Async - await – Promise – Callback Summery

بسم الله الرحمن الرحيم أول حاجه ده شرح للاهم توبيكس فى الجافا سكريبت يعتبر

##
#######

Sync Code VS Async Code

أول حاجه ال Sync code ده الكود العادى الى بيمشى بالترتيب عادى خطوه بخطوه
ذى ده مثلا

```
console.log("Hello 1");
console.log("Hello 2");
console.log("Hello 3");
/*
Scripts.js:1   Hello 1
Scripts.js:2   Hello 2
Scripts.js:3   Hello 3
*/
```

لاحظ ان نتيجه الكود ان كل حاجه جايه بترتيبها عادى خالص الكود راح لقى السطر الاول نوع sync راح رن السطر الاول بترتيبه و جابه و نفس الحكايه التانى و الثالث

Async By setTimeout

طيب تعالى نجرب وسطهم بقه كود نوعه Async فانا هستخدم ال setTimeout عشان دى حاجه من نوع ال Async عشان اوريك الكود هيمشى اذاى فتفهم من كلامى ال Async code بيمشى اذاى و ادى المثال و هشرحهولك

```
console.log("Hello 1");
setTimeout(function(){ console.log("Hello from setTime out 1")}, 1000)
console.log("Hello 2");
setTimeout(function(){ console.log("Hello from setTime out 2")}, 500)
console.log("Hello 3");
/*
Scripts.js:1    Hello 1
Scripts.js:3    Hello 2
Scripts.js:5    Hello 3
Scripts.js:4    Hello from setTime out 2
Scripts.js:2    Hello from setTime out 1
*/
```

بص ياسيدى الى حصل ان الكومبيلر دخل على الملف ده لقى ان اول سطر فيه كود نوعه Sync راح قاله اشطه اشتغل يا معلم و اظهرلى فى الكونسول

راح داخل على السطر التانى لقاه من نوع Async راح راكنه على جنب كده و قاله بص روح اشتغل فى اوضه لوحدك دلوقتى و هجيبلك لما اخلص احطك على الكونسول لما تكون خلصت شغلك عشان انت محتاج شغل خاص يا اسطى هشوف دلوقتى ال sync code الغلبان الى فى ايدى و بعد كده اشوفك عملت ايه
يبقى ال Async بيتخزن فى كونتيكست لوحده الكومبلير بيروحله بعد ما بيخلص ال Sync code الى فى ايده

الكومبيلر بعد ما ساب السطر التانى فى اوضه لوحده راح رايح داخل بقه على السطر التالت لقاه Sync code قاله انت غلبان اشتغل على طول ذى زميلك الى فى السطر الاول و اطلع اظهر فى الكونسول يا اسطى

دخل بقه على السطر الرابع لقاه Async قاله لا انت محتاج تروح فى اوضه مع صحابك ال Async الى ذيك انت محتاج شغل خاص و اشوفك هتحتاج وقت اد ايه عشان تشتغل و حوارات يلا روح على الاوضه عدل استنانى لما اخلص ال Sync code الغلابه هجيلك
بعد كده دخل على السطر الخامس لقاه sync code راح قاله انت غلبان اشتغل على طول يا اسطى ذى صحابك الى فى السطر الاول و الثالث و بقى عندنا دلوقتى
الكونسول مبين نتيجه كود السطر الاول و الثالث و الخامس ذى ما انا وريتك فى الكومينت الى مع الكود

```
Scripts.js:1    Hello 1
Scripts.js:3    Hello 2
Scripts.js:5    Hello 3
```

بعد ما خلاص الكومبيلر خلص الغلابه كده بقه هيروح على الاوضه الى فيها كل الاكواد ال Async فلما هيروح هناك هيعمل ايه بقه هيرتبهم على حسب الى هيخلص الاول و بما ان الكود الى السطر الرابع هيخلص فى نص ثانيه و الكود الى فى السطر الثانى هيخلص بعد ثانيه اذا هيروح مشغل الكود الى فى السطر الرابع الاول و بعد كده الكود الى فى السطر التانى ذى ما انت شايف كده

```
Scripts.js:4    Hello from setTime out 2
Scripts.js:2    Hello from setTime out 1
```

يبقى ملخص الكلام ده كله ان ال Sync code بيتحط فى كونتيكس رقم 1 مثلا و ال Async code بيتحط فى كونتيكست رقم 2
الكومبيلر بيروح رايح مشغل الكونتيكست 1 بتاع ال Sync code الاول و يخلصه بنفس ترتيبهم لان دول مفيهمش اوقات ولا حاجه دول بيتنفذوا بالترتيب من غير وجع قلب

بعد كده يروح رايح مشغل الكونتيكست 2 الى هو فيه Async code و جوه كونتسكست 2 ده بيروح يشوف كل كود فيه المده بتعته اد ايه و بياخد مش بالترتيب بقه لأ ده بياخد بالى مدته اقصر يعنى مهما اختلف الترتيب هو هيجيب الكود الى يخلص فى نص ثانيه الاول و بعد كده الكود الى هيخلص فى ثانيه و بعد كده الكود الى هيخلص فى 2 و هكذا

اذا ال sync code بيعتمد على الترتيب الى هو مكتوب بيه
اما ال Async code بيعتمد على الزمن او المده الى بيحتاجها الكود عشان يظهر كل اما صغرت كل اما ظهرت بدرى لانه هو فيه الى يخلص الاول يظهر الاول

المقصود بالكونتيكست 1 هنا هو ال Main thread الى بيتنفذ فيها ال sync code
الكونتيكست 2 هو ال Event loop و دى الى يتنفذ فيها ال Async code ولا يمكن تنفيذها ابدا الا بعد ال Main thread اى بعد ما اخلص ال Sync codes كلهم

ايه فايده ال Async code يا اسطى ؟؟
ده فيدته يا اسطى ان لو نفترض انى برجع من السيرفر داتا كتير جدا المفروض انه هياخد كده وقت كتير جدا وليكن خمس دقايق مثلا مثلا يعنى هههه فلو انا عملت الكود الى بيرجع الداتا ده انه يشتغل Sync و كنت حاطه فى اول ما يفتح الصفحه مثلا فكده بالشفاء الصفحه كلها هتفتح بعد خمس دقايق فقالك نحل المشكله دى اذاى ؟؟
راحوا عاملين ال Async و ده لان هيروح مشغل الصفحه كلها الاول و بعد كده يرجع ينفذ ال Async code لوحده مع نفسه و يقوله رجع راحتك انا كده كده شغلت باقى الصفحه مش باقى غيرك انت و بالتالى اصبح دلوقتى ان استرجاع الداتا حتى لو فى مده كبيره مش هيخلى الصفحه تقف و تنح فى وشك على ما الداتا ما ترجع لأ ده الصفحه هتشتغل قدامك و بعد كده مش هيبقى باقى غير ارجاع الداتا بس و بكده حلو مشكله كبيره و الكلام ده مطبق فى الفيس بوك اول ما بتخش الموقع بتلاقيه اشتغل بس البوستات بتعد ثوانى كده على ما تظهر لان ارجاع الداتا بتاعت البوستات شغاله بال Async

امثله للحاجات ال Async code
setTimeout
setInterval
addEventListener
Fetch
Subscribe

عمتا اى حاجه بتعتمد على وقت او اكشن تانيه يبقى دى Async اما لو حاجه عاديه ذى السويتش و الايف كونديشن و الفانكشنز العاديه لأ الحاجات دى اشطه بتبقى Sync عاديه خالص

امثله على setTimeout , addEventListener مع شرحهم عشان تفعم ال Async code كويس

```
setTimeout(function () {
    console.log("Hello from setTimeout after 1 second ")
}, 1000)
```

هتلاحظ دايما ان ال Async function بتبقى فيها Callback جواها لفانكشن تانيه و عشان و هتلاقى مع ال callback function دى شرط انها تتنفذ و هنا فى setTimeout الشرط ان يعدى وقت و هنا انا كاتبله ثانيه عشان فانكشن الكول باك تبتدى تتنفذ و يظهرلك رساله ازيك من السيت اوت بعد ثانيه

اما فى حاجه ذى ال addEventListener فهتلاقى فانكشن الكول باك و معاها الايفنت الى المفروض يحصل للعنصر الى انا محددهوله و ساعتها ال callback function تتنفذ و ده الكود اهه و هفهمهولك

```
document.addEventListener("click", function () {
    console.log("Hello After Click on Document")
})
```

هنا بقولك بص انا عندى حاجه اسمها document و دى هى ال body بتاع الموقع يعتبر يعنى من الاخر دى الصفحه بقوله بقه لما ادوس يعنى اعمل click عليها نفذلى الفانكشن الى هتطلعلك فى الكونسول ازيك بعد ما دوست على الدوكمنت هههههههههه

من الاخر الكود ال Async هو الكود المؤجل لحين تنفيذ شرط معين سواء مرور وقت او حدوث ايفنت معين يشغل ال callback function الى المفروض هى دى الى بحط فيها الكود الى يتنفذ

##
#######

Callback function

بص انت اول ما تسمع كلمه callback تعرف ان الفانكشن الى شغاله و جواها callback دى وش كده شغاله Async code لان اصلا ال callback اتعمل عشان يحل مشكله لل Async codes و انا هوريك دلوقتى المشكله و حلها بال callback و ادى المشكله الى ظهرت يا سيدى فى اكواد ال Async و اجبرتهم انهم يخترعوا ال callback و طبعا هشرحلك المشكله ايه عشان تفهم لما اجى اشرح الحل

اول حاجه انا هعمل كود عادى و اوريك انه بيشتغل عادى

```
function GetData() {
  return "this is my data which i need to get from func";
}

let data = GetData();
console.log(data);
```

ده كود Sync و هيرجع الاسترينج الى راجع فى ال return عادى خالص

طيب لو جيت احوله ل Async code المفروض انى مثلا احط الريتيرن دى داخل فانكشن setTimeout

```
function GetData() {
    setTimeout(function () {
        return "this is my data whic i need to get from Async func";
    }, 1000)
}
```

```
let data = GetData();
console.log(data); // undefined
```

اول ما حطيت الريتيرن جوه فانكشن ال setTimeout ظهرت مشكله ال Async و هو انا السيت تايم اوت خلت الريتيرن تستنى لحد ما ال Main thread يخلص شغله و بالتالى هو هيروح دلوقتى اصبح كأن فانكشن ال GetData فاضيه اصلا و بالتالى هترجه undefined للفاريبول data و لما الكونسول يظهرها هيظهرها undefined برضوا لان لسه برضوا ال Async code مشتغلش لان ذى ما انت عارف لازم ال Sync code يخلص الاول فكده يا سيدى ظهرت المشكله ان كده انا مش هعرف اشوف اى حاجه جوه ال setTimeout الا بعد ما المعلمين الى تحت الى هما sync code دول يخلصوا و هما اصلا معتمدين على نتيجه ال setTimeout

طيب اوفق راسين فى الحلال اذاى بقه و اخلى ال setTimeout يبعت القيمه الى راجعه منه لل sync code قبل ما هما يشتغلوا ؟؟

منا هنا ظهر بقه حاجه اسمها Callback و ده هو الى حل المعضله دى انك تبعت حاجه من Async code لحاجه sync code

ادى كود جل المشكله دى و هشرحه اكيد

```
function GetData(MycallBack) {
  setTimeout(function () {
     MycallBack("this is my data which i need to get from Async func");
   }, 1000)
}

GetData(function MBcallBackfunc(data) {
    console.log(data);
});

// this is my data which i need to get from Async func
```

هنا بقه بقوله ايه ؟؟

بقوله انا هعمل فانكشن اسمها getData هتكون بتاخد برامیتار و البرامیتار ده نوعه function و الى هو هيبقى ال callback function من الاخر و هنا البرامیتار انا مسميه MycallBack

تانى حاجه هروح انادى للفانكشن الى اسمها Getdata و طبعا هكتب جواها بقه ال CallBackFunc الى هى هتبقى الفانكشن الى ببعتها ك param للفانكشن الاصليه الى اسمها GetData و ده عشان انا هستخدم الفانكشن الكول باك دى جوه الفانكشن الاصليه اصلا

بما ان لحد دلوقتى الاتنين فانكشن Getdata , MBCallFunction و النداء بتاع فانكشن ال Getdata دول Sync code فهما هيتخذنوا فى ال Main thread و هيبقوا جاهزين انهم يشتغلوا عادى

فدلوقتى الى هيحصل بالظبط ان الفانكشن getdata اتسجلت فى ال Main thread و النداء بتعها اتحزن هو كمان فى Main thread و المفروض ان كده call ده هيشتغل عشان يشتغل ال Getdata ف وهو بينادى عليها هيلاقى callbackfunction الى هى MBCallBackfunc مبعوته كبامیتار فهيروح يقولها انتى Sync تعالى اخزنك عشان تبقى تشتغلى بعدين و طبعا ال call لل Getdata كده سليم و هيشتغل

اول ما هيشتغل هيخش جوه ال getdata هيبص يلاقى ان مفيش اى كود تانى من نوع ال Sync فكده مش باقى الا ال setTimeout الى هى من نوع ال Async code فهيقول امرى لله اشغلك بقه و أول ما هيخش يشغلها هيروح يلاقى فانكشن ال callback هناك موجوده فهيروح بعد ثانيه واحده مشغلها و بما انه هيشغلها خلاص فهتروح بعد الثانيه المحدده للسيت انتيرفال نادهه على MycallBack و بالتالى هتعبت الجمله الكبيره دى فى برامیتار ال data و ساعتها بقه الجمله دى تشتغل عادى ذى الفل يا اسطى و بكده بعد اللفه بنت ستين الكلب دى ال callback اتشرحت و ابقى تف على وشى لو متجننتش بسببها

خد بقه تاسك حاولى الكود ده لكود Async مع ال callback

```
function GetData() {
    return [{Name:"Youssef
Pro" , Age:25},{Name:"Mostafa Yasser" , Age:2.5}] ;
}

let data = GetData();
console.log(data);
```

ادى الحل يا اسطى

```
function GetData(callback) {
    setTimeout(function () {
        callback( [{Name:"Youssef
"Pro" , Age:25},{Name:"Mostafa Yasser" , Age:2.5}]) ;
     },2000)
}
GetData(function (data) {
    console.log(data)
});
```

من الاخر كده الفكر انك بتخلى نداء الفانكشن الاساسيه جواها فانكشن بتظهر الداتا و الفانكشن الكول باك دى مش هتخليها تشتغل الا بعد ما تخش فى ال Async code و بكده بقى ال sync code تحايلنا عليه و غشيناه و خلياه يستنى ال callback على ما يرجع من ال Async code فاصبح ال sync ذى الاهبل مستنى نتيجه ال Async code فهمت حاجه ؟؟

تعالى بقى نشوف مشالك ال callback

طيب دلوقتى لو عملنا اتنين فانكشن ذى دول كده هيكون ايه النتيجه ؟؟

```
function GetArray(callback) {
    setTimeout(function () {
        callback( [1,2,3,4,5] ) ;
     },2000)
}

function GeString(callback) {
    setTimeout(function () {
        callback( "Youssef Pro") ;
     },1000)
}

GetArray(function (data) {
    console.log(data)
});

GeString(function (data) {
    console.log(data)
});
/*
Scripts.js:18     Youssef Pro
Scripts.js:14     [1, 2, 3, 4, 5]
 */
```

النتيجه هتكون ذى ما انت شايف الفانكشن بتاعت الاسترينج بتاخد ثانيه و الفانكشن بتاعت الارى هتتنفذ فى 2 ثانيه و بالتالى فانكشن الاسترينج هى الى هتتنفذ الاول و بعد كده فانكشن الارارى

طيب دلوقتى افرض انا بقه عاوز اجيب الارى الاول ثم الاسترينج من غير ما اغير فى ال timing اعمل ايه بقه ؟؟

قالك بسيطه يا سيدى نادى على ال call بتاع فانكشن ال array و جواه ال call بتاع الاسترينج و بالتالى هو هينفذ فانكشن الارى خلاص 2 ثانيه و بعد كده هيروح ينفذ الفانكشن بتاعت الاسترينج بعد ثانيه و هيبقى كتابه الكود كده

```
function GetArray(callback) {
    setTimeout(function () {
        callback( [1,2,3,4,5] ) ;
     },2000)
```

```
}

function GeString(callback) {
    setTimeout(function () {
        callback( "Youssef Pro") ;
     },1000)
}

GetArray(function (ArrayData) {
    console.log(ArrayData)
    GeString(function (stringData) {
        console.log(stringData)
    });
});
/*
Scripts.js:14     [1, 2, 3, 4, 5]
Scripts.js:16     Youssef Pro
 */
```

طيب الحركه دى كده سيئه جدا جدا و كده انا لو عامل 100 فانكشن المفروض على كده لو كل واحده هتاخد ثانيتين يبقى انا
هستنى لا يقل عن 3 دقايق عشان كله يخلص و كمان هتعمل كم مهول من الفانكشنات جوه بعضها الى هو كول باك جوه كول باك جوه كول باك هههههه tree of dom
فكده خره اوى و بكده ظهرت مشاكل ال callback من الحته دى

حل مشكله ال Tree of Dom عن طريق ال Named CallBacks

##
Named Callback

عشان يحلوا مشكله الكول باك جوه الكول باك جوه الكول باك قالك نعمل حاجه اسمها Named Callback و الطريقه دى هى انى اعمل لكل ال callback function الامبلمنتيشن فى فانكشن خارجيه لوحده خالص و الاسم بس هو الى هستخدمه فى المكان القديم الى كان بيحتوى على كل ال callback function

دى الطريقه القيدمه فى عمل ال callback و طبعا بعده على طول هوريك ال Named callback عشان تفهم ايه الى جرى
الكول باك القديم

```
function GeString(callback) {
    setTimeout(function () {
        callback( "Youssef Pro") ;
     },1000)
}

GeString(function (stringData) {
    console.log(stringData)
});
```

بعد تطبيق ال Named Call هيكون بقه شكل الكود كده هفصل الكول باك فانكشن بره لوحدها و انا ديها كبراميتار جوه ال GetString بالشكل ده

```
function GeString(callback) {
    setTimeout(function () {
```

```
        callback( "Youssef Pro") ;
     },1000)
}

GeString(GeStringCallback);

function GeStringCallback (stringData) {
    console.log(stringData)
}
```

و عشان تفهم فيادتها اكتر انا هعمل كود بالكول باك العادى عباره عن 3 فانكشنز عاملين tree و هوريك بعد ما عملنالهم Named callback

دى الطريقه القديمه العاديه

```
function GetGrandfather(callback) {
    setTimeout(function () {
        callback( "Ahmed Yasser") ;
     },1000)
}
function Getfather(callback) {
    setTimeout(function () {
        callback( "Ayman Ahmed Yasser") ;
     },1000)
}

function GetChild(callback) {
    setTimeout(function () {
        callback( "Mostafa Ayman Ahmed Yasser") ;
     },1000)
}

GetGrandfather(function (GrandFather) {
    console.log(GrandFather);
    Getfather(function (Father) {
        console.log(Father);
        GetChild(function (Child) {
            console.log(Child);
        });
    });
});
```

دى بقه الطريقه بتاعت ال Named Callback هتلاقى وش كده الكود بقى مرتب احسن

```
function GetGrandfather(callback) {
    setTimeout(function () {
        callback("Ahmed Yasser");
    }, 1000)
}

function Getfather(callback) {
    setTimeout(function () {
        callback("Ayman Ahmed Yasser");
    }, 1000)
```

```
}

function GetChild(callback) {
    setTimeout(function () {
        callback("Mostafa Ayman Ahmed Yasser");
    }, 1000)
}

GetGrandfather(GrandFather_Callback);


function Child_Callback(Child) {
    console.log(Child);
}

function Father_Callback(Father) {
    console.log(Father);
    GetChild(Child_Callback);
}

function GrandFather_Callback(GrandFather) {
    console.log(GrandFather);
    Getfather(Father_Callback);
}
```

ذى ما انت شايف ايوه احنا حلينا مشكله ال tree of DOM بس للاسف الموضوع بهوق أوى مننا و بقيت بعمل لكل كول باك فانكشن فكده هكتب كود كتير اوى و انا اتخنقت يا جدعان فقالك انا عشان احل مشكله انى بقيت اعمل فانكشنز كتير كده يبقى خلاص ننزل بالجديد خالص بقه الى هيحل المشاكل المعفنه دى كلها ألا و هو ال promises انا هتكلم عن البروميسيس بعد الكول باك ايرور هاندلينج

##
#######

Callback error Handling

هنا المفروض هعلمك اذاى تهندل الايرور فى ال callback فذى ما انت شايف انا ببعت الايرور فى ال callback فى البراميتار التانى و بقوله يا صديقى لو فى ايرور مبعوت اعمله throw و اظهره فى الكونسول اما لو الايرور جايلك ب null يبقى مفيش ايرورز يبقى اظهرلى بقه الداتا الى هى اسمى الى هو محمود بدوى D:

```
function GeString(callback) {
    setTimeout(function () {
        Errors = null;
        // Errors = {
        //     ErrMsg: "My Custom Error"
        // } ;
        callback("Youssef Pro", error = Errors );
    }, 1000)
}
GeString(function (stringData, error) {
    if (error) {
        throw error.ErrMsg;
    } else {
```

```
        console.log(stringData)
    }
});
/*
Scripts.js:14     Youssef Pro
 */
```

طبعا انا هنا عامل الكود على انه مفيهوش ايرورز لو انت شيلت الكومنت عن الاوبجيكت بتاع الايرور هتلاقى الكود ضرب ايرور فى وشك على طول

##
Promise

البروميس دى اتعملت عشان لقوا ان الناس بتكتب كود كتير جدا عشان يعملوا شويه فانكشنز بسيطه بال Callback فقرروا يعملوا حاجه تلخص معاهم الكود و يبقى حاجه صغننه كده و هيبقى الكود الكبير الى كتبناه مكون من 3 فانكشنز الى فات ده هيتعمل فى عده سطور صغيرين
و هتهندل فيه الايرورز بسهوله و هتبعت نتايجه بسهوله و حاجه اخر شقلطه على الاخر و من غير تعب
تعالى الاول نعمل حاجه صغيره للبروميس عشان افهمك عليها السنتاكس
ده شكل الكول بالكول بالك الى هلخصهولك دلوقتى بال promise

```
function GetString(callback) {
    setTimeout(function () {
        Errors = null;
        callback("Youssef Pro", Errors );
    }, 1000)
}
GetString(GetString_CallBack);

function GetString_CallBack(stringData, error) {
    if (error) {
        throw error;
    } else {
        console.log(stringData)
    }
}
```

لما تعمل بالبروميس بقه هيبقى شكله كده

Promise

```
let MyPromise = new Promise(function (resolve,reject) {
    error = null;
setTimeout(function () {
    if(error== null){
        resolve("Youssef Pro");
    }else{
        reject("there is an error yasta")
    }
    }, 1000)
})
```

```
MyPromise.then((Response)=>{
    console.log(Response);
}).catch((err)=>{
    console.log("Errors  :  " + err );
})
console.log(MyPromise);
```

شايف الكود صغير اذاى و بقى انضف بص يا سيدى نفهمك الكود بقه
هنا بعرف حاجه اسمها بروميس الى هو وعد و بقوله داخل الكونستراكتور بتاع البروميس خد callback و الكول باك ده بياخد اتنين callback برضوا اولهم resolve الى هو يعنى شغل أو حل و التانى reject الى هو يعنى ارفض
داخل ال set time out بقوله لو مفيش ايرور يبقى نفذ ال resolve و ابعت البراميتار الى جواه لفانكشن ال then عشان يعرض النتيجه بتعتك دى جواه عن طريق callback function بتاخد ال response و تعرضه و لو بقه فى ايرور يبقى خش فى الريجيشكت و ابعت الايرور لل catch

##
#######

Promise as a function

كل الى انا هعمله هنا انى بدل ما هعمل البروميس و احطها جوه variable لا ده انا هعملها جوه فانكنش و اخلى البروميس يعمل ريتيرن من الفانكشن و بكده انا ممكن استخدم مميزات الفانكشن مع مميزات البروميس فعلى سبيل المثال هوريك الكود اهه و اشرحه احسن

```
function MyPromiseFunc(Param , errorMsg) {
    return new Promise(function (resolve,reject) {
        error = null;
    setTimeout(function () {
        if(error== null){
            resolve(Param);
        }else{
            reject(errorMsg)
        }
        }, 1000)
    })
 }

let xProm = MyPromiseFunc("Youssef Pro","there is an error yasta");
xProm.then((Response)=>{
    console.log(Response);
}).catch((err)=>{
    console.log("Errors  :  " + err );
})

MyPromiseFunc("Ayman Ahmed Yasser","there is an error yasta").then((Res
ponse)=>{
    console.log(Response);
}).catch((err)=>{
    console.log("Errors  :  " + err );
})
```

أهه ذى ما انت شايف انا عملت البروميس و ندهت عليها مرتين عن طريق الفانكشن call خدت بالك انى ممكن انادى على الفانكشن بشكل مباشر و بعدها then و بعد كده كاتش
بيقولك بقه انى ممكن اعمل كذا then ورا بعض الى هى then chain يعنى سلسله thenat ورا بعض هههههههه و دى بقه محتاجه اوريك مثال و افهمهلوك بقه

```
function MyPromiseFunc(Param , errorMsg) {
    return new Promise(function (resolve,reject) {
        error = null;
    setTimeout(function () {
        if(error== null){
            resolve(Param);
        }else{
            reject(errorMsg)
        }
        }, 1000)
    })
 }

let xProm = MyPromiseFunc("Youssef Pro","there is an error yasta");
xProm.then((Response)=>{
    console.log(Response); // Youssef Pro
}).then((Response)=>{
    console.log(Response); // undefined
}).catch((err)=>{
    console.log("Errors  :  " + err );
})
```

اهه ذى ما انت شايف عملت اول then طلعت الاسم محمود بدوى و رحت عامل كمان then بس التانيه دى طلعت undefined عارف ليه ؟؟
لان then الثانيه بتشتغل على نتيجه then الاولى الى هى من الاخر then بتشتغل على نتيجه then الى قبلها و بما انى انا عامل فى then الاولى كونسول و مفيش اى return فهى كده بترجع فعلا undefined
طيب لو انا بقه عاوز اخلى then التانيه تشتغل اعمل ايه ؟؟
كل الى هتعمله انك هتضيف return response داخل ال then الاولى وبكده انت بقيت ذى الفل يا باشا لان كده انا لما بعمل ريتيرن من البروميس الاولى كده انا ببعت الى راجع من then الاولى لل then الثانيه تشتغل عليه و ممكن ترجعه ذى ما هو او تشتغل عليه تعمل فيه اى حاجه براحتها و دلوقتى تعالى اورريك المثال بقه بعد ما هضيف ليه الريتيرن

```
function MyPromiseFunc(Param , errorMsg) {
    return new Promise(function (resolve,reject) {
        error = null;
    setTimeout(function () {
        if(error== null){
            resolve(Param);
        }else{
            reject(errorMsg)
```

```
        }
        }, 1000)
    })
 }

let xProm = MyPromiseFunc("Youssef Pro","there is an error yasta");
xProm.then((Response)=>{
    console.log(Response); // Youssef Pro
    return Response;
}).then((Response)=>{
    console.log( "Mostafa "+ Response); // Mostafa Youssef Pro
}).catch((err)=>{
    console.log("Errors  :  " + err );
})
```

شوفت انا لما عملت ريتيرن راح واخد الى محمود بدوى الى راجع من ال then الاولى و حطها داخل التانيه و راح مرجع الكلام ده فى الريسبونس بتاع التانيه و لانى عاوز زود على الريسبونس اسم مصطفى فرحت ظاهر النتيجه بقه بعد البروسيسنج ده فى الكونسول فظهر الاسم مصطفى محمود بدوى

و ممكن اكتب الكلام ده بطريقه تانيه و هى انى جوه ال then الاولى اعمل الريتيرن عباره عن promise برضوا و تعالى اوريك مثال

```
function MyPromiseFunc(Param , errorMsg) {
    return new Promise(function (resolve,reject) {
        error = null;
    setTimeout(function () {
        if(error== null){
            resolve(Param);
        }else{
            reject(errorMsg)
        }
        }, 1000)
    })
 }

let xProm = MyPromiseFunc("Youssef Pro","there is an error yasta");
xProm.then((Response)=>{
    console.log(Response); // Youssef Pro
    return new Promise(function (resolve , reject) {
        setTimeout(function () {  resolve("Mostafa " + Response) } , 10
00)
    })
}).then((Response)=>{
    console.log(  Response); // Mostafa Youssef Pro
}).catch((err)=>{
    console.log("Errors  :  " + err );
})
```

الفكره هنا انى بعمل ريتيرن ل promise جديده و جوه البروميس الجديده دى انا ببقى باعت اسم مصطفى + نتيجه البروميس الاولى من داخل ال then الاولى و بالتالى فى ال then التانيه بيبقى مبعوتلها الاسم مصطفى محمود بدوى جاهز للعرض على طول يعنى من الا خر بعمل كل البروسيسنج فى الاولى و ابعت النتيجه بس فى التانيه على عكس الى فاتت ببعت النتيجه من الاولى و اعمل عليه بروسيسنج فى التانيه و بعد كده اعرضه

ملحوظه بقه مهمه يا كبير لو انت دلوقتى عامل 100 then و طلع فى ال then رقم 6 مثلا error ايه الى هيحصل ؟؟
الى هيحصل ان انت هتشوف كل نتايج الخمسه الى اشتغلوا سلام و عند ال then السادسه الى فيها ايرور او reject هتلاقى الكود نط راح مشغل ال catch و طلع رسالته و سابه خالص من باقى ال 100 then

##
Multi Promise

لو عندك 3 promises و عاوز تشغلهم المفروض تشغلهم اذاى ؟
فى كذا فانكشن هشرحهملك بيشغلوا البروميسيس كلهم مع بعض
ذى مثلا

Promise.all()

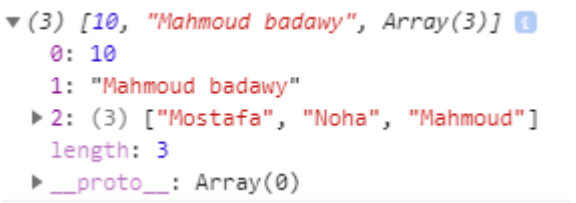
و دى فانكشن بتاخد array of promises و بتشغل كله مره واحده مع بعضه بعد مرور اطول وقت فى كل الفانكشنز الى موجوده و بيطلعوا بنفس ترتيبهم الى محطوطين فيه فى الاراى و بتطلع النتايج كلها فى array برضوا
و من عيوبها ان لو فى بروميس واحده طلعت ايرور بتشغلش اى واحده خالص منهم ساعتها

```
function MyPromiseFunc(Param , errorMsg , Time) {
    return new Promise(function (resolve,reject) {
        if(errorMsg == null){
            setTimeout(function () {
                resolve(Param);
            }, Time)
        }else{
            setTimeout(function () {
                reject(errorMsg);
            }, Time)
        }
    })
 }

let Promise_1 = MyPromiseFunc(10,null,1000);
let Promise_2 = MyPromiseFunc("Youssef Pro",null,2000);
//Promise_2 = MyPromiseFunc("Youssef Pro","error in prom 2",2000);
let Promise_3 = MyPromiseFunc(["Mostafa" , "Noha" ,"Ayman"],null,1500);

Promise.all([Promise_1,Promise_2,Promise_3]).then((Response)=>{
    console.log(Response);
    }).catch((err)=>{
    console.log("Errors  :  " + err );
})
```

النتيجه هتبقى بالشكل ده



جرب بقه تشيل الكومينت الى انا حاطه على promise 2 هتلاقى الايرور اشتغل و وقف شغل ال 3 بروميس كلهم

الطريقه التانيه

()Promise.race

دى يا سيدى بتاخد برضوا array of promises بس بتطلع فقط البروميس الى بتاخد اقل وقت و خلاص و نتيجه الكود هتبقى لو هنطبها على الكود الى فات ذى كده النتيجه هتبقى 10 لان البروميس الاولى هى اقل بروميس بتستهلك وقت

```
function MyPromiseFunc(Param , errorMsg , Time) {
    return new Promise(function (resolve,reject) {
        if(errorMsg == null){
            setTimeout(function () {
                resolve(Param);
            }, Time)
        }else{
            setTimeout(function () {
                reject(errorMsg);
            }, Time)
        }

    })
 }

let Promise_1 = MyPromiseFunc(10,null,1000);
let Promise_2 = MyPromiseFunc("Youssef Pro","Error in prom 2",2000);
let Promise_3 = MyPromiseFunc(["Mostafa" , "Noha" ,"Ayman"],null,1500);

Promise.race([Promise_1,Promise_2,Promise_3]).then((Response)=>{
    console.log(Response);
    }).catch((err)=>{
    console.log("Errors  :  " + err );
})
```

الطريقه التالته بقه

()Promise.allSettled

و دى يا اسطى ذى all بالظبط بس الميزه الى فيها انها بترجع نتيجه كل بروميس لوحدها و بتقولك حالتها يعنى
لو فى واحده خلصت و اشتغلت هتقولك
انها fulfilled و القيمه كذا و لو فى واحده فيها ايرور هتقولك انها rejected و القيمه بتاعت الايرور بقه و لو
فى واحده لسه مخلصتش شغل هتقولك انها لسه بتعمل binding و القيمه لسه مجتش
بس عيبها ان لسه مش كل البراوزارس بيدعموها دلوقتى
تعالى بقى اوريك الكود يا اسطى

()Promise.allSettled

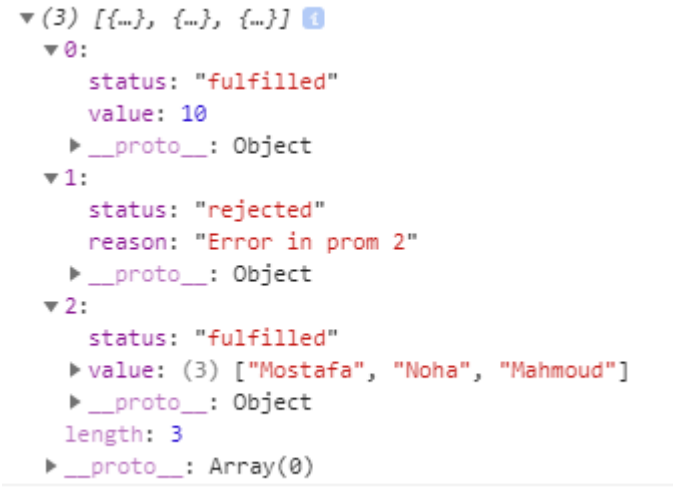
ادى الكود يا اسطى

```
function MyPromiseFunc(Param , errorMsg , Time) {
    return new Promise(function (resolve,reject) {
        if(errorMsg == null){
            setTimeout(function () {
                resolve(Param);
            }, Time)
        } else {
            setTimeout(function () {
                reject(errorMsg);
            }, Time)
        }
    })
 }

let Promise_1 = MyPromiseFunc(10,null,1000);
let Promise_2 = MyPromiseFunc("Youssef Pro","Error in prom 2",2000);
let Promise_3 = MyPromiseFunc(["Mostafa" , "Noha" ,"Ayman"],null,1500);

Promise.allSettled([Promise_1,Promise_2,Promise_3]).then((Response)=>{
    console.log(Response);
    }).catch((err)=>{
    console.log("Errors  :  " + err );
})
```

لاحظ انى فى البروميس التانيه ببعت ايرور يا اسطى
و ادى النتيجه اهه عشان تشوف الى حل



بعد كل التطور الى انت شايفه ده كله هما برضوا حسوا انهم بيكتبوا كود كتير اوى اوى و الافضل يخترعوا حاجه توفر عليهم كتابه الكود دى كلها و كان الحل اخير فى احدث حاجه طلعت الى هى Async , await

##
async - await

دى بقه جت عشان تقلل كتابه الكود و احنا بنعمل استدعاء ال promises و ده لانهم حسوا انهم بيكتبوا كود كتير جدا فقالك احنا نعمل حاجه بتعمل نفس شغل ال then , catch فى البروميس بس بسينتاكس بسيط و بكود اصغر بكتير من البروميس العاديه الفرق بينها و بين ال promise العاديه انى فى البروميس العاديه بقعد اعمل then , catch و قرف عشان اوصل لنتيجه البروميس

انما فى async , await هستخدم الاتنين كى ورد دول و شكرا

من المعروض ان async دى بتتحط قبل اى فانكشن هتشغلها و ال await بتتحط قبل البروميس الى هتشغله برضوا اذا الطريقه الجديده هيبقى شكل الكود كده

```
function MyPromiseFunc(Param , errorMsg , Time) {
    return new Promise(function (resolve,reject) {
        if(errorMsg == null){
            setTimeout(function () {
                resolve(Param);
            }, Time)
        }else{
            setTimeout(function () {
                reject(errorMsg);
            }, Time)
        }

    })
 }

let Promise_1 = MyPromiseFunc(10,null,1000);
let Promise_2 = MyPromiseFunc("Youssef Pro",null,2000);
let Promise_3 = MyPromiseFunc(["Mostafa" , "Noha" ,"Ayman"],null,1500);
```

```
async function RunAllPromises (Param1,Param2,Param3) {
    try {
        const prom1 = await Param1;
        console.log(prom1);
        const prom2 = await Param2;
        console.log(prom2);
        const prom3 = await Param3;
        console.log(prom3);
    } catch (error) {
        console.log(error);
    }
}

RunAllPromises(Promise_1,Promise_2,Promise_3);
```

ذى ما انت شايف عملت فانكشن و قبلها كلمه async و بعت للفانكشن 3 براميترز الى هما هبعت فيهم البروميسيس اصلا و جواها عملت try , catch عشان القط اى ايرور فى اى بروميس و جوه ال try عملت فاريبولز بسجل فيها الى راجع من كل بروميس عن طريق ال await و بظهر النتيجه فى الكونسول ياباشا و شكرا على كده كل حاجه هتظهر بالترتيب بتعها داخل ال try حتى لو فى واحده منهم هتخلص قبل التانيه و ترتيبها بعديها بيتم تأجيلها لحد ما الترتيب العادى يخلص يعنى كان الكود ماشى sync بالترتيب على حسب ال line مش على حسب الوقت فهمت قصدى ؟؟

ايوه انا فهمت كده بس مش فاهم مع ان البروميس نفسه async ليه ال Async await خلاه شبه ال sync ؟

ده ياباشا لأن ال wait اول ما الكومبيلر بيروح عنده بيروح قايل للكومبيلر بص يا اسطى استنى متمشيش من هنا الا لما تخلصنى بس انت ممكن برضوا تروح تعمل حاجه تانيه عادى بس المهم انت منتش خارج من عندى الا لما تخلص و اطلع نتيجه و بعد كده لو معاك نتايج تانيه عادى طلعها بس كله بترتيب ال line بتاعه يعنى من الاخر الطريقه دى بتخلى البروميس يشتغل Async بس بشكل sync فاهم حاجه :D

و ده الى بيميز ال async await فى البروميس عن ال subscribe و بسبب الحركه دى فبستخدم البروميس اكتر لو انت مثلا محتاج تشغل حاجه فى اول البرنامج و بيعتمد عليها باقى البرنامج ذى مثلا انك تجيب ال configuration بتاعت المشروع

انت ساعتها بتروح تجيبها بال async await promise و ده لان انا هروح اجيب الاعدادات دى فى الاول و مش هيمشى من مكانه الا لما يكون واخد معاه الاعدادات فعلا و رايح بيها على باقى البرنامج

انما لو استخدمت ال observable عن طريق ال subscribe مثلا فهتلاقى ان هو بيروح يجيب السيسنج و بما انها بتاخد وقت فبيسيها شغاله ويروح يجيب نتيجه اى حاجه تانيه غيرها فلو نتيجه الحاجه التانيه دى معتمده على ال configuration الى لسه مطلعتش فساعتها الكونفيجريشن هتروحلها ب undefined و بالتالى الفانكشن التانيه دى هتبقى اخدت حاجه معتمده عليها بقيمه ال undefined فهتضرب على عكس ال promise مش هتودى للفانكشن التانيه حاجه الا لما تخلص شغلها مع ال configuration و بالتالى هتوديلها الكونفيجريشن الى هى معتمده عليها سليمه و بالتالى الفانكشن هتشتغل تمام و هترجع نتايج سليمه

من الاخر ملخص الهرى ده كله

لو انت شايف ان الفانكشن مهمه و معتمد عليها فانكشنز تانيين هيتنفذوا بعديها يبقى اشتغل بال promise

اما لو شايف ان الفانكشن دى ينفع تتعمل لوحدها جنب الحاجات التانيه لانهم مش معتمدين عليها فاستخدم ال observable عن طريق ال subscribe اشطه كده يلا سلام اشوفك فى شرح تانى و على فكره انا شارح ال observable فى ملخص ال angular روح ذاكرها بقه

بص على ده كده هتفهم الفرق أوى برضوا

https://stackoverflow.com/questions/37364973/what-is-the-difference-between-promises-and-observables?fbclid=IwAR35zaF0bI_vAoc-WEnSr5U71HmhFZeecOR_UJetUjpS839TzrClfU0qK0E

و ده شرح تانى عشان تفهم البروميس برضوا لو مفهمتوش من فوق

##
promise

دى معناها وعد و دى شغاله على طول مع أوامر Async و عمتا طريقه شغلها كالأتى
الفانكشن بتشتغل و بالمفروض يترجع نتيجه يا اما ترجع ايرور
لو رجعت نتيجه بترجعها فى ال Resolve لو رجعت ايرور بترجعه فى reject اعتبرها ذى الفانكشن العاديه
السليم يرجه فى Return و البايظ يرجع فى ال throw Exeption هنا بقه Resolve , Reject
طريقه كتابتها بقه كده
اول حاجه بتروح تعمل البروميس نفسه و جواه بتعمل الفانكشنز الى انت عاوز تنفذها و جوه الفانكشن الى انت عاوز تنفذها دى هتكتب فانكشن الريسولف و هتبعت فيها براميتار و الريجيكت برضوا هتبعت فيها براميتارز عشان بعد كده هتستخدمهم و هيبقى الكتابه كده
ده كده انا عملت فاريبول من نوع بروميس فيه بقوله خش نفذلى الفانكشن الى جواك دى بعد ثانيتين و لو الفانكشن نجحت نفذلى ال Resolve و لو فشلت نفذلى ال Reject

```
var MyPromise = new Promise(
          //Arrow Function for Naming Resolve and Reject functions
          (MyResolve, MyReject) => {

              setTimeout(function () {
                  var Result = "Promise Logic Here";
                  // var Result = null; // Remove comment and Try
                  if (Result != null) {
                      // if  Success
                      MyResolve("Hello From Resolve");
                  } else {
                      // if  Error
                      MyReject("Hello From Reject");
                  }

              }, 2000)
          }
      );
```

طيب دلوقتى عشان ينفذ ال Resolve or Reject المفروض انا كده هاخد الفاريبول بتاعت البروميس و اشغل عليها اتنين فانكشن تانين then , Catch ذى try , Catch ال Resolve تقابلها ال try و ال reject تقابلها ال Catch و ادى ياعم باقى الكود عشان البروميس تشتغل بقه

```
    MyPromise.then(
          // Arrow Func (call Back) Take Only One Param
          // so If you need Alot of Variable Send all in one Object
          (Only_1_Param) => {
              console.log(Only_1_Param);
          }
      ).catch(
          (Only_1_Param) => {
```

```
                console.log(Only_1_Param);
            }
        )
```

##
#######

Nested Promises

لو عاوز تعمل بروميس لما تخلص تشغل البروميس الى بعدها و هكذا تعملها اذاى ؟؟ بالنيستيد بروميس
طب و دى تتكتب اذاى ؟؟ بتتكتب كده يا اسطى

```
var MyPromise = new Promise(
            (MyResolve, MyReject) => {
                setTimeout(function () {
                    var Result = "Promise Logic Here";
                    if (Result != null) {
                        MyResolve("Hello From Resolve");
                    } else {
                        MyReject("Hello From Reject");
                    }
                }, 2000)
            }
        );

        var MyPromise_2 = new Promise(
            (MyResolve, MyReject) => {
                setTimeout(function () {
                    var Result = null; // Remove comment and Try
                    if (Result != null) {
                        MyResolve("Hello From Resolve 2 ");
                    } else {
                        MyReject("Hello From Reject 2 ");
                    }
                }, 4000)
            }
        );

        MyPromise.then(
            (Only_1_Param) => {
                console.log(Only_1_Param);
                MyPromise_2.then(
                    (Only_1_Param) =>{
                        console.log(Only_1_Param);
                    }
                ).catch(
            (Only_1_Param) => {
                console.log(Only_1_Param);
            })}
).catch(
            (Only_1_Param) => {
```

```
                console.log(Only_1_Param);
            }
        )
```

ذى ما انت شايف عمنا 2 بروميس و ناديت على الاولى بعد ثانيتن و بعد كده ناديت على التانيه بعد ثانتين كمان عشان كده قولتله اندهها بعد 4 ثوانى من بدايه التشغيل

طيب انا بصراحه الطريقه دى فى كتابه الكود مش عجبانى و حاسسها ملعبكه حاجه بسيطه اى نعم هى سهله و حلوه بس الاحسن ان احنا نستدعى الاتنين برومايسيس بطريقه السلسله و دى عن طريق انى ارجع فى ال then بتاعت الاولى البروميسيس التانيه فى return و ده معناه انى كانى بقوله خلى كل الجزء ده بيساوى Promise_2 و بالتالى انت دلوقتى تقدر تستدعى ياباشا ال then بتاعت بروميس 2 و تخلى كله ليه كاتش واحده هيرجع فيها الرساله بتعته باختلافهم فى التعريف و ادى طريقه الكتابه الحلوه

```
MyPromise.then(
            (Only_1_Param) => {
                console.log(Only_1_Param);
                return MyPromise_2
            }
        ).then(
            (Only_1_Param) => {
                console.log(Only_1_Param);
            }
        ).catch(
            (Only_1_Param) => {
                console.log(Only_1_Param);
            }
        )
```

##
########

Promises With Async _ Await

دى طريقه بقه احسن من الاتنين الى فاتوا فى كتابه البروميس لانها بتختصر كتابه كود كتير
دلوقتى انت عامل اتنين برومبس و عاوز تشغل الاولى ثم التانيه دى هتعملها اذاى بسهوله فشيخه ؟؟
كل الى هتعمله انك هتعمل فانكشن و هتديها data Key اسمها Async و البروميس الى عاوز تشغلها هتكتب قبلها await و تكتب اسمها هتلقيها اشتغلت و مهمه ال Await ان هى بتخلى البروميس تشتغل و متخرجش منها الا لما تخلصها و بعد ما تخلصها تشتغل فى البروميس التانيه و ادى مثال بالكود اهه

```
    async function MyFunc(){
            let  MyProm_1 = await MyPromise;
            console.log(MyProm_1);
            return MyPromise_2;
    }
    MyFunc().then((msg)=>{console.log(msg);}
    ).catch((err)=>{console.log(err)});
```

بص يا سيدى الفكره انى عملت ويت للبروميس الاول بحيث انه مفيش اى حاجه تتنفذ الا لما هو يخلص و فعلا اول ما الثانيتن الاولنين خلصوا راح ظاهر الكونسول الى انا مرجه فيه نتيجه البروميس الاول و الى هى يا اما ريسولف يا اما ريجيكت طيب و بعدها عملت ايه ؟؟ عملت ريتيرن ؟؟ طب عملت ريتيرن لأيه ؟؟ عملت ريتيرن للبروميس التانى و ده عشان يبقى كانى بقوله ان الفانكشن دى نتيجتها فى الاخر انها ترجع البروميس التانى و بالتالى ساعتها اقدر اقوله الفانكشن دوت then دوت catch الى هو كانى بقوله اعملى بقه ثين و كاتش بتوع البروميس التانيه الى نتجت عن الكول للفانكشن الى انا فيها عامل ويتنج و بشغل البروميس الاولى

يارب تكون فهمت بعد العك ده كله :D: D