Keyword Var, Let dan Const

Keyword var, let dan const sama-sama untuk membuat atau mendeklarasikan variabel.  
Lalu apa perbedaan var, let dan const? Kapan waktu terbaik kita menggunakannya?

Tiga keyword tersebut memiliki perbedaan:

1. var dan let isinya dapat diubah kembali, sedangkan const (konstanta) isinya tidak bisa diubah.
2. var akan selalu terkena hoisting, apa itu hoisting? penjelasannya ada di bawah ya.
3. var adalah function scope sedangkan let dan const adalah block scope.

Block adalah kode apa pun yang ada dalam kurung kurawal {} seperti fungsi, pernyataan kondisional dan perulangan.

Contoh penggunaan var dan let silahkan perhatikan kode di bawah ini:

var contoh1 = 'isi awal';

let contoh2 = 'isi awal';

console.log(contoh1); // isi awal

console.log(contoh2); // isi awal

contoh1 = 'isinya sudah diubah';

contoh2 = 'isinya sudah diubah';

console.log(contoh1); // isinya sudah diubah

console.log(contoh2); // isinya sudah diubah

Kode di atas akan berjalan dengan baik, tapi coba kalian ubah var atau let di kode baris ke 1 atau ke 2 dengan keyword const dan jalankan ulang kodenya, pasti kalian akan menemukan pesan error atau kode tidak menghasilkan output dengan maksimal.

**Apa itu hoisting?**  
Untuk memahami apa itu hoisting silahkan perhatikan terlebih contoh di bawah ini:

angka = 10; // mengisi nilai variabel angka

console.log(angka);

var angka; // mendeklarasikan variabel angka

Kenapa saat kode di atas saat dijalankan tidak menghasilkan error?, padahal kita baru bikin variabel angka nya di baris ke 5. Jadi setiap variabel yang menggunakan keyword var akan dipindahkan oleh JavaScript ke paling atas, jadi kode di atas sama seperti kode di bawah ini:

var angka; // mendeklarasikan variabel angka

angka = 10; // mengisi nilai variabel angka

console.log(angka);

Jika kita menggunakan let seperti contoh di atas maka kita akan mendapatkan error atau output yang tidak sesuai, karena let tidak di-hoisting oleh JavaScript, perhatikan kode di bawah ini:

angka = 10; // mengisi nilai variabel angka

console.log(angka);

let angka; // mendeklarasikan variabel angka

Jika kode di atas kalian jalankan maka kalian akan mendapat error atau output tidak sesuai.

**Apa itu scope?**  
Scope dalam JavaScript adalah konsep yang digunakan untuk membatasi pengaksesan suatu variabel.

**Apa itu global scope?**  
Global scope adalah scope di luar block.

**Apa itu local scope?**  
Local scope adalah scope di dalam block (terdiri dari function scope dan block scope).

**Apa itu function scope dan block scope?**  
Untuk mengilustrasikan perbedaannya, perhatikan kode berikut:

// global scope

var contohVar = "ini contohVar global scope";

let contohLet = "ini contohLet global scope";

const contohConst = "ini contohConst global scope";

console.log("memanggil variabel di global scope:");

console.log(contohVar);

console.log(contohLet);

console.log(contohConst);

function test() {

// function scope (karena di dalam function)

var contohVar = "ini contohVar function scope";

let contohLet = "ini contohLet function scope";

const contohConst = "ini contohConst function scope";

console.log("\nmemanggil variabel di function scope:");

console.log(contohVar);

console.log(contohLet);

console.log(contohConst);

}

test();

if (true) {

// block scope (karena di dalam block if)

var contohVar = "ini contohVar block scope";

let contohLet = "ini contohLet block scope";

const contohConst = "ini contohConst block scope";

console.log("\nmemanggil variabel di block scope:");

console.log(contohVar);

console.log(contohLet);

console.log(contohConst);

}

console.log("\nmemanggil variabel di global scope:");

console.log(contohVar);

console.log(contohLet);

console.log(contohConst);

Jika kode di atas dijalankan, di baris ke 7 dan ke 38 hasilnya beda padahal sama-sama di global scope, kenapa begitu? itu adalah kekurangan keyword var, jadi sebenarnya di baris ke 27 itu kita dianggap menghapus contohVar yang ada pada global scope dan dibuat ulang di baris itu.