Principles of Distributed Ledgers

Tutorial 2: Introduction to Smart Contracts

Daniel Perez, **Paul Pritz, Sam Werner**

October 29, 2024

Imperial College London

Goals of this tutorial

Our overarching goal is to understand the basics of smart contracts development

- Get local setup ready to develop smart contracts
- Learn the basics of using a wallet (MetaMask)
- Develop a simple fungible token contract
- Deploy our contract on Optimism (low transaction fees blockchain)
- Develop a simple non-fungible token contract (optional)

Development flow for smart contracts

1. Write high-level code
2. Test the code (using Foundry)
3. Optimise the code for gas efficiency
4. Compile the contract into Bytecode
5. Send a transaction to deploy the contract
6. Interact with the contract by sending transactions to the generated address

Development flow for smart contracts

1. Write high-level code
2. Test the code (using Foundry)
3. ~~Optimise the code for gas efficiency~~
4. Compile the contract into Bytecode
5. Send a transaction to deploy the contract
6. Interact with the contract by sending transactions to the generated address

Fungible tokens

- Fungible tokens are tokens that cannot be distinguished from one another
- They are interchangeable and divisible
- In the real-world, fiat currencies are fungible

Fungible tokens on Ethereum are often implemented using the ERC-20 standard. It defines a common interface to

- Transfer tokens
- Allow other parties to transfer tokens
- Check balance for tokens
- Emit events for token transfers

Non-fungible tokens

- Non-fungible tokens are tokens that are unique
- They are often used to represent collectibles

Non-fungible tokens on Ethereum are often implemented using the ERC-721 standard, which defines functions similar to the ones of ERC-20.

Development framework

- A development framework is a set of tools that help you develop smart contracts, e.g. compiling, dependency management, testing, deploying, etc.
- In this course, we will use Foundry. A popular alternative is Hardhat.

Deployment

We will deploy the contract on Optimism, an EVM-compatible blockchain with low transaction fees (a so-called Layer 2). We will be covering the workings of L2s in the following lecture.

1. Get an account used for deployment
2. Fund the account
3. Export the account private key
4. Get an API key for Optimistic Etherscan (optional, provided for this tutorial)
5. Decide on the constructor arguments
6. Run the command to deploy and verify the contract

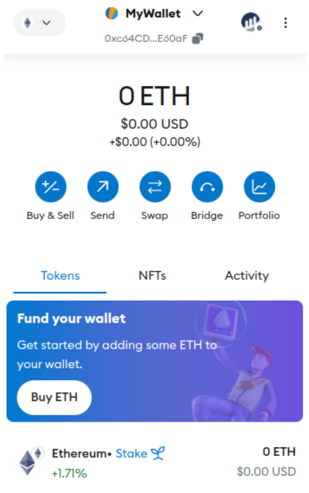
MetaMask

- MetaMask is a popular Ethereum wallet that can be installed as a browser extension.
- You can either generate a new private key via MetaMask or import an existing one (e.g., paste key or import from hardware wallet). In this tutorial, you simply generate a new private key.
- MetaMask gives you a *Secret Recovery Phrase* (seed phrase): 12 words representing your private key in a more human-readable format **(don't lose it!)**
- You can execute transactions and sign messages via MetaMask (e.g., interact with smart contracts, send ETH, etc.).

MetaMask networks and accounts

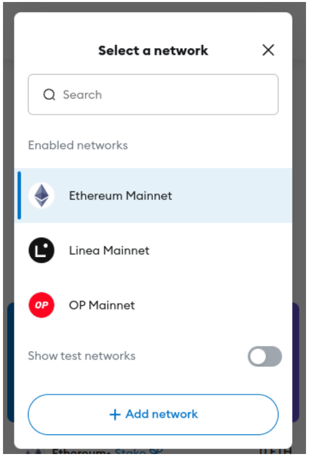
- MetaMask supports different EVM-based chains (e.g., Optimism). Optimism requires Optimism ETH to pay for transactions.
- You can easily add a new network to your MetaMask. Your address(es) will be the same on that network (note: this does not apply to contract accounts).

MetaMask: switch networks

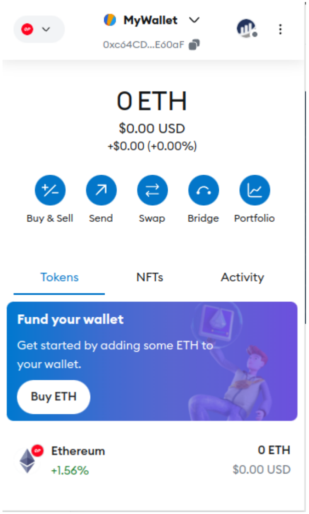

MyWallet
0xc64CD...E60aF
0 ETH
$0.00 USD
+$0.00 (+0.00%)

Fund your wallet
Get started by adding some ETH to your wallet.

Ethereum • Stake
+1.71%
0 ETH
$0.00 USD


Select a network

Enabled networks
Ethereum Mainnet
Linea Mainnet
OP Mainnet
Show test networks


MyWallet
0xc64CD...E60aF
0 ETH
$0.00 USD
+$0.00 (+0.00%)

Fund your wallet
Get started by adding some ETH to your wallet.

Ethereum
+1.56%
0 ETH
$0.00 USD

Optimistic Etherscan

- You can view transactions on Optimistic Etherscan
- You can also interact with contracts by connecting MetaMask on Optimistic Etherscan
- Example: USDC token contract on Optimism

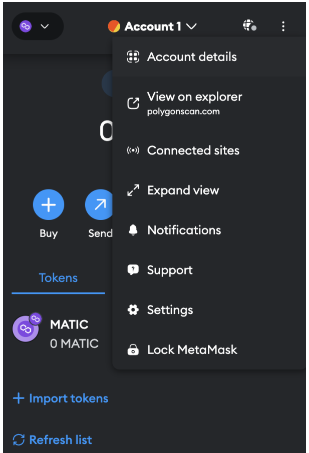
Funding

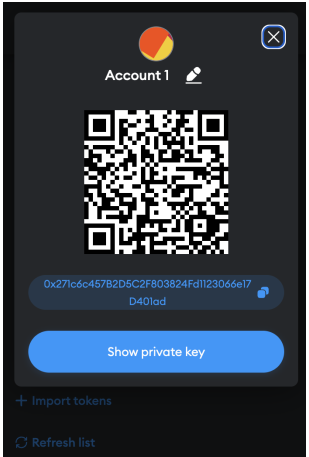
- Some amount of Optimism ETH is required to pay for transaction fees (e.g., to deploy a contract)
- The typical flow is to buy cryptocurrency on a centralized exchange (e.g. Coinbase) and send it to a MetaMask (or other) wallet
- For the purpose of this tutorial, we prepared a faucet (app that sends funds) so that you can receive enough money to deploy
- It allows you to get funds by connecting MetaMask and logging in with your Imperial credentials

`https://faucet.pdl.wiki/`

MetaMask: exporting private key

MetaMask stores your private key. You need to export it so that you can use it for contract deployment in your development environment.





Deployment command

The ERC20 contract can be deployed using the following command:

```
forge create \
    --constructor-args TOKEN_NAME TOKEN_SYMBOL \
    --private-key PRIVATE_KEY \
    --rpc-url RPC \
    --verify \
    --chain optimism \
    --etherscan-api-key ETHERSCAN_API_KEY \
    src/ERC20.sol:ERC20
```

This will also publish the source code of the contract on Optimistic Etherscan.

NOTE: Constants need to be replaced when executing the command. Values can be found in the tutorial's PDF