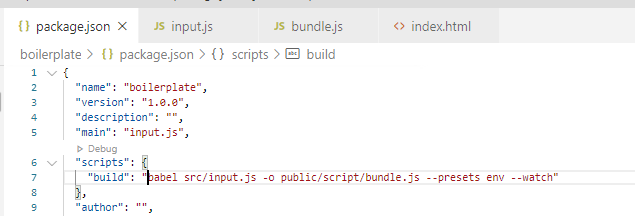
Run

Babel abc\source.js -o add\dest.js --presets env

This will generates and copy the js into this mentioned file

1. Also this process can be automated.

Inside package.js file include this command under script section

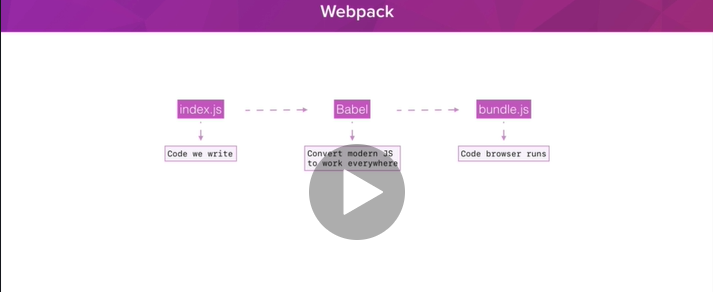


Run command call

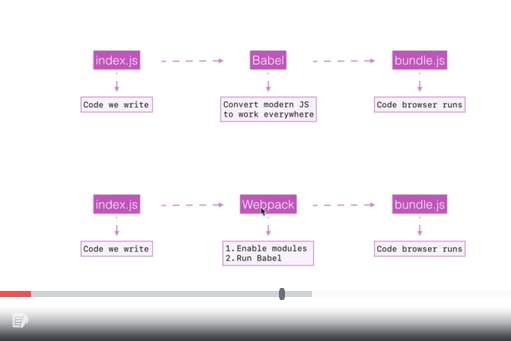
$npm run build will do the same thing

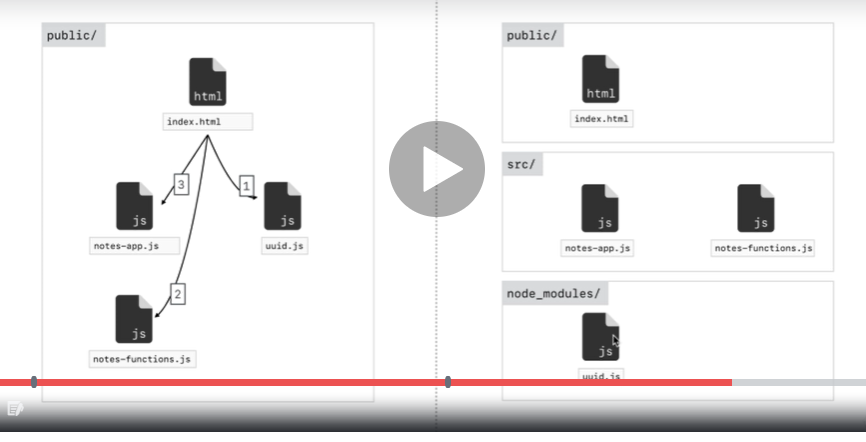
With watch live server keeps track of source file changes.

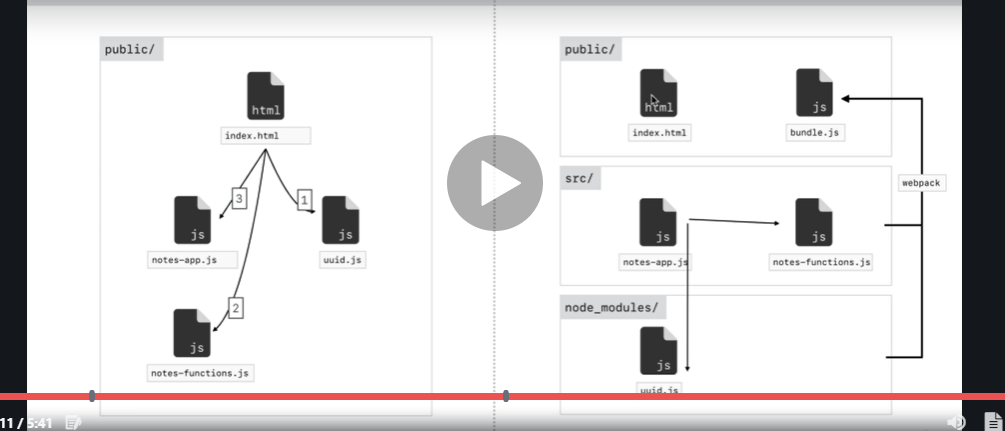
Web Pack:



Js Module based system:







Public contains the things that can be accessed. Src has code written by use

Node\_module has external Apis

Web pack will club everything and generate the single file called bundle.js that will be used in the page.

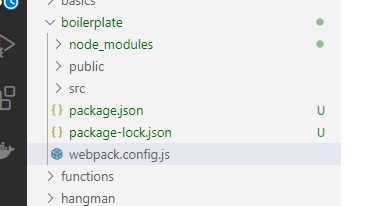
This will eliminate the multiple calls to get the multiple JavaScript files. There by helping in improving the application response time and reduce the bandwidth consumption

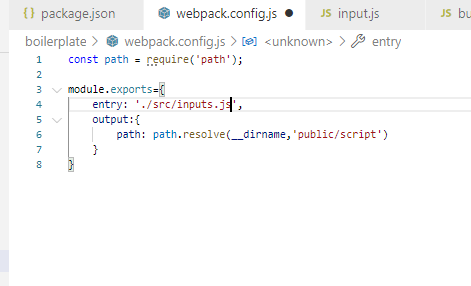
Install webpack:

1. $npm install [webpack@4.5.0](mailto:webpack@4.5.0) [webpack-cli@2.0.14](mailto:webpack-cli@2.0.14)
2. Add webpack command in Package.js file



1. Add new file under root of the project





Npn run webpack

JavaScript modules:

With this approach we can control what content to be imported and exported from the JavaScript file. When one JavaScript file uses the content from another JavaScript file. With this we can even eliminate the need of defining multiple script tags in design screen. We can import the required js files or function from another js file using import statements.

Index.js

Utility.js

If index need utility

Include this in index import ‘./utility’

Also we can achieve a kind of abstraction between two JavaScript files by importing and exporting certain functions.

**Code I want to export:**

Export let add = (a,b)=>a+b;

**From the referring js file:**

Import {lsit of methods} from ‘js file’

With this we can share the code between two or more file

There are 2 export statement possible in the js file

1. Name export
2. Default export

And file can have only one default export.

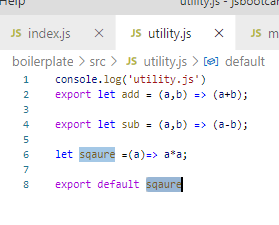
Syntax

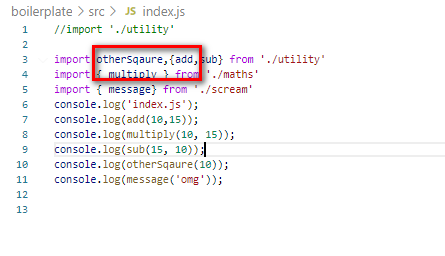
let func = ()=>

export default func

in the import

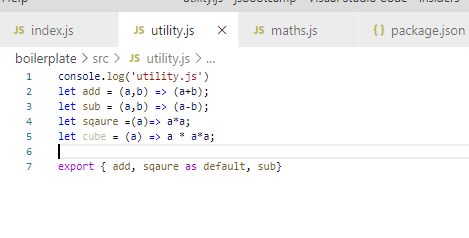
import func1, {name imports} ‘jas file’





A js file can a at the most one default export and any number of name exports.

Better export for single export statement



Adding babel to webpack:

This is achieved with the help of webpack loader this will tell us how to customize webpack to use babel

Here we use babel loader

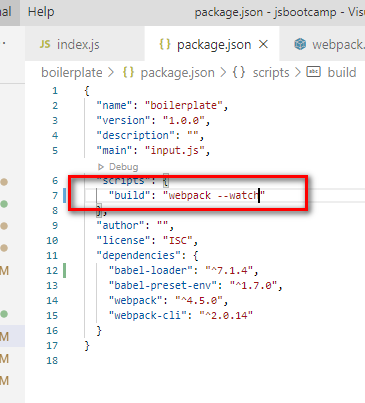
which will help you to load babel inside the web pack.

Install babel loader:

$npm install [babel-loader@7.1.4](mailto:babel-loader@7.1.4)

**Webpack dev server:**

Automating the process of the script minification with webpack.



For this you need to make sure the webpack is up and running.

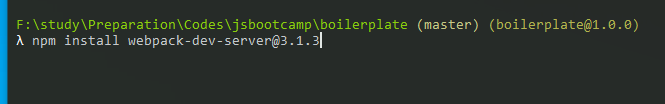
Webpack devserver:

Eliminate the need for running webpack in one tab and live server in another tab.

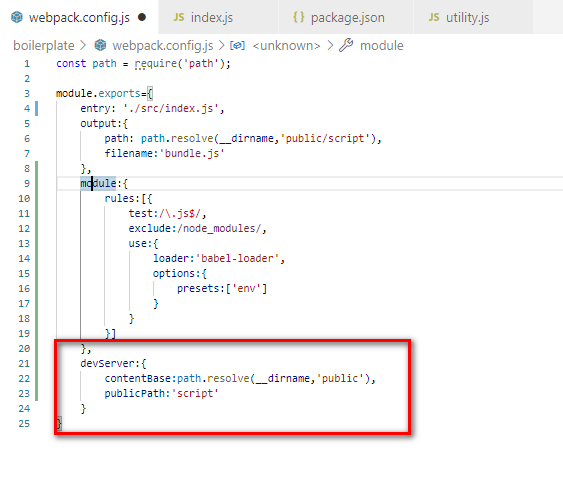
Single tool that runs everything in one tool

Step: install webpack server:

**Npm install** [**webpack-dev-serve@3.1.3**](mailto:webpack-dev-serve@3.1.3)



**Setup dev server in webpack config to tell which content to serve**



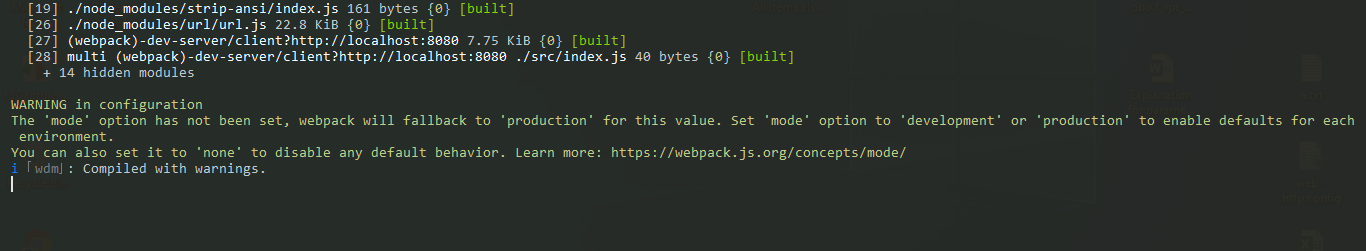
**Setup shorthand command in script file.**



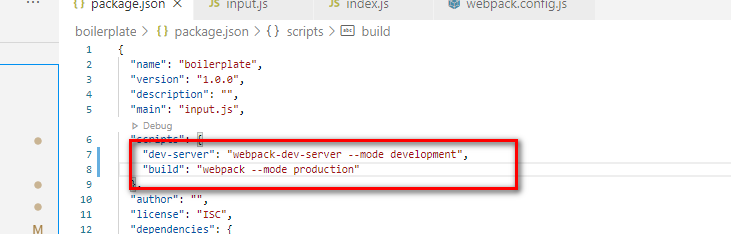
$nmp run dev-server

And its used only for the development process we would not need this for the production code

Environment and source:



To eliminate above warning we may need a mode for dev server



Dev server is virtual implementation so it supplies virtual script without having a physical file

Where as webpack makes a physical files

When data merged from multiple js files into single then there is a problem with debugging the js file when there is any error.

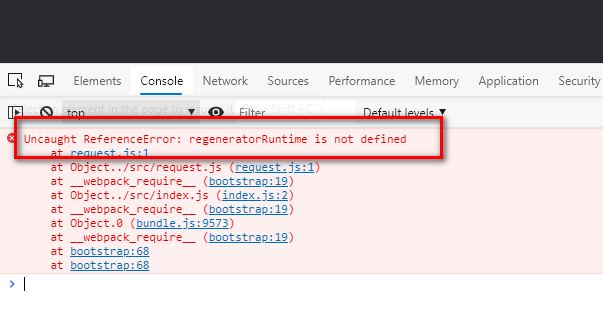
This is because source file and line number will vary in the final script file

To maintain the source map for the al js file we need to make a configuration in our webpack config.

So we need to enabled dev tool from the webpack configuration



Asyn /awaiti issue:



Fixed with the help of babel polyfill

This library will go through the code being converted and check if anything need to be generated before making the conversion and does it before the conversion happens

npm install [babel-polyfill@6.26.0](mailto:babel-polyfill@6.26.0)

after that specify this polyfill in webpack configuration



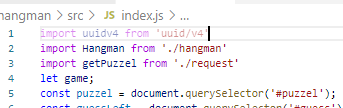
How to use third part library inside the project:

For this we may be needing command to install the library.

$ Npm install uuid

Import the library method inside the SJ file where you want to use.

import uuidv4 from 'uuid/v4'



With web pack we introduce a module system in our javascript development where is each java script file has its own scope. And we cannot share methods variable between js file.

We need to use import export statements to share the code between the files.

Rest parameter:

Similar to what we have in c# params where in we can pass the list of values as argument for the single variable and its represented with 3 dots …

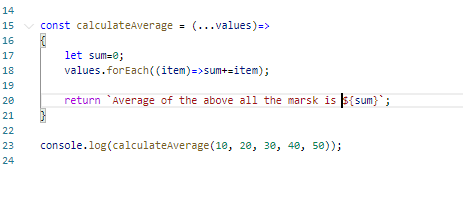
Eg:

CalculateSum(…list)

{

}

Call: CalculateSum(1,2,3,4,5,6);



With rest we are converting set of individual parameter into and array

**Spread operator:**

Opposite of spread operator

With spread operator we are converting and array into a individual parameter list.



This we can use to make true clone of the array

Cosnt a=[1,2,3,4,5]

Cont b=a;

Here both a and b are referring to same array so any changes done will affect both the arrays.

On the other hand

Cosnt b =[…a] this will make the true copy of the array a

So changes done to one array will not alter the other array.

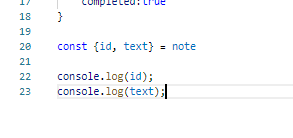
Object spread:

We need a babel plugin for this

npm install [babel-plugin-transform-object-rest-spread@6.26.0](mailto:babel-plugin-transform-object-rest-spread@6.26.0)

**Destructuring:**

With de structuring we extract the value of the property from the object and save it in a local variable.



DE structuring array

