**JAVASCRIPT LANJUTAN | 6.1 Destructuring Assignment**

Destructuring Assignment atau Destructuring variabel jadi fitur ini fitur yg menurut pak dhika cuku unik, kalo dari definisi nya : *“Expression pada javascript yang membuat kita dapat membongkar nilai dari array atau properti dari object kedalam variabel yang terpisah.”*

-MDN Web Docs.

Destructuring variabel ini merupakan sebuah expression pada javascript atau sintaks pada javascript ya. Jadi dia bisa membokar atau unpacking array atau object kalo mau lait detail nya kunjungi link berikut :

<https://developer.mozilla.org/enUS/docs/Web/JavaScript/Reference/Operators/Destructuring_assignment> .

Contoh sederhana nya itu misalnya gini :

const coba = ['satu', 'dua', 'tiga'];

const [a, b, c] = coba;

console.log(a); *//"satu"*

console.log(b); *//"dua"*

console.log(c); *//"tiga"*

Jadi kita punya sebuah variabel yg dia adalah array yg didalam nya saya punya 3 buah string nah itu kita bisa buat elemen” itu menjadi nilai pada variabel terpisah, kalo sebelum nya kan kita harus bikin satu-satu variabel nya gini :

const coba = ['satu', 'dua', 'tiga'];

const a = coba[0];

const b = coba[1];

const c = coba[2];

console.log(a); *//"satu"*

console.log(b); *//"dua"*

console.log(c); *//"tiga"*

Nah itu gambaran nya kalo tidak menggunakan Destructuring variabel, nah dengan menggunakan Destructuring variabel ini kita cukup gunakan tanda kurung siku untuk array ya yg ini [ ], lalu kita simpan nama-nama variabel yg ingin kita gunakan lalu kita Assignment dengan array coba itu.

Nah sehingga secara otomatis isi dari expression Destructuring itu akan dipetakkan taip-tiap elemen pada array nya sehingga kalian tidak perlu assignment satu persatu lagi variabel nya. Jadi disitu nya kalo saya console.log(a) dia langsung otomatis tampil array yg index ke-0 yaitu string “satu” gitu.

Nah begitu pula dengan object, object juga secara sederhananya sama misalnya saya punya object literal seperti ini :

const mhs = {

    nama: 'Sandhika Galih',

    umur: 33,

    email: 'sandhikagalih@unpas.ac.id'

};

Nah kalo kita mau petakkan tiap-tiap property nya sebagai variabel terpisah kan biasa nya kita bikin variabel nama nya nama di isi dengan mhs.nama gitu, kek gini misal nya :

const mhs = {

    nama: 'Sandhika Galih',

    umur: 33,

    email: 'sandhikagalih@unpas.ac.id'

};

const nama = mhs.nama;

const umur = mhs.umur;

const email = mhs.email;

console.log(nama);  *// "Sandhika Galih"*

console.log(umur); *// 33*

console.log(email); *// "sandhikagalih@unpas.ac.id"*

Nah kalo sekarang gausah seperti itu kalo sekarang kita tinggal bikin pake sintaks Destructuring Assignment nya kurung kurawal, jadi kalo kurung kurawal pake destructuring object untuk ngebongkar object kalo kurung siku untuk ngebongkar array, jadi nya gini kalo untuk object :

const mhs = {

    nama: 'Sandhika Galih',

    umur: 33,

    email: 'sandhikagalih@unpas.ac.id'

};

const { nama, umur, email } = mhs;

console.log(nama);  *// "Sandhika Galih"*

console.log(umur); *// 33*

console.log(email);  *// "sandhikagalih@unpas.ac.id"*

Nah disitu saya punya {nama, umur, email} yg di assignment ke objcet mahasiswa dan langsung secara otomatis menjadi variabel terpisah yg isi nya sudah ngambil dari object nya.

LETS CODE

Misalnya kita punya variable yg nama nya perkenalan didalam nya array isi nya string, nah sebelum menggunakan destructuring misalnya saya ingin mengambil isi array nya sebagai satu variable terpisah kan tadi cara nya kita bikin misalnya variable salam ngambil array pekenalan index ke-0, gini :

const perkenalan = ['Halo', 'nama', 'saya', 'Sandhika Galih'];

const salam = perkenalan[0];

const nama = perkenalan[3];

console.log(nama);

Nah menggunakan teknik biasa mengambil index dari array seperti itu gaada masalah ya, nah kalo menggunakan destructuring kita cukup bikin variable tapi bentuk nya pake kurung siku, kita langsung aja petakkan tiap-tiap elemen nya kedalam variable terpisah yg nama variable nya bebas lalu kita assignmet ke array perkenalan, gini :

const perkenalan = ['Halo', 'nama', 'saya', 'Sandhika Galih'];

const [salam, satu, dua, nama] = perkenalan;

console.log(nama);

Nah itu namanya destructuring array jadi yg [salam, satu, dua, nama] itu dia akan langsung jadi variable jadi di unpack gitu atau di bongkar jadi ketika saya tulis console.log(nama) langsung jadi variable nama yg nanti muncul di console nya yg ‘Sandhika Galih’ itu.

**// Skipping items / Menskip elemen nya**

Nah sekarang kita juga bisa melompati elemen-elemen yg ada pada array misalnya saya gamau semuanya di petakkan kedalam variable, saya ingin hanya variable salam dan nama saja variable satu dan dua nya gamau ditaro situ.

Nah ini caranya sama tinggal satu dan dua nya saya hapus tapi koma nya jangan ( , ) begini :

const perkenalan = ['Halo', 'nama', 'saya', 'Sandhika Galih'];

const [salam, , , nama] = perkenalan;

console.log(nama);

Jadi nulis nya [salam, , , nama] harus tetap ada koma nya gitu dan sekarang string “nama” dan string “saya” gamasuk kedalam variable terpisah jadi kalo saya console.log(dua) itu dia eror karena emang gaada.

**// Swap items / Menukar isi array**

Nah dan lain nya juga kita bisa melakukan Swap atau menukar isi dari array ini udah pernah kita coba divideo sebelum nya, jadi kalo misalkan saya punya 2 variable misalkan saya gunakan a dan b, gini :

let a = 1;

let b = 2;

Nah untuk menukar isi nya biasanya kita butuh sebuah variable tambahan buat menampung nilai sementara tapi dengan menggunakan destructuring ini kita bisa langsung aja menggunakan [a,b] = [b,a]; begitu aja isi variable nya jadi tertukar gini :

let a = 1;

let b = 2;

[a, b] = [b, a];

Nah itu langsung isi nya ketuker ya, jadi kalo saya tulis sebelum di assignment itu console.log pasti isi nya nilai asli variable tersebut, nah saya tulis juga sesudah di assignment maka nilai nya jadi berubah gini :

let a = 1;

let b = 2;

console.log(a);

console.log(b);

[a, b] = [b, a];

console.log(a);

console.log(b);

Nah jadi bisa juga sintaks destructuring ini digunakan untuk menukar isi dari variable.

**// Return value pada function** .

Nah lalu kita juga bisa gunakan sebagai return value pada function jadi return pada function nya bisa mengembalikan array dan array nya langsung ditangkap oleh sintaks destructuring itu, contohnya saya punya function namanya coba yg isi nya return array, kalo return array kan boleh ya jadi mengembalikan array gini :

function coba() {

    return [1, 2];

}

Nah kalo misalkan saya punya sebuah variable namanya a di isi dengan function coba() nah nanti ( a ) itu tipe nya array jadi kalo saya console.log(a), gini :

function coba() {

    return [1, 2];

}

const a = coba();

console.log(a);

Nah kalo saya mau ambil angka 2 di array itu berarti saya tinggal tulis a index ke-1 jadi nya console.log(a[1]); gitu. Nah tapi dengan menggunakan destructuring variable ini kita bisa langsung simpan di variable nya begini :

function coba() {

    return [1, 2];

}

const [a, b] = coba();

console.log(a);

Tuh jadi langsung dipetakkan kedalam variable nya [a, b] gitu, jadi 1 masuk ke variabl a dan 2 masuk ke variable b.

**// Rest parameter**

Dan kalo misalkan kalian mau ngebongkar nya banyak nilai array nya tapi kalian nanti nya gatau akan ada berapa elemen nah kalian bisa gunakan Rest parameter yg titik 3 itu loh di video sebelum nya kita sudah menggunakn.

Jadi kalo misalkan kalian punya variable a dan b destructuring nay 2 nilai nya kalo saya ngambil array nya 2 itu ga masalah kan gini : const [a, b] = [1, 2]; tuh tapi gimana kalo array nya ada 3 jadi nya [1, 2, 3] gitu berarti kita harus tambah variable nya juga dong, terus kalo nilai array nya sampai 10 gimana berarti kita harus nampung 10 variable juga dong.

Nah disini kita bisa gunakan rest Parameter, jadi mau disimpan dimana nih misal kita simpan di variable pertama deh, kalian mau simpan dimana aja bebas ya asal pastikan isi array nya lebih banyak dari pada destructuring nya, jadi kalian tinggal tambahkan titik 3kali lalu nama variable nya apa, gini :

const [a, ...values] = [1, 2, 3, 4, 5, 6, 7];

console.log(a);

console.log(values);

Jadi angka 1 akan masuk kedalam variable a dan sisanya masuk kedalam array yg namanya values jadinya sisanya jadi array lagi jadi sama kita bisa manfaatin rest Parameter.

**// OBJECT**

Misalnya kita punya varible isi nya gini :

const mhs = {

    nama: 'Sandhika Galih',

    umur: 33

};

Nah sekarang gimana kalo kita mau menangkap tiap-tiap property nya sebagai variable biasa kan gini :

const mhs = {

    nama: 'Sandhika Galih',

    umur: 33

};

const nama = mhs.nama;

const umur = mhs.umur;

Nah kalo sekarang cukup gunakan kurung kurawal lalu tuliskan nama variable nya sesuai dengan nama property dari object nya, jadi ini harus sama nama nya kalo tadikan array bebas ya, gini :

const mhs = {

    nama: 'Sandhika Galih',

    umur: 33

};

const { nama, umur } = mhs;

Jadi itu gaboleh sembarang nama variable nya, dia harus sama dengan yg di object itu sendiri. Nah keren nya kalo object itu kita bisa lakukan destructuring tanpa melakukan deklarasi object nya kaya di atas, kalo itukan kita bikin dulu object nya pake object literal ya.

**// Assignment tanpa declaration object** .

Nah kita bisa langsung melakukan destructuring pada object, kita langsung membuat nya disitu terus kita ga perlu tanda const tapi syarat nya kita harus kasih kurung buka dan tutup, gini :

({ nama, umur } = {

    nama: 'Sandhika Galih',

    umur: 33

});

console.log(nama);

Kalo kaya gitu gausah deklarasi object, kalo object nya simpel bisa lakukan itu tapi kalo object nya banyak bisa gunakan deklarasi object saja.

Sekarang gimana kalo kita mau ngasih nama baru ga sesuai sama nama property nya misalkan nama dari porperty nya terlalu panjang atau bahkan kalian ingin namanya yg laen, oke kita ambil contoh lagi yang tadi.

**// Assignment ke variable baru**

const mhs = {

    nama: 'Sandhika Galih',

    umur: 33

};

const { nama, umur } = mhs;

console.log(nama);

Nah itukan sama ya nama variable nya, nah kalo mau beda kalian cukup titik 2 spasi nama barunya mau apa misalkan gini :

const mhs = {

    nama: 'Sandhika Galih',

    umur: 33

};

const { nama: n, umur: u } = mhs;

console.log(u);

Jadi sekarang kalo mau manggil nama jangan nama tapi ( n ) kalo mau umur berarti ( u ), oke jadi itu di assign ke variable baru.

**// Memberikan Default Value** .

Dan selanjutnya kita juga bisa memberikan default value, ini seperti di dalam PHP, didalam function kita bisa memberikan nilai default untuk destructuring juga sama bisa, kita ambil contoh yg tadi begini :

const mhs = {

    nama: 'Sandhika Galih',

    umur: 33

};

const { nama, umur } = mhs;

console.log(nama);

Kalo misalnya didalam destructuring nya saya kasih dia email lalu saya console.log email kira-kira yg terjadi apa dan yg tampil adalah undefined, gini :

const mhs = {

    nama: 'Sandhika Galih',

    umur: 33

};

const { nama, umur, email } = mhs;

console.log(email);

Yang tampil undefined itu karena di cek kedalam object nya gaada property email nya kalo ada langsung tampil tapi karena gaada undefined deh. Nah kita bisa kasih langsung disitu nya sebagai nilai default, gini :

const mhs = {

    nama: 'Sandhika Galih',

    umur: 33

};

const { nama, umur, email = 'email@default.com' } = mhs;

console.log(email);

Jadi dia cek dulu ada email ga kalo gaada pake yg nilai default nya tapi kalo saya kasih email di object nya maka yg tampil yg di object nya gini :

const mhs = {

    nama: 'Sandhika Galih',

    umur: 33,

    email: 'sandhikagalih@unpas.ac.id'

};

const { nama, umur, email = 'email@default.com' } = mhs;

console.log(email);

Jadi kita bisa kasih default value dan keduanya bisa kita gabungkan sama yg sebelum nya itu, kita bisa memberi nilai default dan juga assign ke variable baru :

const mhs = {

    nama: 'Sandhika Galih',

    umur: 33,

    email: 'sandhikagalih@unpas.ac.id'

};

const { nama: n, umur: u, email: e = 'email@default.com' } = mhs;

console.log(email);

Nah jadi kita mempersingkat atau mengganti nama variable yg seharusnya nama itu sam dengan nama property jadi nya kita persingkat, tapi memberi nama satu huruf seperti itu tidak bagus ya gays.

Jadi sekarang itu kalo dijalanin EROR karena email tidak di kenali karena yg dikenali adalah E gitu ganti console.log nya jadi console.log(e);

/**/ Rest Parameter Object** .

Nah sama juga ya kalo misalkan kalian mau bikin dia menggunakan rest parameter ya, misalkan pertama parameter nya nama lalu yg kedua baru rest parameter :

const mhs = {

    nama: 'Sandhika Galih',

    umur: 33,

    email: 'sandhikagalih@unpas.ac.id'

};

const { nama, ...value } = mhs;

console.log(value);

Nah jadi property nama masuk ke variable nama dan property sisanya masuk ke rest parameter yg si value itu dan hasil akhirnya juga object bukan array karna itu awal nya object kan, jadi value itu bentuk nya object.

Jadi kalo nanti kalian gatau nanti didalam object yg di assign nya ada berapa property bisa di tangkap pak rest parameter, sekali lagi nama rest parameter bebas asal ada titik 3 di depannya ( …variable ).

**// Mengambil filed pada object, setelah dikirim sebagai parameter untuk function**.

Ini ketika kita mau mengambil field pada object yg object itu kita kirim sebagai parameter untuk function, kita ambil contoh gini :

const mhs = {

    id: 123,

    nama: 'Sandhika Galih',

    umur: 33,

    email: 'sandhikagalih@unpas.ac.id'

};

Cerita nya kan setelah kita ngambil data dari API biasanya kan tuh ada ( id ) nya kan ya, Nah sekarang sebetulny kita bisa bikin function untuk ngambil id dari mahasiswa gitu ya, misalkan getIdMhs() gitu.

Nah nanti function itu akan ngambil object mahasiswa nya yg nanti akan mereturn mhs.id, jadi kalo saya console.log gini :

function getIdMhs(*mhs*) {

    return *mhs*.id;

}

console.log(getIdMhs(mhs));

Maka hasil dari console nya harusnya 123 ya sesuai isi dari id nya ga aneh ya itu biasa aja, yg dikirim object yg diterima function nya object baru yg dikembalikan salah satu property dari object nya aja.

Nah sekarang dengan menggunakan destructuring ini kita bisa simpan sebagai parameternya itu ga object padahal dikirim nya itu object, jadi yg dikirim nya itu object tapi yg ditangkap disitu hanya id nya saja tapi pake kurung kurawal, Jadi sisa property nya itu di abaikan sehingga yg saya return bisa id nya saja :

const mhs = {

    id: 123,

    nama: 'Sandhika Galih',

    umur: 33,

    email: 'sandhikagalih@unpas.ac.id'

};

function getIdMhs({ *id* }) {

    return *id*;

}

console.log(getIdMhs(mhs));

Jadi dikirimkan object masuk dibagian parameter yg id itu, itu di bongkar dulu yg diambil hanya id nya atau kalo butuh yg lain silahkan tambahkan disitu, misalkan butuh nama gini :

const mhs = {

    id: 123,

    nama: 'Sandhika Galih',

    umur: 33,

    email: 'sandhikagalih@unpas.ac.id'

};

function getIdMhs({ *id*, *nama* }) {

    return `id: ${*id*}, ini punya ${*nama*}`;

}

console.log(getIdMhs(mhs));

Nah begitu jadi gasemua, jadi yg dikirim nya itu object lengkap lalu yg diambil nya hanya beberapa saja boleh.