**JAVASCRIPT LANJUTAN 13. Fetch Refactor (Async Await)**

Disini kita akan perbaiki code kita atau istilah nya refactoring sehingga kode ketika lebih rapih lebih terstruktur dan lebih mudah dibaca. Jadi kita buka code sebelum nya tapi yg Jquery nya udah saya apus jadi di folder ini hanya ada yg fetch saja.

**Pertama kita akan perbaikin saat menampilkan card nya :**

Sebetulnya code kita gaada masalah ya sudah berjalan dengan normal tapi kita bisa perbaiki supaya lebih mudah dibaca lagi. Gini :

const searchButton = document.querySelector(".search-button");

searchButton.addEventListener("click", function () {

   const inputKeyword = document.querySelector(".input-keyword");

});

Sampe sini masih sama ya, nah kalo sebelum nya kan kita langsung tulisin fetch() nya ya didalam situ, nah alangkah lebih baik nya fetch itu kita tarik atau kita abstraksi kedalam fungsi yg berbeda dan nanti saat kita menampilkan card nya kedalam document kita juga akan simpan kedalam fungsi yg berbeda juga, supaya gampang baca code nya.

Gini deh :

const searchButton = document.querySelector(".search-button");

searchButton.addEventListener("click", function () {

  const inputKeyword = document.querySelector(".input-keyword");

  const movies = getMovies(inputKeyword.value);

  updateUI(movies);

});

Jadi nanti kita buat fungsi nya dulu diluar itu, jadi enak tinggal kita panggil aja fungsi nya terus jalan deh gampang dimengerti nya juga kan. Pertama kita buat function getMovies gini :

**// Fungsi getMovies()**

const searchButton = document.querySelector(".search-button");

searchButton.addEventListener("click", function () {

  const inputKeyword = document.querySelector(".input-keyword");

  const movies = getMovies(inputKeyword.value);

  console.log(movies);

  updateUI(movies);

});

function getMovies(*keyword*) {

  return fetch("http://www.omdbapi.com?apikey=78c0d1a4&s=" + *keyword*)

    .then((*response*) => *response*.json())

    .then((*response*) => *response*.Search);

}

Jadi artinya kalo kita cari nih misal Disney maka yg muncul 10 film disney, coba deh console.log(movies) nya kek di atas gitu. Nah pas di cek ternyata yg kembali itu adalah promise yg pending kan kita mau nya data film nya ya, padahal disitu kita udah menggunakan then.

Nah perhatikan nih ketika kita berusaha untuk bikin code kita semakin modular menggunakan cara seperti itu yg kalian mesti pahami cara kita itu adalah cara yg Synchronous, jadi begitu getMovies() dijalankan dia langsung jalan dan ketika pertama kali getMovies() dijalankan karena dia promise maka pending kan setelah data balik harusnya hasilnya resolve tapi keburu kita masukin ke console.log.

Jadi kalo kalian paham mengenai event loop itu udah kelewat keburu di tampilin karena codingan kita itu codingan yg synchronous, javascript nya gatau kalo getMovies() adalah sebuah fungsi yg Asynchronous.

Nah disinilah kita bisa memanfaatkan keyword Async dan Await ini penting ya walaupun penjelasan lebih detail nya ada di video selanjut nya ya. Nah ini sekalian aja karena kita pengen ngerapihin codingan kita.

Jadi cara nya kita harus kasih tau dulu ke javascript nya bahwa fungsi callback itu didalam nya bakalan ada sesuatu yg Asynchronous, cara ngasih tau nya itu di depan keyword function nya kasih keyword async dan terus kita kasih await juga supaya javasciprt nya tau yg mana sih sciprt yg asynchronous nya, gini :

const searchButton = document.querySelector(".search-button");

searchButton.addEventListener("click", async function () {

  const inputKeyword = document.querySelector(".input-keyword");

  const movies = await getMovies(inputKeyword.value);

  console.log(movies);

  updateUI(movies);

});

Tuh liat sebelum fungsi getMovies() ada keyword await yg tanda nya dia ngasih tau bahwa fungsi itu bersifat Asynchronous, jadi await gunanya itu biar nunggu dulu sampai promise nya jadi resolve baru deh masukin ke variable movies baru deh cetak ke console.log. Jadi kalo kalian coba lagi udah berhasil ngambil data film nya.

**// fungsi updateUI()**

Kalo udah gitu tinggal kita buat deh fungsi selanjut nya gini :

function updateUI(*movies*) {

  let cards = "";

*movies*.forEach((*e*) => {

    cards += showCard(*e*);

  });

  const movieContainer = document.querySelector(".movie-container");

  movieContainer.innerHTML = cards;

}

Jadi sebenernya sama aja isi nya kalo fungsi yg updateUI() sama codingan yg sebelum nya itu bedanya dia menerima parameter dan codingan nya dimasukin ke fungction gitu.

Jadi udah gitu aja dan kasus nya sama aja tapi code kita jadi lebih rapih dan mudah dibaca dan lebih masuk akal. Jadi singkat nya adalah itu cara kita untuk membuat coding kita terlihat seperti Asynchronous padah Synchronous.

**Tahap kedua kita menampilkan showDetail :**

Dicodingan yg sebelum nya event handler kita berada didalam method then punya nya fetch yg pertama, nah sekarang kita mau bikin supaya diluar indepeden ya supaya kita ga ngeganggu fungsi yg lain nya.

Ketika tombol detail di klik sayang nya kita gabisa langsung nulisin event handler nya gini :

const mdButton = document.querySelectorAll(".modal-detail-button");

sayang nya itu gabisa kita lakukan sekarang karena baris itu akan dijalankan ketika javscript nya di eksekusi pertama kali, jadi ketika kita baru ngebuka halaman nya baris itu udah di eksekusi, dan ketika baris itu udah di eksekusi coba ada ga element yg punya nama class tersebut. Sekarang gaada karena masih kosong container nya karena kan kita baru aja ngebuka halaman nya belum cari nama filmnya.

const mdButton = document.querySelectorAll(".modal-detail-button");

mdButton.forEach((*btn*) => {

*btn*.addEventListener("click", function () {

    console.log("ok");

  });

});

Ini saya coba buat ketika tombol nya diklik menggunakan looping, kalo kalian coba klik showDetail nya dan ternyata tidak terjadi apa-apa harusnya kan tampilih ‘ok’ di console nya, ini karena tombol event hendler nya sudah berjalan ketika tombol showDetail nya belum ada, jadi nya kita gabisa pake cara itu ya.

**Cara ngakalin nya supaya bisa menggunakan**

**// event binding**

Jadi kita ngasih event ke element yg awal nya belum ada tapi ketika dia ada event nya masih tetap berjalan, cara nya kalian bisa simpan event handler nya kedalam element document, document tuh semua element yg ada di websaite nya, gini :

*// event binding*

document.addEventListener("click", function (*e*) {

  console.log(*e*.target);

});

Jadi kalo misalnya kita klik tulisan ‘WPU Movie DB’ nah nanti liat di console nya yg muncul itu element h1 nya liat sendiri deh, jadi nya kalo gitu ketika kita klik tombol showDetail nya yg muncul element button itu sendiri, jadi nya kita bisa gini :

*// event binding*

document.addEventListener("click", function (*e*) {

  if (*e*.target.classList.contains("modal-detail-button")) {

*// lakukan sesuatu yg kalian ingin kan*

*// lakukan showDetail nya*

  }

});

Jadi jika e.target itu nama class nya (contains atau berisi) atau mengandung kata “modal-detail-button’ maka jalankan perintah berikut, tinggal perintah nya itu kaya codingan sebelum nya aja udah. Gini :

*// event binding*

document.addEventListener("click", function (*e*) {

  if (*e*.target.classList.contains("modal-detail-button")) {

    const imdbid = *e*.target.dataset.imdbid;

    const movieDetail = getMovieDetail(imdbid);

    updateUIDetail(movieDetail);

  }

});

Nah itu udah tinggal jalanin aja, untuk fungsi nya seperti ini :

function getMovieDetail(*imdbid*) {

  return fetch("http://www.omdbapi.com?apikey=78c0d1a4&i=" + *imdbid*)

    .then((*response*) => *response*.json())

    .then((*r*) => *r*);

}

Nah yg parameter ‘r’ itu kenapa gaada Search nya seperti fungsi yg getMovie() sebelum nya karena ini bukan untuk search melainkan hanya menampilkan data nya ya gitu. Ini untuk yg updateUIDetail() :

function updateUIDetail(*m*) {

  const movieModal = showMovieDetail(*m*);

  const movieContainer = document.querySelector(".modal-body");

  movieContainer.innerHTML = movieModal;

}

Seperti itulah codingan nya dan isi fungsi nya seperti itu saja, nahh kita kelupaan ingat asyn dan await kita masih butuh itu untuk membuat showDetail, gampang aja kita tambahin jadi gini :

*// event binding*

document.addEventListener("click", async function (*e*) {

  if (*e*.target.classList.contains("modal-detail-button")) {

    const imdbid = *e*.target.dataset.imdbid;

    const movieDetail = await getMovieDetail(imdbid);

    updateUIDetail(movieDetail);

  }

});

Nah udah selesai kita merapikan codingan nya menjadi lebih masuk akal ya.