به نام خدا

-------------------------------------------------------------------------------------------------------------------------

Data types in js :

-number

-string

-boolean

-BigInt

-object

-Array

-null

----------------------------------------------------------------------------------------------------------------------------------------------------------

How to convert datatypes in js :

-Number() => داده را به تایپ نامبر تبدیل میکند . => ساده ترش اینه که یه + قبل اسم داده بزاریم .

-String() => داده رو به استرینگ تبدیل میکنه

-Boolean() => داده رو به تایپ بولین تبدیل میکنه

----------------------------------------------------------------------------------------------------------------------------------------------------------

Logical operators :

\*isNaN() => checks that is data a number or not .

== ---> بررسی برابری

< or > => بررسی کوچکتری یا بزرگتری

=== 🡪 بررسی برابری مقدار و تایپ داده

----------------------------------------------------------------------------------------------------------------------------------------------------------

Bitwise operator :

! 🡪 عملگر نقیض

&& 🡪 and

|| 🡪 Or

----------------------------------------------------------------------------------------------------------------------------------------------------------

Ternary condition :

جهت نوشتن شرط تک خطی

برای مثال :

Let accessAlowed = (age>18) ? true : false ;

دستور بعد از ؟ در صورت صحیح بودن شرط اجرا میشود و دستور بعد از : در صورت نادرست بودن شرط اجرا خواهد شد .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Switch(x)

Case "firs condition for x " :

….

Breake ;

Case "second condition for x " :

….

Breake ;

Case "third condition for x " :

….

Breake ;

Default : 🡪 در صورتی که هیچ کیسی درست نباشد این دستور اجرا خواهد شد .

....

Breake;

----------------------------------------------------------------------------------------------------------------------------------------------------------

Declaration functions :

Index 🡪 از صفر شروع میشه

Length 🡪 طولش رو میگیره

charAt() 🡪 character at (here) 🡪 به عنوان ورودی شماره ایندکس رو میگیره

concat () 🡪 ورودی ها رو بهم میچسبونه

trim() 🡪 اسپیس های ابتدا و انتها رو پاک میکنه

toLowerCase() 🡪 همه حروف ورودی رو کوچیک میکنه

toUpperCase() 🡪 همه حروف دیتای ورودی رو بزرگ میکنه

search() 🡪 یک استرینگ رو میگیره و ایندکس حرف اول رو در صورت وجود به عنوان خروجی بر میگردونه

indexOf() 🡪 مانند متود سرچ عمل میکنه

includes() 🡪 دیتای ورودی رو میگیره و اون رو بررسی میکنه و برای وجود یا عدم وجودش یک بولین بر میگردونه

slice(start?,end?)

substr(start? , length?)

substring(start? , end? )

----------------------------------------------------------------------------------------------------------------------------------------------------------

Math methods :

PI 🡪 π

Pow(2,4) 🡪 2 رو به توان 4 میرسونه

Sqrt() 🡪 جذر

Abs() 🡪 قدر مطلق

Min() 🡪 مینیموم دیتا های ورودی رو ریترن میکنه

Max() 🡪 ماکسیمم دیتا های ورودی رو ریترن میکنه

Floor() 🡪 جزء صحیح

Trunc() 🡪 قطع کردن عادی

Round() 🡪 گرد میکنه

Ceil() 🡪 همیشه به سمت بالا گرد میکنه

----------------------------------------------------------------------------------------------------------------------------------------------------------

Assignment operators :

+= 🡪 num +=5 🡪 num = num + 5

-= 🡪 num -=5 🡪 num = num – 5

\*= 🡪 num \*=5 🡪 num = num \* 5

/= 🡪 num /= 5 🡪 num = num / 5

\*\*= 🡪 num \*\*= 5 🡪 num = num \*\* 5

Number ++ 🡪 number += 1

++number 🡪 number += 1

Number -- 🡪 number -= 1

--number 🡪 number-= 1

----------------------------------------------------------------------------------------------------------------------------------------------------------

Loops :

For(begin point ; condition ; step ){

//loop body

}

While (condition ) {

//codes so called loop body

}

Do{

//body

}while( condition ) 🡪 حداقل یکبار اجرا میشه در هر صورت

----------------------------------------------------------------------------------------------------------------------------------------------------------

Arrays :

آرایه ها کمک میکنند که چند تا موجودیت رو تو یه متغیر تعریف کنیم . در یک ارایه از هر نوعی در کنار هم میتوان استفاده کرد .

["Ali" , 10 , true ]

Arrays methods :

Push() 🡪 اضافه کردن یک آیتم به انتهای ارایه

Pop() 🡪 حذف کردن آخرین آیتم آرایه . این متود پارامتری به عنوان ورودی نمی گیرد .

Shift() 🡪 پارامتری نمیگیرد و اولین آیتم آرایه را حذف میکند .

unshift() 🡪 یک پارامتر را دریافت کرده و اون رو به ابتدای آرایه اضافه میکند .

foreach() 🡪 فانکشنی رو از ما میگیره و برای هر عضو از ارایه مد نظر اون رو اجرا میکنه

includes() 🡪 سرچ میکنه داخل ارایه ، همیشه کارساز نیست ،نمیتونه به عنوان ورودی یک ابجکت بگیره

some() 🡪 مانند فورایچ یک فانکشن میگیره و ترو یا فالس برمیگردونه ، وقتی به یک ترو برسه متوقف میشه

Every () 🡪 یک شرط رو به شکل همگانی چک میکنه و نتیجه رو برمیگردونه ، در صورتی ک به حتی یک شرط فالس برسه متوقف میشه

Splice() 🡪 با توجه به ایندکس ورودی هر آیتمی رو میشه حذف یا اضافه کرد .

Splice(start , count , items we need to add/remove(optional))

findIndex() 🡪 ایندکس ورودی رو در آرایه به ما برمیگردونه

map() 🡪 این متود همواره یک آرایه به ما بر میگردونه . یک فانکشن خاص رو روی همه اعضا اعمال میکنه و نتیجه رو در غالب یک آرایه برمیگردونه .

Filter() 🡪 یک فانکشن به عنوان فیلتر ورودی میگیره و اعضایی که ازون فیلتر رد میشن رو در غالب آرایه به ما برمیگردونه .

isArray() 🡪 چک میکنه آیا ورودی یک آرایه هست یا نه

indexOF() 🡪 ورودی که بهش میدیم رو در آرایه یا استرینگ سرچ میکنه و ایندکسش رو بهمون برمیگردونه ، در صورت وجود نداشتن آیتم ورودی در آرایه ، عدد -1 ایندکس خروجی خواهد بود .

Join() 🡪 تک نک آیتم هارو بهم میچسبونه .

Reverse() 🡪 کل آرایه رو برعکس میکنه .

Split () 🡪 تقریبن بر عکس متود جوین میتونه عمل کنه .

Find() 🡪 داده ای که بهش میدیم رو جست و جو میکنه و اطلاعات کل اون آبجکت یا آرایه رو بهمون برمیگردونه . تا زمانی که به یک ترو برسه اجرا میشه

Eval() 🡪 یک استرینگ به شکل ورودی میگیره و اگه یه کد باشه اون رو اجرا هم میکنه ازین جهت امنیت پایینی داره .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Objects :

Let user = new object() // " object constructor " syntax .

Let user = {} // " object literal " suntax .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Primitive data types :

-number

-bigInt

-Boolean

-symbol

-Null

-undefined

-string

Reference:

` -Object

-array

-Function

----------------------------------------------------------------------------------------------------------------------------------------------------------

Anonymous Functions :

Example : user.filter(یک تابع بی نام )

IIFE : Immediately Involked Function Expression 🡪 تابع خودخوان

Syntax :

(function(){

//codes

})()

----------------------------------------------------------------------------------------------------------------------------------------------------------

کد نویسی اصولی :

* زمانی که میخوایم یک آرایه رو پیمایش کنیم بهتره فقط و فقط از متود هاش استفاده کنیم برای مثال استفاده از foreach برای حلقه های for و while و do while .
* از eval استفاده نشه .
* همیشه برای ورودی های هر فانکشن مقدار default در نظر بگیرید .
* نام گذاری اصولی فانکشن ها بر اساس عملکردشون
* هر فانکشن همیشه فقط یک کار انجام بده نه چند کار

----------------------------------------------------------------------------------------------------------------------------------------------------------

Set Interval :

setInterval(function () {} , delay time (ms))

این تابع هر ms یکبار اجرا میشه .

clearInterval(interval name )

----------------------------------------------------------------------------------------------------------------------------------------------------------

False and truthy values :

-Boolean() 🡪 مقداری رو میگیره و فالسی و تروثی بودن اون رو برمیگردونه

-true , false

-undefined / NaN 🡪 false value .

[ ] / { } 🡪 true

----------------------------------------------------------------------------------------------------------------------------------------------------------

setTimeout :

زمانی که میخایم یک فانکشی بعد از یک مدت زمان معین اجرا بشه

Syntax :

setTimeout(function () { } , time(ms) , … arguments )

----------------------------------------------------------------------------------------------------------------------------------------------------------

Flag in JS :

فلگ ها یک مقدار بولین هستند و در شرط ها کاربرد دارند .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Let and const :

ما میتونیم متغیر هایی که با var تعریف کردیم رو هم آپدیت بکنیم هم دوباره تعریف بکنیم اما متغیر هایی که با let ساخته میشن رو فقط یکبار میتونی تعریف بکنی (مزیت ) .

Let از قانون hoistins پیروی نمیکنه .

متغیر هایی که با var تعریف میشن همیشه گلوبال هستن حتی اگه تو یه اسکوپ باشن .

اگه متغیری رو با const تعریف کنیم دیگه مقدارش آپدیت نمیشه و ثابته .

اگه متغیری با const تعریف بشه باید تو همون خط مقدار دهی بشه .

DOM in JS :

Document Object Model

----------------------------------------------------------------------------------------------------------------------------------------------------------

Methods for selecting an element :

-getElementById(" " ) 🡪 یونیک و برای انتخاب یک المنت با آیدی خاص

-getElementsByClassName( " " ) 🡪 همیشه آرایه برمیگردونه

-grtElementsByTagName( " " ) 🡪 همیشه آرایه برمیگردونه

-queryselector(' ' ) 🡪 ورودیش مثل سلکتور های سی اس اس هست و اولین المنت که اون آیدی یا کلاس رو داشته باشه انتخاب میکنه .

-querySelectorAll( ' ' ) 🡪 همه المان های ورودی رو برمیگردونه و یک آرایه هست .

----------------------------------------------------------------------------------------------------------------------------------------------------------

نحوه دسترسی به attribute ها و تغییر اون ها :

getAttribute(' ' ) 🡪 اسم اون صفت رو میگیره

setAttribute('name' , " value" ) 🡪 اضافه کردن صفت

----------------------------------------------------------------------------------------------------------------------------------------------------------

Access style of element :

Elementname.style.css\_property = "value"

----------------------------------------------------------------------------------------------------------------------------------------------------------

Access to the content of an HTML Element :

1.elementName.innetHTML

2.elementName.innerText

3.elementName.textContent

برای دسترسی به محتوا و value تگ هایی که تو فرم ها استفاده میشن از .value استفاده میکنیم .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Build elements of HTML with JS :

-createElement('element TagName ')

----------------------------------------------------------------------------------------------------------------------------------------------------------

How to append elements in DOM :

-append(' ' )

parentElements.append(' ')

----------------------------------------------------------------------------------------------------------------------------------------------------------

Events in JS :

-keypress

-keydown

-keyup

-change 🡪 معمولا برای تگ های انتخاب کردن انجام میشه

-onblur

-onfocus

-onselect

-ondblclick

-onload 🡪 این رویداد زمانی اجرا میشه که کل صفحه لود شده باشه .

-DOMContentLoaded 🡪 فقط منتظر میمونه دام لود بشه و منتظر لود فایل های لینک شده نمیمونه

-paste

-cut

-copy

-contextMenu

-onTouchStart

-onTouchMove

-onTouchEnd

-onTouchCancel

----------------------------------------------------------------------------------------------------------------------------------------------------------

addEvenetListener( , )

دو ورودی میگیره اولی اسم ایونت بدون on و ورودی دوم فانکشنی که میخایم اجرا بشه .

removeEventListener(eventName)

----------------------------------------------------------------------------------------------------------------------------------------------------------

classList

methods 🡪

add()

remove()

toggle(class name ) 🡪 اگه اون کلاس رو داشته باشه حذفش میکنه و اگه نداشته باشه بهش اضافه میکنه .

----------------------------------------------------------------------------------------------------------------------------------------------------------

has Attribute and remove Attribute :

void 🡪 یعنی چیزی رو بر نمیگردونه

----------------------------------------------------------------------------------------------------------------------------------------------------------

Third way of adding an event :

targetID.Event = function

----------------------------------------------------------------------------------------------------------------------------------------------------------

Event object \*

اطلاعات ورودی از کاربر رو به شکل یک object در اختیار ما قرار میده .

Id.addEventListener('event' , function(event){

//codes

}

----------------------------------------------------------------------------------------------------------------------------------------------------------

preventDefault()

برای جلوگیری از رفتار دیفالت و پیشفرض یک عنصر از متود preventDefault() استفاده میکنیم .

TargetId.preventDefault()

Example :

-form 🡪 ریلود شدن

-a 🡪 لینک شدن

----------------------------------------------------------------------------------------------------------------------------------------------------------

Consept of Node and its variants :

-node name

-node type

To add a new element to the HTML DOM, you must create the element (element node) first, and then append it to an existing element.

The appendChild() method appended the new element as the last child of the parent.

If you don't want that you can use the insertBefore() method

To remove an HTML element, use the remove() method

To replace an element to the HTML DOM, use the replaceChild() method

----------------------------------------------------------------------------------------------------------------------------------------------------------

How tov DOM Navigation :

elemName.parentNode 🡪 به والد اون المنت دسترسی پیدا میکنه

elemName.parentElement 🡪 به والد اون المنت دسترسی پیدا میکنه

elemName.previousElementSibling 🡪 المنت قبلی رو انتخاب میکنه

elemName.nextElementsSibling 🡪 المنت بعدی رو انتخاب میکنه

این پراپرتی ها فقط برای المنت ها هستند و روی آرایه ها کاربردی نداره

Children 🡪 array 🡪 نود هایی که فرزند اون هستن و یک المنت هستند رو برمیگردونن

childNodes 🡪 array 🡪 همه نود ها رو برمیگردونه

hasChildNodes 🡪 Boolean 🡪 چک میکنه ببینه فرزند داره یا نه

firstChild 🡪 فرزند اول رو انتخاب میکنه

lastChild 🡪 فرزند آخر رو انتخاب میکنه

firstElementChild 🡪 اولین فرزندی که یک المنت است رو انتخاب میکنه

lastChildElement 🡪 آخرین فرزندی که یک المنت است رو انتخاب میکنه

----------------------------------------------------------------------------------------------------------------------------------------------------------

Custom context menu :

Oncontexmenu = " return false " 🡪 منوی راست کلیک دیفالت رو میبنده .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Managing clipboard :

-onpaste 🡪 هنگام پیست ران میشه

-oncopy 🡪 هنگام کپی ران میشه

-oncut 🡪 هنگام کات ران میشه

Event.preventDefault() 🡪 جهت جلوگیری از پیست و کپی و کات میتواند استفاده کرد

----------------------------------------------------------------------------------------------------------------------------------------------------------

Properties of event object :

pageX and pageY in event object 🡪 میتوان ازین مقدار ها برای منو کلیک راست کاستوم استفاده کرد

----------------------------------------------------------------------------------------------------------------------------------------------------------

dataset in DOM

<element data-name="x" >

Elementname.dataset.name : x

----------------------------------------------------------------------------------------------------------------------------------------------------------

Methods and properties in working with media/audio/video elements in JS :

Let audioelem = document. …

audioElem.play() 🡪 پخش

audioElem.pause() 🡪 استپ

audioElem.duration() 🡪 مدت زمان بر حسب زمان ثانیه

audioElem.currentTime () 🡪 تایم کنونی رو برمیگردونه

audioElem.playbackRate() 🡪 تنظیم سرعت پخش

----------------------------------------------------------------------------------------------------------------------------------------------------------

Async and defer :

تقریبن کارشون یکیه ولی defer بهتره

فقط برای تگ های اکسترنال استفاده میشن .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Drag and Drop Events :

برای درگ کردن از اتریبیوت dragable = "true" استفاده میکنیم .

Event 🡪 - onDragStart - OnDragOver

-Datatransfer.setdata("name" , value)

-Datatransfer.getData("name")

-Dragged 🡪

1.ondragstart

2.ondrag

3.ondragEnd

-Dropped 🡪

1.onDragEnter

2.onDragOver

3.onDragLeave

4.onDrop

----------------------------------------------------------------------------------------------------------------------------------------------------------

Local storage

Syntax 🡪 localStorage.setItem("key","value")

Methods :

-setItem()

-getItem()

فقط و فقط دیتا هایی ذخیره میشن که استرینگ باشن .

لوکال استورج همه دیتا ها رو به استرینگ تبدیل میکنه .

JSON.stringify("key")

JSON.parse()

----------------------------------------------------------------------------------------------------------------------------------------------------------

Animation Events :

-onAnimationStart

-onAnimationEnd

-onAnimationIteration 🡪 هنگام تکرار شدن

----------------------------------------------------------------------------------------------------------------------------------------------------------

در روش document.elem.style فقط به استایل های اینلاین اون المنت دسترسی پیدا میکنه نه استایل های اکسترنال

-getComputedStyle

----------------------------------------------------------------------------------------------------------------------------------------------------------

Error handling

-try

-catch

-finally

-throw

----------------------------------------------------------------------------------------------------------------------------------------------------------

CSS text

-setProperty

-getProperty

برای تغییر متغیر های CSS

----------------------------------------------------------------------------------------------------------------------------------------------------------

Date Object

-getFullYear()

-getMonth()

-gatDate() 🡪 Day

-getHours()

-getMinutes()

-getSeconds()

-getMiliSeconds()

-getTime() 🡪 ms since jan 1 1970

-getDay() 🡪 week day as 0-6

-DateNow()

----------------------------------------------------------------------------------------------------------------------------------------------------------

Resize Events

-window.innerwidth

-wondow.innerheight

----------------------------------------------------------------------------------------------------------------------------------------------------------

Insert Adjacent Html

Element.insertAdjacentHTML('position' , "Text")

Positions :

-afterBegin

-afterEnd

-beforeBegin

-beforeEnd

Text:

An HTML code in string.

----------------------------------------------------------------------------------------------------------------------------------------------------------

Delete keyword:

برای پاک کردن یک پراپرتی از یک Obj از کلمه کلیدی delete استفاده میکنیم .

Delete obj.property

به هیچ عنوان از delete برای پاک کردن دیتای یک آرایه نباید استفاده کنیم .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Fragment

Let fragment = document.createDocumentfragment

DocumentFragments are DOM Node objects which are never part of the main DOM tree. The usual use case is to create the document fragment, append elements to the document fragment and then append the document fragment to the DOM tree. In the DOM tree, the document fragment is replaced by all its children.

Since the document fragment is *in memory* and not part of the main DOM tree, using document fragments could result in better performance in some older engines.

----------------------------------------------------------------------------------------------------------------------------------------------------------

Console Methods :

Console.log()

Console.error("")

Console.info(' ')

Console.warn(' ')

Console.assert('شرط ' , ' چیزی که باید اجرا بشود ' )

Console.table(object)

Console.time() و Console.timeEnd() 🡪 ورودی اختیاری میگیرنئ و همواره باید با هم استفاده شوند . مانند تایمر عمل میکند .

Console.group() و console.groupEnd() 🡪 ورودی ها اختیاری و از جنس رشته . همواره با هم استفاده می شوند .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Bom concept

BOM 🡪 Browser Object Model

-window

-location(URL)

-Navigator 🡪 دسترسی به اطلاعات مرورگر کاربر حتی وبکم

-Screen

-History

هرچیزی در js زیر مجموعه BOM محسوب میشه .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Width and Height in BOM

Window.innerwidth 🡪 عرض داخلی بادی سایت

Window.outerwidth 🡪 عرض کلی صفحه مرورگر

Window.innerHeight

Window.outerHeight

-screenTop

-screenLeft

----------------------------------------------------------------------------------------------------------------------------------------------------------

URL search params

برای گرفتن اطلاعات سرچ از URL و بارگزاری اطلاعات روی یک سینگل پیج

Let locationSearch = location.search

//bad way

//locationSearch.slice(1).split('=')[1]

//good Way

Let locationSearchParams = new URLSearchParams(locationSearch)

let userIdParam = locationSearchParams.get('ID')

سپس با استفاده از متود find از آرایه دیتا OBJ مورد نظر را پیدا کرده و روی صفحه بارگزاری میکنیم .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Reload methods :

History.go(0)

Location.href = location.href

Location.reload

----------------------------------------------------------------------------------------------------------------------------------------------------------

How to check user connection

-ononlineEvent

🡪 این دو فقط چک میکنن که آیا کاربر به مودمی متصل هست یا نه

-onOfflineEvent

----------------------------------------------------------------------------------------------------------------------------------------------------------

Work with console

Document.querySelector 🡪 $(' ')

Document.querySelectorAll 🡪 $$(' ')

Document.designMode = 'on' / 'off'

Document.body.contentEditable = true/false

----------------------------------------------------------------------------------------------------------------------------------------------------------

کانفیگ ها و آپشن های ایونت لیسنر که به عنوان ورودی سوم ثبت میشن :

Once : true 🡪 ایونت فقط یکبار فایر میشه

elementaddEventListener('eventName' , function {} , {config} )

----------------------------------------------------------------------------------------------------------------------------------------------------------

Event Deligation (bubbling and Capturing)

Bubbling 🡪 when an event happens on an element , it first runs the handlers out , then all the way up on other ancestors

Capturing 🡪 برعکس بابلینگ

Event capturing

اگر از کانفیگ capture : true استفاده بشه بجای bubbling از capturing استفاده میشه

----------------------------------------------------------------------------------------------------------------------------------------------------------

Stop propagation

هنگامی که میخایم نه بابلینگ و نه کپچرینگ اتفاق بیافته از این متود استفاده میشه .

Event.stopPropagation()

----------------------------------------------------------------------------------------------------------------------------------------------------------

Callback Functions :

تابعی که به عنوان ورودی به یک تابع دیگه ارسال میشه تابع callback نام داره .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Sync and async :

Synchronous and Asynchronous

Asynchronous معماری غیر مسدود کننده است، بنابراین اجرای یک کار به دیگری وابسته نیست. وظایف می توانند به طور همزمان اجرا شوند. Synchronous یک معماری مسدود کننده است، بنابراین اجرای هر عملیات به تکمیل عملیات قبل از آن بستگی دارد. هر کار قبل از رفتن به تکرار بعدی نیاز به پاسخ دارد.

----------------------------------------------------------------------------------------------------------------------------------------------------------

CallBack Usage :

از توابع callback برای sync کردن لود اطلاعات استفاده میشه برای اینکه زمان و موقع اجرا تابع اجرا کننده تصمیم بگیره .

Pure functions

توابعی که به ازای ورودی یکسان خروجی یکسانی هم داشته باشند .

اثرات جانبی نداشته باشند ( استفاده از متغیر ها در اسکوپ داخلی بلامانع است )

فقط یک کار مشخص انجام بدهند .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Composition function concept

Trim() 🡪 اسپیس های انتهای یک متن

وقتی از تابع های تو در تو استفاده میکنیم اصطلاحا یک composition fnc است .

----------------------------------------------------------------------------------------------------------------------------------------------------------

HOF concept

Higher Order Functions

تابعی که یک تابع دیگه رو به عنوان ورودی بگیره یا تابعی رو به عنوان خروجی بازگشت بده میشه یک HOF .

----------------------------------------------------------------------------------------------------------------------------------------------------------

JS strict mode

"use strict "

در خط اول این دستور رو مینویسیم . در اسکوپ های جدا هم میشه استفاده کرد .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Arrow functions :

() => {

//codes

}

-اگر می خواهیم فقط یک خط کد اجرا بشه نیازی به {} نداریم .

یعنی : () => //codes

اگر فقط یک آرگومان ورودی داشته باشیم نیازی به () هم نیست ، مانند :

Let logger = val => console.log(val)

اگر ورودی نداشته باشیم حتمن باید از () استفاده بشه .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Spread syntax in Array

میتونیم به کمک spread syntax (… element ) بیاییم یک آرایه کلی رو بهش بدیم و اون رو برای ما item by item بکنه .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Argument Keywords :

کلمه کلیدی arguments در Arrow fnc ها قابل دسترسی نیست .

تبدیل شبه آرایه arguments به آرایه : let name = array.from(arguments)

----------------------------------------------------------------------------------------------------------------------------------------------------------

جایگزین Arguments در Arrow functions :

Rest operators :

Const sum = (a,b,c, …aName)

رست اپراتور همیشه باید اخرین ورودی تابع باشد .

Syntax like spread but with different function .

Spread syntax 🡪 for making a copy of an array

----------------------------------------------------------------------------------------------------------------------------------------------------------

Array Dest :

Let user = [1,'ali',12]

Let userID = user[0]

Let userName= user[1]

Let userAge = user[2]

Use this instead : let [userId,username,userAge] = user

----------------------------------------------------------------------------------------------------------------------------------------------------------

Obj Destructing :

Let user = {

Id : 1 ,

Name : "Ali" ,

Age : 12

}

Let {id,name,age} = user 🡪 ترتیب فاقد اهمیت است ولی باید مطابق با آبجکت اصلی باشد .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Templete String : ~ lorem ipsum is ${sth}~

----------------------------------------------------------------------------------------------------------------------------------------------------------

For of loop :

همانند فورایچ برای آرایه ها

//like-arrays

//Iterable

Let number = [1,2,3,4,5,6]

For ( let Item of number ) {

//codes

}

----------------------------------------------------------------------------------------------------------------------------------------------------------

For In loop :

Let user = {

Id : 1,

Name : "Ali",

Age : 19

}

For (let item in user ) {

Console.log(~user.${item} : ${user.[item]}~)

}

----------------------------------------------------------------------------------------------------------------------------------------------------------

String padding :

Let username = "amin"

Console.log(username.padEnd/padStart(maxChar , fillWithWhat))

----------------------------------------------------------------------------------------------------------------------------------------------------------

Numeric separator :

جدا کننده اعداد

Let num = 9\_000 🡪 output = 9000

----------------------------------------------------------------------------------------------------------------------------------------------------------

And Bitwise operator in depth

Let age = 19

If(age > 18 ) {

Alert("You Logged In")

}

Instead :

Age > 18 && alert(" You Logged In " )

----------------------------------------------------------------------------------------------------------------------------------------------------------

Modules :

Import {names} from 'file address'

میتوان با استفاده از کلمه کلیدی as یک داده Import شده را rename کرد .

Import {is login as x } from ' ./…'

Is login 🡪 not defined

<script src = ' ' type = "module" ></script>

----------------------------------------------------------------------------------------------------------------------------------------------------------

Export defaults :

هر فایل جاوااسکریپتی فقط یک export default میتواند داشته باشد .

در هنگام import میتوان با یک نام دلخواه آن را ایمپورت کرد .

----------------------------------------------------------------------------------------------------------------------------------------------------------

What is type coercion in js :

تایپ کوئرشن 🡪 تبدیل نوع اجباری

Type coercion( Implicit-explisit)

TC Explisit 🡪 تبدیل تایپ با متود هایی همچون نامبر ، استرینگ ، بولین و ...

TC Implicit 🡪 تبدیل تایپ خودکار همچون اعمال ریاضی روی دیتاهای بولیان و استرینگ و ...

----------------------------------------------------------------------------------------------------------------------------------------------------------

RegEx 🡪 regular expression

صرفا برای کار با استرینگ ها

Str.match( / ……. / )

عمدتا برای جست و جو داخل متن

----------------------------------------------------------------------------------------------------------------------------------------------------------

کلمه کلیدی this

در جایگاه event handling همانند event.target عمل میکند .

< li onClick = " removeHandler(this)" >….</li>

Function removeHandler(el) {

El.remove()

Console.log(el)

}

🡺 log : < li onClick = " removeHandler(this)" >….</li>

This in functions 🡪 اطلاعاتی راجب ویندو بما میدهد

This in objects 🡪 اشاره به خود آبجکت دارد .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Call , blind , apply

کار با this و تغییرات آن در توابع به کمک این متود ها

showUser.call(user , 20 , 'Tabriz')

🡺 Same

showUser.apply(user , [20 , 'Tabriz'])

call & blind 🡪 در نحوه ارسال ورودی یکسان هستند

هر bind یک HOF است و نیاز به فراخوانی مجدد دارد .

Arrow Fnc 🡪 نیازی به کال و بایند و اپلای ندارند .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Constructor functions :

از arrow fncها نمیتوان به عنوان تابع سازنده استفاده کرد .

Constructor exp :

Function User(username , userAge , UserJob) {

This.name : userName ,

This.age : userAge ,

This.Job : UserJob ,

This.getName = function () {

Return this.name

}

This.getAge = function () {

Return this.age

}

This.getJob = function () {

Return this.job

}

This.setName = function () {

This.name = newname

}

This.setAge = function () {

This.Age = newAge

}

This.setJob = function () {

This.Job = newJob

}

}

Let user Ali = new User("Ali" , 18 , Web Developer )

Console.log(userAli) 🡪 {Ali , 18 , web developer }

----------------------------------------------------------------------------------------------------------------------------------------------------------

Nullish Operator 🡪 ??

عملگر or ||

|| دیتا ها رو یکی یکی از سمت چپ بررسی میکنه و به محض اینکه به یک مقدار trusty برسه اون رو به ما برمیگردونه . این عملگر از روی تمامی فالس ها رد میشه .

؟؟ فقط از روی null و undefined رد میشه.

Let developingPort = testPort || port ?? 3000 🡺 syntax error

Let developingPort = (testPort || port) ?? 3000 🡺 correct

----------------------------------------------------------------------------------------------------------------------------------------------------------

Chaining :

Let userStreet = user.address.city?.street 🡪 در صورت وجود نداشتن بخش سیتی برای ما بطور اتوماتیک آندیفایند برمیگردونه .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Object Entries :

Object.enteries(' like Array name') 🡪

خروجی یک ابجکت با آرایه

Let use : {

Id:1 , name:"Ali" , age: 20

} 🡪 { [ ' id ' , ' 1 ' ] , [ ' name ' , ' Ali ' ] , [ ' age ' , ' 20 ' ] }

----------------------------------------------------------------------------------------------------------------------------------------------------------

Set :

در این نوع از دیتا ها که مشابه آرایه هاست نمیتوانیم دیتا های تکراری ذخیره کنیم .

Let anything = new set()

Anything.add(sth)

Anything.delete(sth)

Anything.clear() 🡺 deletes all data in a set

Length at arrays 🡪 size at sets

Anything.has(sth) 🡪 works like " contain " command in Arrays .

We can use foreach() loop in sets like arrays .

We don't have map , filter , indexOf , find and etc in sets .

Let setScores = new Set(array name ) 🡪 gives us a set from an existed array .

Let arrayNum = [ … set name ] 🡪 gives us an array from an existed set .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Map :

Maps are like objects and follow " key : value " structure .

Let userMap = new Map( )

userMap.set(' id ' , 1 ) 🡪 like add command in sets .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Weakset and Weakmap :

Weakset 🡪 don't have size .

Weakset 🡪 just takes objects .

Weakmap 🡪 ? 🡪 use document for more information .

----------------------------------------------------------------------------------------------------------------------------------------------------------

What is proxy in java Script :

In JS proxies ( proxy object ) are used to wrap an object and redefine various operations into the object such as reading , insertion , validating , etc . Proxy allows you to add custom behavior to an object or a function .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Get Trap in proxies :

Let user = {

Id : 1 ,

First Name : " Ali " ,

Last Name : ' Doosti ' ,

Age : 20

}

User = new Proxy(user , {

Get : function ( target , property ) {

Return property in target ? target [ property ] : null

}

})

----------------------------------------------------------------------------------------------------------------------------------------------------------

Set trap :

Fires on object list updating .

Read more on documentary .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Symbol :

We can't read the value of a symbol at all . ( we don’t have access to it )

We use symbols when we want to have a unique data .

Let x = Symbol ( ) 🡺 we don’t need "new" key word to create new symbol .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Shared symbol :

Symbole.for () 🡪 symbol.for('sth') === symbol.for("sth") 🡪 true .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Object value short hand :

Let id = 1 ,

Let username = ' Ali " ,

Let age = 22 ,

Let user = { id , username , age }

Console.log(user) 🡪 id : 1 , username : ' ali ' , age : 22

----------------------------------------------------------------------------------------------------------------------------------------------------------

Multiple cords in ternary ( شروط تک خطی )

Let score = 10 ,

Let status = شرط : شرط : شرط : شرط

شرط ها به نوبت بررسی میشن و هر کدوم که صحیح بود نتیجش در status ذخیره میشه .

Status = score >0 && score >= 5 ? " 0 " 🡪 output : 0

----------------------------------------------------------------------------------------------------------------------------------------------------------

ASI : Auto Semicolon Injection

استثنا ها :

زمانی که ابتدای خط با [ ] شروع بشه موظفیم انتهای خط قبلی ; بزاریم .

هنگامی که خط با ( ) شروع بشه موظفیم انتهای خط قبلی ; بزاریم .

----------------------------------------------------------------------------------------------------------------------------------------------------------

Mutable and immutable in JS :

Mutable : an editable data 🡪 Reference type data 🡪 objects , functions , arrays , dates and etc

Immutable : type of data that we can't edit it. 🡪 primitive type 🡪 Num , str , Boolean , null , undefined , symbol

تغییر ندادن دیتا های خام : بهتر است تا حد امکان دیتاهای خام رو تغییر ندهیم و در صورت نیاز به تغییر اون ها رو توی یک متغیر دیگه ذخیره کنیم .

برای جلوگیری از میوتیت شدن اطلاعات در استفاده از متود هایی مانند Array.reverse() از اسپیرد سینتکس استفاده کنیم . [ …Array ] . reverse()

---------------------------------------------------------------------------------------------------------------------------------------------------------

Call back hell

برای فرار از جهنم کال بک ها باید از promise ها استفاده کرد . هنگامی که در انتهای کد کلی پرانتز و آکولاد باز و بسته است که خوانایی کد رو پایین میاره .

---------------------------------------------------------------------------------------------------------------------------------------------------------

Promise

// promise = ' قول دادن ' ; )) (resolved , rejected )

Let promise = new promises ( a function )

نمیتوان با یک پرامیس مثل یک تابع رفتار کرد .

myPromise() 🡪 false

myPromise.them(() => console.log(" success 😊) " ) 🡪 first callback when triggers on resolved part.

myPromise.then( success , reject ) 🡪call back handling

myPromise

.then( success) 🡪 on reesolved

🡺 call backs name

.catch ( error) 🡪 on reject

---------------------------------------------------------------------------------------------------------------------------------------------------------

Pass params to promise Handlers :

میتونیم به هندلر های خودمون ورودی ارسال کنیم و توی توابع callback از then و catch اون هارو دریافت کنیم .

---------------------------------------------------------------------------------------------------------------------------------------------------------

Promise phases :

اگر پرامیس به یک return رسید بقیه کد های بادی خونده نمیشه .

1. Pending
2. Fulfilled ( resolved )
3. Rejected

----------------------------------------------------------------------------------------------------------------------------------------------------

Promise finally handler :

myPromise

.then () 🡪 in fulfilled phase

.catch() 🡪 in rejected phase

.finally () 🡪 it would run at end in all terms .

اگر بخواهیم یک کد هم در .then () و هم در .catch() اجرا بشه بجای تکرار کد از .finally( ) استفاده میکنیم .

----------------------------------------------------------------------------------------------------------------------------------------------------

Chaining handlers :

myPromise

.then( response => {

Console.log( response) ;

Return response.split(" ") 🡺 the output of this return would use as input of next " .then " .

} )

.then ( response2 => { } )

---------------------------------------------------------------------------------------------------------------------------------------------------------

Ajax :

Ajax 🡪 Asynchronous Java Script and XML

Ajax بین بک اند و فرانت اند ارتباط ایجاد میکند ولی صفحه ریفرش نمیشود .

---------------------------------------------------------------------------------------------------------------------------------------------------------