Desarrollo Blockchain Ethereum con Solidity

Módulo 6 - Implementación con JavaScript

Implementación con JavaScript

Estructura

- Las referencias a los contratos se hacen de la misma forma que en los archivos de migración con **artifacts.require(...)**
- Web3js está disponible en todos los test, por lo tanto podemos utilizar dicha librería sin agregar la referencia (por ejemplo, **web3.eth.getBalance(...)**)
- La estructura general responde a los tests de Mocha js, sólo que se utiliza la palabra *"contract"* en vez de *"describe"*

Estructura - Ejemplo con then

```
const MetaCoin = artifacts.require("MetaCoin");

contract("MetaCoin", accounts => {
  it("should put 10000 MetaCoin in the first account", () =>
    MetaCoin.deployed()
      .then(instance => instance.getBalance.call(accounts[0]))
      .then(balance => {
        assert.equal(
          balance.valueOf(),
          10000,
          "10000 wasn't in the first account"
        );
        }));

  });
```

Estructura - Ejemplo con async/await

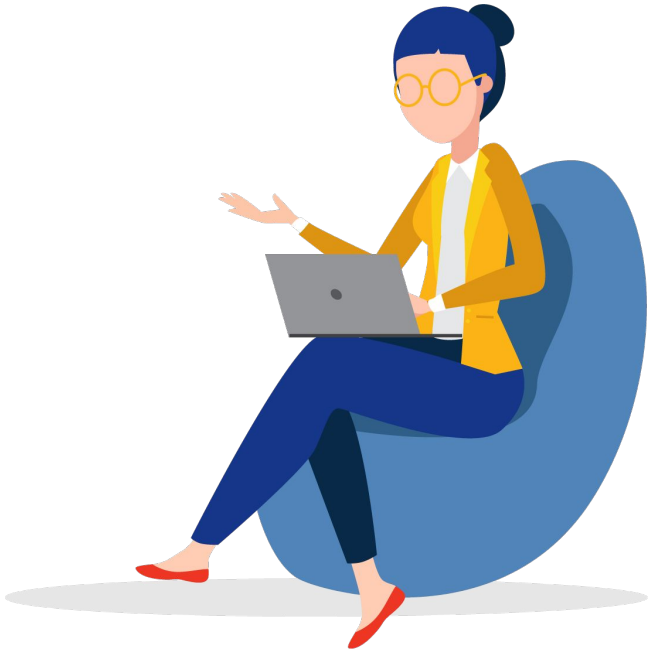
```
const MetaCoin = artifacts.require("MetaCoin");

contract("2nd MetaCoin test", async accounts => {
  it("should put 10000 MetaCoin in the first account", async
() => {
    const instance = await MetaCoin.deployed();
    const balance = await
instance.getBalance.call(accounts[0]);
    assert.equal(balance.valueOf(), 10000);
    });

  });
```

Ejercitación

1. Armar un entorno de pruebas para el contrato de Storage de ejemplo.
2. Considerar agregar permisos de acceso de Owner al contrato para probar no sólo el almacenamiento si no también que sólo los usuarios permitidos puedan transaccionar.
3. Probar los escenarios en contratos ya implementados y también en contratos nuevos.
4. No olvidar probar el valor por defecto del contrato.



¡Muchas gracias!

¡Sigamos trabajando!