· github : divyaranjan10

DATE

CHAP 9 : Block Scope and Shadowing in JS

· What is a Block?
→ So, the multiple JS statements formed in a group that is enclosed with curly brackets, is called a block.

Ex:
```
{
      var a = 10;
      let b = 20;
      const c = 30;
    }
```

"Block Scope means all the variables or functions we can access inside the specified block.

** Block values are stored inside separate memory than global. They are stored in block. (the reason let and const are called block scope). [Summary points by Jagrut Sharma]

DATE

· Shadowing in JS: It occurs when a variable declared in a certain scope has the same name as a variable in an outer scope. When this happens, the outer variable is said to be shadowed by inner variable.

· For ex: (9.1)

```
var a = 100;
{
    var a = 200;
    console.log (a);
}
console.log (a);
```

· Output:

```
200
200
```

(9.1) So, in this example, 'variable a of inner scope' ~~showed~~ shadowed the 'variable a of outer scope'.

· Shadowing in let: (9.2)

```
let a = 100;
{
    let a = 200;
    console.log (a);
}
console.log (a);
```

· Output

```
200
100
```

DATE________

(9.1) So, in this example, first 200 is printed 'let is block scoped', and later 100 is printed as ~~not~~ now 'let a of outer scope' is global.

* The ~~other~~ shadowing is same in const as let *

* Shadowing is not just a concept of block, it shows the same behavior in function *

Ex:

```
const c = 100;
function x () {
      const c = 30;
      console.log (c);
}
x();
console.log(c);
```

Output:

```
30
100
```

DATE

• Illegal shadowing in JS: Shadowing let with var is illegal shadowing and gives error.

Ex:
```
let a = 100;
{
    var a = 20;
}
```

* Block scope also follows lexical scope *

Ex 1

```
const a = 100;
{
  const a = 200;
  {
    const a = 300;
    console.log(a);
  }
}
```
• Output : 300

Ex 2

```
const a = 100;
{
  const a = 200;
  {
    console.log(a);
  }
}
```
• Output : 200

* All the scope rules that follows on normal functions will work on arrow-functions too *