**JAVASCRIPT LANJUTAN | 5.2 Template Literals (Latihan)**

Sebelum nya kita inget” dulu Template Literal ini bisa melakukan hal-hal ini ya :

* Menggunakan `` back tick
* Multi-line String
* Embedded Expression
* HTML Fragments
* Expression Interpolation
* Tagged Template

Nah penjelasan itu semua udah ada di video sebelum nya ya. Nah sekarang kita akan lihat lebih lanjut mengenai HTML Fragments jadi nanti kita akan liat bagaimana penggunaan template literal ini untuk membentuk sebuah elemen HTML yg nanti nya akan kita simpan di DOM.

Dan nanti didalam nya kita bisa simpan banyak hal, kita bisa simpan pengulangan pengokondisian bahkan nanti kita bisa simpan Expression didalam Fragments nya itu bersarang.

Sekarang yg pertama kita akan coba kita liat dulu bagaimana cara membuat HTML Fragments misalnya contoh kasus saya punya sebuah object sederhana seperti ini :

const mhs = {

    nama: 'Sandhika Galih',

    umur: 33,

    nrp: '043040023',

    email: 'sandhikagalih@unpas.ac.id'

};

Nah misalnya saya punya variabel lagi untuk element nya yg akan dirangkai sebagai HTML Fragments kita bikin menggunakan back tick atau template literal gini :

const el = `<div class="mhs">

              <h2>${mhs.nama}</h2>

              <span class="nrp">${mhs.nrp}</span>

            </div>`;

console.log(el);

tapi kita tidak akan tampilkan di console, misalkan kita mau tampilkan nya kedalam DOM seperti ini :

const el = `<div class="mhs">

              <h2>${mhs.nama}</h2>

              <span class="nrp">${mhs.nrp}</span>

            </div>`;

document.body.innerHTML = el;

Nah nanti yg nongol di layar hanya nama sama nrp nya aja, itulah yg paling sederhana, nah sekarang coba bayangkan gimana kalo misalkan kalian punya object mahasiswa nya banyak ada lebih dari satu mahasiswa. Kan mungkin aja seperti query ke databes atau kalian minta data ke sebuah API data mahasiswa di kembalikan nya banyak.

Jadi sekarang topik ke-2 nya adalah looping, bagaimana mengulang data yang didapatkan jadi misalkan saya punya variabel mhs lagi tapi sekarang isi nya array yg didalam nya ada object, seperti ini :

const mhs = [

    {

        nama: 'Sandhika Galih',

        email: 'sandhikagalih@unpas.ac.id'

    },

    {

        nama: 'Doddy Ferdiansyah',

        email: 'doddy@unpas.ac.id'

    },

    {

        nama: 'Erik',

        email: 'erik@unpas.ac.id'

    }

];

Sekarang gimana cara kita untuk menampilkan data kedalam DOM nya menggunakan HTML Fragments yg kita looping langsung didalam template literal nya, jadi sama ya kita punya div terus kita melooping cara looping nya kita menggunakan Expression yg didalam nya kita panggil mathod map() ya.

Itu higher order function ke array mahasiswa gitu lalu kita looping menggunakan map() jadinya mhs.map(), gini :

const el = `<div class="mhs">${mhs.map(*m* =>

    `<ul>

        <li>${*m*.nama}</li>

        <li>${*m*.email}</li>

    </ul>`)}

</div>`;

document.body.innerHTML = el;

Jadi memang lumayan rumit kalo dipikir” tapi coba pahamin perlahan ya. Nah disitu data nya tampil tapi pas tampil ada tanda koma nya ( , ) karena bentuk nya array nah untuk menghilangkan nya kalian bisa pake method join() kita chaining ya lalu isi dengan string kosong supaya koma nya ilang :

const el = `<div class="mhs">${mhs.map(*m* =>

    `<ul>

        <li>${*m*.nama}</li>

        <li>${*m*.email}</li>

    </ul>`).join('')}

</div>`;

document.body.innerHTML = el;

Itulah kalo mau menggunakan looping ya.

**PENGKONDISIAN**

Sekarang gimana kalo mau melakukan pengkondisian, sebetulnya kita bisa pakai ternary seperti di video sebelum nya, misalnya kalian punya data daftar lagu ada judul lagu dan ada penyanyi nya, gini :

const lagu = {

    judul: 'Tetap Dalam Jiwa',

    penyanyi: 'Isyana Saraswati'

}

Kalo kita mau menampilkan kedalam HTML Fragments gampang ya gini :

const el = `<div class="lagu">

        <ul>

            <li>${lagu.penyanyi}</li>

            <li>${lagu.judul}</li>

        </ul>

</div>`

document.body.innerHTML = el;

Kalo gitu gaada yg aneh kan, Cuma problem nya ada lagu yg punya fituring jadi misalnya data yg kalian dapatkan ada property feat di object nya, gini :

const lagu = {

    judul: 'Kau Adalah',

    penyanyi: 'Isyana Saraswati',

    feat: 'Rayi Putra'

}

const el = `<div class="lagu">

        <ul>

            <li>${lagu.penyanyi}</li>

            <li>${lagu.judul} (feat. ${lagu.feat})</li>

        </ul>

</div>`

document.body.innerHTML = el;

NAH tapi masalah nya adalah ga semua lagu ada feat nya jadi ketika lagu nya yg sebelum nya gaada feat nya, terus kalo property feat nya ilang dan kita masih tulis feat di html nya nanti feat nya undefined yg ada.

Nah gimana kalo kita kasih pengkondisian, jadi saya jalankan expresi lagi yg isi nya jika ada lagu.feat maka tambahkanstring pake template literal lagi, gini :

const el = `<div class="lagu">

        <ul>

            <li>${lagu.penyanyi}</li>

            <li>${lagu.judul} ${lagu.feat ? `(feat. ${lagu.feat})` : ''}</li>

        </ul>

</div>`

document.body.innerHTML = el;

Jadi ketika object lagu itu ada property feat dia bakal tampil terus kalo gaada feat nya dia ga tampil apapun.

**BERSARANG / Nested**

Misalnya kita punya object mahasiswa lagi :

const mhs = {

    nama: 'Sandhika Galih',

    semester: 5,

    mataKuliah: [

        'Rekayasa Web',

        'Analisis dan Perancangan Sistem Informasi',

        'Pemrograman Sistem Interaktif',

        'Perancangan Sistem Berorientasi Object'

    ]

};

Nah gimana cara menampilkan semua data itu kedalam html Fragments, gini dulu :

const el = `<div class="mhs">

    <h2>${mhs.nama}</h2>

    <span class="semester">Semester : ${mhs.semester}</span>

    <h4>Mata Kuliah :</h4>

    ${cetakMataKuliah(mhs.mataKuliah)}

</div>;

Nah untuk mata kuliah nya kita bisa menggunakan langsung Higher Order Function di dalam variabel element nya jadi digabung atau seperti contoh di atas itu kita menggunakan Function jadi tinggal manggil nama function nya aja.

Jadi function nya itu seperti ini :

function cetakMataKuliah(*mataKuliah*) {

    return `<ol>

                ${*mataKuliah*.map(*m* => `<li>${*m*}</li>`).join('')}

            </ol>`;

};

Dan disitu kita hilangkan koma nya menggunakan join() itu di chaining ya. Nah kunci nya itu pahamin telusurin perlahan karena semua harus plan plan dulu ya.