**JAVASCRIPT LANJUTAN | 10.1 Callback**

Di video kali ini kita akan membahas lebih detail lagi apa itu Callback ya dan kenapa kita harus pake dan nantinya kita juga harus paham kenapa harus kita hindari. Definisi *“Function yang dikirimkan sebagai parameter pada function yang lain.”* Sumber

<https://developer.mozilla.org/en-US/docs/Glossary/Callback_function>

Kan kita tahu bahwa function tuh bisa menerima parameter yg salah satu nya bisa menerima parameter function lagi, ada lai sumber yg menyebutkan bahwa callback itu adalah *“Sebuah function yang dieksekusi setelah fungsi lain selesai dijalankan.”* Sumber

<https://codeburst.io/javascript-what-the-heck-is-a-callback-aba4da2deced>

LETS CODE!!

**Synchronous Callbak**

Kita langsung impelementasikan pada kode sederhana kita, saya akan kasih contoh sebuah callback dimana tadi katanya callback adalah function yg dikirimkan sebagai parameter pada function yg lain, contoh nya misal saya punya function yg namanya halo() dia menerima parameter nya string berupa nama.

Nah nanti simpelnya saya Cuma mau nulis alert yg menampilkan ‘Halo nama’ gini :

        function halo(*nama*) {

            alert(`Halo ${*nama*}`);

        }

Nah nanti function itu akan kita kirim kan sebagai parameter dari function yg lain, jadi misalkan saya punya sebuah function lagi namanya tampilkan pesan, nah dia akan menerima parameter berupa function nah kita ambil contoh aja nama parameternya callback sebetulnya namanya boleh apa aja.

Nah nanti isi nya gini didalam nya saya mempunyai sebuah variable namanya nama isinya sebuah prompt() jadi meminta user buat memasukan namanya sendiri, lalu variable nama nya akan masuk sebagai parameter dari function halo(), jadi saya bisa tulis gini :

    function halo(*nama*) {

        alert(`Halo ${*nama*}`);

    }

    function tampilkanPesan(*callback*) {

        const nama = prompt('Masukan nama : ');

        callback(nama);

    }

    tampilkanPesan(halo);

Nah jadi parameter tampikanPesan() adalah function lalu function nya dipanggil didalam function, nah cara memanggil function halo nya kenapa gapake tutup kurung kan harusnya gini halo(), nah itu karena kan kita mengirimkan function sebagai parameter dan didalam nya memanggil function jadi anggapan pas kita ngirim halo kan ke callback jadi anggap aja halo nya ke ganti jadi namanya callback kan kalo kita ngirim function nya gini halo() entar didalam nya jadi halo()(nama); masa gitu, nah kalo kita kirim halo aja jadinya gini halo(nama); jadi perumpaan nya begitu.

Atau kalo dari pak dhika kita manggil halo nya gaperlu pake kurung karena kalo kurung itu langsung dieksekusi nah kalo tulis halo aja dia ngga, dia nunggu nantinya dipanggil di function tampilkanPesan() nya, pas manggil nya yg penulisan callback(nama) gitu.

Nah itu contoh kasus sederhana aja dan itu kita sebuh Synchronous Callback, callback function yg synchronous dan kaya nya synchronous callback ini akan jarang kita gunakan nanti kita butuh nya yg Asynchronous callback, Cuma itu supaya kalian paham aja cara pake sederhana nya.

Nah bahkan kita bisa aja ga bikin function baru yg halo() itu, kita bisa bikin Anonymous function, jadi langsung dijalankan seperti ini :

        function tampilkanPesan(*callback*) {

            const nama = prompt('Masukan nama : ');

            callback(nama);

        }

        tampilkanPesan(*nama* => alert(`Halo, ${*nama*}`));

Nah itu kalo kita jalanin sama aja ya tapi lebih simpel code nya, itu adalah Synchronous callback sebuah function yg parameter nya adalah function juga.

Nah yg kita butuh kan sebetulnya pemahaman mengenai Asynchronous callback, kita ke contoh berikut nya misalkan kita punya variable yg dia tipe nya arry yg didalam nya ada object anggap aja nanti kedepan nya itu jadi JSON gitu ya. Gini :

    const mhs = [

        {

            "nama": "Sandhika Galih",

            "nrp": "043040023",

            "email": "sandhikagalih@unpas.ac.id",

            "jurusan": "Teknik Informatika",

            "idDosenWali": 1

        },

        {

            "nama": "Doddy Ferdiansyah",

            "nrp": "133040123",

            "email": "doddy@gmail.com",

            "jurusan": "Teknik Informatika",

            "idDosenWali": 2

        },

        {

            "nama": "Erik",

            "nrp": "10404001",

            "email": "sandhikagalih@unpas.ac.id",

            "jurusan": "Teknik Industri",

            "idDosenWali": 2

        }

    ];

Jadi disitu ada array yg didalam nya ada objcet kan array itu kurung siku kalo object itu kurung kurawal nah disitu ada idDosenWali supaya kita tau nanti dosen wali nya siapa tapi itu kalian abaikan dulu ya.

Jadi ceritanya saya punya data dalam bentuk array of object certianya saya mau menampilkan semua nama mahasiswa nya saja kedalam console, masih inget ya pake higher order function kita pake forEach() ya, gini :

mhs.forEach(*e* => console.log(*e*.nama));

Nah itu gaada masalah ya kita belum masuk ke Asynchronous nya ya, tapi kalian liat disitu kalo kita pake higher order function biasa nya parameter nya itu udah callback karena dia function juga kan forEach() itu function nah parameter nya di isi Arrow function juga sama kaya yg tadi : tampilkanPesan( nama => alert(`Halo` ${nama})); jadi higher order function itu biasanya dia juga callback function.

Sekarang gini kalo misalnya sebelumnya saya kasih console.log(‘mulai’) dan setelah nya saya kasih juga console.log(‘selesai’), gini :

        console.log('mulai');

        mhs.forEach(*e* => console.log(*e*.nama));

        console.log('selesai');

ini kalo kalian inget dari video sebelum nya itu Synchronus ya jadi script nya di eksekusi dari atas kebawah jadi kalo kita jalanin pertama muncul ‘mulai’ terus nama-nama nya baru ‘selesai’.

Nah kedepannya kita mungkin aja ngambil data nya ga seperti yg ada di variable mhs dan ga sesimpel itu dan ngambil datanya ga didalam array gitu bisa aja kita ngambil datanya lewat API ke sever punya orang lain dan biasanya butuh waktu.

Dan kalo ngambil data nya cukup lama berarti console.log(‘selesai’) itu tampilnya nunggu dulu forEach nya selesai masih inget ya karena dia blocking, karena dia Synchronous maka ke blok gitu gabisa tampil console.log(‘selesai’) nya sebelum forEach() nya selesai.

Nah coba kalo kita simulasikan kita bikin seolah-olah ngambil data nya lama, nah sebelum kita nampilin nama nya kita bikin looping yg ga masuk akal ya, gini :

        console.log('mulai');

        mhs.forEach(*e* => {

            for (let i = 0; i < 1000000; i++) {

                let date = new Date();

            }

            console.log(*e*.nama);

        });

        console.log('selesai');

Jadi disitu kita bikin pengulangan yg ga masuk akal sampe 1juta kali tapi jangan looping Foreber ya nanti looping nya ga selesai”, looping Foreber itu kaya nampilin console.log atau alert() nah jadi nya disitu looping nya menjalankan variable saja yg kek seakan-akan membuat sebuah variable terus yg banyak jadinya masih berjalan looping nya.

Nah jadi kalo kalian jalankan itu ada perloadingan dulu kaya lama ga langsung tampil hasilnya coba sendiri, kalo kalian tambah 0 lagi di pengulangannya itu jadi tambah lama banget, gini :

        console.log('mulai');

        mhs.forEach(*e* => {

            for (let i = 0; i < 10000000; i++) {

                let date = new Date();

            }

            console.log(*e*.nama);

        });

        console.log('selesai');

Jadi keluar hasil nama nya itu satu-satu ga sekali gus tampil 3 nama gitu, baru terakhir muncul ‘selesai’ nya jadi nunggu dulu, coba bayangkan kalo ga selesai” loopingnya maka script setelah nya ga ga akan di eksekusi karena dia Synchronous.

**Asynchronous Callback**

Nah sekarang kita ubah simulasi kode kita menjadi Asynchronous ya, jadi harus nya kalian paham kenapa kita butuh Asynchronous, nah cerita nya saya mau bikin sebuah function yg ketika function itu dijalankan diaakan merequest data menggunakan AJAX.

Jadi cerita nya pak dhika itu udah punya 2 buah data dalam bentuk JSON disitu di file terpisah yg harus nya data itu diambil lewat API tapi itu simulasi aja ya, jadi ada data mahasiswa yg kaya tadi sama persis yg tadi di variable mhs isi nya array yg didalam array ada data mahasiswa tapi kali ini pak dhika bentuk kedalam file JSON.

Jadi saya punya function namanya getDataMahasiswa() nantinya dia akan menerima parameter nya 3, yg pertama url nya jadi itu alamat tempat penyimpanan data nya atau nanti itu bisa diganti dengan API terus parameter keduanya adalah function callback ketika success jadi kalo udah success ngapai, dan parameter ke-3 ketika eror gimana, success dan eror dia itu callback.

Sekarang saya akan menjalankan AJAX kita bikin let namanya xhr aja tapi saya mau bikin object AJAX nya XMLHttpRequest() hati-hati huruf bersar kecil nya, lalu selanjut nya ( xhr.onreadystatechange ) nah itu di isi dengan function, jadi ketika State nya udah siap jalankan fungsi itu terus nanti kita cek jika readyState nya adalah 4. Gini :

        function getDataMahasiswa(*url*, *success*, *eror*) {

            let xhr = new XMLHttpRequest();

            xhr.onreadystatechange = function(){

                if (xhr.readyState === 4) {

                }

            }

        }

Nah kalian ga usah khwatir dulu itu maksudnya apa itu adalah ketika kita mau melakukan Request secara Asynchronous menggunakan ajax, jadi kalo readyState nya 4 artinya itu udah ready ya karena perubahan State itu beruah dari 0 sampe 4, masing-masing State punya keadaan nya sendiri-sendiri 4 itu pokoknya udah state terakhir kalo udah siap.

Lalu jika status nya itu 200 (xhr.status === 200), artinya 200 itu oke kita bisa mengakses sebuah halaman dalam hal ini halaman yg file mahasiswa.json dan dsen.json kalo sudah oke maka saya butuh respon nya nah tapi itu mau saya bungkus kedalam function callback, gini :

        function getDataMahasiswa(*url*, *success*, *eror*) {

            let xhr = new XMLHttpRequest();

            xhr.onreadystatechange = function () {

                if (xhr.readyState === 4) {

                    if (xhr.status === 200) {

                        success(xhr.response);

                    }

                }

            }

        }

Jadi kalo udah sukses jalanin function success nya, tapi kalo eror status nya sama dengan 404 ya itu fice not found ya, jadi kalo kita salah nulisin tujuan ke file nya nanti kita nya eror maka jalanin aja function eror() nya gausah ada respon nya kalo eror tinggal kasih pesan eror nya. Gini :

        function getDataMahasiswa(*url*, *success*, *eror*) {

            let xhr = new XMLHttpRequest();

            xhr.onreadystatechange = function () {

                if (xhr.readyState === 4) {

                    if (xhr.status === 200) {

                        success(xhr.response);

                    } else if (xhr.status === 404) {

                        eror();

                    }

                }

            }

        }

Tinggal kalo udah kita jalanin ajax nya kalian bisa tulis xhr.open() nah hati-hati ya nyimpen nya itu diluar xhr.onreadystatechange nya kita tulis xhr.open() yg didalam nya ada beberapa parameter method nya mau apa yg (‘get’) lalu url nya kemana.

Nah url nya sesuai dengan yg nanti dikirim di parameter nya baru terakhir kita send(), gini :

        function getDataMahasiswa(*url*, *success*, *eror*) {

            let xhr = new XMLHttpRequest();

            xhr.onreadystatechange = function () {

                if (xhr.readyState === 4) {

                    if (xhr.status === 200) {

                        success(xhr.response);

                    } else if (xhr.status === 404) {

                        eror();

                    }

                }

            }

            xhr.open('get', *url*);

            xhr.send();

        }

Nah itu ajax menggunakan vanila js ya atau javascript murni nah sekarang kita panggil function nya, cara manggil nya langsung aja parameter nya 3 yg pertama url nya mengarahkan ke file json nya lalu parameter ke-2 nya adalah callback function Success dan parameter ke-3 adalah callback function eror, gini :

getDataMahasiswa('data/mahasiswa.json', success, eror);

Nah kalian kalo mau tulis seperti itu bisa tapi artinya kita harus punya function succes dan function eror diluar itu, kalo kalian gamau punya 2 function itu disaat memanggil nya kalian bisa langsung aja bikin anonymous function :

        getDataMahasiswa('data/mahasiswa.json', () => {

        }, () => {

        });

Nah jadi gitu, nah untuk yg function success itukan dia punya parameter ( xhr.response ) ya jadi anggap aja kita kasih parameter di function anonymous nya misal namanya results, coba kita liat dulu isi nya :

        getDataMahasiswa('data/mahasiswa.json', *results* => {

            console.log(*results*);

        }, () => {

        });

Coba kita jalanin berhasil ga data didapatkan dari file json nya, kalo kita buka console nya ternayta ada eror nih, katanya akses ke file mahasiswa.json nya diblok oleh CORS nah itu artinya kita harus jalankan kedalam sebuah web server, yg paling gampang kalian simpen file ini file javascript file HTML nya beserta folder data nya kedalam web server, kalo kalian punya XAMPP atau apche kalian bisa simpen kedalam folder htdocs nya.

Nah karena saat ini gua belum ngerti untuk menajalankan web server sebenarnya ada alternatife lain selain menggunakan web server yaitu menggunakan ektension vsCode yaitu live server nah saya udah download ektension nya jadi kalian tinggal ke file HTML terus klik kanan kek biasa gitu terus open with live server terus liat console nya nah udah tampil deh.

Nah tampil nya masih dalam bentuk teks, kita itu butuh nya tampil dalam bentuk object jadi caranya si parameter results itu harus kita parsing dulu menggunakan sebuah method parse(), jadi json.parse(), gini :

        getDataMahasiswa('data/mahasiswa.json', *results* => {

            console.log(JSON.parse(*results*));

        }, () => {

        });

Nah kalo udah gitu yg tadi nya teks udah berubah jadi object, nah sekarang kita ingin menampilkan hanya nama nya saja, jadi caranya kalian looping aja di situ dan si results nya kita parsing dulu masukin ke variable, gini :

        getDataMahasiswa('data/mahasiswa.json', *results* => {

            const mhs = JSON.parse(*results*);

            mhs.forEach(*m* => console.log(*m*.nama));

        }, () => {

        });

Nah terus sekarang apa dong bedanyayg kita lakukan ini dengan yg cara sebelumnya kan sama-sama menampilkan nama ya, yg cara sebelum nya juga sama menampilkan nama ya nah bedanya sekarang udah Asynchronous.

Nah buat ngecek nya kalian bisa kasih aja console.log sederhana aja sebelum manggil function nya dan sesudah nya, gini :

        function getDataMahasiswa(*url*, *success*, *eror*) {

            let xhr = new XMLHttpRequest();

            xhr.onreadystatechange = function () {

                if (xhr.readyState === 4) {

                    if (xhr.status === 200) {

                        success(xhr.response);

                    } else if (xhr.status === 404) {

                        eror();

                    }

                }

            }

            xhr.open('get', *url*);

            xhr.send();

        }

        console.log('mulai');

        getDataMahasiswa('data/mahasiswa.json', *results* => {

            const mhs = JSON.parse(*results*);

            mhs.forEach(*m* => console.log(*m*.nama));

        }, () => {

        });

        console.log('selesai');

Kalo sebelum nya kan ‘mulai’ tampil data nama baru ‘selesai’ nah kalo sekarang karena udah Asynchronous jadinya tampil di console ‘mulai’ terus ‘selesai’ dan baru tampil data nama nya. Nah jadi getDataMahasiswa() itu masuk kedalam web API kalo kalian inget di video sebelum ini ya nah lalu dia ke event loop jadi si console.log(‘selesai’) nya bisa tampil duluan dari pada data nama nya. Jadi kita bisa menjalankan code setealahnya tidak perlu menunggu dulu gitu.

**Jquery**

Nah kalo mau lebih gampang lagi dari pada kalian menggunakan vanila javascript atau javascript murni seperti di atas itu, nah kita bisa aja gunakan dari bantuan Jquery, jadi nanti kita panggil Jquery nya di file html ya.

Biar gampang coba kita cari tau cara manggil jquery, kalian coba googling aja jquery cdn jadi ita akan konek ke jquery secara online, disitu ada banyak versi kita akan pakai yg versi 3 kalian pilih antara yg uncomeressed atau minified jangan pilih yg slim karena yg slim gaada ajax nya, begitu pula kalo nanti kalian pake bootstrap nah bootstreap secara default udah bawa jquery tapi jquery nya yg slim jadi kalo kalian mau jalanin ajax harus diganti tuh.

Nah disitu kalian tinggal klik uncomeressed disitu udah ada script nya tinggal salin terus tempel di body HTML nya, gini :

<body>

    <script *src*="https://code.jquery.com/jquery-3.4.1.min.js"

*integrity*="sha256-CSXorXvZcTkaix6Yvo6HppcZGetbYMGWSFlBw8HfCJo=" *crossorigin*="anonymous"></script>

    <script *src*="script.js"></script>

</body>

Nah disimpan nya sebelum script punya kita, jadi sekarang kita udah bisa pake JQUERY, kalo pake jquery lebih simpel cara nulis nya.

Kalo pake jquery kalian cukup panggil sebuah method yg namanya ajax, jadi cara nulis nya jquery ($) jadi jquery itu tanda dolar nya, jquery jalanin ajax() yg parameter nya object, nah property nya ada macem-macem yg pertama ada url nya yg isi nya ngarah ke file mahasiswa.json.

Lalu kalo success mau ngapain kalian bisa jalanin sama kaya tadi lalu kalo eror mau ngapain gini :

        console.log('mulai');

        $.ajax({

            url: 'data/mahasiswa.json',

            success: () => {

            },

            eror: () => {

            }

        });

        console.log('selesai');

Nah kalo success dia menerima parameter hasilnya atau apalah nama parameter nya misal results boleh, tapi keren nya yg udah dibalikin ke parameter nya itu udah langsung JSON jadi kalian ga perlu pake method JSON.parse() gitu.

Jadi kita bisa langsung aja kasih namanya mhs karena isi nya mhs itu udah object dan liat itu udah Asynchronous jadi kalo misalkan kita mau ngambil nama nya aja udah kita tinggal looping aja, gini :

        console.log('mulai');

        $.ajax({

            url: 'data/mahasiswa.json',

            success: (*mhs*) => {

*mhs*.forEach(*m* => console.log(*m*.nama));

            },

            eror: () => {

            }

        });

        console.log('selesai');

Nah itu lebih simpel pake json dari pada vanila javascript, nah kalo eror kita mau nampilin apa nih jadi kita bisa kasih parameter apa saja namanya di eror itu return nya berupa eror nya, terus misalnya kita mau tampilin dalam bentuk text gitu jadi nya ( e.responseText ), gini :

        console.log('mulai');

        $.ajax({

            url: 'data/mahasiswa.jso',

            success: (*mhs*) => {

*mhs*.forEach(*m* => console.log(*m*.nama));

            },

            eror: (*e*) => {

                console.log(*e*.responseText);

            }

        });

        console.log('selesai');

Nah itu contoh kalo ada yg salah jadinya eror, yg disalahin typo di url nya nama file nya typo itu, nah jadi kalo eror tampil nya bacaan 404, katanya Cannot GET data/mahasiswa.jso kan harus nya json bukan jso ya, nah itu jadi ke console.log(e.responseText) callback nya.

Jadi itu kalo kalian mau pake jquery ya keliatan nya lebih simpel tapi kalian inget ketika kalian pake jquery artinya kalian pake library external, sebetulnya nanti didalam vanila javascript sudah ada fungsi baru untuk menggantikan ajax yg kasus sebelum ini nah nama nya feth.

Intinya di video kali ini kalian udah bisa lihat bahwa penggunaan callback untuk Asynchronous javascript seperti itu. Dan saat ini kasus nya masih berupa json data nya nah nanti selanjut nya kita akan coba bikin aplikasi yg lebih reall lagi kita akan benar-benar konek ke API sehingga kita bisa lihat bagaimana cara pakai callback dan nanti nya kita akan liat kelemahan dari callback.