JS

CLOSURE AND SCOPE

¿QUÉ ES EL SCOPE?

El alcance que determina la
accesibilidad de las variables
(Donde las podemos leer)

Tipos de Scope

01

GLOBAL

Variables declaradas fuera de funciones. accesibles en cualquier parte tienen alcance global (Global scope)

02

BLOCK

Con las palabras reservadas let y const, no se puede acceder a dichas variables creadas en un bloque de codigo desde afuera del bloque. Var no tiene alcance de bloque

03

LOCAL

Variables declaradas dentro de una función cuyo alcance solo es dentro de dicha función (Function Scope)

04

FUNCTION

Cada función crea un nuevo alcance, dentro de cada función; var, let y const se comportan de manera similar (Function Scope)

CLOSURE

Teniendo en cuenta que en javascript todo hereda de la clase Object, se puede decir que una función es un objeto, por lo que puede ser guardada dentro de una variable

```
const a = "Hey!";

function global() {
  const b = "¿Qué";

  function local() {
    const c = "tal?"
    return a + b + c;
  }

  return local;
}
```

return local es como si "regresara una variable" y no un metodo a no estar ejecutandose en paréntesis (aunque si se ejecuta)

CLOSURE

Ejemplo

```
function moneyBox() {
  let saveCoins = 0;

  return function savaMoney(coins) {
    return (saveCoins += coins);
  };
}

const MauBox = moneyBox();
const DiegoBox = moneyBox();
```

Lo importante aquí es la persistencia del dato saveCoins, por lo que al ejecutar la función más de una vez, el dato persistirá en las constantes

CLOSURE

Resumen: podemos decir que un closure es una forma o tecnica de hacer que nuestros datos persistan sin necesidad de variables globales, es decir, trabajamos con funciones anidadas que operan variables con el mismo ambito lexico y que regresan uno o mas valores