Agenda

- Tools needed for the workshop

- How smart contracts work

- Quick overview of Solidity

- Build smart contract together

- Deploy smart contract on an Ethereum test network

- Interact with smart contract

Tools needed

- Set up a Node development environment on your machine
  - You should be able to run **node** and **npm** commands from your Terminal
- **Hardhat - `npm install --save-dev hardhat`**
- **MetaMask Wallet**: Make sure you have some Gorli Testnet ETH in your wallet. Use a Goerli Faucet to fund your account.
- **Infura**: Sign up for a free account and create a Gorli network endpoint
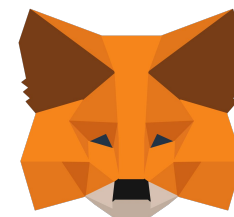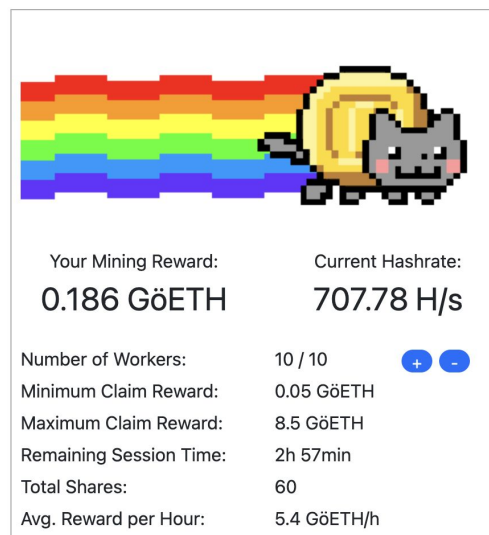
Full details in README found on

https://github.com/lanasta/deploy-smart-contract

Show test networks
Select this to show test networks in network list

ON

Goerli PoW Faucet

Your Mining Reward:	Current Hashrate:
0.186 GöETH	707.78 H/s

Number of Workers:	10 / 10
Minimum Claim Reward:	0.05 GöETH
Maximum Claim Reward:	8.5 GöETH
Remaining Session Time:	2h 57min
Total Shares:	60
Avg. Reward per Hour:	5.4 GöETH/h

How smart contracts work

- Runs in a blockchain network

    - Decentralized, trustless, transparent, immutable

- Self-execute when certain conditions are met, removes the need for a third-party (eg. bank or other centralized authorities)

- Widely used in business management, legal processes, financial agreements, health systems

- In enterprise use cases, a private blockchain would be of interest

- In healthcare, smart contracts can automate processes like payment and insurance claims

    - Saves time, costs, reduce chances of human error

Smart contract concepts

- **Ethereum Virtual Machine(EVM)**: the execution environment for smart contracts in Ethereum and the fundamental consensus mechanism. Acts as a "world computer", accessible from anywhere throughout the world through participating Ethereum nodes.

- **Gas fees**: required as it is expensive to have thousands of computers (usually referred to as nodes in blockchain) all over the world validate smart contracts. Paid to miners who use their computing power to process and confirm transactions.

- **Block explorer**: A visualization tool that allows anyone to explore the state of a particular blockchain network and see information about blocks, transactions, balances, etc e.g. https://etherscan.io/

- **Addresses**: Each smart contract and each Ethereum account has an address that is represented by a 42-character hexadecimal address e.g. 0x92bc44d5318309EE2abF1539BF71dE1b7d7bE3b8

- **Application Binary Interface(ABI)**: defines the methods and variables that are available to interact with in a smart contract.

What bytecode looks like

Source:
https://ethereum.org/en/developers/docs/smart-contracts/compiling/

Corresponding ABI for the Solidity code

THE EVM

For the EVM to be able to run your contract it needs to be in **bytecode**. Compilation turns this:

```
1   pragma solidity 0.4.24;
2
3   contract Greeter {
4
5       function greet() public constant returns (string) {
6           return "Hello";
7       }
8
9   }
```

into this

```
1   PUSH1 0x80 PUSH1 0x40 MSTORE PUSH1 0x4 CALLDATASIZE LT PUSH2 0x41
    JUMPI PUSH1 0x0 CALLDATALOAD PUSH29
    0x100000000000000000000000000000000000000000000000000000000 SWAP1
    DIV PUSH4 0xFFFFFFFF AND DUP1 PUSH4 0xCFAE3217 EQ PUSH2 0x46 JUMPI
    JUMPDEST PUSH1 0x0 DUP1 REVERT JUMPDEST CALLVALUE DUP1 ISZERO PUSH2
    0x52 JUMPI PUSH1 0x0 DUP1 REVERT JUMPDEST POP PUSH2 0x5B PUSH2 0xD6
    JUMP JUMPDEST PUSH1 0x40 MLOAD DUP1 DUP1 PUSH1 0x20 ADD DUP3 DUP2
    SUB DUP3 MSTORE DUP4 DUP2 DUP2 MLOAD DUP2 MSTORE PUSH1 0x20 ADD
    SWAP2 POP DUP1 MLOAD SWAP1 PUSH1 0x20 ADD SWAP1 DUP1 DUP4 DUP4 PUSH1
    0x0 JUMPDEST DUP4 DUP2 LT ISZERO PUSH2 0x9B JUMPI DUP1 DUP3 ADD
```

```
1   [
2       {
3           "constant": true,
4           "inputs": [],
5           "name": "greet",
6           "outputs": [
7               {
8                   "name": "",
9                   "type": "string"
10              }
11          ],
12          "payable": false,
13          "type": "function"
14      }
15  ]
```

Gas fees

Sub-units of Ether

Unit	wei value	wei	ether value
wei	1 wei	1	10^-18 ETH
kwei	10^3 wei	1,000	10^-15 ETH
mwei	10^6 wei	1,000,000	10^-12 ETH
gwei	10^9 wei	1,000,000,000	10^-9 ETH
microether	10^12 wei	1,000,000,000,000	10^-6 ETH
milliether	10^15 wei	1,000,000,000,000,000	10^-3 ETH
ether	10^18 wei	1,000,000,000,000,000,000	1 ETH

Source: https://www.investopedia.com/terms/w/wei.asp

Transaction Details ‹ ›

Sponsored: ◑ OpenOcean: The most efficient aggregator on Ethereum & 16 other networks. Swap your crypto now!

Overview   State   Comments

⑦ Transaction Hash:	0xe2333f02a9dde721c1163a3c170cc7e7a1f58341e2e6403277365426f486a0a5 ⧉
⑦ Status:	✓ Success
⑦ Block:	15440635 1 Block Confirmation
⑦ Timestamp:	⊙ 54 secs ago (Aug-30-2022 01:34:59 PM +UTC)	⊙ Confirmed within 30 secs
⑦ From:	0xea674fdde714fd979de3edf0f56aa9716b898ec8 (Ethermine) ⧉
⑦ To:	0x9dca330f5469374950dc23664ab10ee6a7fdd06c ⧉
⑦ Value:	0.099465521973971281 Ether ($157.18)
⑦ Transaction Fee:	0.000545644077951 Ether ($0.86)
⑦ Gas Price:	0.000000025983051331 Ether (25.983051331 Gwei)
⑦ Gas Limit & Usage by Txn:	250,000	21,000 (8.4%)
⑦ Gas Fees:	Base: 24.983051331 Gwei	Max: 50.715594201 Gwei	Max Priority: 1 Gwei
⑦ Burnt & Txn Savings Fees:	🔥 Burnt: 0.000524644077951 Ether ($0.83) 🎁 Txn Savings: 0.00051938340027 Ether ($0.82)

Source: Etherscan.io transaction details

Solidity Overview

1. **Version pragma**: solidity compiler version compatible with the code at the time of the development. Future versions may contain incompatible changes.

2. **Contract keyword**: the contract name follows after, encapsulates the smart contract logic

3. **State variables**: types include signed integers, unsigned integers, Boolean, addresses, enums, and bytes

4. **Function declarations**: can take in parameter, specify `view` or `pure` functions, return type and function visibility types – `private, internal, external, or public`

Source: https://www.geeksforgeeks.org/introduction-to-solidity/

```
pragma solidity >=0.4.0 <0.6.0;
contract SimpleStorage {
    uint storedData;
    function set(uint x) public {
        storedData = x;
    }
    function get() public view returns (uint) {
        return storedData;
    }
}
```

Source: https://www.tutorialspoint.com/solidity/solidity_quick_guide.htm

Solidity global variables

Name	Returns
blockhash(uint blockNumber) returns (bytes32)	Hash of the given block - only works for 256 most recent, excluding current, blocks
block.coinbase (address payable)	Current block miner's address
block.difficulty (uint)	Current block difficulty
block.gaslimit (uint)	Current block gaslimit
block.number (uint)	Current block number
block.timestamp (uint)	Current block timestamp as seconds since unix epoch
gasleft() returns (uint256)	Remaining gas
msg.data (bytes calldata)	Complete calldata
msg.sender (address payable)	Sender of the message (current caller)
msg.sig (bytes4)	First four bytes of the calldata (function identifier)
msg.value (uint)	Number of wei sent with the message
now (uint)	Current block timestamp
tx.gasprice (uint)	Gas price of the transaction
tx.origin (address payable)	Sender of the transaction

Source: https://www.tutorialspoint.com/solidity/solidity_variables.htm

Let's write and deploy a smart contract together!