• github : divyaranjan10

DATE

CHAP 8: let & const in JS, Temporal Dead Zone

" let and const declarations are hoisted (but they are in temporal dead zone for the time being)."

```
console.log(b) -> It will throw an reference
var a = 100;      error (Cannot access 'b'
let b = 200;      before initialization)
```

↳ Why? : let and const are hoisted but it's memory is allocated at other place than window which cannot be accessed before initialization. [Summary points by Jagruti Sharma]

Temporal Dead Zone : So, it is the time since the 'let' variable is hoisted and till it's initialized some value, that time between them is TDZ.

DATE

"Window. variable or this. variable will not give value of variable defined using let or const." [Summary points by Tarpat Sharma]

"We cannot redeclare the same variable with let / const (even with using 'var' the second time)." For ex: let a = 20; let a = 30; (X)

- Diff b/w const and let

⇒ It (const) is even more strict than let as const variable declaration and initialization must be done on the same line, unlike in 'let' you can declare but can initialize its value later.

For ex: const a;

a = 100; "It will threaw an error"

[Missing initializer in const declaration]

But in case of 'let':

let b;

b = 100; "This will work"

DATE

- Difference b/w Syntax Error vs Type Error vs Reference Error

a) Reference Error: When JS engine tries to find a specific variable inside the memory space and can't access it, then it gives a reference error.

b) Type Error: Occurs when we change type that is not supposed to be changed.

For ex: const a = 100;

a = 200;

c) Syntax Error: when proper syntax (way of writing a statement) is not used.

- How to avoid Temporal Dead Zone?

=> Initializing variables at the top is good idea, helps shrinks TDZ to zero.