Configuración y despliegue de contratos inteligentes en la blockchain de Ethereum y redes compatibles con la EVM

Juan Antonio Lleó

INTRODUCCIÓN: OBJETIVOS

El objetivo de este taller es conocer lo mínimo necesario para configurar y desplegar contratos inteligentes del tipo ERC-20, en Ethereum o en cualquier red compatible con el estándar de la Máquina Virtual de Ethereum (EVM).

Evidentemente, es apenas un punto de inicio pero que puede dar una idea sobre el tema y facilitarnos profundizar en todo ello, si tuviéramos interés.

INTRODUCCIÓN: ESQUEMA

1. INTRODUCCIÓN: ETHEREUM
2. METAMASK
3. FAUCETS
4. EXPLORADORES DE BLOCKCHAINS
5. CONTRATOS INTELIGENTES
6. EDICION Y DESPLIEGUE: REMIX
7. PRACTICA: EJEMPLOS DE SMART CONTRACTS

INTRODUCCIÓN: ETHEREUM I

Ethereum es, después de Bitcoin, la más importante blockchain en la actualidad.

Dispone de una Máquina Virtual, la EVM que le permite ejecutar código con el que interactuar con la blockchain de Ethereum, lo que convierte al ecosistema Ethereum en una especie de ordenador global.

La EVM se ejecuta en los Nodos de Ethereum, que se encargan también de almacenar y actualizar una copia de toda la cadena de bloques de Ethereum.

Existen otras blockchains que usan esta misma tecnología, lo que se conoce como Forks. Cualquier contrato que funcione en Ethereum puede desplegarse también en estas otras cadenas, o en sus versiones de 2ª Capa (Layer 2).

INTRODUCCIÓN: ETHEREUM II

Cada usuario que participe en la red de Ethereum puede tener, al menos una dirección, del tipo:

J A LLEO PROFESOR (Nombre para identificarla):
0x5A92752eD040472ECA7384BA934C4109bD09478E

donde se almacena el estado de su cuenta y con ella se pueden consultar todos los movimientos, desde su creación

Y cada contrato Inteligente tiene también una única dirección, que se le asigna en el momento de su despliegue.

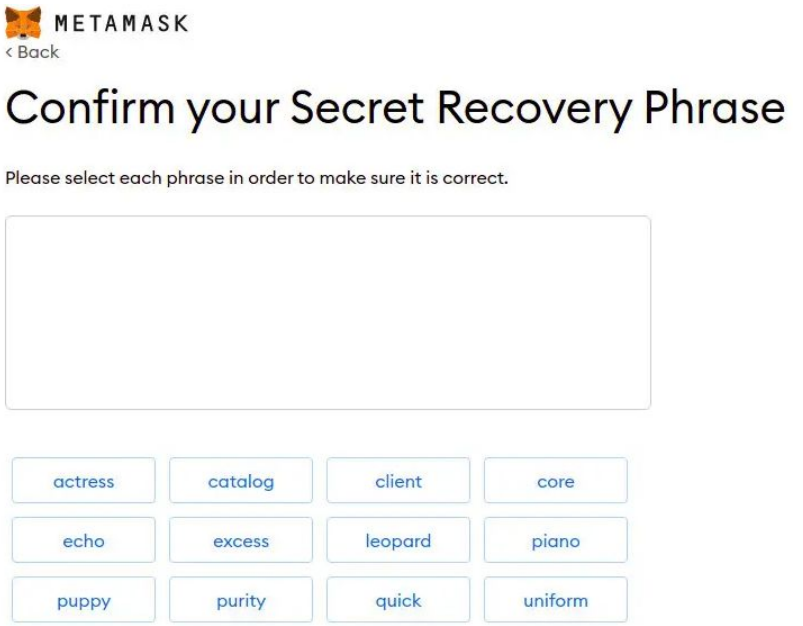
Iremos viendo más características de Ethereum a lo largo del taller.

INTRODUCCIÓN: METAMASK I

Para poder desplegar un contrato (Smart Contract), por ejemplo uno para crear tokens ERC-20, en Ethereum o en cualquier red compatible (EVM), se necesita una cartera (wallet), donde guardemos los datos de nuestra dirección, las claves para poder gestionarlos y los tokens necesarios para poder pagar el coste de las transacciones.

La más usada es Metamask, pero existen otras opciones.
Al conectar con un navegador, te sugiere un enlace para instalarlo en ese navegador.

INTRODUCCIÓN: METAMASK II



Lo más importante a la hora de instalar Metamask es escoger una buena contraseña para acceder a la cartera y guardar con mucho cuidado la semilla para poder recuperar la cuenta.
La frase Secreta de Recuperación son doce palabras que tienen un orden concreto que hay que respetar,

INTRODUCCIÓN: METAMASK III

MAINNET Y TESTNET:

A las redes principales de blockchain se las conoce como Mainnet.

Pero también existen redes de prueba, las testnets que es lo más recomendable para aprender y cuando tenemos que desarrollar proyectos, que las pruebas sean más económicas.

Un ejemplo de redes de pruebas en uso actualmente en Ethereum son Goerli y Sepolia.

INTRODUCCIÓN: FAUCETS

Los Faucets son sitios en los que podemos conseguir tokens de prueba (test) para poder operar en las redes de prueba (Testnets). Últimamente cada vez es más difícil de conseguir los tokens y algunas redes están saturadas, como Goerli, que es una de las principales testnet de Ethereum.

Sobre todo, vamos a usar la red Arbitrum Sepolia, pero se podría usar cualquier otra de las redes de testnet en las que consigamos fondos para pagar las transacciones:

- Sepolia PoW Faucet: https://sepolia-faucet.pk910.de/#/

- QUICKNODE (VARIOS FAUCETS): Hay que tener >=0.001 ETH en la wallet: https://faucet.quicknode.com/

EXPLORADORES DE ETHEREUM: ETHERSCAN

Los exploradores de red nos permiten consultar el contenido de una blockchain y en algunos casos, interactuar con los contratos inteligentes.

Cada red, evidentemente, necesita su propio explorador. Una misma dirección puede usarse en cualquier red compatible con EVM, ya sea mainnet o testnet.

Si introducimos una dirección de una wallet o de un contrato, accederemos a toda la información de la misma, desde el momento en el que se creó.

El más usado es Etherscan, pero hay bastantes opciones más:

https://etherscan.io/

CONTRATOS INTELIGENTES I

Los contratos inteligentes no son más que fragmentos de código que, al compilarse y desplegarse, se pueden ejecutar en la EVM de cualquier blockchain compatible.
El lenguaje más usado es Solidity, pero existen otras opciones.
Su sintaxis no difiere mucho de los lenguajes más usados en la actualidad, pero su ejecución en la EVM hace que sea deseable tener nociones de la misma para sacar un mayor partido.
Es muy importante tener en cuenta aspectos relacionados con la seguridad de los contratos.

CONTRATOS INTELIGENTES II

Uno de los ejemplos más sencillos de contrato inteligente en Solidity es el típico ¡Hola Mundo!, que tendría este aspecto:

HELLO WORLD (VERSION SENCILLA): HelloWorldSimple.sol
https://solidity-by-example.org/hello-world/

```
// SPDX-License-Identifier: MIT

// compiler version must be greater than or equal to 0.8.20 and less than 0.9.0

pragma solidity ^0.8.20;

contract HelloWorld {

    string public greet = "Hello World!";

}
```

EDICION Y DESPLIEGUE: REMIX I

Para que podamos crear un contrato, es deseable usar una herramienta que nos facilite esa tarea, desde presentarnos el código de una forma que podamos entender mejor, con formato y colores adecuados, gestión de errores, etc.

Lo normal es que también nos ofrezcan la posibilidad de compilar el código, según la versión sobre la que vayamos a desplegarlo y nos facilite el despliegue, ya sea en un entorno local, en la nube o en una blockchain, testnet o mainnet.

Remix es una opción muy interesante para todo ello, podemos acceder desde un navegador y nos puede servir tanto para aprender como para desarrollar y desplegar pequeños proyectos. Para desarrollos más importantes, es recomendable usar herramientas más potentes.

CONTRATOS INTELIGENTES: PRACTICAS

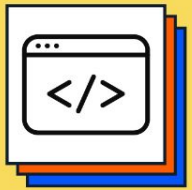
EJEMPLOS:

1. **HOLA MUNDO:** Sencillo e interactivo
   HelloWorldSimple.sol
   HelloWorld.sol (Interactivo)

2. **STORAGE:** Almacena y consulta un valor numérico
   Storage.sol

3. **TOKEN ERC-20** (EN UN SOLO FICHERO SOLIDITY):
   CodemotionSimpleERC20Token.sol

4. **OPEN ZEPPELIN:** CONTRATO ERC-20 (PROFESIONAL)
   OZ_ERC20_Mintable.sol



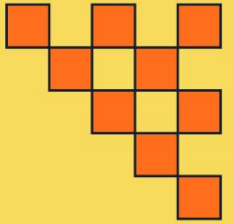
WORKSHOP FEST

¡Contacta conmigo!

@Juan_A_Lleo

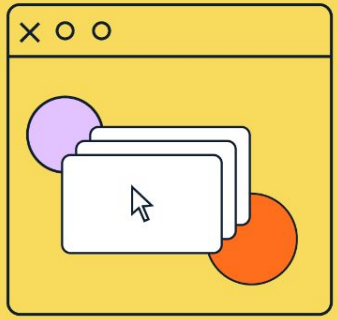
@juanantoniolleo

Juan Antonio Lleó



WORKSHOP FEST

¡Gracias!



WORKSHOP FEST

¡Recuerda puntuar el Workshop!