Primeira Criptomoeda Ethereum

Este repositório contém a implementação de um desafio para a criação de uma criptomoeda ERC-20 simples utilizando a plataforma Ethereum e o Hardhat para desenvolvimento, teste e deploy de contratos inteligentes.

Propósito do Desafio

O desafio consiste em criar uma criptomoeda na rede Ethereum que siga o padrão ERC-20, permitindo ao deployer definir um suprimento inicial (total supply) e alocar todo esse valor ao seu endereço. O projeto foi desenvolvido como parte do aprendizado de desenvolvimento de contratos inteligentes.

Estrutura de Arquivos

```
Primeira_Criptomoeda_Ethereum/
│
├── contracts/
│   └── MinhaPrimeiraCriptomoeda.sol   # Contrato inteligente da criptomoeda
│
├── scripts/
│   └── deploy.js              # Script para fazer o deploy do contrato
│   └── interact.js            # Script de interação simples com o contrato
│
├── test/
│   └── TesteCriptomoeda.js          # Teste unitário para verificar o total supply
│
├── node_modules/               # Dependências do projeto
├── hardhat.config.js           # Configuração do Hardhat
├── package.json                # Gerenciador de pacotes e dependências
├── README.md                   # Arquivo de documentação
└── .gitignore                  # Ignorar arquivos desnecessários no Git
```

Tecnologias Utilizadas

- Ethereum: Plataforma blockchain para contratos inteligentes.
- Solidity: Linguagem de programação utilizada para escrever o contrato inteligente.
- Hardhat: Ferramenta de desenvolvimento Ethereum que auxilia na compilação, deploy, e testes dos contratos.
- Chai: Biblioteca de testes para fazer assertions no teste unitário.

Bibliotecas Instaladas

As bibliotecas e ferramentas utilizadas no projeto são:
- hardhat: Ferramenta principal para desenvolvimento no Ethereum.
- chai: Utilizado para testar o contrato com assertions.
- ethers: Biblioteca para interagir com contratos Ethereum.
- mocha: Ferramenta de testes automatizados.

Instalação das dependências:

npm install

Passos para Execução

1. Compilação do Contrato

Para compilar o contrato inteligente, utilize o seguinte comando:
```npx hardhat compile```

2. Deploy do Contrato

Para realizar o deploy do contrato na rede local Hardhat, utilize:
```npx hardhat run scripts/deploy.js --network hardhat```

3. Testes

Os testes podem ser executados com o comando:
```npx hardhat test```

Estruturação de Pastas

- O contrato está localizado em `contracts/MinhaPrimeiraCriptomoeda.sol`.
- O script de deploy encontra-se em `scripts/deploy.js`.
- Um script simples para interagir com o contrato está em `scripts/interact.js`.
- O teste automatizado está em `test/TesteCriptomoeda.js`.

Resultados Obtidos

- **Compilação**: O contrato foi compilado com sucesso.
- **Deploy**: O contrato foi implantado na rede Hardhat com o deployer definido e o total supply alocado corretamente.
- **Testes**: Os testes foram concluídos com sucesso, validando que o total supply foi atribuído ao deployer.

Comandos Principais Utilizados

```
npx hardhat compile         # Para compilar o contrato
npx hardhat run scripts/deploy.js --network hardhat   # Para fazer o deploy
```
```

```
npx hardhat test            # Para rodar os testes
```

Repositório GitHub

Acesse o repositório completo do projeto no GitHub:
[Primeira_Criptomoeda_Ethereum](https://github.com/IOVASCON/Primeira_Criptomoeda_Ethereum.git)