

Blockchain

Smart Contract & Solidity

2024

目錄

- Introduction to Smart Contract
- EVM 與智能合約
- Solidity





Introduction to Smart Contract

A Simple Smart Contract

- Storage Example。
- Line 1 : source code 使用GPL 3.0
- Line 2 : 使用solidity 0.8.26以上或0.9.0以下版本
- Contract : Solidity 內的contract是保存在以太坊區塊鏈上特定位址的程式碼（功能）和資料（狀態）的集合。
- 合約定義了可用於修改或擷取變數值的函數 set 和 get。

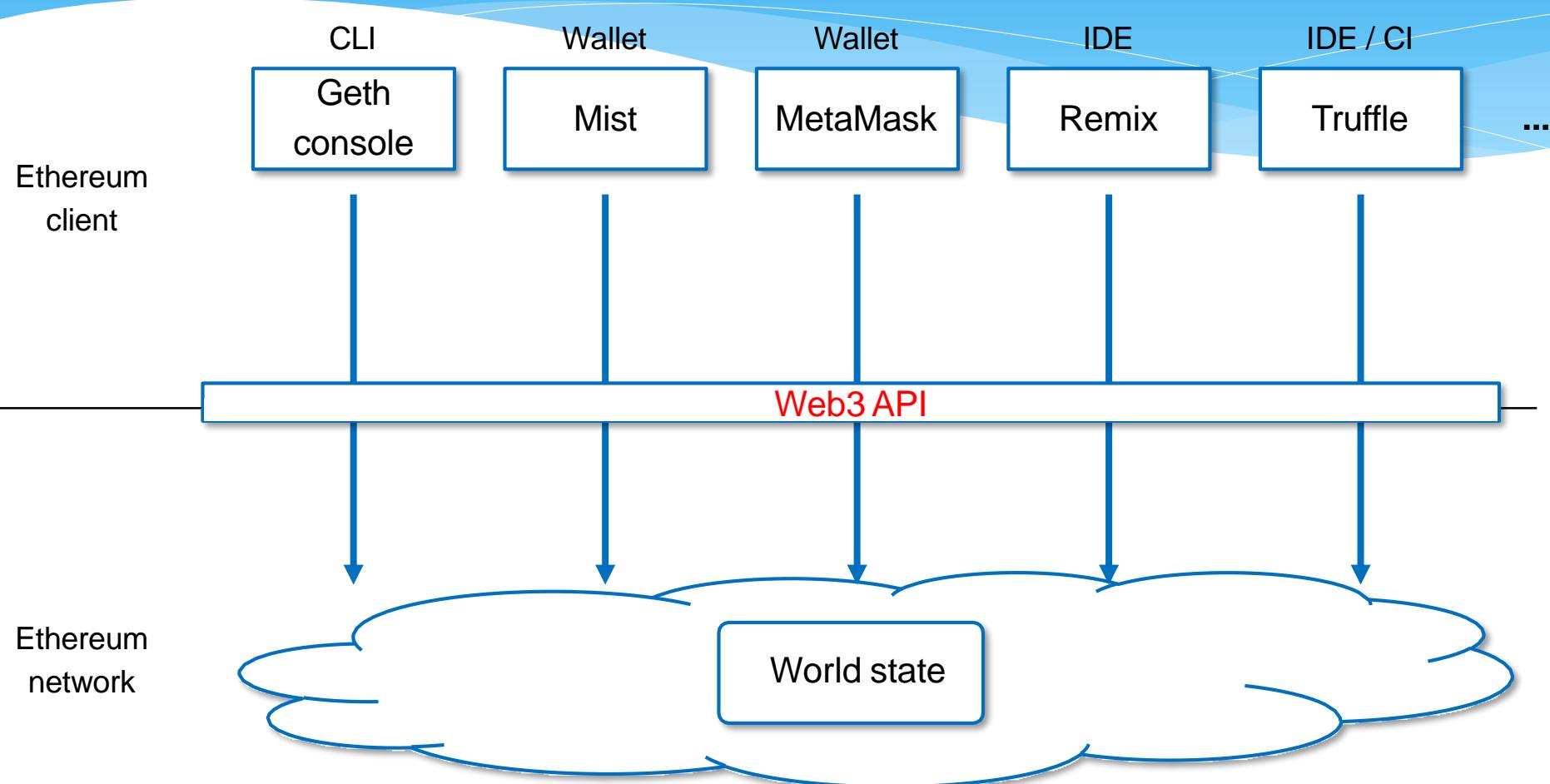
```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

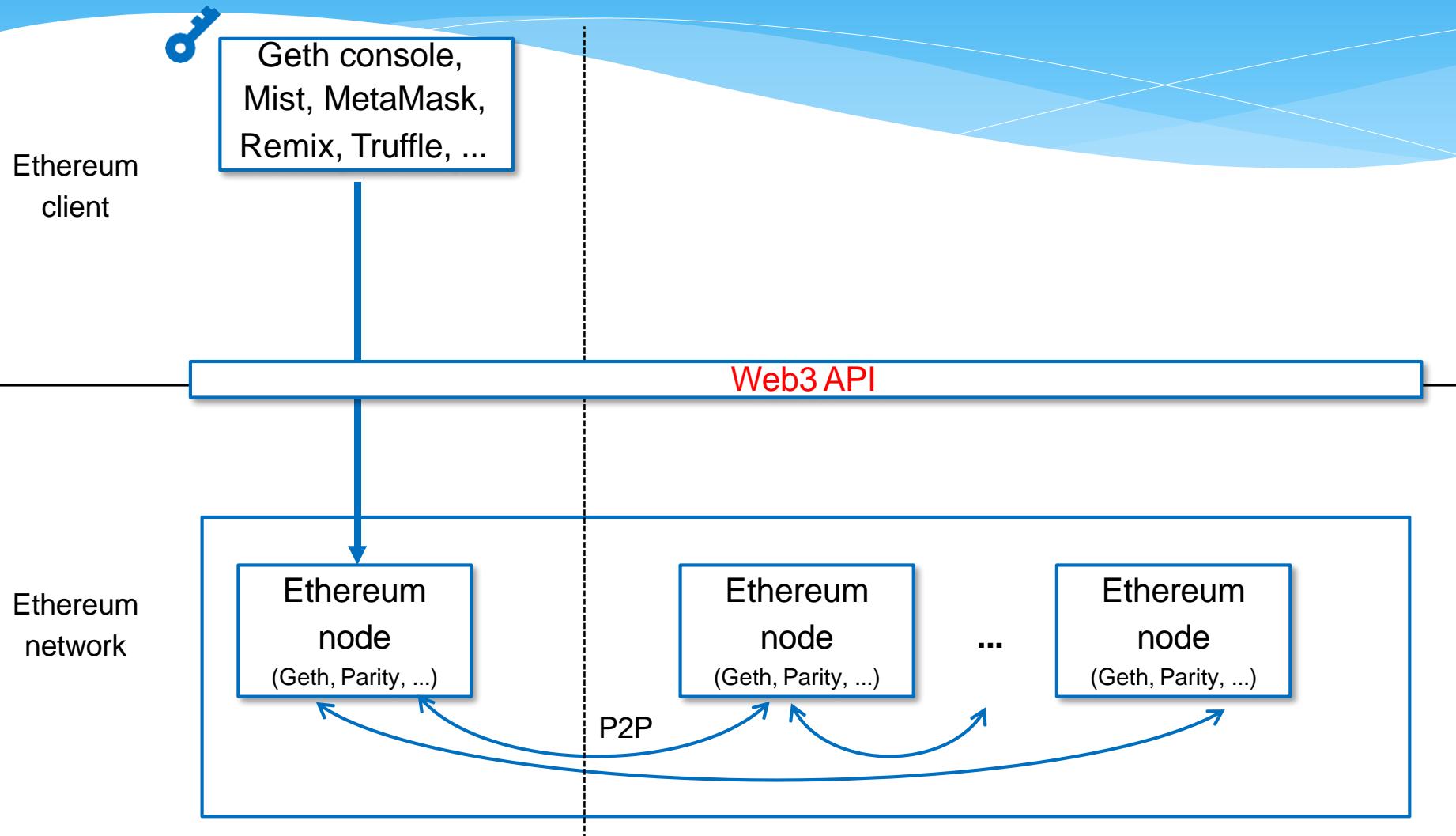
    function get() public view returns (uint) {
        return storedData;
    }
}
```

Web3 API and client



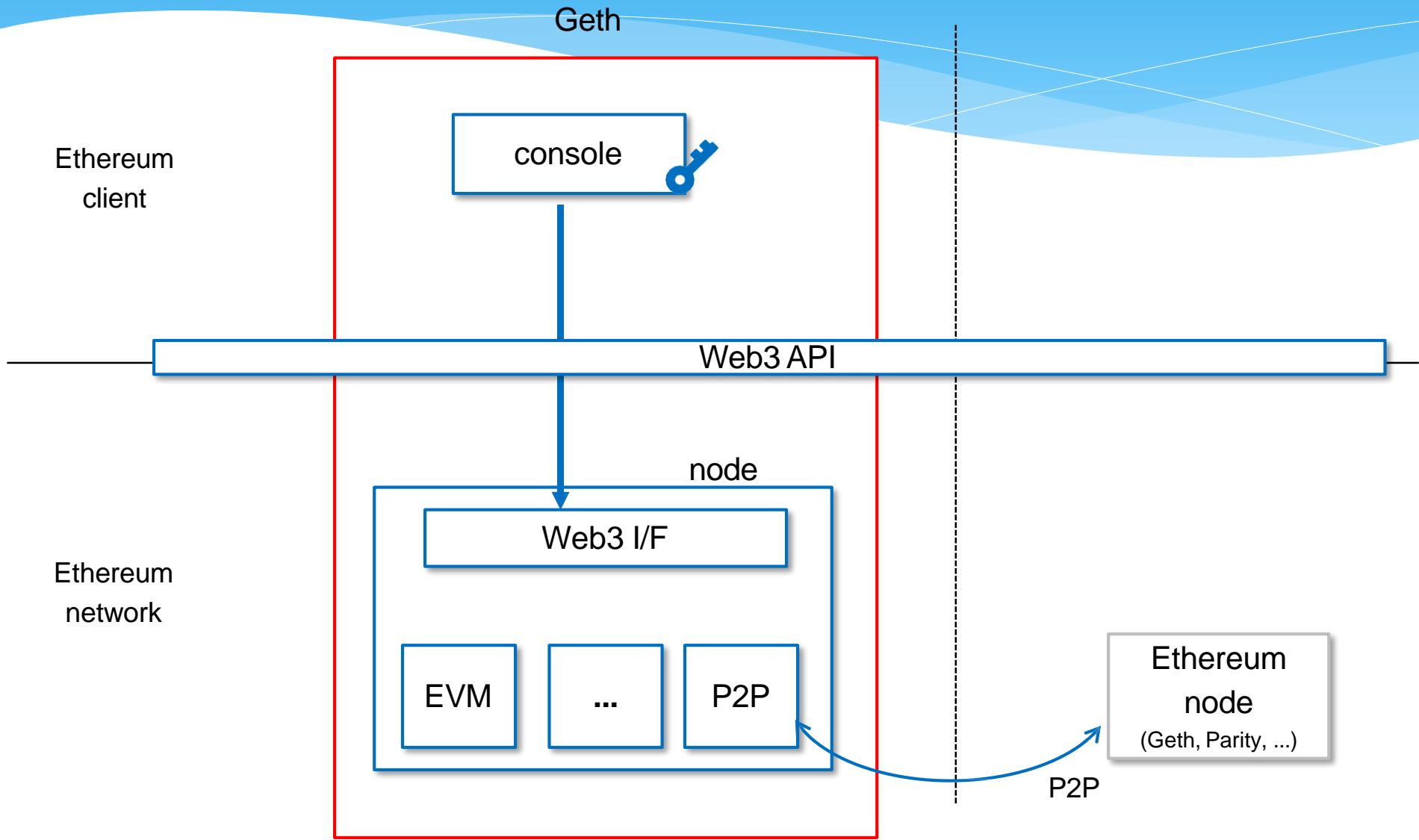
Ethereum clients access to Ethereum network via Web3 API.

Web3 API and client

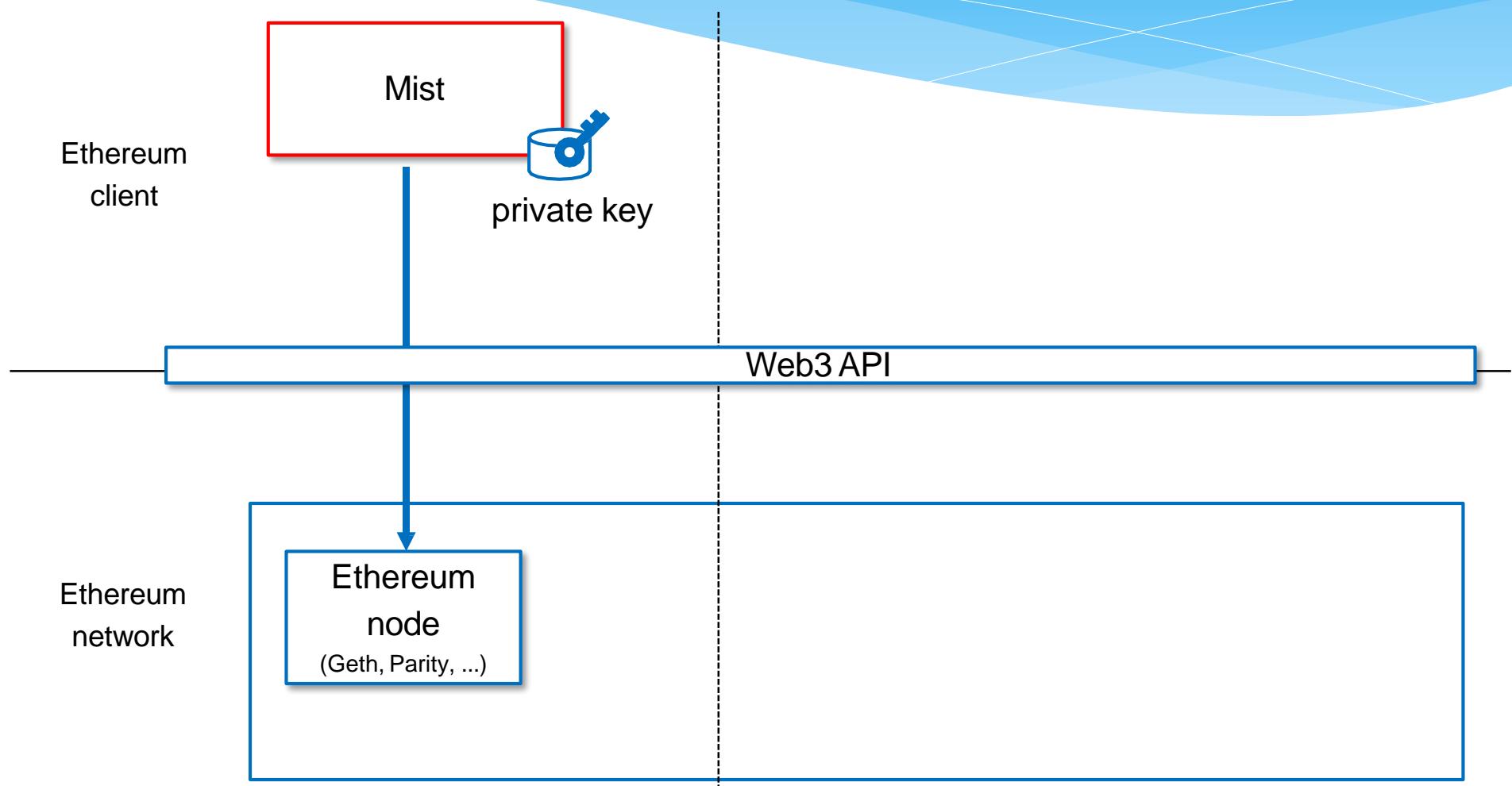


Ethereum clients access to Ethereum network via Web3 API.

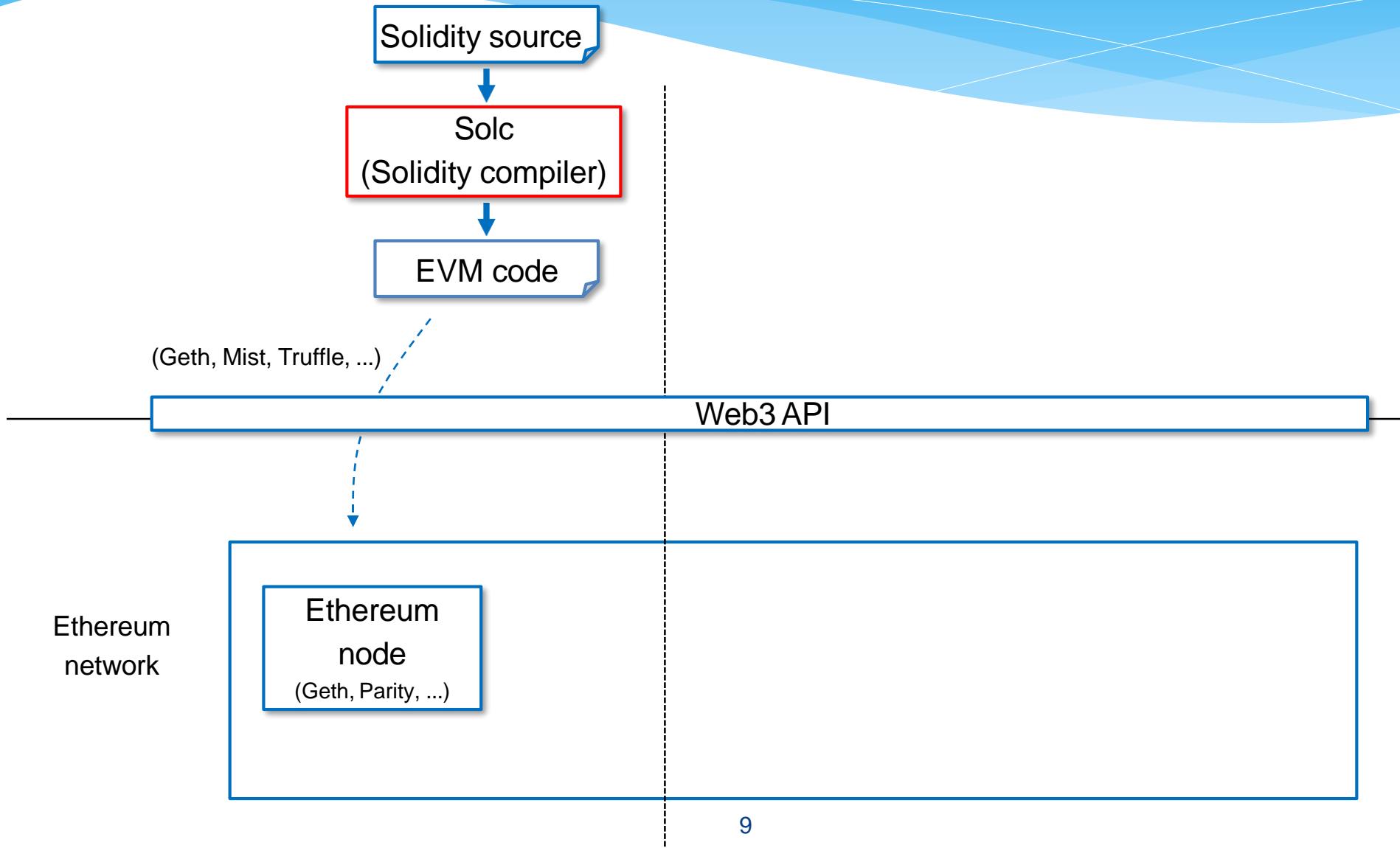
Geth



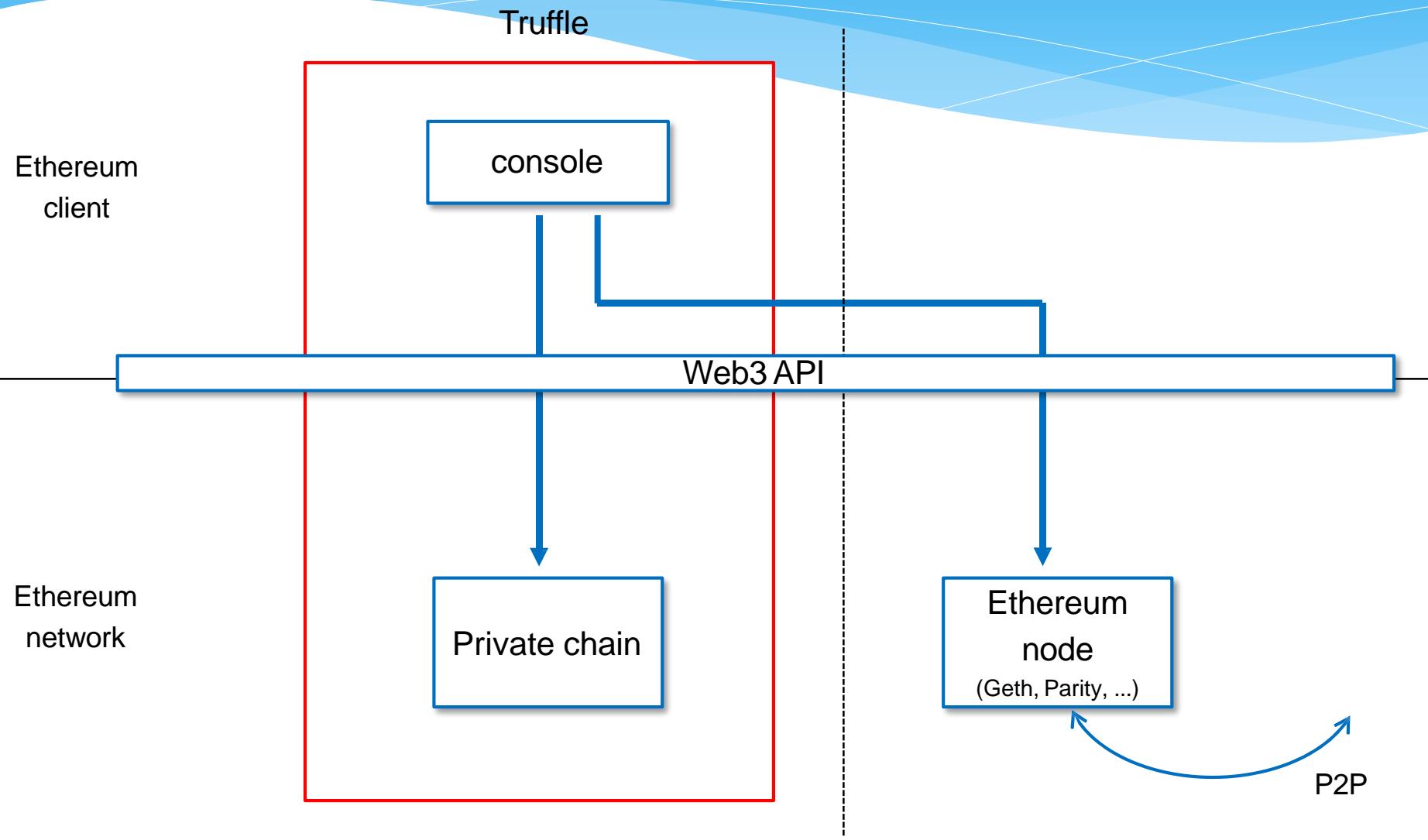
Mist



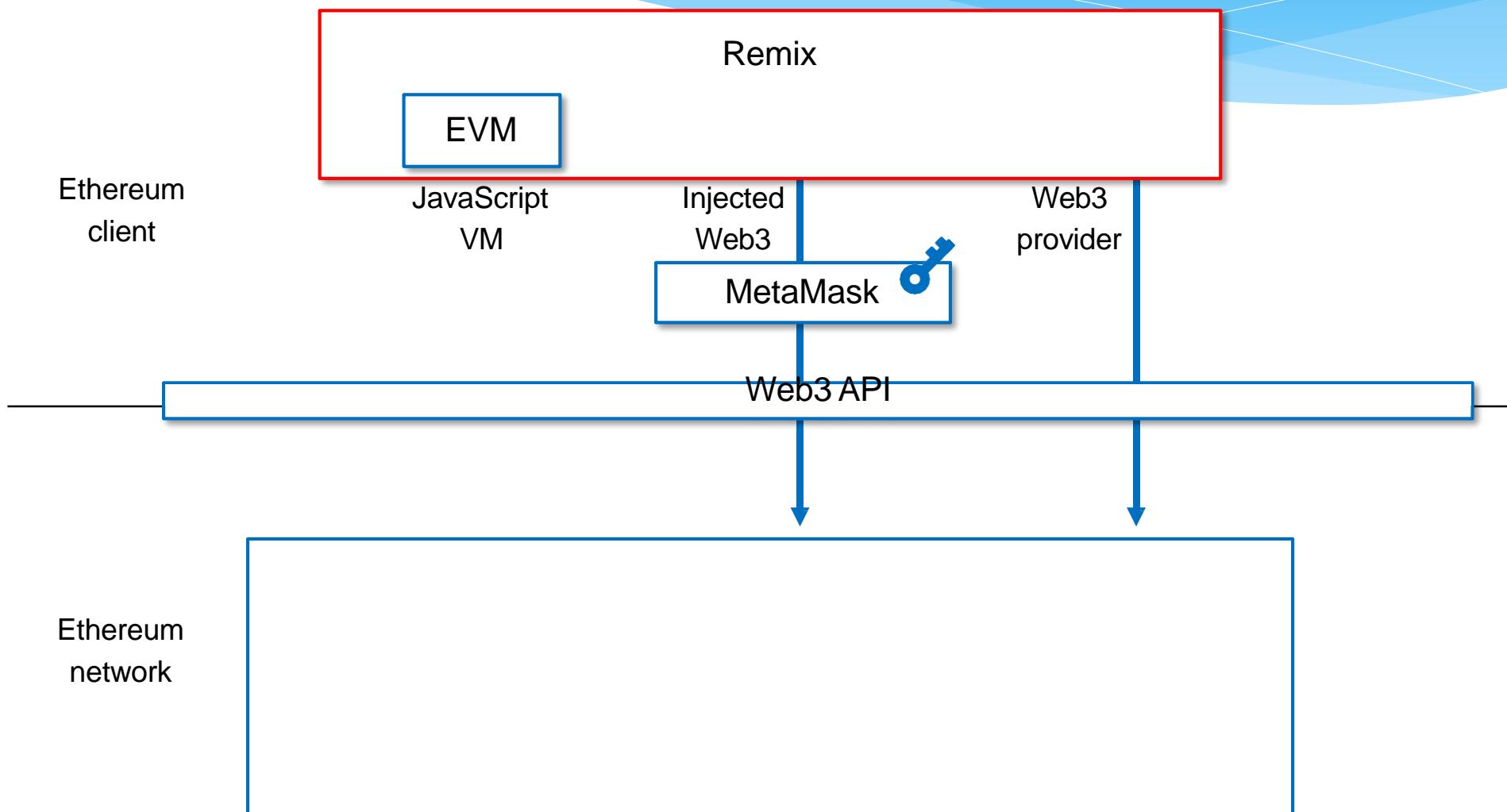
Solc



Truffle



Remix





**Start to Deploy
Smart Contract**

EVM 與智能合約

EVM

- 以太坊虛擬機（EVM）是以太坊中智慧合約的運行時環境。
- 它是沙箱，而且實際上是完全隔離的，
- EVM 內運行的程式碼無法存取網路、檔案系統或其他程序。
- 智能合約甚至對其他智能合約的存取也受到限制。

Account

World state

Account 1

Account 2

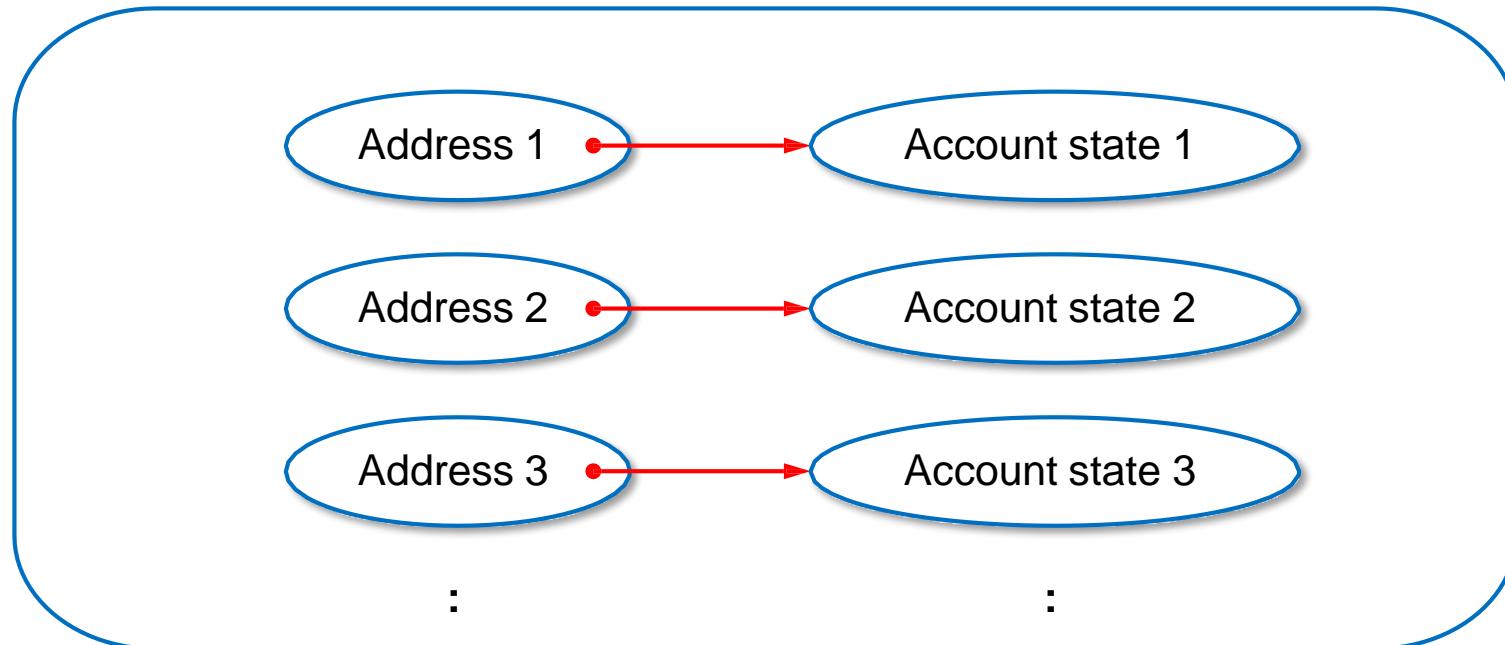
Account 3

:

An account is an object in the world state.

World state

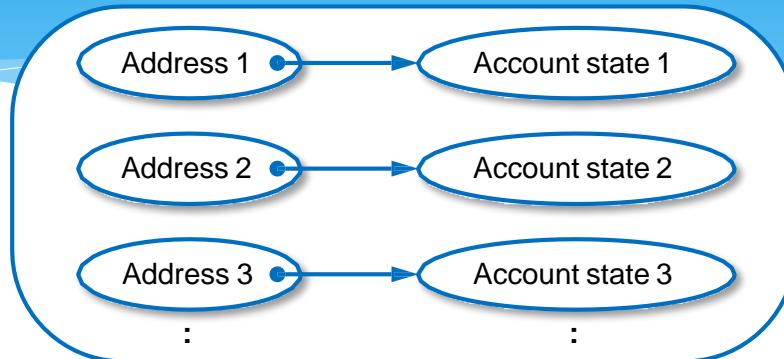
World state σ_t



1. The world state can be seen as the global state that is constantly updated by transaction executions.
2. The world state is a mapping between address and account state.

Several views of World state

Mapping view



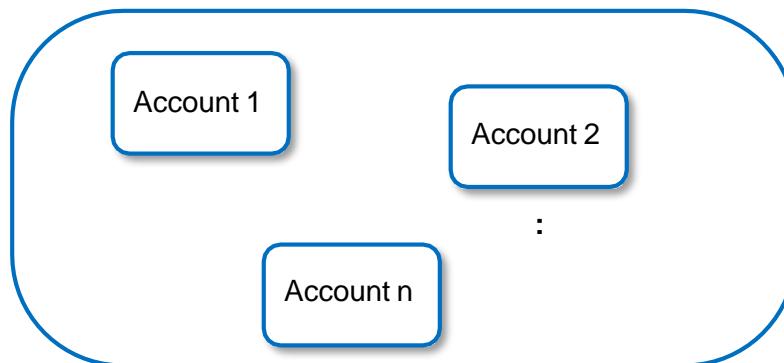
World state

Table view

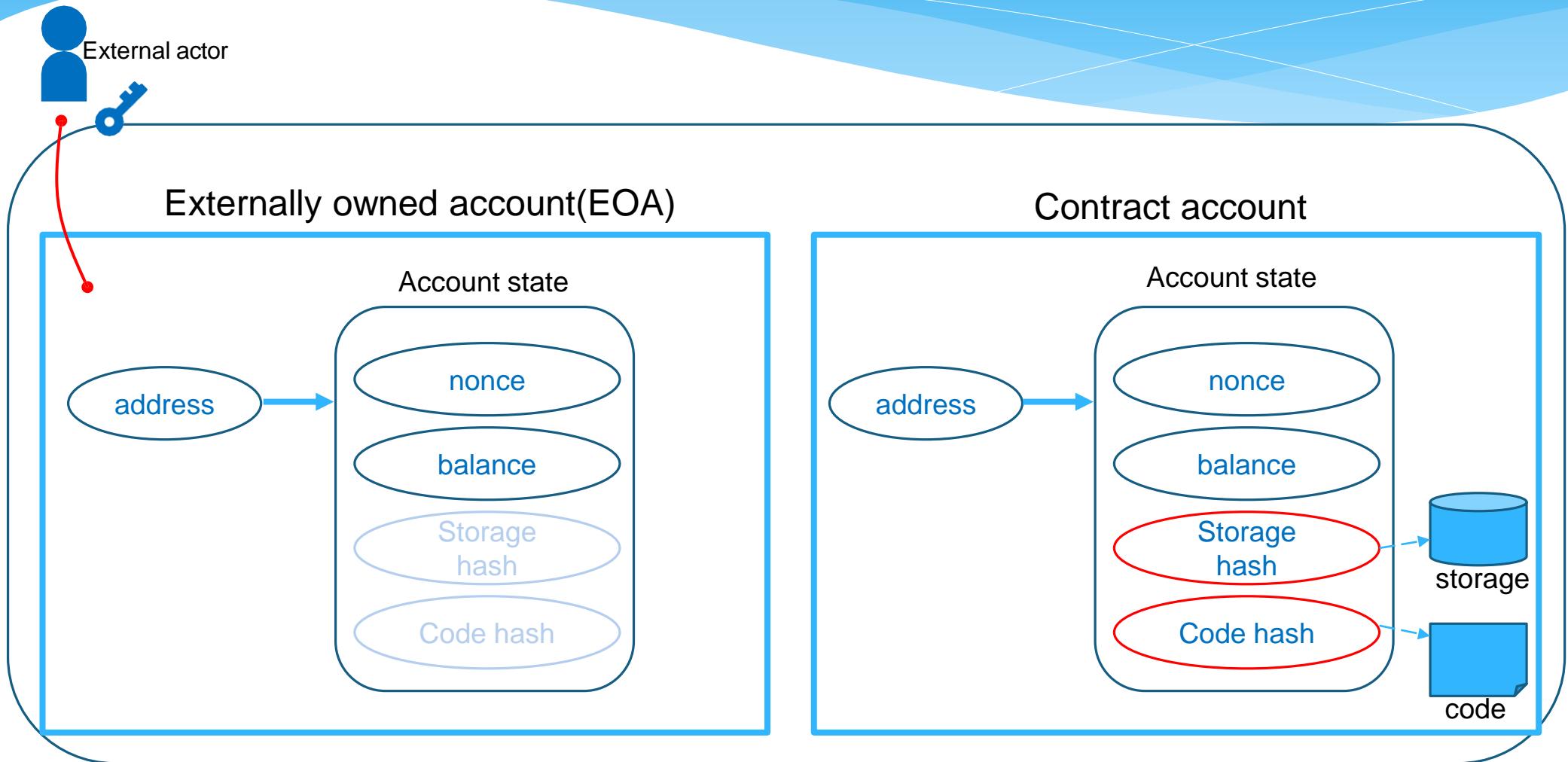
Address 1	Account state 1
Address 2	Account state 2
:	:
Address n	Account state n

World state

Object view



Two types of account

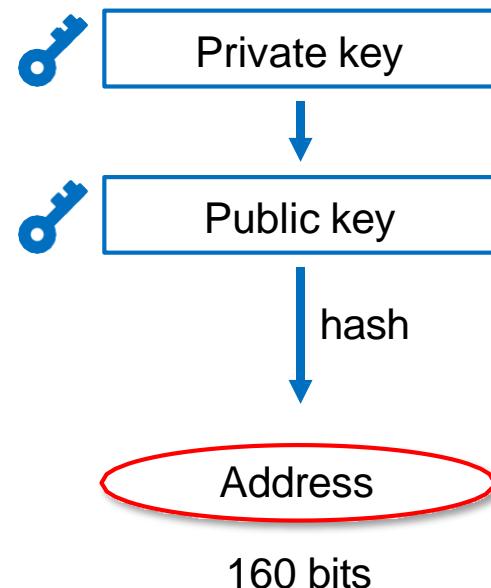


EOA is controlled by a private key.
EOA cannot contain EVM code.

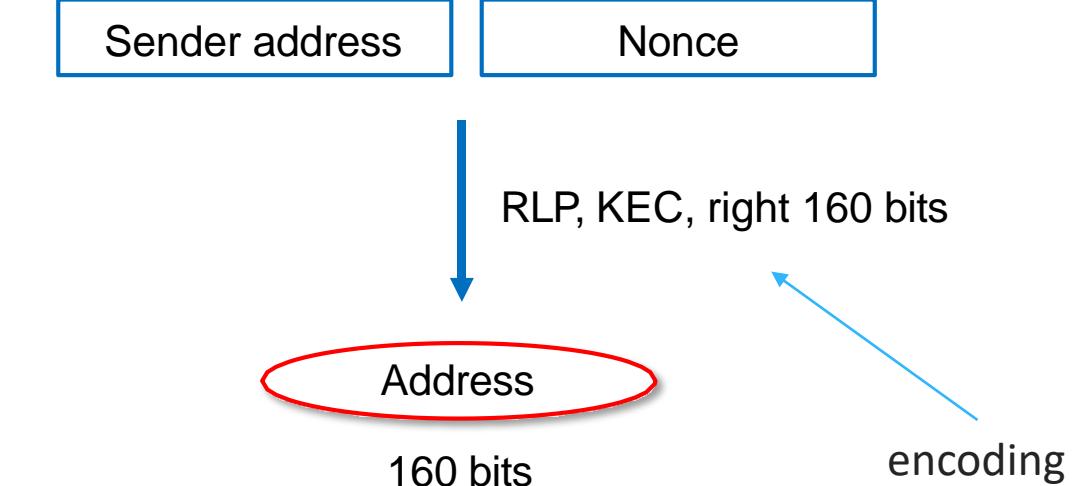
Contract contains EVM code.
Contract is controlled by EVM code.

Address of account

Externally owned account (EOA)

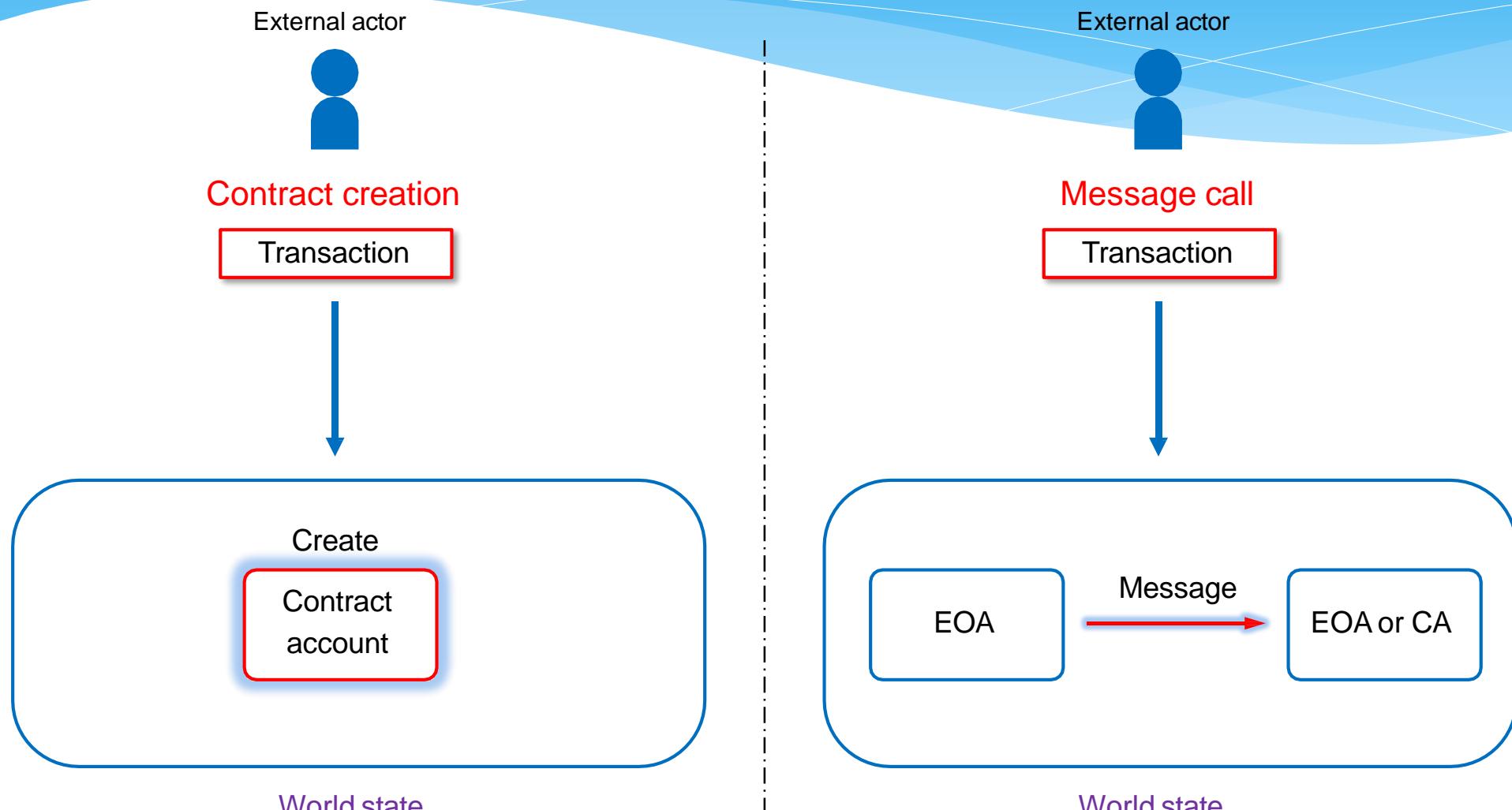


Contract account



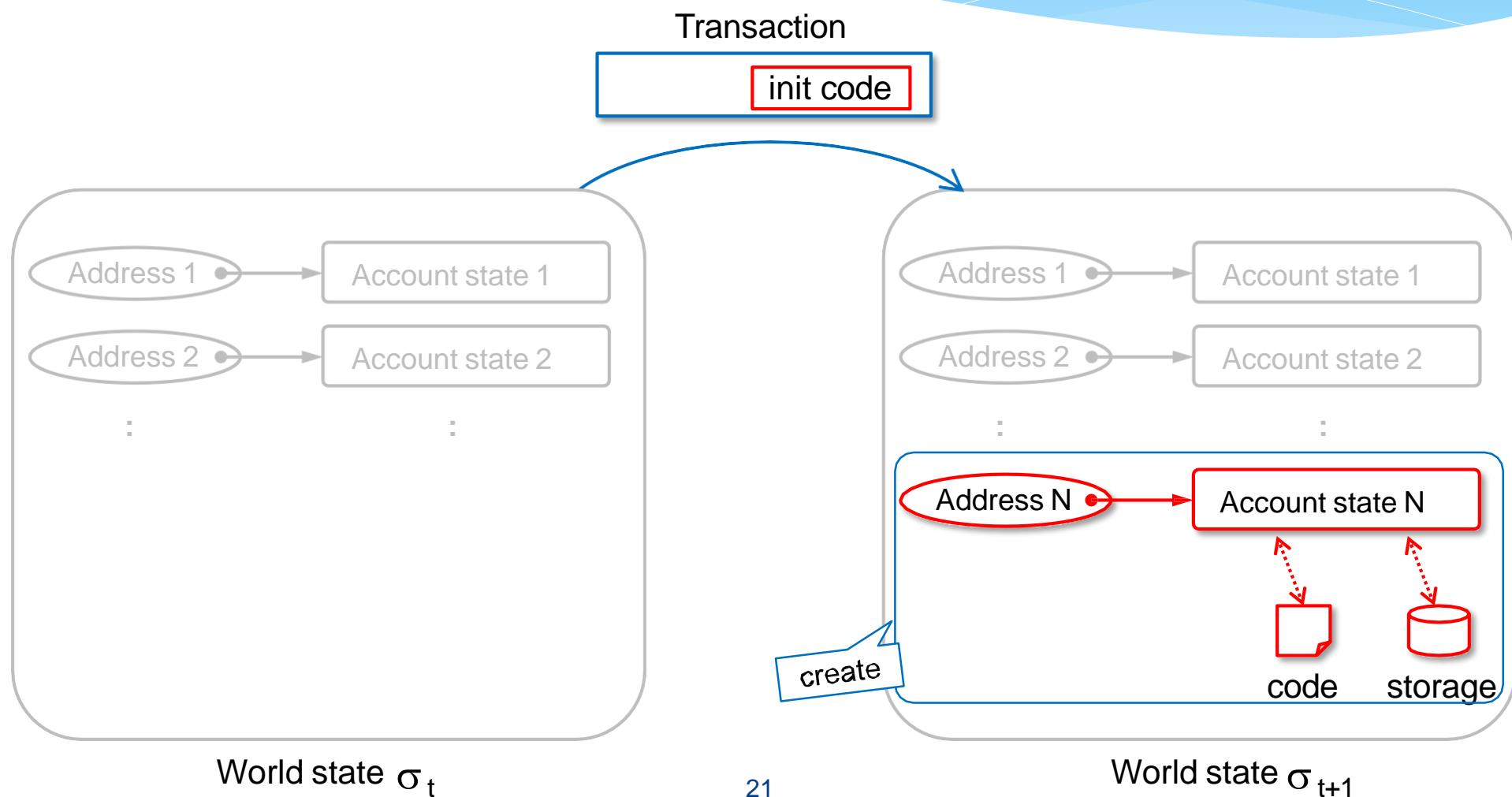
A 160-bit code used for identifying accounts.

Two types of transaction

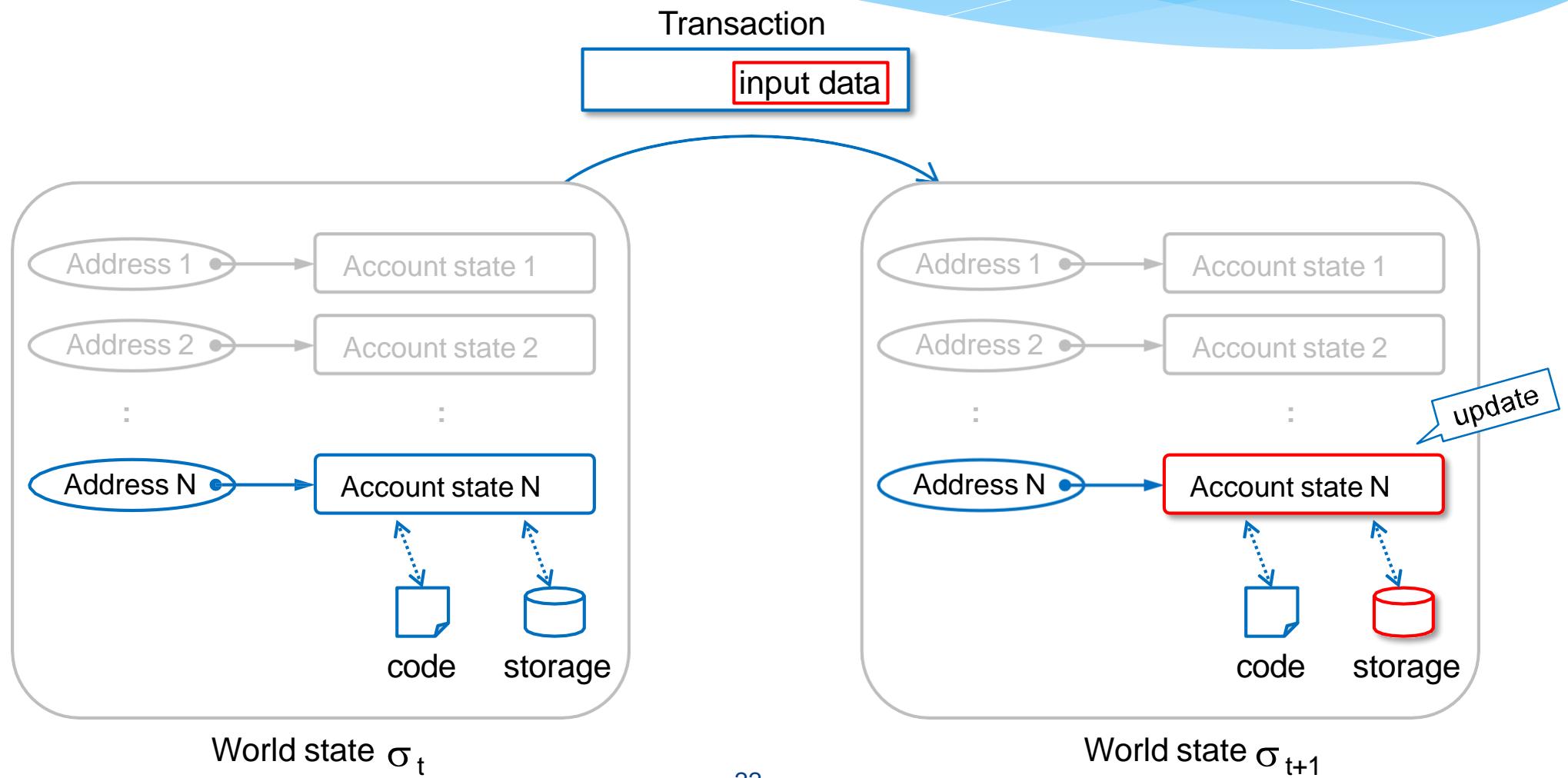


There are two practical types of transaction, contract creation and message call.

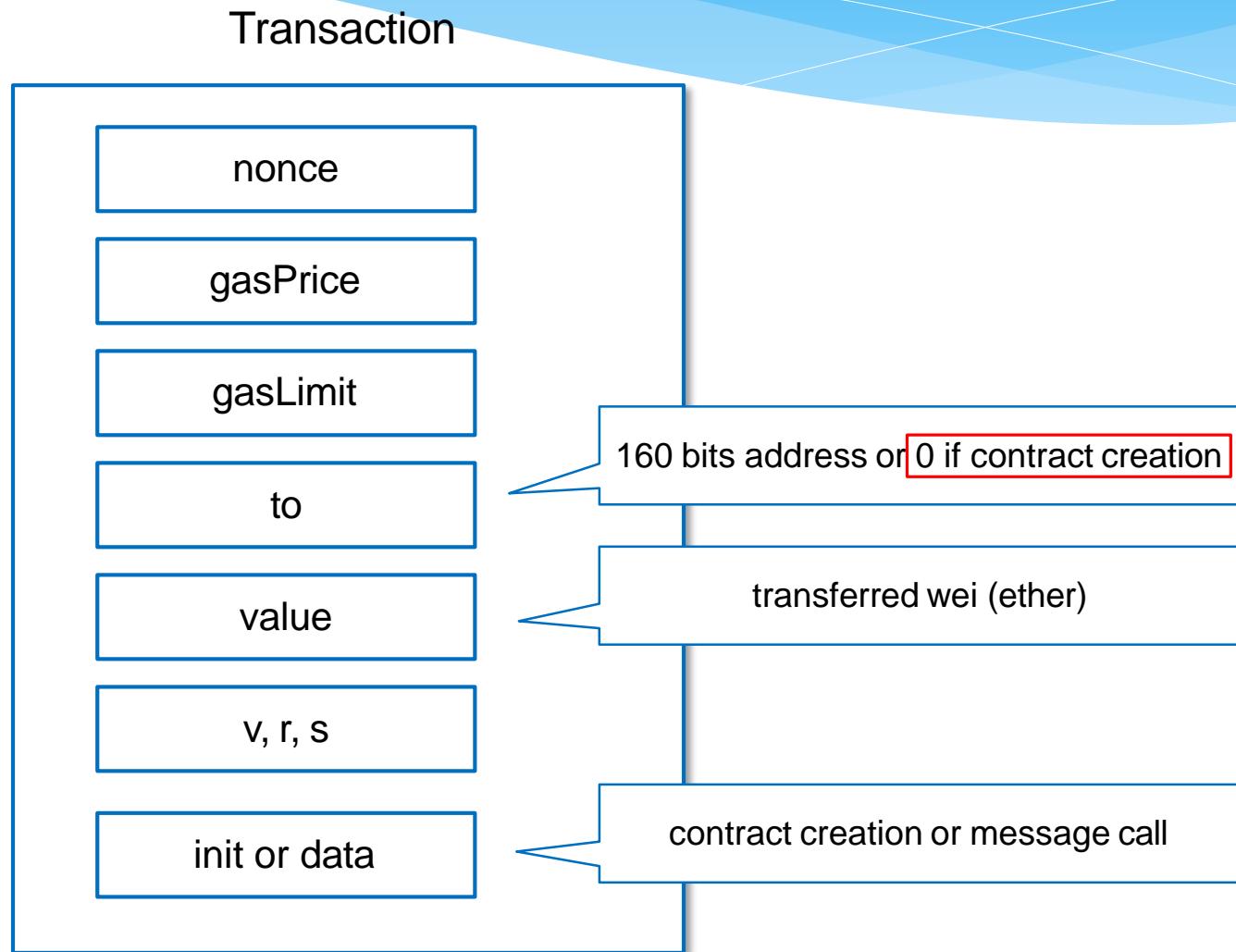
Contract creation



Message call

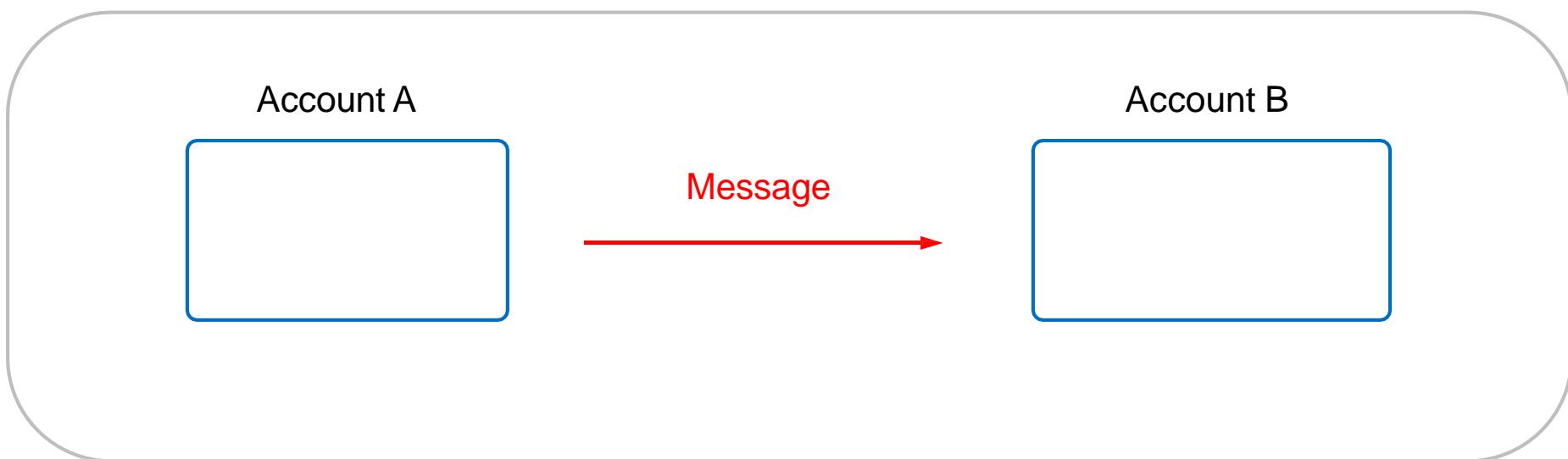


Field of a transaction



Message

World state

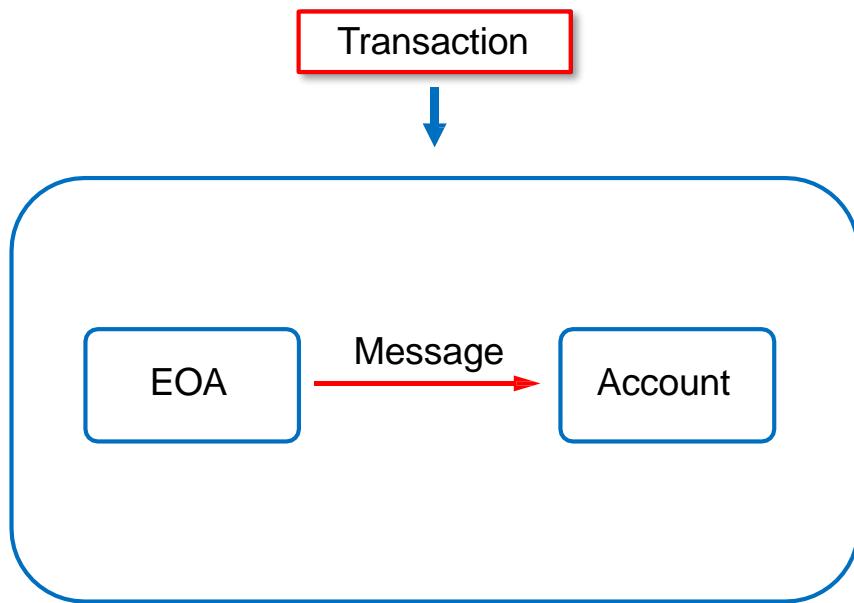


Message is passed between two Accounts.

Message is Data (as a set of bytes) and Value (specified as Ether) .

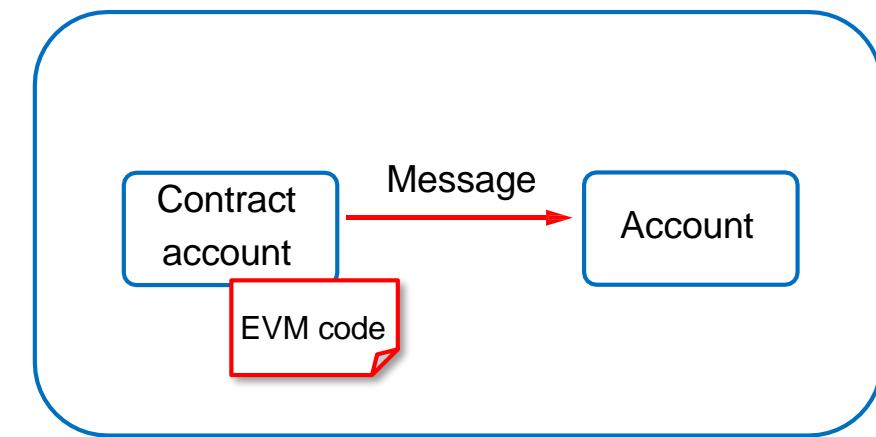
Message

Triggered by transaction



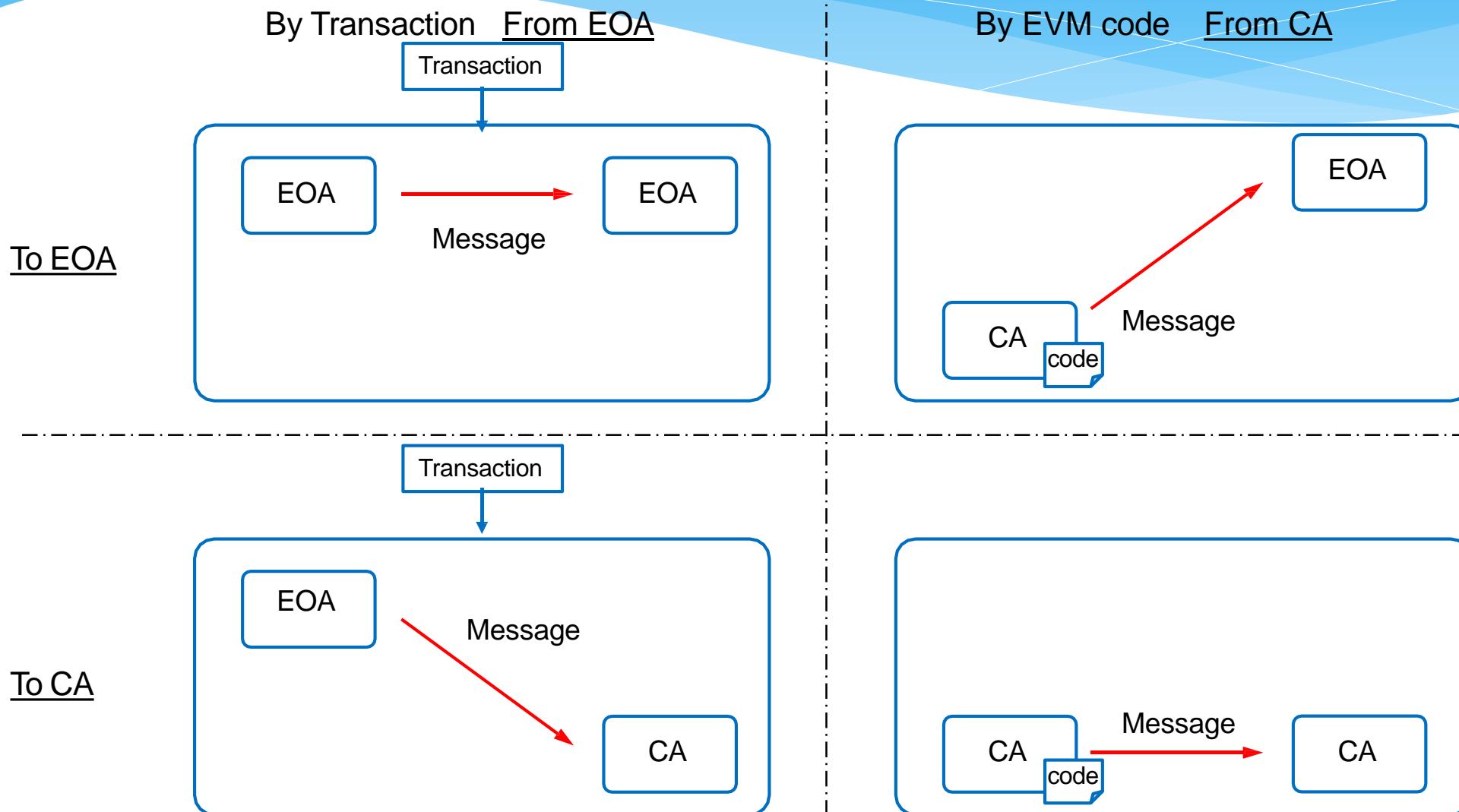
Transaction triggers an associated message.

Triggered by EVM code

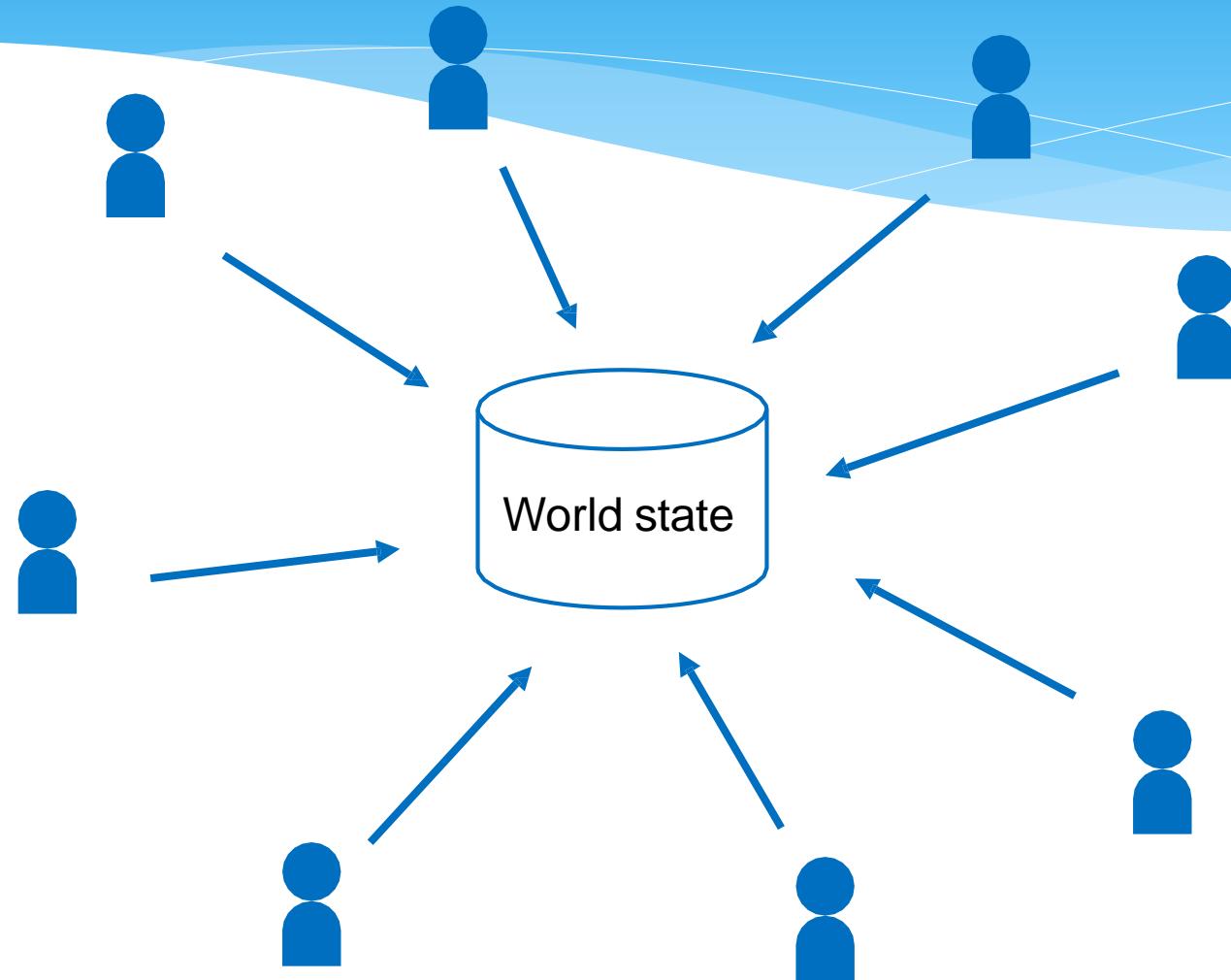


EVM can also send a message.

Four cases of Message

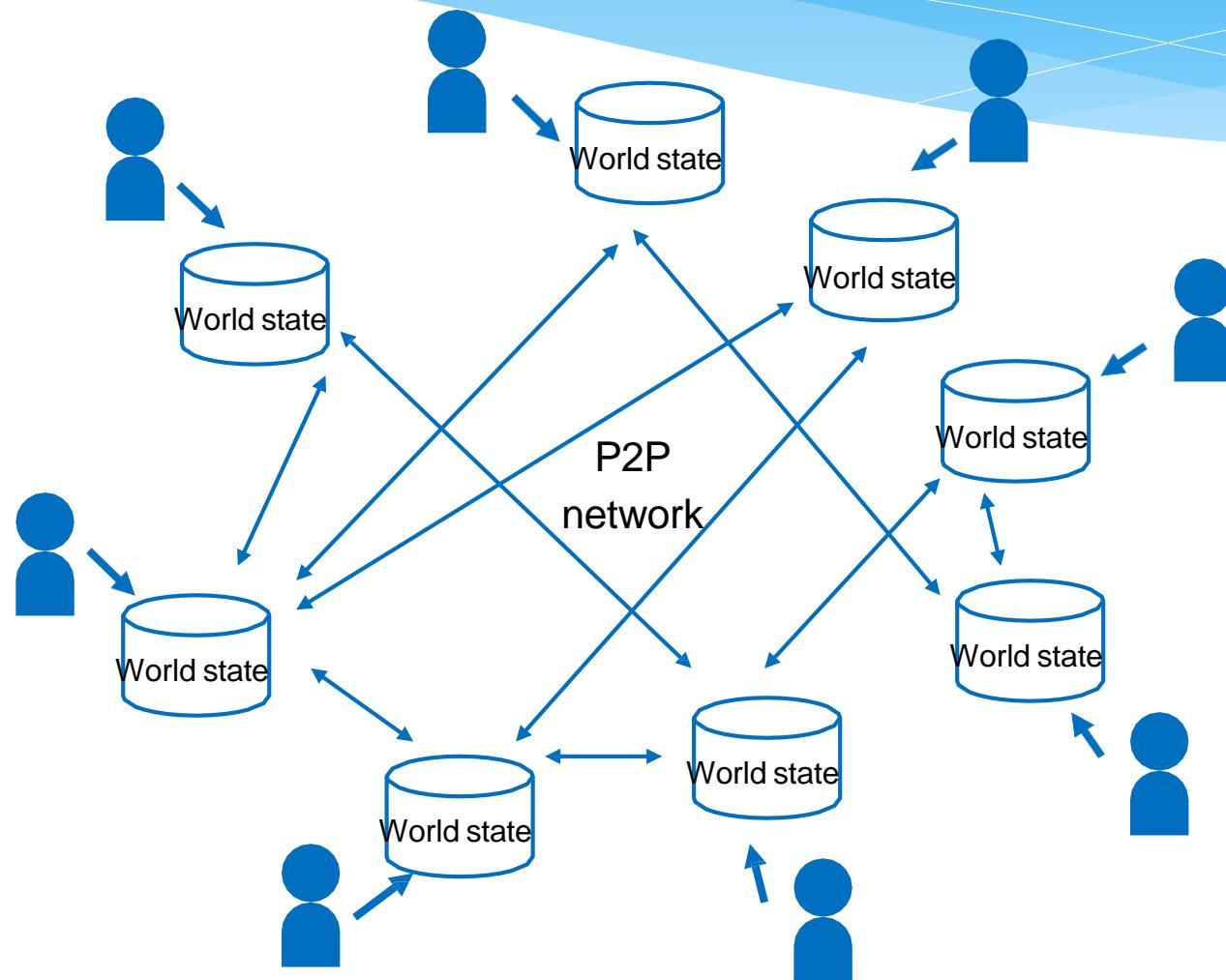


Globally shared, transactional database



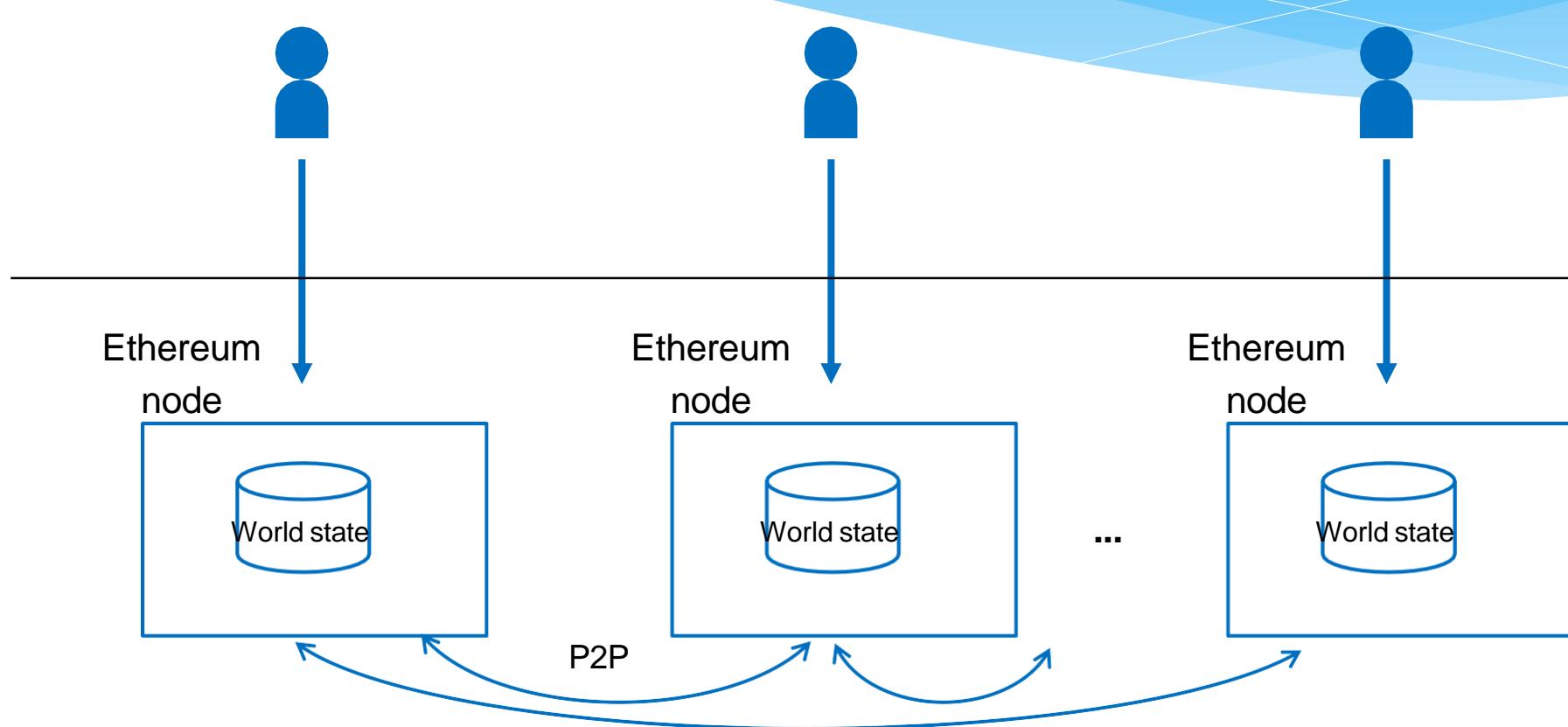
A blockchain is a globally shared, transactional database.

Decentralised database



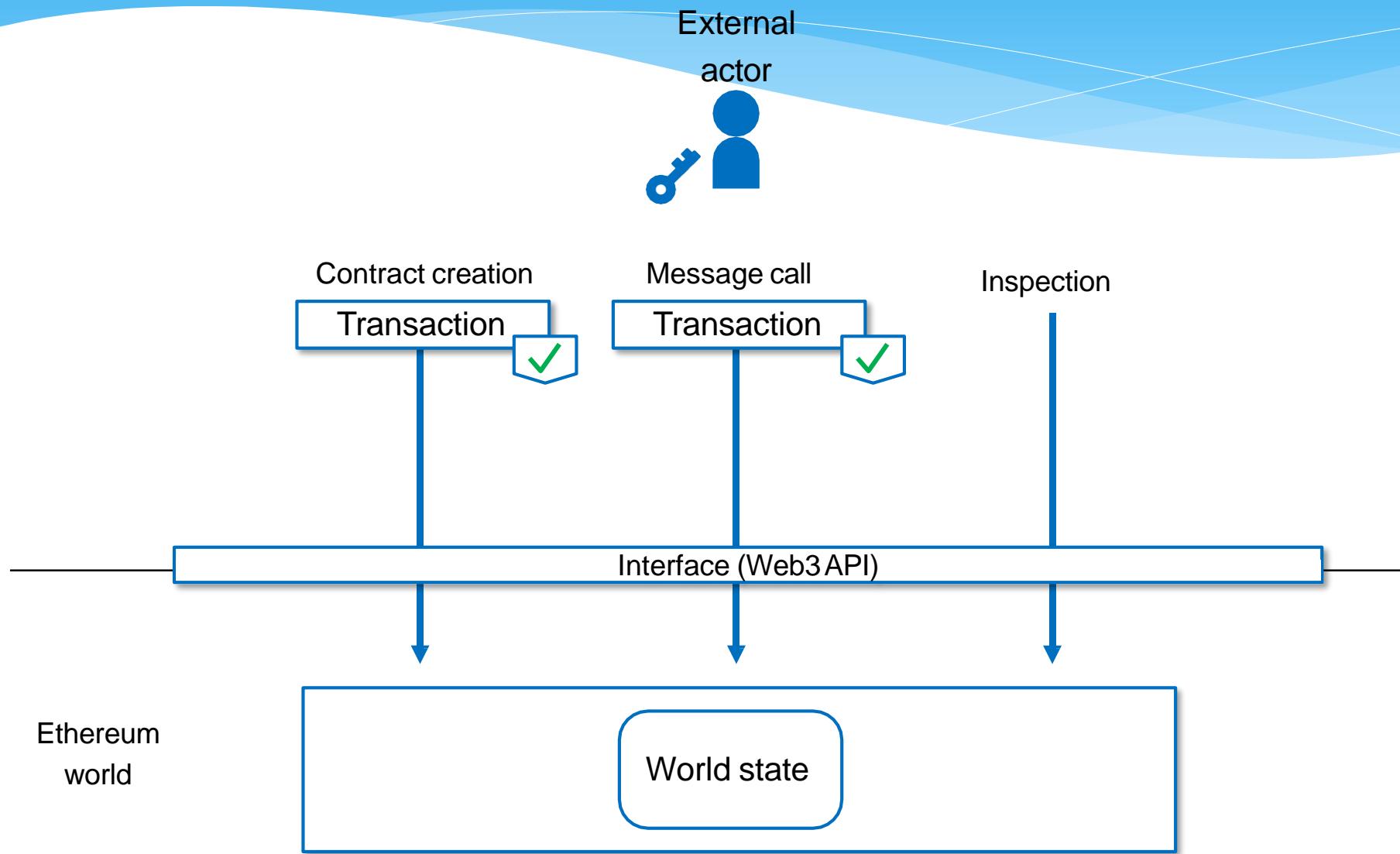
A blockchain is a globally shared, **decentralised**, transactional database.

P2P network inter nodes

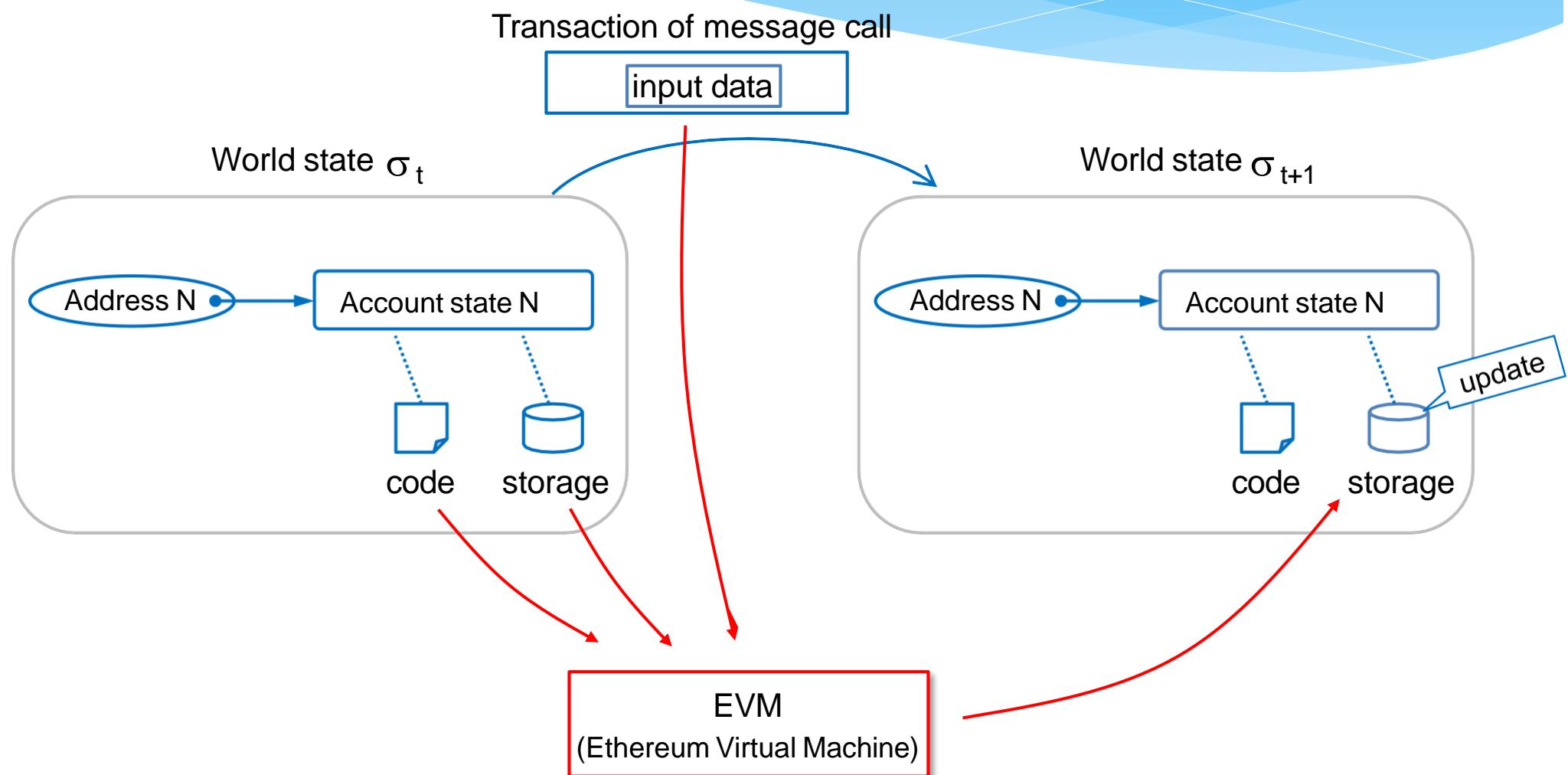


Decentralised nodes constitute Ethereum P2P network.

Interface to a node



Ethereum virtual machine(EVM)



Ethereum virtual machine(EVM)

Code

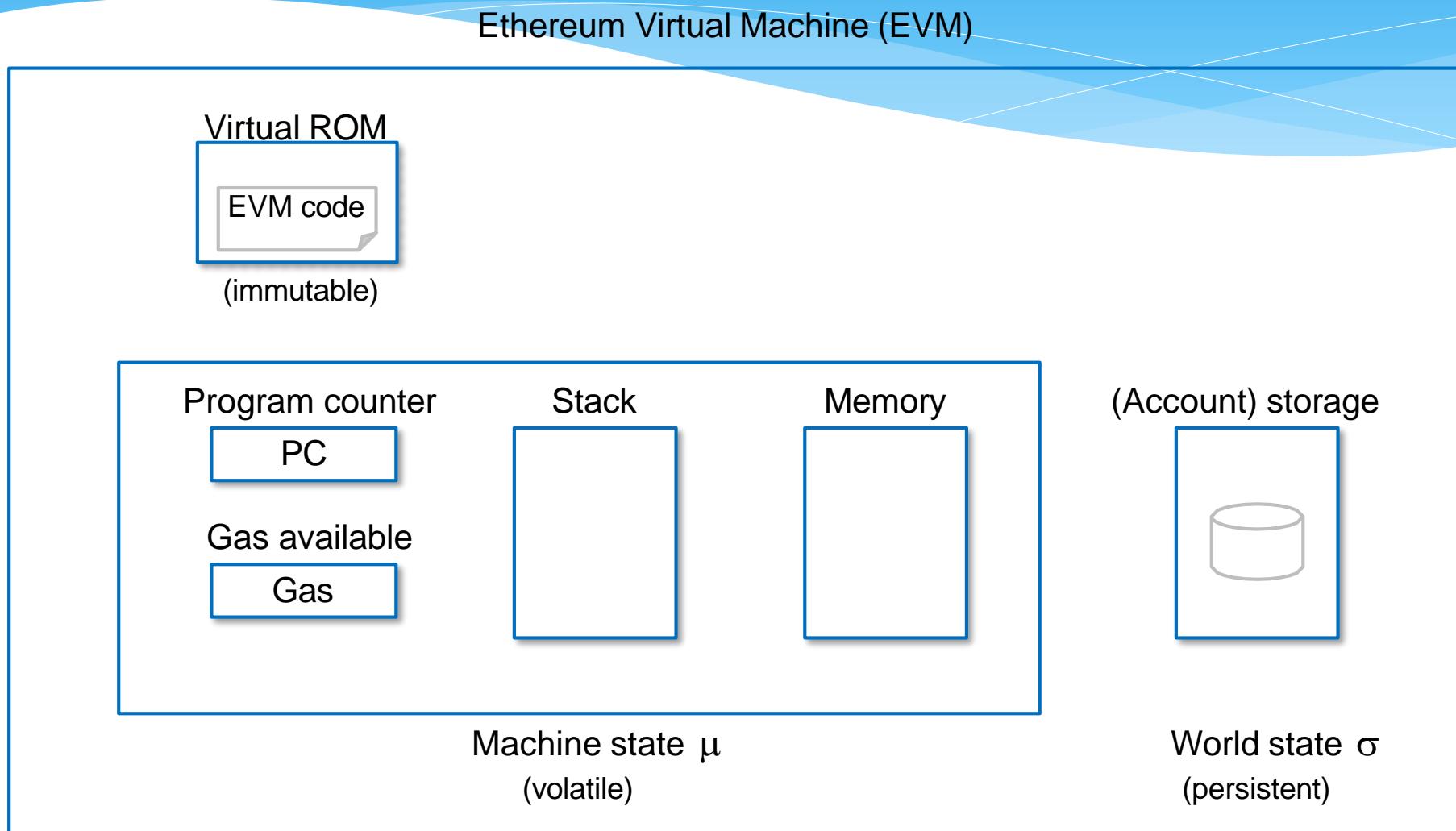
EVM code

Virtual machine

EVM (Ethereum Virtual Machine)

The Ethereum Virtual Machine is the runtime environment for smart contracts in Ethereum.

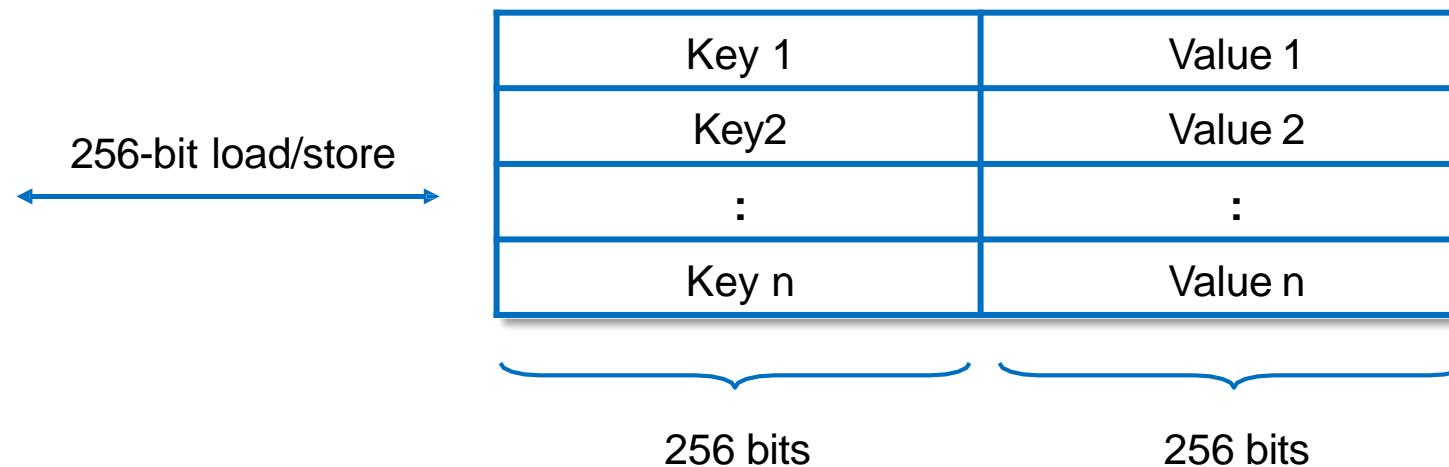
EVM architecture



The EVM is a simple stack-based architecture.

Account storage

(Account) storage



Storage is a key-value store that maps 256-bit words to 256-bit words.
Access with SSTORE/SLOAD instructions.
All locations in storage are well-defined initially as zero.

EVM code

Assembly view

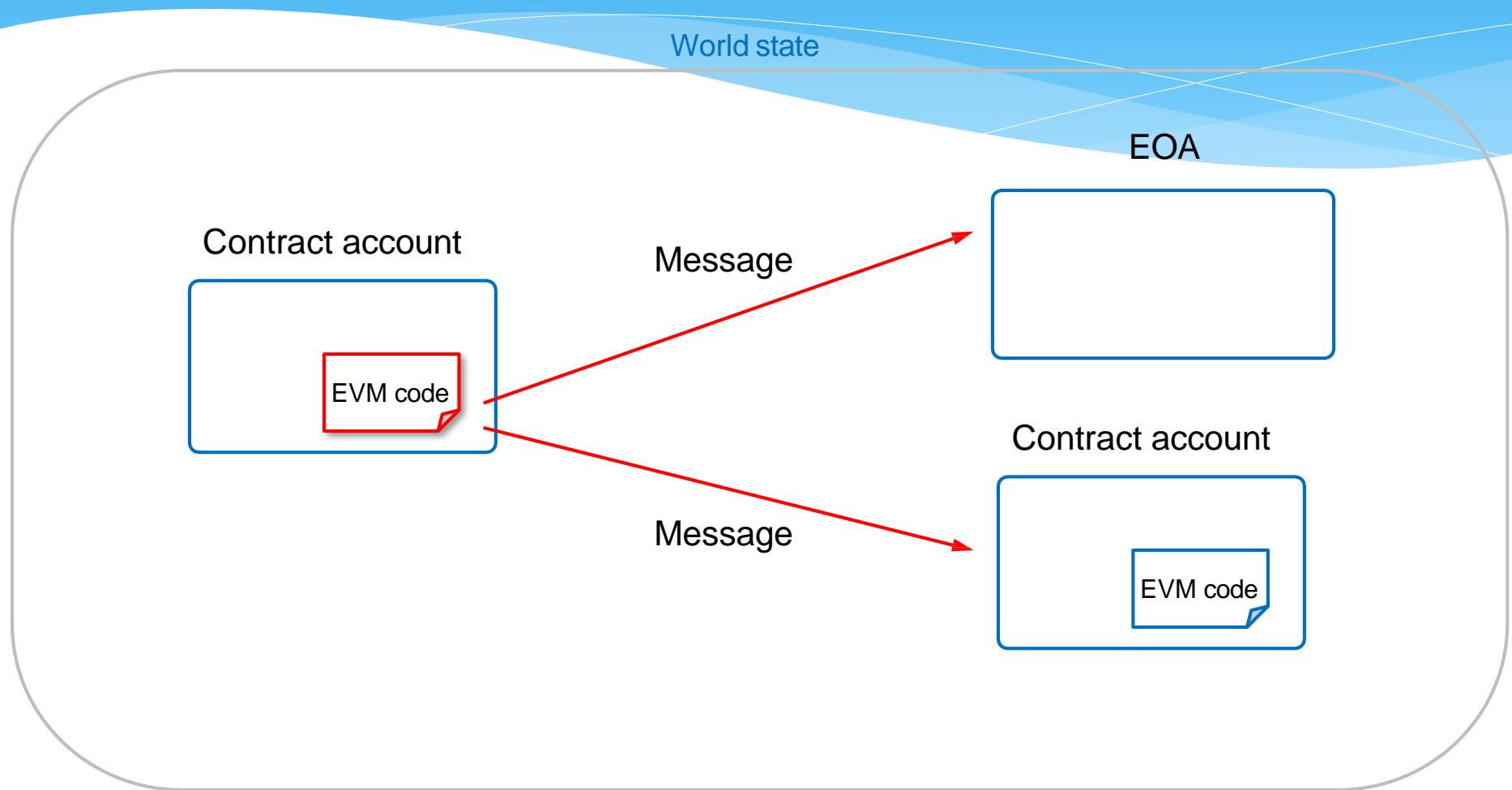
```
PUSH1 e0  
PUSH1 02  
EXP  
PUSH1 00  
CALLDATALOAD  
:
```

Bytecode view

```
0x60e060020a600035...
```

EVM Code is the bytecode that the EVM can natively execute.

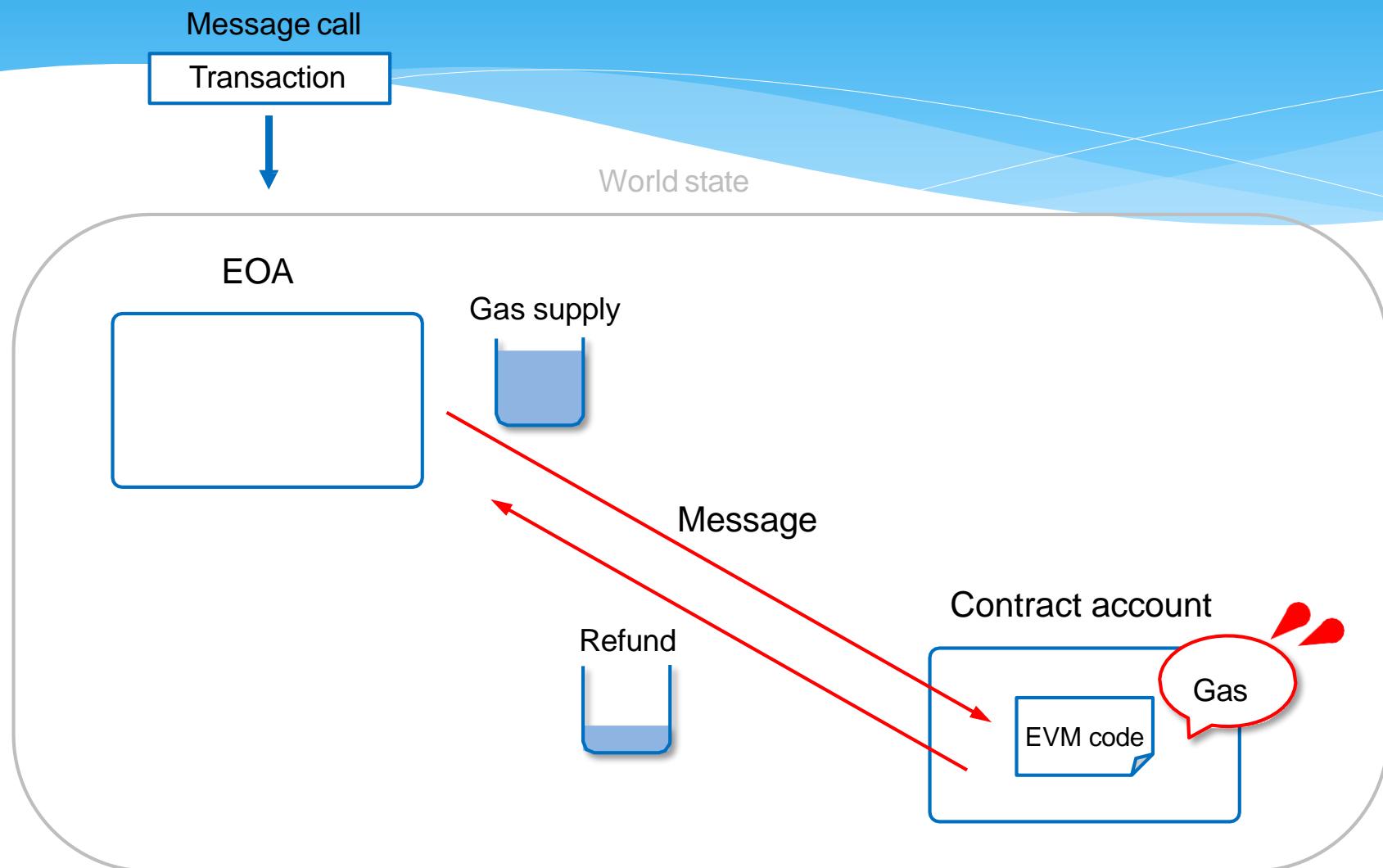
Message call



EVM can send a message to other account.

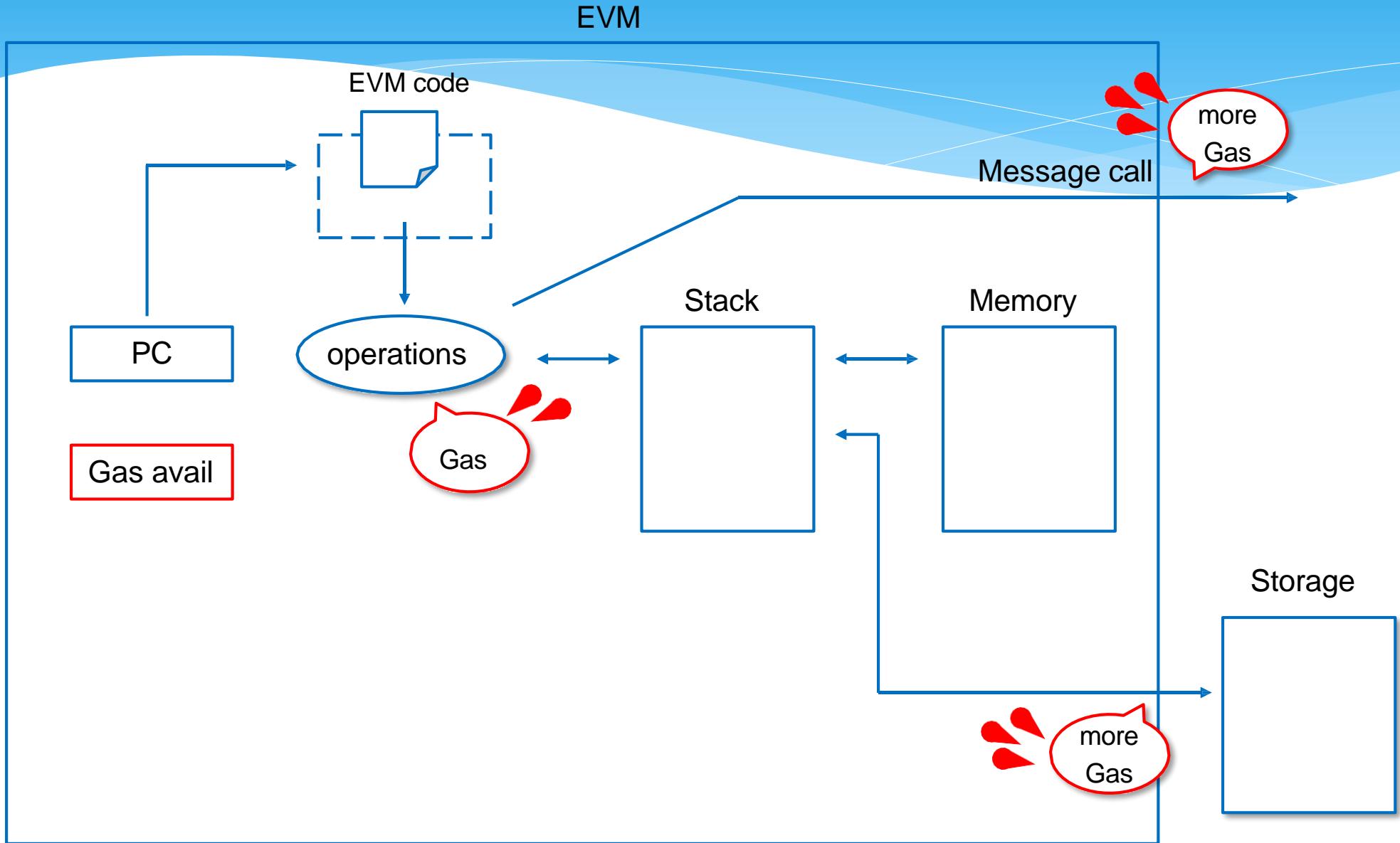
The depth of message call is limited to less than 1024 levels.

Gas and fee

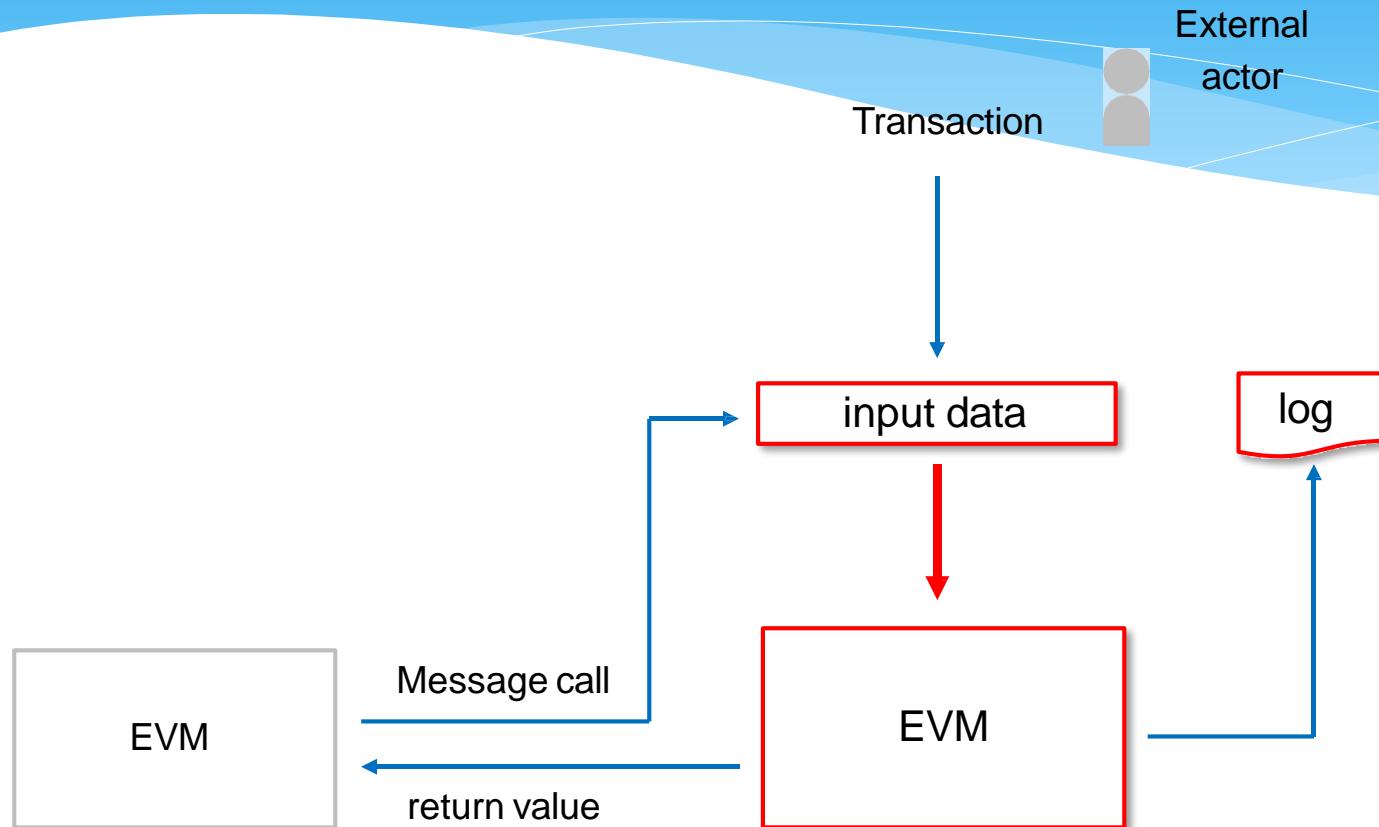


All programmable computation in Ethereum is subject to fees (denominated in gas).

Gas



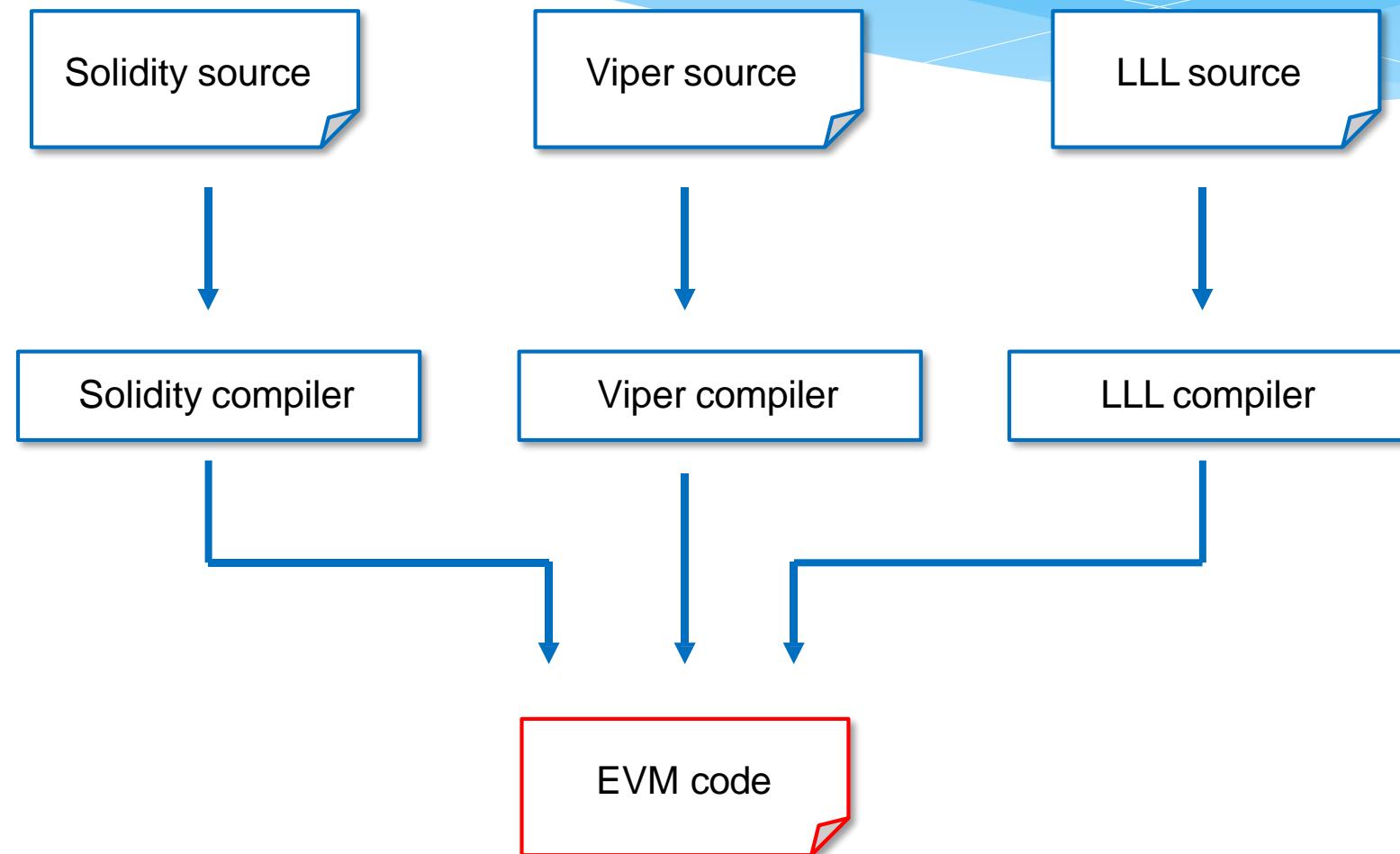
Input and Output of EVM



EVM can input external data from a message call.

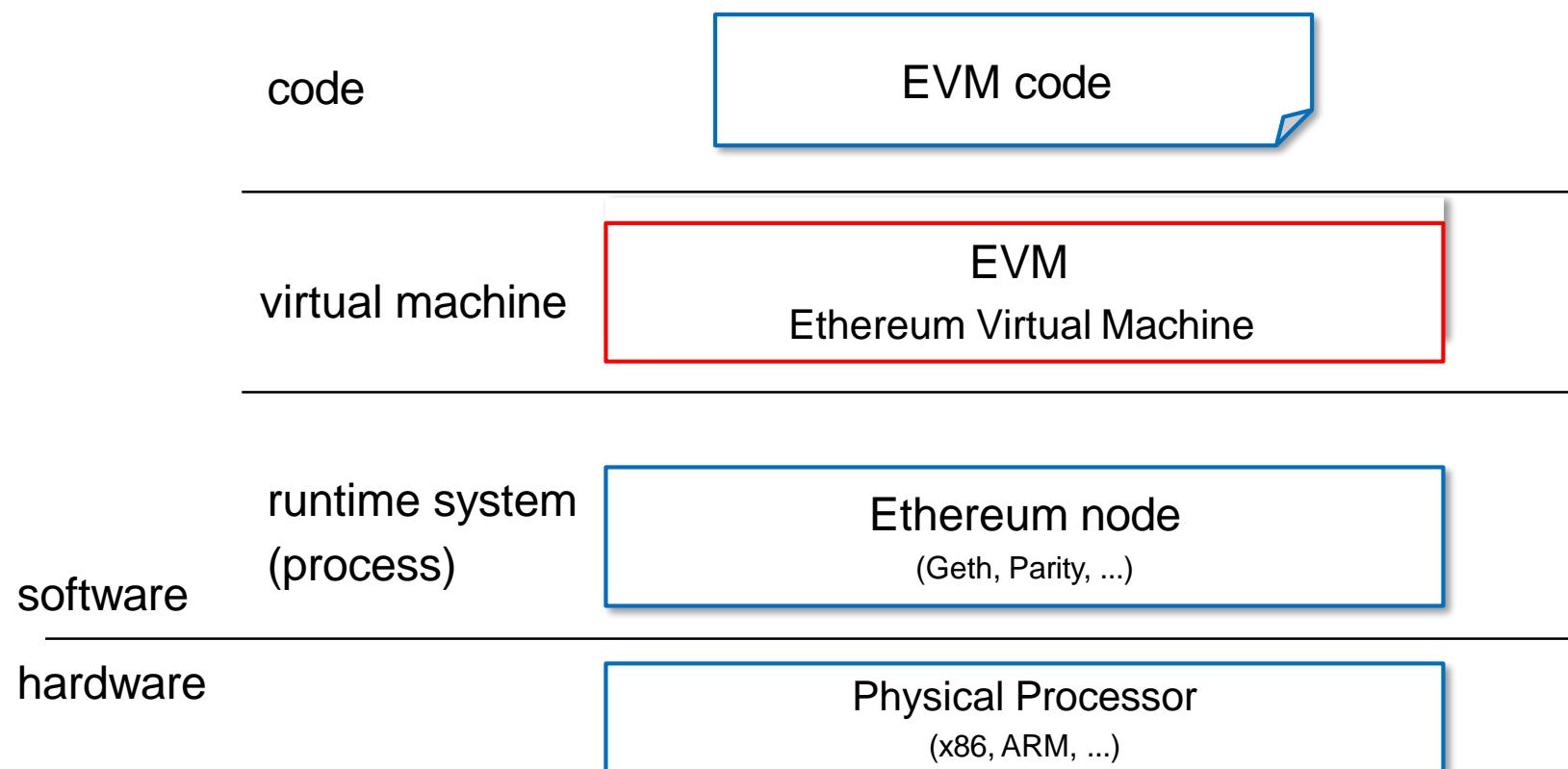
EVM can output log. EVM can also return values to Caller EVM.

EVM code generation



Ethereum virtual machine code

Ethereum virtual machine layer





Solidity

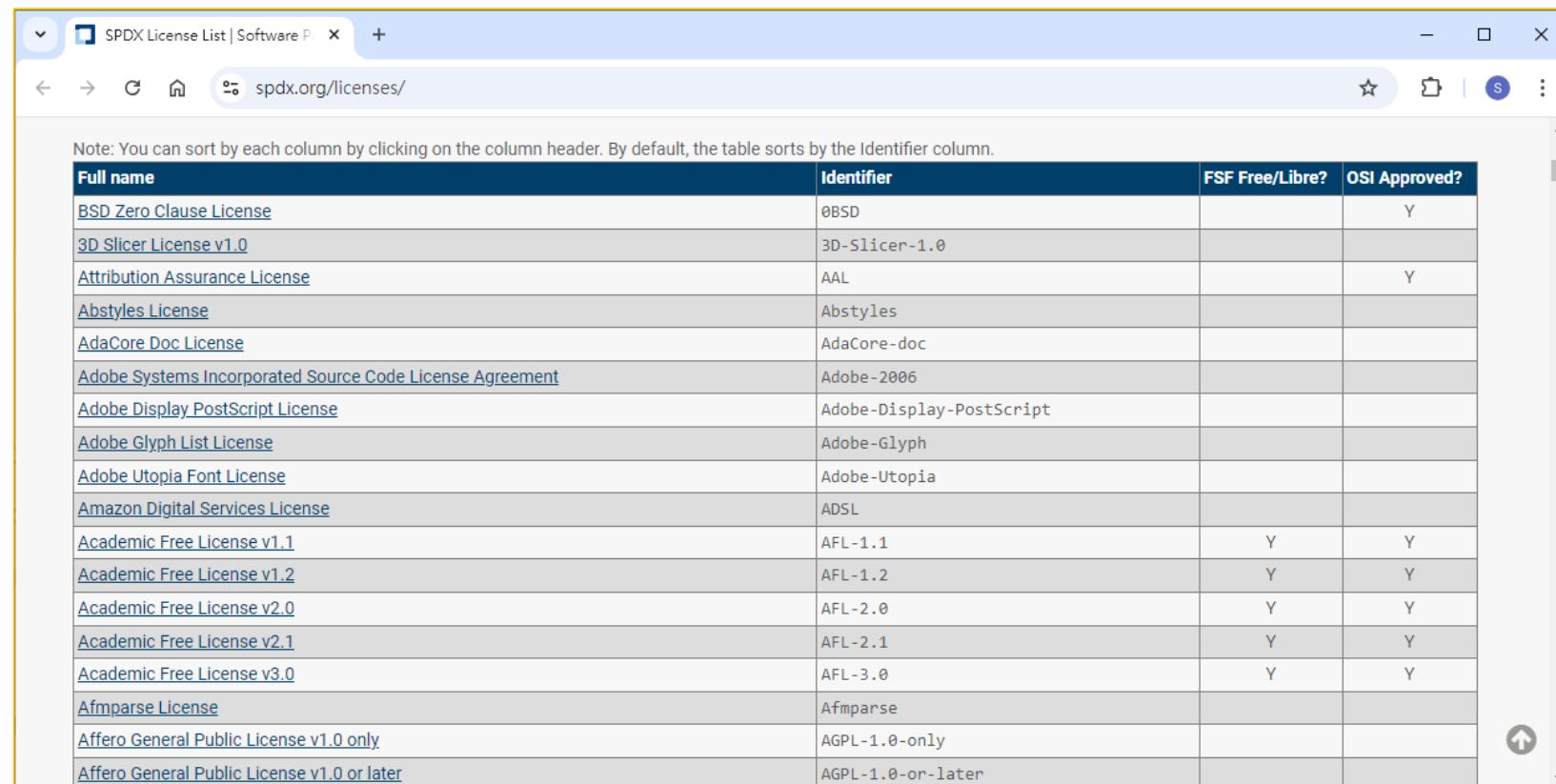
SPDX license識別碼

- SPDX(System Package Data Exchange)是一種開放標準，能夠表示具有 SBOM (Software Bill of Materials/軟體物料清單) 等軟體元件的系統，以及支援一系列風險管理使用案例的其他人工智慧、資料和安全參考。
- SPDX 規格是提供完全免費的國際開放標準(ISO/IEC 5692:2021)。
- 建立合約前，必須在最前面聲明採用那一個識別碼(identifier)。

```
// SPDX-License-Identifier: MIT  
pragma solidity ^0.8.4;
```

SPDX license List

■ 完整的SPDX清單於 <https://spdx.org/licenses/>



The screenshot shows a web browser window with the title "SPDX License List | Software P" and the URL "spdx.org/licenses/" in the address bar. The page content is a table of license information. A note at the top says: "Note: You can sort by each column by clicking on the column header. By default, the table sorts by the Identifier column." The table has four columns: "Full name", "Identifier", "FSF Free/Libre?", and "OSI Approved?". The "Full name" column contains links to individual license pages. The "Identifier" column contains short codes. The "FSF Free/Libre?" and "OSI Approved?" columns contain "Y" or "N" indicating their status.

Full name	Identifier	FSF Free/Libre?	OSI Approved?
BSD Zero Clause License	0BSD		Y
3D Slicer License v1.0	3D-Slicer-1.0		
Attribution Assurance License	AAL		Y
Abstyles License	Abstyles		
AdaCore Doc License	AdaCore-doc		
Adobe Systems Incorporated Source Code License Agreement	Adobe-2006		
Adobe Display PostScript License	Adobe-Display-PostScript		
Adobe Glyph List License	Adobe-Glyph		
Adobe Utopia Font License	Adobe-Utopia		
Amazon Digital Services License	ADSL		
Academic Free License v1.1	AFL-1.1	Y	Y
Academic Free License v1.2	AFL-1.2	Y	Y
Academic Free License v2.0	AFL-2.0	Y	Y
Academic Free License v2.1	AFL-2.1	Y	Y
Academic Free License v3.0	AFL-3.0	Y	Y
Afmparse License	Afmparse		
Afferro General Public License v1.0 only	AGPL-1.0-only		
Afferro General Public License v1.0 or later	AGPL-1.0-or-later		

備註說明

- 單行備註使用雙斜線//在說明的前面。
- 多行備註使用 /*...*/ 把備註包括在裏面。

```
// This is a single-line comment.
```

```
/*
This is a
multi-line comment.
*/
```

版本宣告

- 每次建立合約前，必須聲明solidity採用的版本。

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.4;

//pragma solidity >=0.8.4 <0.9.0;
//Demo updated build Aug/10/2024
contract MyDemo {
    // Basic variables
    address _owner;
}
```

- 程式碼格式類似js，以分號;來區隔指令。
- ^0.8.4表示compiler只適用於0.8.4<=X<0.9.0之間。

Importing other Source Files

- 使用import來載入其他的.sol檔案。

```
import "filename";  
  
// import '@openzeppelin/contracts/token/ERC721/IERC721.sol';
```

- 建立一個新的全域符號**symbolName**，其成員是「filename」中的所有全域符號。

```
import * as symbolName from "filename";  
  
// import {IERC20} from "./IERC20.sol";
```

狀態變數(State Variables)

- 狀態變數會被永久存入合約中，類似於寫入database。

```
// 宣告變數的方式，也與js 非常類似
// contract建立一個合約外層，並在合約內建立變數
contract MyDemo {
    // 建立一個變數amount，指定其為uint資料型態，並給予賦值
    uint amount = 10; // State variable
}
```

加減運算

■ solidity的數學計算，明確直白，就是一般數學符號進行計算。

```
1 + 1  
2 - 1  
10 * 2  
6 / 3  
2 ** 2 (平方運算)
```

因此我們也可以宣告變數來當作運算用數字

```
contract MyDemo {  
    uint qty = 5;  
    uint price = 10;  
    uint totalPrice = qty * price;  
}
```

String Literals and Types

- String literals can only contain printable ASCII characters, which means the characters between and including 0x20 ~ 0x7E.
- 另支援以下特殊字元。

- \<newline> (escapes an actual newline)
- \\ (backslash)
- \' (single quote)
- \" (double quote)
- \n (newline)
- \r (carriage return)
- \t (tab)
- \xNN (hex escape)
- \uNNNN (unicode escape)

// 宣告變數的方式如下
string tokenName;

Unicode Literals

- While regular string literals can only contain ASCII, Unicode literals – prefixed with the keyword `unicode` – can contain any valid UTF-8 sequence. They also support the very same escape sequences as regular string literals.

```
//宣告變數的方式如下  
string memory test = unicode"Hello 😊";
```

Address Literals

- Hexadecimal literals that are between 39 and 41 digits long and do not pass the checksum test produce an error.
- You can prepend (for integer types) or append (for bytesNN types) zeros to remove the error.

```
//宣告變數的方式如下
```

```
address fromAddress = 0xCad3a6d3569DF655070DEd06cb7A1b2Ccd1D3AF;
```

結構體(struct Types)

- solidity提供一種名為結構體(struct)的資料型態表示方法。
- 結構體允許生成複雜的資料型態，同時擁有多個屬性。
- 下列程式碼中，結構體內的uint、string為資料型態。

```
//宣告變數的方式如下
```

```
struct Student { // struct
    string name;
    uint age;
}
```

陣列資料型態(Array)

- solidity有兩種陣列資料型態，分別是靜態與動態。
- 所有陣列都由連續的記憶體位置組成。最低地址對應於第一個元素，最高地址對應於最後一個元素。

// 靜態

```
uint[10] fixedArray; //固定長度為10的靜態數字陣列  
string[20] stringArray; //固定長度為20且是string類型的靜態陣列  
uint balance[3] = [1, 2, 3]; //固定長度為3,且設定初始值  
uint balance[] = [1, 2, 3]; //和上面的作法是一樣的
```

// 動態

```
uint[] dynamicArray; //長度不固定的動態數字陣列
```

- 公開陣列，提供保存公開資料，僅供其他合約讀取資料，而不能寫入。

```
// 使用public宣告公開陣列
```

```
myAccount[] public my;
```

Function

- Functions are the executable units of code. Functions are usually defined inside a contract, but they can also be defined outside of contracts.

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.1 <0.9.0;

contract SimpleAuction {
    function bid() public payable { // Function
        // ...
    }
}

// Helper function defined outside of a contract
function helper(uint x) pure returns (uint) {
    return x * 2;
}
```

Events

- Events are convenience interfaces with the EVM logging facilities ◉

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity ^0.8.22;

event HighestBidIncreased(address bidder, uint amount); // Event

contract SimpleAuction {
    function bid() public payable {
        // ...
        emit HighestBidIncreased(msg.sender, msg.value); // Triggering event
    }
}
```

revert

- A direct revert can be triggered using the revert statement and the revert function.
- The revert statement takes a custom error as direct argument without parentheses:
`revert CustomError(arg1, arg2);`
- For backward-compatibility reasons, there is also the revert() function, which uses parentheses and accepts a string:
`revert(); revert("description");`
- The error data will be passed back to the caller and can be caught there. Using revert() causes a revert without any error data while revert("description") will create an Error(string) error.

Errors

- Errors allow you to define descriptive names and data for failure situations. Errors can be used in `revert` statements. In comparison to string descriptions, errors are much cheaper and allow you to encode additional data. You can use NatSpec to describe the error to the user.

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity ^0.8.4;

/// Not enough funds for transfer. Requested `requested`,
/// but only `available` available.
error NotEnoughFunds(uint requested, uint available);

contract Token {
    mapping(address => uint) balances;
    function transfer(address to, uint amount) public {
        uint balance = balances[msg.sender];
        if (balance < amount)
            revert NotEnoughFunds(amount, balance);
        revert("Not enough Ether provided.");
        balances[msg.sender] -= amount;
        balances[to] += amount;
        // ...
    }
}
```

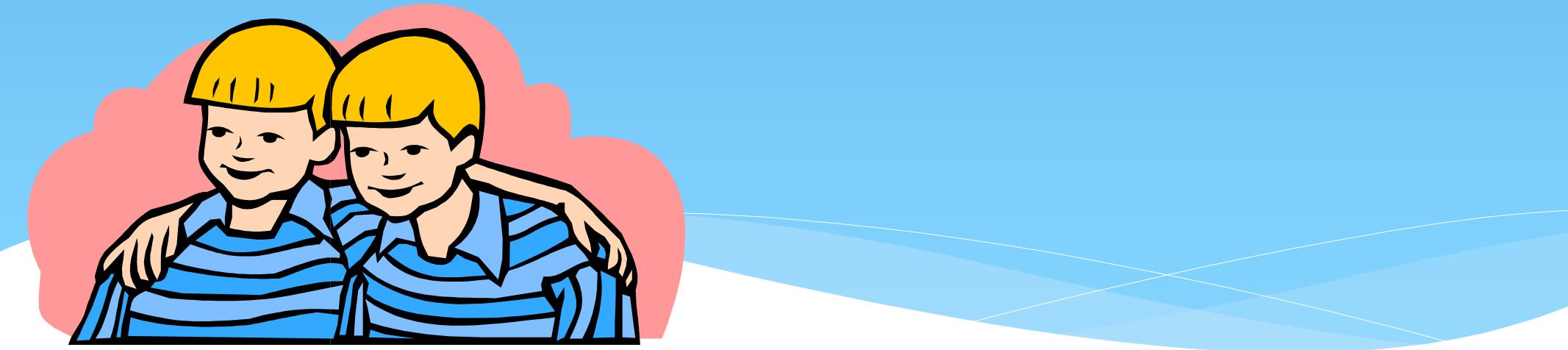
Enum Types

- Enums can be used to create custom types with a finite set of ‘constant values’ (see Enums in types section).

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.4.0 <0.9.0;

contract Purchase {
    enum State { Created, Locked, Inactive } // Enum
}
```

Thank you !





Start to Deploy Smart Contract

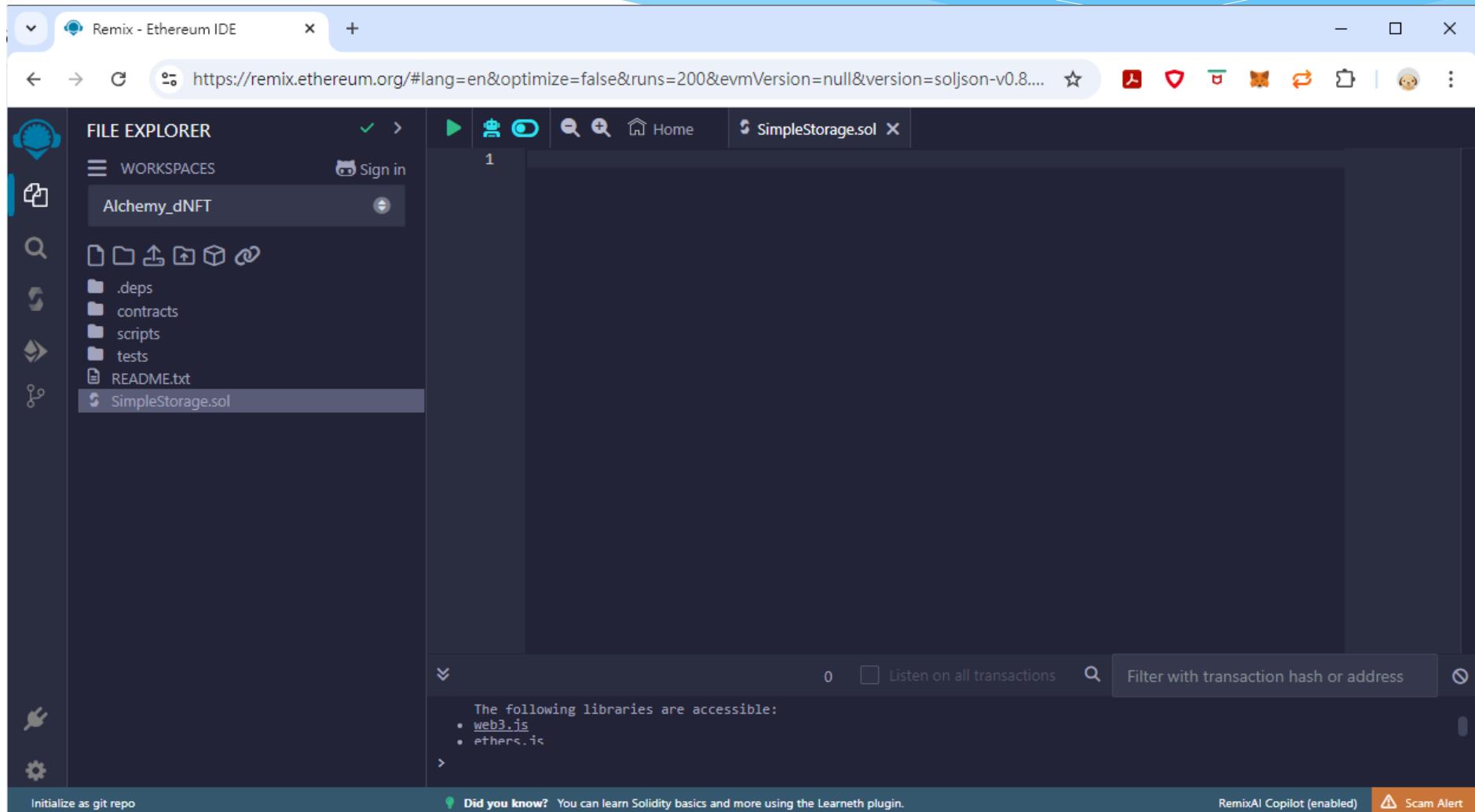


1. Deploy Smart Contract to Sepolia Testnet

Create new file

The screenshot shows the Remix Ethereum IDE interface. On the left, the **FILE EXPLORER** panel displays a workspace named **Alchemy_dNFT**. A red box highlights the **New File** icon (a document with a plus sign) in the sidebar of the file explorer. The central area features the **REMIX** logo and the tagline *The Native IDE for Web3 Development.* Below it, there are several buttons: **Start Coding**, **ZK Semaphore**, **ERC20**, **Uniswap V4 Hooks**, **NFT / ERC721**, and **MultiSig**. To the right, the **Featured** section includes a cartoon character, a **v0.54.0 RELEASE HIGHLIGHTS** section with three bullet points, and a **What's New** button. At the bottom, there are sections for **Recent Workspaces**, **Featured Plugins**, and a footer with links like **Did you know?**, **RemixAI Copilot (enabled)**, and **Scam Alert**.

SimpleStorage.sol



SimpleStorage.sol

The screenshot shows the Ethereum IDE interface with a prominent red box highlighting a modal dialog titled "Pasted Code Alert". The dialog contains a warning message about the risks of running untrusted code, including the potential loss of funds. It also provides a link for more information and an "OK" button at the bottom right. The background shows the file explorer with a project named "Alchemy_dNFT" and a Solidity file named "SimpleStorage.sol" selected.

Pasted Code Alert

⚠ You have just pasted a code snippet or contract in the editor.
Make sure you fully understand this code before deploying or interacting with it.
Don't get scammed!

Running untrusted code can put your wallet at risk . In a worst-case scenario, you could lose all your money.

If you don't fully understand it, please don't run this code.

If you are not a smart contract developer, ask someone you trust who has the skills to determine if this code is safe to use.

See [these recommendations](#) for more information.

OK

SimpleStorage.sol

The screenshot shows the Ethereum IDE interface with the file `SimpleStorage.sol` open. The code is highlighted with a red box.

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

The interface includes a FILE EXPLORER sidebar with a workspace named `Alchemy_dNFT` containing files like `.deps`, `contracts`, `scripts`, `tests`, `README.txt`, and `SimpleStorage.sol`. The bottom status bar displays network information (`wcuju.j2`, `ethers.js`) and transaction monitoring options.

Compile Smart Contract

The screenshot shows the Ethereum IDE interface with the title "Remix - Ethereum IDE". The left sidebar contains the "SOLIDITY COMPILER" section, which includes a "COMPILER" dropdown set to "0.8.26+commit.8a97fa7a", a checkbox for "Auto compile" (which is checked and highlighted with a red box), and a "Compile SimpleStorage.sol" button (also highlighted with a red box). Below these are "Advanced Configurations" and "Compile and Run script" options. The main workspace displays a Solidity code for a "SimpleStorage" contract:

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

The bottom of the interface features a footer with links for "Initialize as git repo", "Did you know?", "RemixAI Copilot (enabled)", and "Scam Alert".

Deploy and Run Transaction

The screenshot shows the Remix Ethereum IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' sidebar is visible, featuring a red box around the 'ENVIRONMENT' dropdown set to 'Injected Provider - MetaMask' and a red box around the 'Deploy' button. A red arrow points from the 'Deploy' button towards the MetaMask window. The central area displays the Solidity code for the 'SimpleStorage' contract:

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

On the right, the MetaMask extension window is open, showing a welcome screen with the text '歡迎回來!' (Welcome back!) and a lock icon. It also includes a password field and a 'Unlock' button.

Deploy and Run Transaction

The screenshot shows two open windows: the Remix Ethereum IDE and the Salu 1 wallet.

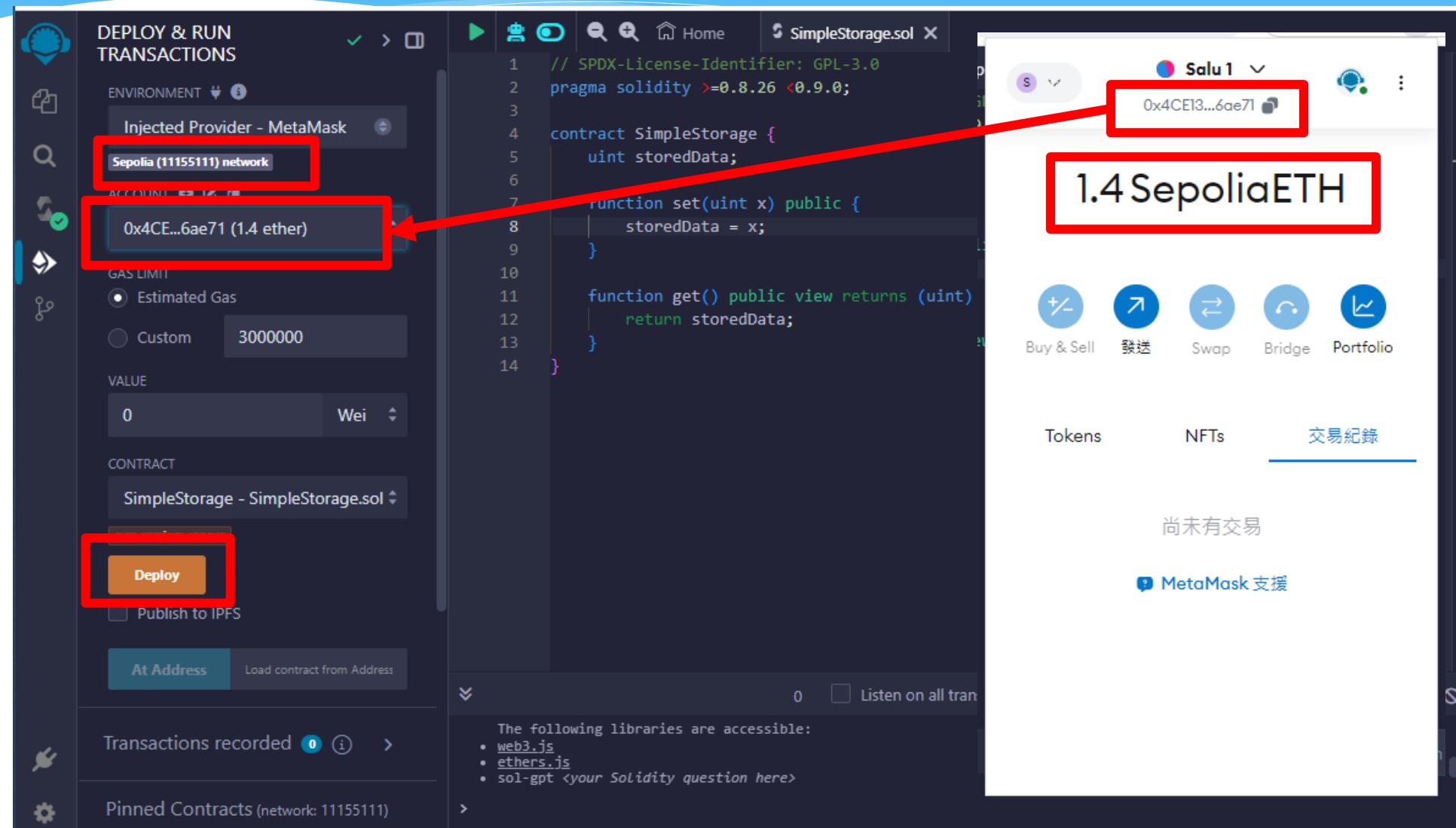
Remix - Ethereum IDE (Left Window):

- DEPLOY & RUN TRANSACTIONS:**
 - ENVIRONMENT:** Injected Provider - MetaMask, Sepolia (11155111) network.
 - ACCOUNT:** A dropdown menu showing the account address.
 - GAS LIMIT:** Estimated Gas (selected), Custom value: 3000000.
 - VALUE:** 0 Wei.
 - CONTRACT:** SimpleStorage - SimpleStorage.sol.
 - Deploy:** A large orange button.
 - Publish to IPFS:** An unchecked checkbox.
 - At Address:** A button.
 - Transactions recorded:** 0.
 - Pinned Contracts:** (network: 11155111)
- Code Area:** Solidity code for SimpleStorage contract.
- Bottom Bar:** Shows accessible libraries: web3.js, ethers.js, sol-gpt <your Solidity question here>.

Salu 1 (Right Window):

- Header:** Salu 1, Account address: 0x4CE13...6ae71.
- Balance:** 1.4 SepoliaETH.
- Buttons:** Buy & Sell, 發送 (Send), Swap, Bridge, Portfolio.
- Tabs:** Tokens, NFTs, 交易紀錄 (Transactions). The Transactions tab is selected.
- Message:** 尚未有交易 (No transactions yet).
- MetaMask Support:** A note about MetaMask support.
- Alert:** A red-bordered alert box stating "Salu 1 isn't connected to remix.ethereum.org" and "Connect account".

Deploy and Run Transaction



Deploy and Run Transaction

The screenshot shows the Remix IDE interface for deploying and running a transaction.

Left Panel (Deploy & Run Transactions):

- ENVIRONMENT:** Injected Provider - MetaMask, Sepolia (11155111) network.
- ACCOUNT:** 0x4CE...6ae71 (1.4 ether).
- GAS LIMIT:** Estimated Gas (selected), Custom value: 3000000.
- VALUE:** 0 Wei.
- CONTRACT:** SimpleStorage - SimpleStorage.sol.
- Deploy** button.
- Publish to IPFS** checkbox.
- At Address** and **Load contract from Address** buttons.
- Transactions recorded**: 0.
- Pinned Contracts** (network: 11155111).

Middle Panel (Solidity Code):

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

Right Panel (MetaMask Confirmation):

MetaMask window showing the Sepolia network. A red box highlights the "Sepolia" network selection and the "建立新合約" (Create New Contract) button.

Estimated changes: No changes predicted for your wallet.

Estimated fee: 0.02444487 SepoliaETH (Market -60 sec, Max fee: 0.03293543 SepoliaETH).

Buttons at the bottom: 拒絕 (Reject) and 確認 (Confirm).

Deploy and Run Transaction

The image displays three panels illustrating the process of deploying and running a transaction.

Left Panel: Deploy & Run Transactions

- ENVIRONMENT:** Injected Provider - MetaMask, Sepolia (11155111) network.
- ACCOUNT:** 0x4CE13...6ae71 (1.4 ether).
- GAS LIMIT:** Estimated Gas (selected), Custom value: 3000000.
- VALUE:** 0 Wei.
- CONTRACT:** SimpleStorage - SimpleStorage.sol.
- Deploy** button.
- Publish to IPFS** checkbox.
- At Address** and **Load contract from Address** buttons.
- Transactions recorded**: 0.
- Pinned Contracts**: (network: 11155111).

Middle Panel: Solidity Code

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

Right Panel: Wallet Interface

- Balances:** 1.3784 SepoliaETH.
- Actions:** Buy & Sell, 發送, Swap, Bridge, Portfolio.
- Transactions:** 部署合約 已確認 (Deployed Contract Confirmation). This transaction is highlighted with a red border.
- Logs:** -0 SepoliaETH, -0 SepoliaETH.
- MetaMask Support**.

Deploy and Run Transaction

The screenshot shows the Truffle UI interface for deploying a Solidity contract. On the left, the 'DEPLOY & RUN TRANSACTIONS' sidebar is visible, showing the environment set to 'Injected Provider - MetaMask' on the 'Sepolia (11155111) network'. The account selected is '0x4CE...6ae71 (1.4 ether)'. The GAS LIMIT is set to 'Estimated Gas' (radio button selected). The VALUE is set to 0 Wei. The CONTRACT dropdown shows 'SimpleStorage - SimpleStorage.sol'. Below it, the EVM version is set to 'cancun'. A large orange 'Deploy' button is prominent. Other options include 'Publish to IPFS' and address selection buttons for 'At Address' or 'Load contract from Address'. At the bottom, there's a 'Transactions recorded' section and a 'Pinned Contracts (network: 11155111)' section.

SimpleStorage.sol

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

部署合約

Status	View on block explorer
已確認	複製交易 ID
來源帳戶	目的帳戶
建立新合約	0x4CE13...6...
交易	
Nonce	0
數量	-0 SepoliaETH
Gas 上限 (單位)	124076
Gas 用量 (單位)	123005
Base fee (GWEI)	174.334649051
Priority fee (GWEI)	1.5
Total gas fee	0.021629 SepoliaETH
Max fee per gas	0.000000237 SepoliaETH
總量	0.02162854 SepoliaETH

View Transaction on Sepolia

The screenshot shows a web browser window displaying a transaction on the Sepolia Testnet. The URL in the address bar is <https://sepolia.etherscan.io/tx/0x90a9c066ddfe726df9f3e867533e684251f23975945785e9190961431610dc7a>. The page header indicates it is the 'Sepolia Testnet'. The main content area shows the transaction details under the 'Overview' tab:

- [This is a Sepolia **Testnet** transaction only]
- Transaction Hash: [0x90a9c066ddfe726df9f3e867533e684251f23975945785e9190961431610dc7a](#)
- Status: Success
- Block: [6807711](#) 46 Block Confirmations
- Timestamp: [10 mins ago \(Oct-03-2024 03:40:48 PM UTC\)](#)
- Transaction Action: [Call 0x60806040 Method by 0x4CE135aB...64E06ae71](#)
- From: [0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71](#)
- To: [\[0x73a61e22f57d9534110d76e8e92657750345bfc5 Created \]](#) ✓

A red box highlights the 'To' field entry: [0x73a61e22f57d9534110d76e8e92657750345bfc5 Created].

Smart Contract Information

The screenshot shows a browser window displaying the Etherscan.io interface for a Sepolia Testnet contract. The URL in the address bar is <https://sepolia.etherscan.io/address/0x73a61e22f57d9534110d76e8e92657750345bfc5>. The page title is "Contract Address 0x73a61e22f57d9534110d76e8e92657750345bfc5".

The main content area includes:

- Contract Address:** 0x73a61e22f57d9534110d76e8e92657750345bfc5 (highlighted by a red box)
- ETH BALANCE:** 0 ETH (highlighted by a red box)
- CONTRACT CREATOR:** 0x4CE135aB...64E06ae71 (highlighted by a red box)
- Multichain Info:** N/A

Below these sections, there are tabs for **Transactions**, **Token Transfers (ERC-20)**, **Contract**, and **Events**. The **Transactions** tab is selected.

The transaction table shows the following data:

Transaction Hash	Method	Block	Age	From	To	Amount	Txn Fee
0x90a9c066ddf...	0x60806040	6807711	14 mins ago	0x4CE135aB...64E06ae71	IN	Contract Creation	0 ETH 0.02162854

A red box highlights the entire transaction row in the table.

View & Publish

The screenshot shows a web browser window displaying the Etherscan.io contract details page for the Sepolia Testnet. The URL in the address bar is <https://sepolia.etherscan.io/address/0x73a61e22f57d9534110d76e8e92657750345bfc5#code>. The page has a header with tabs for "Sepolia Testnet", "OVERVIEW", "More Info", and "Marketplace". Below these are sections for "ETH BALANCE" (0 ETH) and "CONTRACT CREATOR" (0x4CE135aB...64E06ae71 at tx 0x90a9c066...). At the bottom, there are tabs for "Transactions", "Token Transfers (ERC-20)", "Contract" (which is selected), and "Events". A message "Are you the contract creator? Verify and Publish your contract source code today!" is displayed with a red rectangular box highlighting the "Verify and Publish" link. Below this are buttons for "Decompile Bytecode" and "Switch to Opcodes View", and a "Similar Contracts" section containing a large amount of hex code.

View & Publish Contract Source Code

The screenshot shows a web browser window titled "Verify & Publish Contract Sou". The URL in the address bar is <https://sepolia.etherscan.io/verifyContract?a=0x73a61e22f57d9534110d76e8e92657750345bfc5>. The page is titled "Verify & Publish Contract Source Code" and explains that source code verification provides transparency by matching uploaded source code with blockchain data. It shows two steps: "Enter Contract Details" (selected) and "Verify & Publish".

1 Enter Contract Details — **2 Verify & Publish**

Please enter the Contract Address you would like to verify
0x73a61e22f57d9534110d76e8e92657750345bfc5

Please select Compiler Type
Solidity (Single file)

Please select Compiler Version
v0.8.26+commit.8a97fa7a

Uncheck to show all nightly commits

Please select Open Source License Type ⓘ
5) GNU General Public License v3.0 (GNU GPLv3)

I agree to the [terms of service](#)

Continue Reset

View & Publish Contract Source Code

The screenshot shows a web browser window titled "Sepolia Solidity Contract Sour" with the URL <https://sepolia.etherscan.io/verifyContract-solc?a=0x73a61e22f57d9534110d76e8e926577...>. The page is titled "Verify & Publish Contract Source Code". It explains that source code verification provides transparency by matching uploaded code with blockchain data. It's a simple interface for contracts fitting in a single file. A progress bar indicates step 1 (Enter Contract Details) is complete and step 2 (Verify & Publish) is in progress.

Source code verification provides transparency for users interacting with smart contracts. By uploading the source code, Etherscan will match the compiled code with that on the blockchain. [Read more](#).

A simple and structured interface for verifying smart contracts that fit in a single file.

1 Enter Contract Details — 2 Verify & Publish

Upload Contract Source Code

1. If the contract compiles correctly at REMIX, it should also compile correctly here.
2. We have limited support for verifying contracts created by another contract and there is a timeout of up to 45 seconds for each contract compiled.
3. For programmatic contract verification, check out the [Contract API Endpoint](#).

Contract Address:
0x73a61e22f57d9534110d76e8e92657750345bfc5

Compiler Type:
SINGLE FILE / CONCATENATED METHOD

Compiler Version:
v0.8.26+commit.8a97fa7a

Enter the Solidity Contract Code below *

Fetch from Gist

View & Publish Contract Source Code

The screenshot shows a web browser window with the title "Sepolia Solidity Contract Sour" and the URL "https://sepolia.etherscan.io/verifyContract-solc?a=0x73a61e22f57d9534110d76e8e926577...". The page is titled "Sepolia Testnet" and features a search bar for "Search by Address / Txn Hash / Block / Token". The main content area is titled "Enter the Solidity Contract Code below *". A red box highlights the contract code:

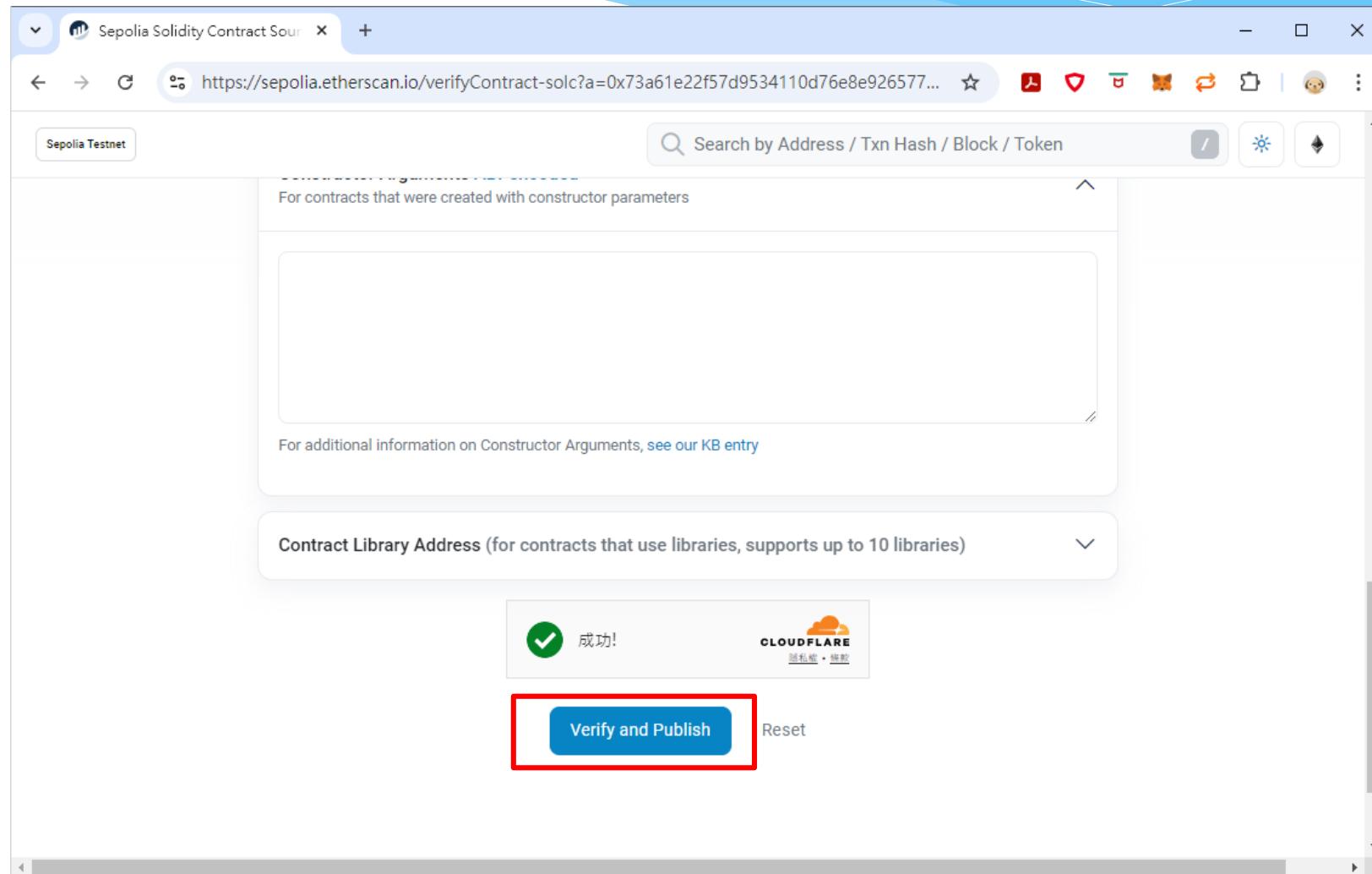
```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }
}
```

There is a "Fetch from Gist" button next to the input field. Below this, there is an "Advanced Configuration" section with three dropdowns: "Optimization" (set to "No"), "Runs (Optimizer)" (set to "200"), and "EVM Version to target" (set to "default (compiler defaults)"). Another red box highlights the "License Type" section, which shows "5) GNU General Public License v3.0 (GNU GPL)".

View & Publish Contract Source Code



View & Publish Contract Source Code

The screenshot shows a web browser window with the title "Sepolia Solidity Contract Sour" and the tab "SimpleStorage | Address 0x73...". The URL in the address bar is <https://sepolia.etherscan.io/verifyContract-solc?a=0x73a61e22f57d9534110d76e8e9265775...>. The page content is titled "Verify & Publish Contract Source Code". It explains that source code verification provides transparency by matching uploaded code with blockchain data. Below this, it says "A simple and structured interface for verifying smart contracts that fit in a single file." At the bottom, there are two steps: "1 Enter Contract Details" and "2 Verify & Publish". A red box highlights the message "Successfully generated Bytecode and ABI for Contract Address [0x73a61e22f57d9534110d76e8e92657750345bfc5]". A red arrow points from the word "click" to this message.

1 Enter Contract Details — 2 Verify & Publish

Successfully generated Bytecode and ABI for Contract Address
[0x73a61e22f57d9534110d76e8e92657750345bfc5]

Learn how to verify your contract on multiple blockchains with a single API key [here](#).

View Contract Source Code

The screenshot shows the Sepolia Testnet interface for a contract named "SimpleStorage". The browser address bar contains the URL <https://sepolia.etherscan.io/address/0x73a61e22f57d9534110d76e8e92657750345bfc5#code>. A red box highlights the address bar and the "Contract Source Code Verified (Exact Match)" message. Another red box highlights the "Contract Source Code (Solidity)" section, which displays the following Solidity code:

```
1 /**
2  *Submitted for verification at Etherscan.io on 2024-10-03
3  */
4
5 // SPDX-License-Identifier: GPL-3.0
6 pragma solidity >=0.8.26 <0.9.0;
7
8 contract SimpleStorage {
9     uint storedData;
10
11     function set(uint x) public {
12         storedData = x;
13     }
14
15     function get() public view returns (uint) {
16         return storedData;
17     }
18 }
```

The interface also shows the Compiler Version as v0.8.26+commit.8a97fa7a, Optimization Enabled as No with 200 runs, and Other Settings as default evmVersion, GNU GPLv3 license.

View & Publish Contract Source Code

The screenshot shows the Etherscan interface for the Sepolia Testnet. The address displayed is 0x73a61e22f57d9534110d76e8e92657750345bfc5. The 'Contract' tab is selected, highlighted with a red box. Below it, another red box highlights the 'Code' button in the navigation bar. The page displays the following information:

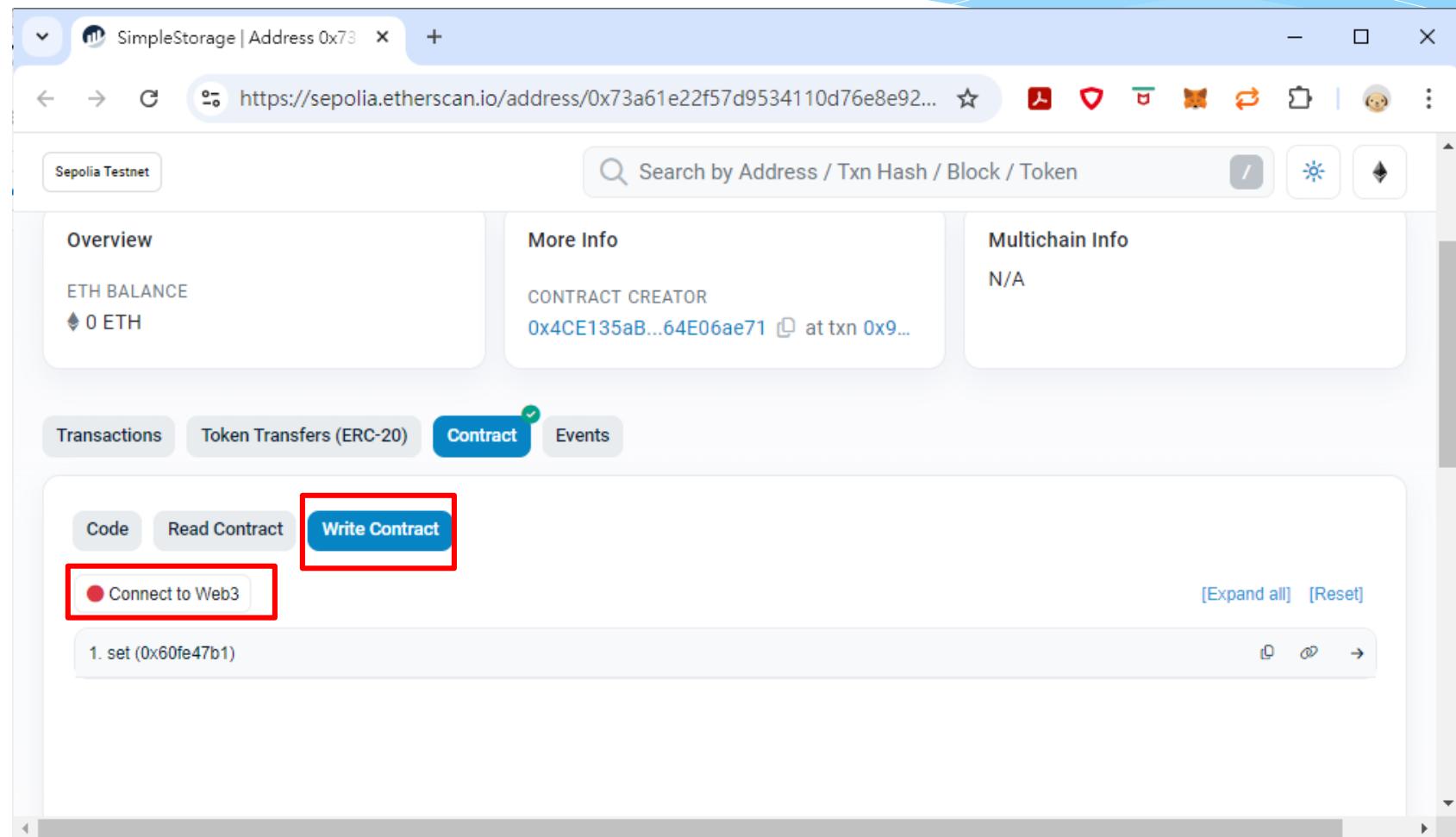
- Overview:** ETH BALANCE: 0 ETH.
- More Info:** CONTRACT CREATOR: 0x4CE135aB...64E06ae71 at txn 0x90a9c066dd...
- Multichain Info:** N/A

Below the tabs, there is a section for verified source code:

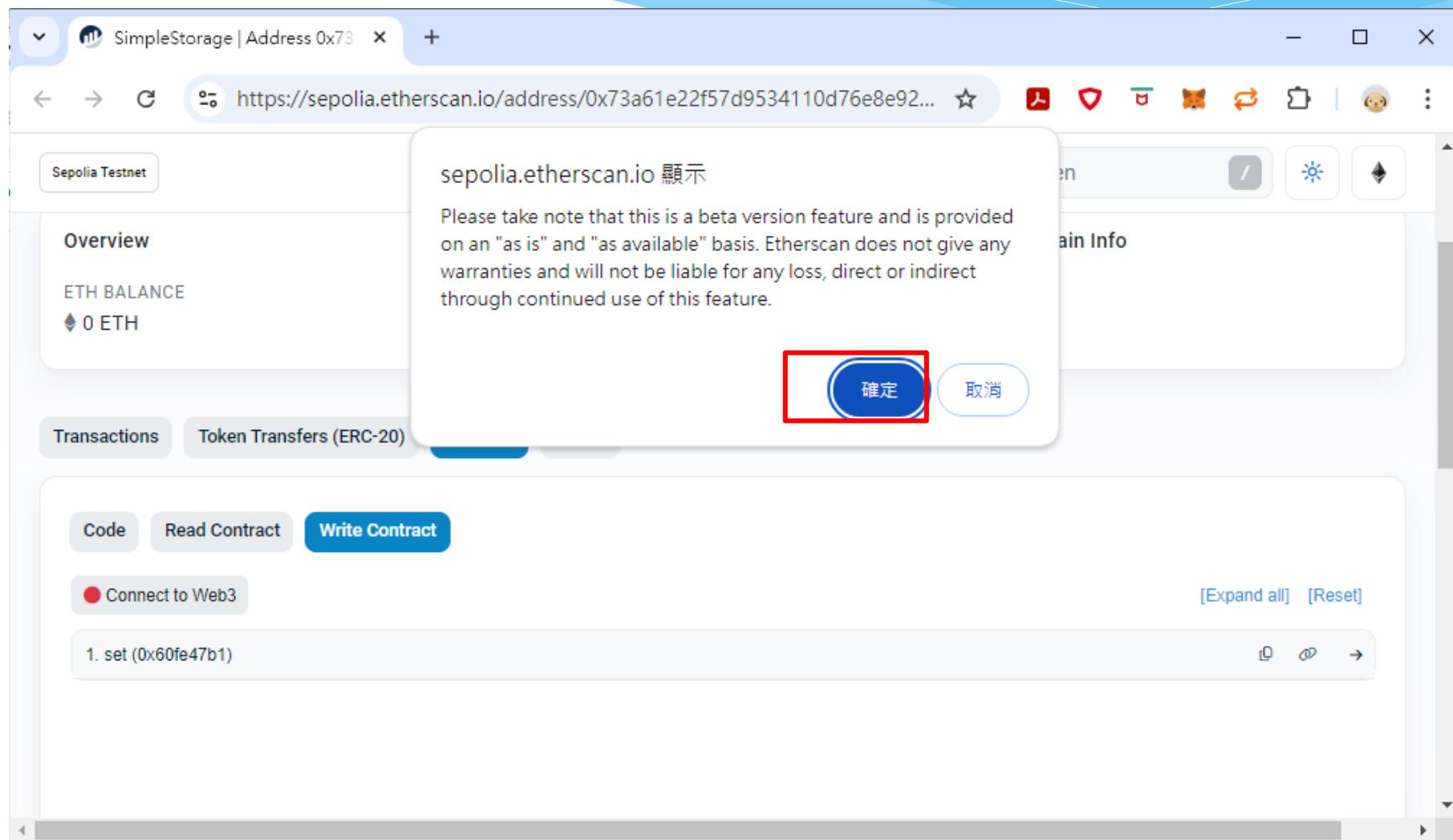
- Contract Source Code Verified (Exact Match)**
- Contract Name: SimpleStorage
- Compiler Version: v0.8.26+commit.8a97fa7a
- Optimization Enabled: No with 200 runs
- Other Settings: default evmVersion, GNU GPLv3 license

At the bottom, there are links for 'Contract Source Code (Solidity)' and various interface options like 'IDE', 'Outline', and 'More Options'.

Write Contract-Connected Web3



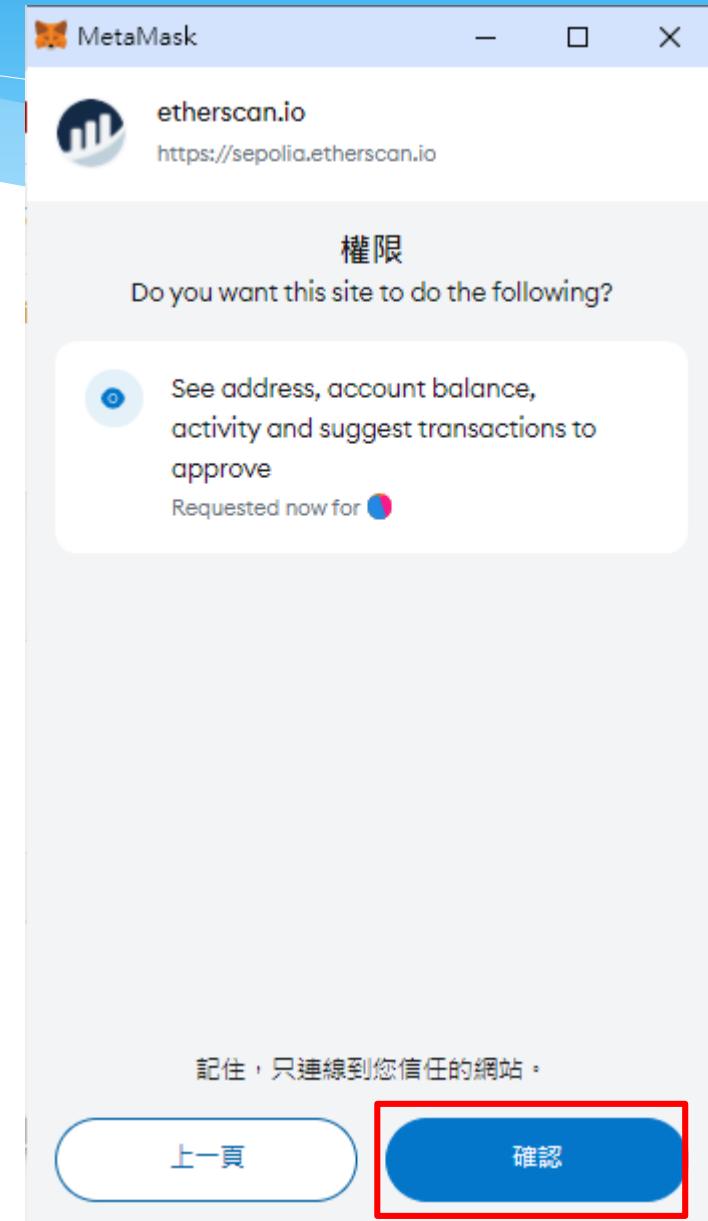
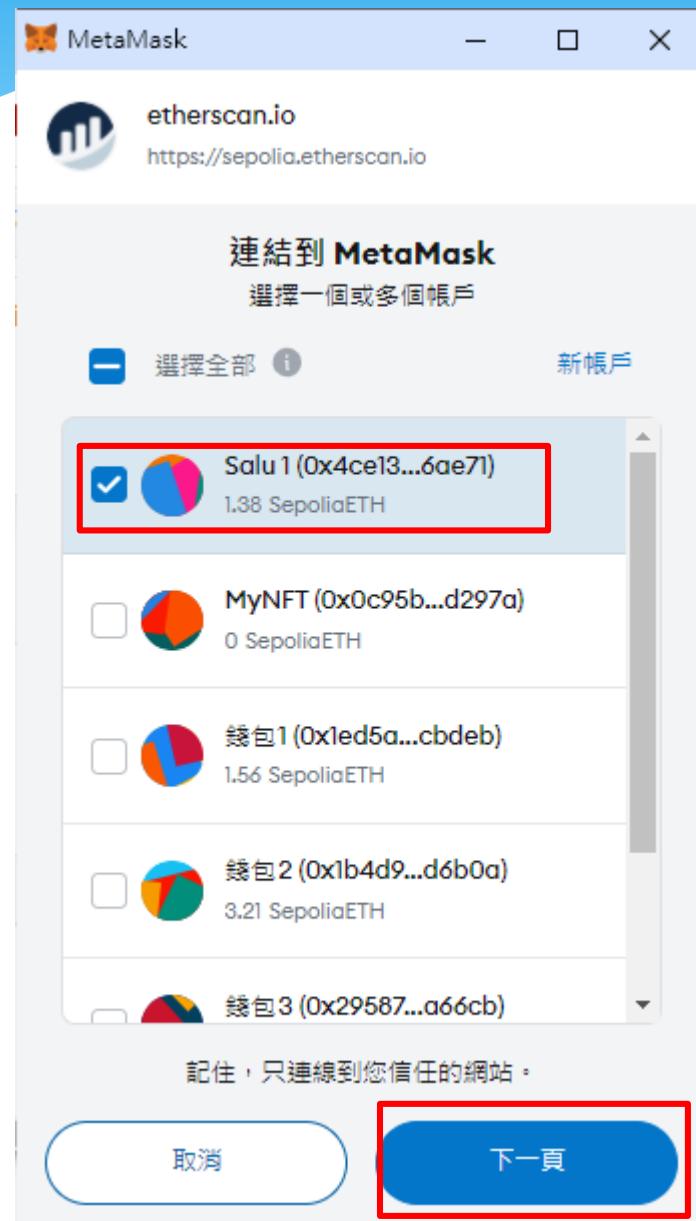
Connected-Web3



Connected-Web3

A screenshot of a web browser window displaying the Etherscan interface for the Sepolia Testnet. The address bar shows the URL: <https://sepolia.etherscan.io/address/0x73a61e22f57d9534110d76e8e92657750345bfc5#code>. A modal dialog titled "Connect a Wallet" is centered over the page, listing three options: MetaMask (Popular), WalletConnect, and Coinbase Wallet. The "MetaMask" option is highlighted with a red border. The background of the browser shows the Etherscan dashboard with sections for Overview, ETH BALANCE (0 ETH), Transactions, and a Write Contract section.

Connected-Web3



Write contract-set function

The screenshot shows the Etherscan interface for the Sepolia Testnet. A transaction has been submitted to the address 0x73a61e22f57d9534110d76e8e92... with the function signature 0x73A61...5bfC5. The transaction details are as follows:

- Estimated changes:** No changes predicted for your wallet.
- Estimated fee:** 0.01143547 SepoliaETH (Market -60 sec), Max fee: 0.01540316 SepoliaETH

The MetaMask confirmation dialog is open, showing the same transaction details. The "確認" (Confirm) button is highlighted with a red box and a red arrow from the transaction hash in the Etherscan interface.

Connected - Web3 [0x4CE1...ae71]

1. set (0x60fe47b1)

x (uint256) +

1688

Write

Estimated changes

No changes predicted for your wallet

Estimated fee

0.01143547 SepoliaETH

Market -60 sec Max fee: 0.01540316 SepoliaETH

拒絕 確認

View your transaction

The screenshot shows a web browser window displaying the Etherscan interface for the Sepolia Testnet. The address being viewed is 0x73a61e22f57d9534110d76e8e92... . The 'Contract' tab is selected. Below it, there are tabs for 'Transactions', 'Token Transfers (ERC-20)', 'Contract' (which has a green checkmark icon), and 'Events'. Under the 'Contract' tab, there are three buttons: 'Code', 'Read Contract', and 'Write Contract', with 'Write Contract' being the active button. A message indicates 'Connected - Web3 [0x4CE1...ae71]'. The main area shows a list of transactions:

- 1. set (0x60fe47b1)
x (uint256) +
1688

At the bottom, there are two buttons: 'Write' and 'View your transaction'. The 'View your transaction' button is highlighted with a red border.

View your transaction

The screenshot shows a web browser window displaying the Sepolia Testnet version of Etherscan. The URL in the address bar is <https://sepolia.etherscan.io/tx/0xc73a292e29fa3fce646e85da2c5d9d03c05f576348cbd04a112e4b1611073019>. The page title is "Sepolia Transaction Hash (Txh)".

The main content area is titled "Transaction Details". It includes tabs for "Overview" (which is selected) and "State". A note at the top states "[This is a Sepolia Testnet transaction only]".

The transaction details listed are:

- Transaction Hash: [0xc73a292e29fa3fce646e85da2c5d9d03c05f576348cbd04a112e4b1611073019](#)
- Status: Success
- Block: [6807962](#) 6 Block Confirmations
- Timestamp: [1 min ago \(Oct-03-2024 04:37:12 PM UTC\)](#)
- Transaction Action: Call [Set](#) Function by [0x4CE135aB...64E06ae71](#) on [0x73A61e22...50345bfC5](#)
- From: [0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71](#)
- To: [0x73A61e22f57D9534110D76E8E92657750345bfC5](#) ✓

View your transaction-More Detail

Sepolia Transaction Hash (Txh) x

sepolia.etherscan.io/tx/0xc73a292e29fa3fce646e85da2c5d9d03c05f576348cbd04a112e4b1611073019

Sepolia Testnet Search by Address / Txn Hash / Block / Token

Transaction Hash: **0xc73a292e29fa3fce646e85da2c5d9d03c05f576348cbd04a112e4b1611073019**

Status: Success

Block: 6807962 45441 Block Confirmations

Timestamp: 7 days ago (Oct-03-2024 04:37:12 PM UTC)

Transaction Action: Call Function by 0x4CE135aB...64E06ae71 on 0x73A61e22...50345bfC5

From: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71

To: 0x73A61e22f57D9534110D76E8E92657750345bfC5

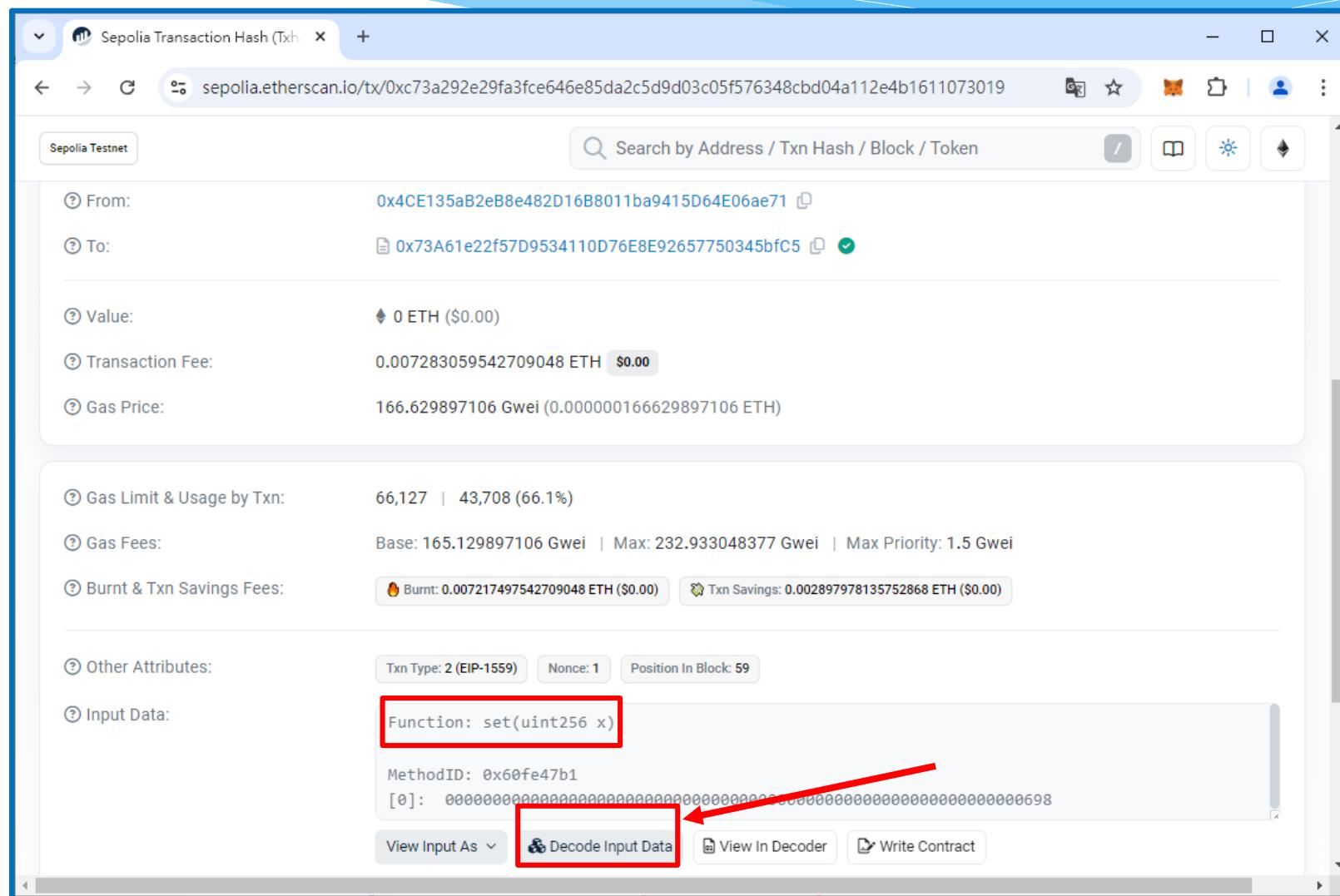
Value: 0 ETH (\$0.00)

Transaction Fee: 0.007283059542709048 ETH \$0.00

Gas Price: 166.629897106 Gwei (0.000000166629897106 ETH)

More Details: + Click to show more

View your transaction-Input Data



View your transaction-Decode Input Data

The screenshot shows a web browser displaying the Sepolia Testnet transaction details at <https://sepolia.etherscan.io/tx/0xc73a292e29fa3fce646e85da2c5d9d03c05f576348cbd04a112e4b1611073019>. The transaction hash is 0xc73a292e29fa3fce646e85da2c5d9d03c05f576348cbd04a112e4b1611073019.

Transaction Details:

- From: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71
- To: 0x73A61e22f57D9534110D76E8E92657750345bfC5
- Value: 0 ETH (\$0.00)
- Transaction Fee: 0.007283059542709048 ETH (\$0.00)
- Gas Price: 166.629897106 Gwei (0.000000166629897106 ETH)

Gas Limit & Usage by Txn: 66,127 | 43,708 (66.1%)

Gas Fees: Base: 165.129897106 Gwei | Max: 232.933048377 Gwei | Max Priority: 1.5 Gwei

Burnt & Txn Savings Fees: Burnt: 0.007217497542709048 ETH (\$0.00) | Txn Savings: 0.002897978135752868 ETH (\$0.00)

Other Attributes: Txn Type: 2 (EIP-1559) | Nonce: 1 | Position In Block: 59

Input Data:

#	Name	Type	Data
0	x	uint256	1688

[Switch Back](#) | [View In Decoder](#)

View your transaction-Input Data

The screenshot shows a web browser displaying the Sepolia etherscan.io transaction details for the hash `0xc73a292e29fa3fce646e85da2c5d9d03c05f576348cbd04a112e4b1611073019`. The transaction details include:

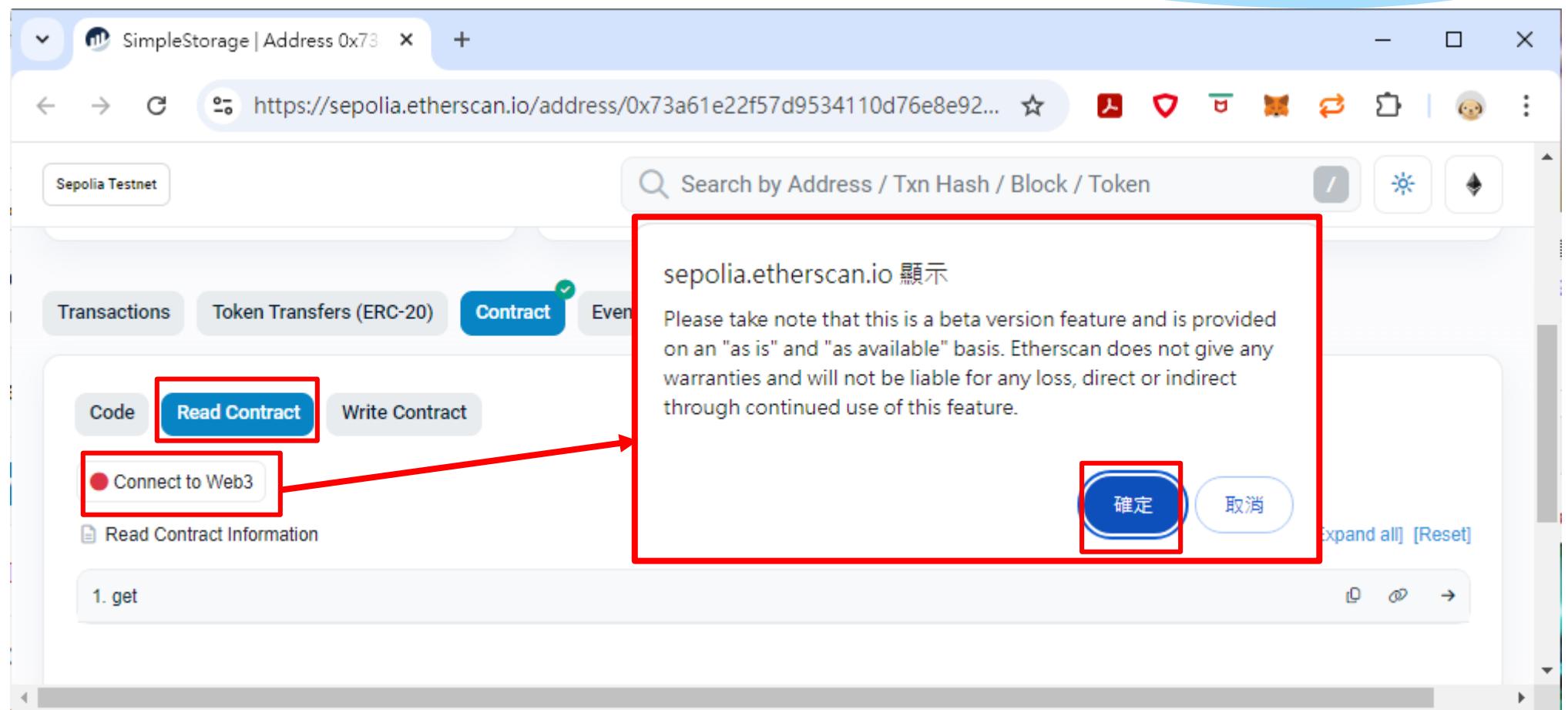
- To: `0x73A61e22f57D9534110D76E8E92657750345bfC5`
- Value: 0 ETH (\$0.00)
- Transaction Fee: 0.007283059542709048 ETH (\$0.00)
- Gas Price: 166.629897106 Gwei (0.00000166629897106 ETH)
- Gas Limit & Usage by Txn: 66,127 | 43,708 (66.1%)
- Gas Fees: Base: 165.129897106 Gwei | Max: 232.933048377 Gwei | Max Priority: 1.5 Gwei
- Burnt & Txn Savings Fees: Burnt: 0.007217497542709048 ETH (\$0.00) | Txn Savings: 0.002897978135752868 ETH (\$0.00)
- Other Attributes: Txn Type: 2 (EIP-1559) | Nonce: 1 | Position In Block: 59
- Input Data: Function: `set(uint256 x)` (highlighted with a red box)

A red arrow points to the `View In Decoder` button at the bottom of the input data section.

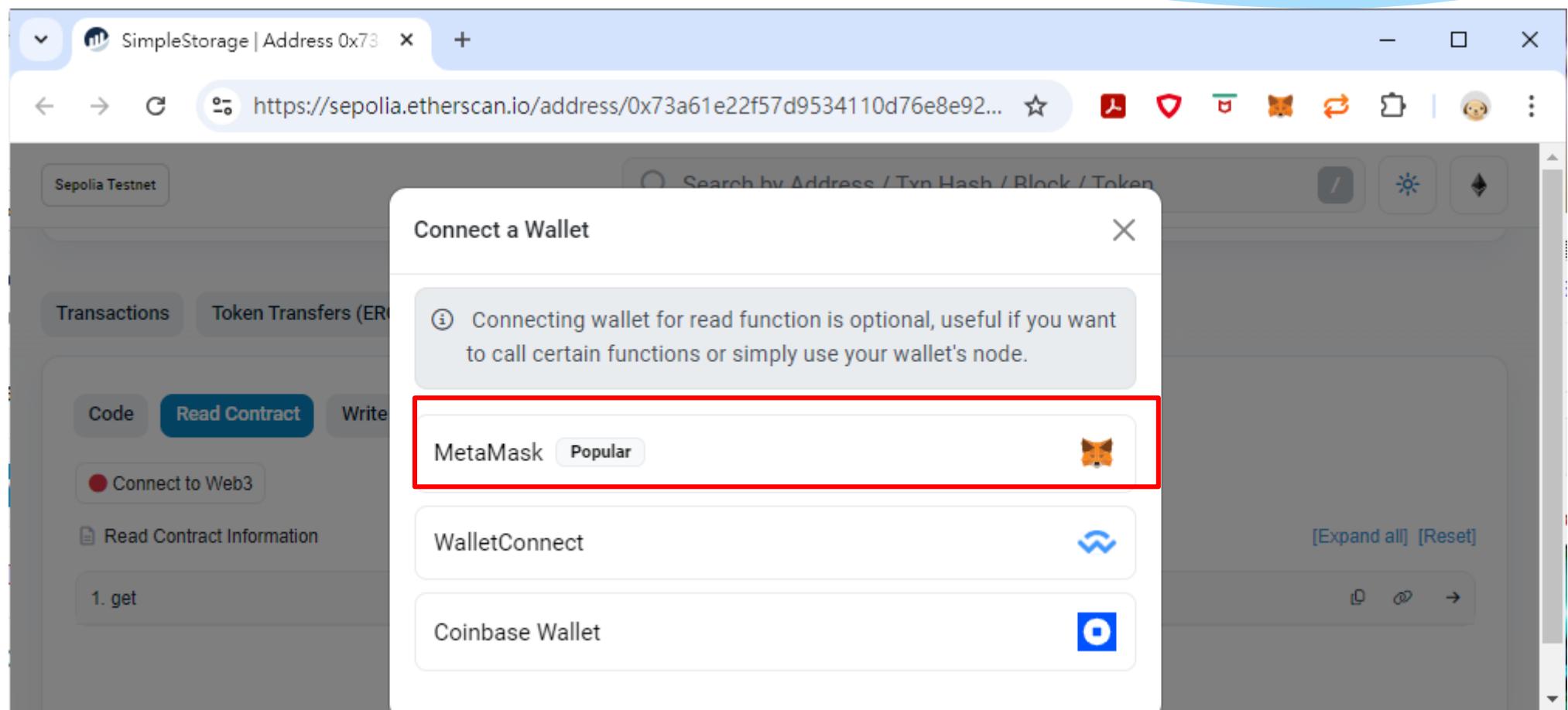
View your transaction-Input Data Decoder

The screenshot shows a web browser window titled "Input Data Decoder | Etherscan". The URL in the address bar is <https://sepolia.etherscan.io/inputdatadecoder?tx=0xc73a292e29fa3fce646e85da2c5d9d03c05f576348cbd04a112e4b1611073019>. The left sidebar has a "Sepolia Testnet" button and a "CSV Export" button, followed by a list of tools: Account Balance Checker, Token Supply Checker, Token Standard Checker, **Input Data Decoder** (which is selected), Similar Contract Search, Vyper Online Compiler, Bytecode to Opcode, Broadcast Transaction, Unit Converter, Base64 Converter, Block & Date Converter, and UTF-8 Converter. The main area has a "Choose option:" section with radio buttons for "from Transaction" (selected), "from Address", "from ABI", and "without ABI". Below it is a "Txn Hash" input field containing the value `0xc73a292e29fa3fce646e85da2c5d9d03c05f576348cbd04a112e4b1611073019`, which is highlighted with a red box. Below the input are "Decode" and "Reset" buttons. Under "Decoded Results:", there is a "Function" button with a "set" dropdown, which is also highlighted with a red box. Below it is a "uint256" input field with the value `x` and a "Wei" dropdown. A red arrow points from the "Function" button to the "1688" value in the "uint256" field, which is also highlighted with a red box.

Connect to Web3



Connect a Wallet-MetaMask



Connected Web3 using account

The image shows a web browser with two tabs open, connected by a large red arrow pointing from the left tab to the right tab.

Left Tab (Etherscan Interface):

- Address: SimpleStorage | Address 0x73...
- Network: Sepolia Testnet
- Buttons: Transactions, Token Transfers (ERC-20), Contract (selected), Write Contract
- Message: Connected - Web3 [0x4CE1...ae71]

Right Tab (MetaMask Wallet Interface):

- Account: Salu1 (Address: 0x4CE13...6ae71)
- Balance: 1.3711 SepoliaETH
- Buttons: Buy & Sell, 發送 (Send), Swap, Bridge, Portfolio
- Tab: 交易紀錄 (Transactions) (selected)
- Transactions:

 - Oct 4, 2024: Set (部署合約), 已確認 (Confirmed), -0 SepoliaETH, -0 SepoliaETH
 - Oct 3, 2024: 部署合約 (Deployed Contract), 已確認 (Confirmed), -0 SepoliaETH, -0 SepoliaETH

Read Contract-get function

The screenshot shows the Etherscan interface for the SimpleStorage contract at address 0x73a61e22f57d9534110d76e8e92... on the Sepolia Testnet. The 'Contract' tab is selected. In the 'Read Contract' section, the '1. get' button is highlighted with a red box and a purple arrow pointing to it with the text 'click'. The button has a value of '1688 uint256'.

SimpleStorage | Address 0x73a61e22f57d9534110d76e8e92...

https://sepolia.etherscan.io/address/0x73a61e22f57d9534110d76e8e92...#readContract

Sepolia Testnet

Search by Address / Txn Hash / Block / Token

Transactions Token Transfers (ERC-20) Contract Events

Code Read Contract Write Contract

Connected - Web3 [0x4CE1...ae71]

Read Contract Information

1. get
1688 uint256

[Expand all] [Reset]



set Contract using Remix

The screenshot shows the Ethereum Remix IDE interface with three main panels:

- Left Panel (Deploy & Run Transactions):** Shows deployment settings for a "SimpleStorage - SimpleStorage.sol" contract. It includes fields for gas limit (3000000), value (0 Wei), and a "Deploy" button. A red box highlights the "set" button and the value input field (labeled 2). A red circle labeled 3 points to the "set" button.
- Middle Panel (Code Editor):** Displays the Solidity code for the SimpleStorage contract:

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

A red box highlights the "set" function (labeled 2).
- Right Panel (MetaMask):** Shows the MetaMask extension interface for the Sepolia test network. It displays a transaction record for "0x52acE...52209 : SET". A red circle labeled 4 points to the "Estimated fee" section, which shows "0.00765218 SepoliaETH". Below it, a confirmation dialog box has "確認" (Confirm) highlighted with a red border.

View transaction on etherscan

The screenshot shows a Solidity development environment with the following details:

- Deploy & Run Transactions:** A sidebar with "Custom" value set to 3,000,000 Wei.
- Contract:** SimpleStorage - SimpleStorage.sol
- EVM Version:** canary
- Buttons:** Deploy, Publish to IPFS, At Address, Load contract from Address.
- Transactions recorded:** 2
- Pinned Contracts:** (network: 11155111) - No pinned contracts found.
- Deployed/Unpinned Contracts:** SIMPLESTORAGE AT 0x52A...
 - Balance:** 0 ETH
 - Actions:** set, set - transact (not payable), get
 - get Result:** 0; uint256: 8888
- Low level interactions:** CALLDATA
- Logs:**
 - txIndex: 35 [block:6810514] from: 0x4ce...6ae71 to: SimpleStorage.(constructor) value: 0 wei data: 0x608...a0033 logs: 0 hash: 0xc7d...04807
call to simplestorage.get
 - 1 txIndex: 48 [call] from: 0x4CE135aB2eB8e482D1688011ba9415D64E06ae71 to: SimpleStorage.get() data: 0x6d4...ce63c
transact to Simplestorage.set pending ...
 - view on etherscan** (button highlighted with a red box)
 - txIndex: 48 [block:6810719] from: 0x4ce...6ae71 to: SimpleStorage.set(uint256) 0x52a...52209 value: 0 wei data: 0x60f...0270f logs: 0 hash: 0x440...4c7a5

View transaction on etherscan

The screenshot shows a web browser window displaying the Etherscan Transaction Details page for a Sepolia Testnet transaction. The URL in the address bar is <https://sepolia.etherscan.io/tx/0x10cf8b33cf95efa31db384947fcd07b642bb2059e91b7882874dd3de9dac5be6>. The transaction hash is highlighted with a red box.

The page header includes the Etherscan logo and navigation links for Sepolia Testnet, Home, Blockchain, Tokens, NFTs, and More. A search bar at the top right allows searching by Address / Txn Hash / Block / Token.

The main content area is titled "Transaction Details" and shows the following information:

- [This is a Sepolia Testnet transaction only]
- Transaction Hash: [0x10cf8b33cf95efa31db384947fcd07b642bb2059e91b7882874dd3de9dac5be6](#)
- Status: Success (highlighted with a red box)
- Block: [6810719](#) | 39 Block Confirmations
- Timestamp: [8 mins ago \(Oct-04-2024 03:05:00 AM UTC\)](#)
- Transaction Action: [Call](#) [Set](#) Function by [0x4CE135aB...64E06ae71](#) on [0x52acEad3...523A52209](#)
- From: [0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71](#)
- To: [0x52acEad3AB674f6567dC4c9969F06E7523A52209](#) ✓

get Contract using Remix

The screenshot shows the Remix IDE interface with two main panes. The left pane is titled "DEPLOY & RUN TRANSACTIONS" and contains the following details:

- Gas limit: 3000000
- Value: 0 Wei
- Contract: SimpleStorage - SimpleStorage.sol
- EVM version: cancan
- Deploy button (highlighted with a red box)
- Publish to IPFS checkbox
- At Address tab (selected)
- Transactions recorded: 2
- Pinned Contracts (network: 11155111): No pinned contracts found for selected workspace & network
- Deployed/Unpinned Contracts: SIMPLESTORAGE AT 0X52A... (Balance: 0 ETH). It shows a "set" button (highlighted with a red box) and a "get" button (highlighted with a red box and circled with a red circle labeled 1).
- Low level interactions: CALLDATA and Transact buttons.

The right pane displays the Solidity code for the SimpleStorage contract:

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

Below the code, the transaction history is shown:

- [block:6810514 txIndex:35] from: 0x4ce...6ae71 to: SimpleStorage.(constructor) value: 0 wei data: 0x608...a0033 logs: 0 hash: 0xc7d...04807
call to Simplestorage.get
- [block:6810719 txIndex:48] from: 0x4ce...6ae71 to: SimpleStorage.set(uint256) 0x52a...52209 value: 0 wei data: 0x60f...0270f logs: 0 hash: 0x440...4c7a5
call to SimpleStorage.set
- [block:6810719 txIndex:48] from: 0x4ce...6ae71 to: SimpleStorage.get() data: 0x6d4...ce63c
call [call] from: 0x4CE135aB2eB8e482D1688011ba9415D64E06ae71 to: SimpleStorage.get() data: 0x6d4...ce63c

Red boxes highlight the "get" button in the Deploy & Run pane, the "get" button in the transaction history, and the transaction details in the history pane. Red circles with numbers 1, 2, and 3 mark specific points of interest: 1 points to the "get" button in the Deploy & Run pane; 2 points to the transaction details in the history pane; 3 points to the "get" button in the transaction details.



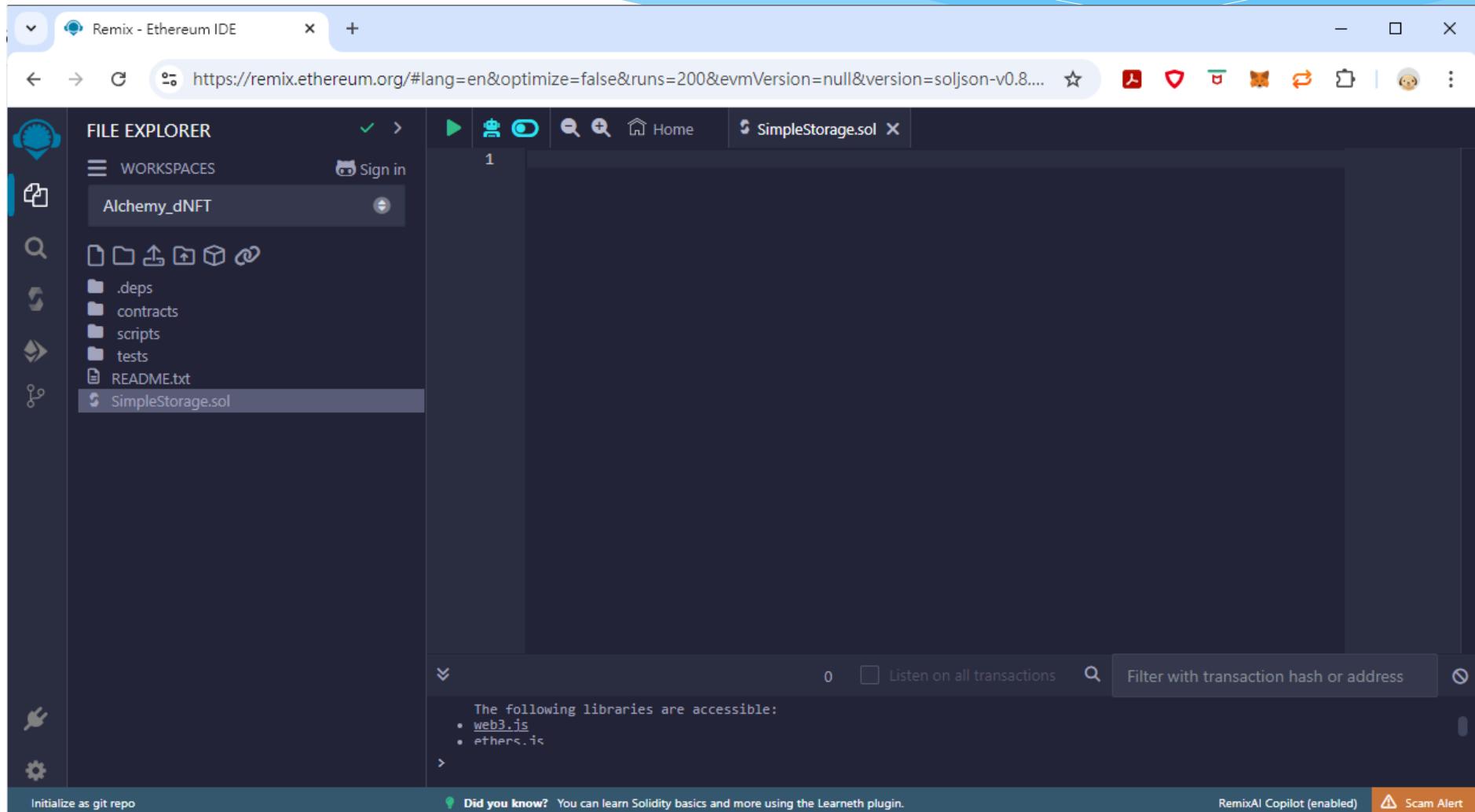


2. Deploy Smart Contract to Holesky Testnet

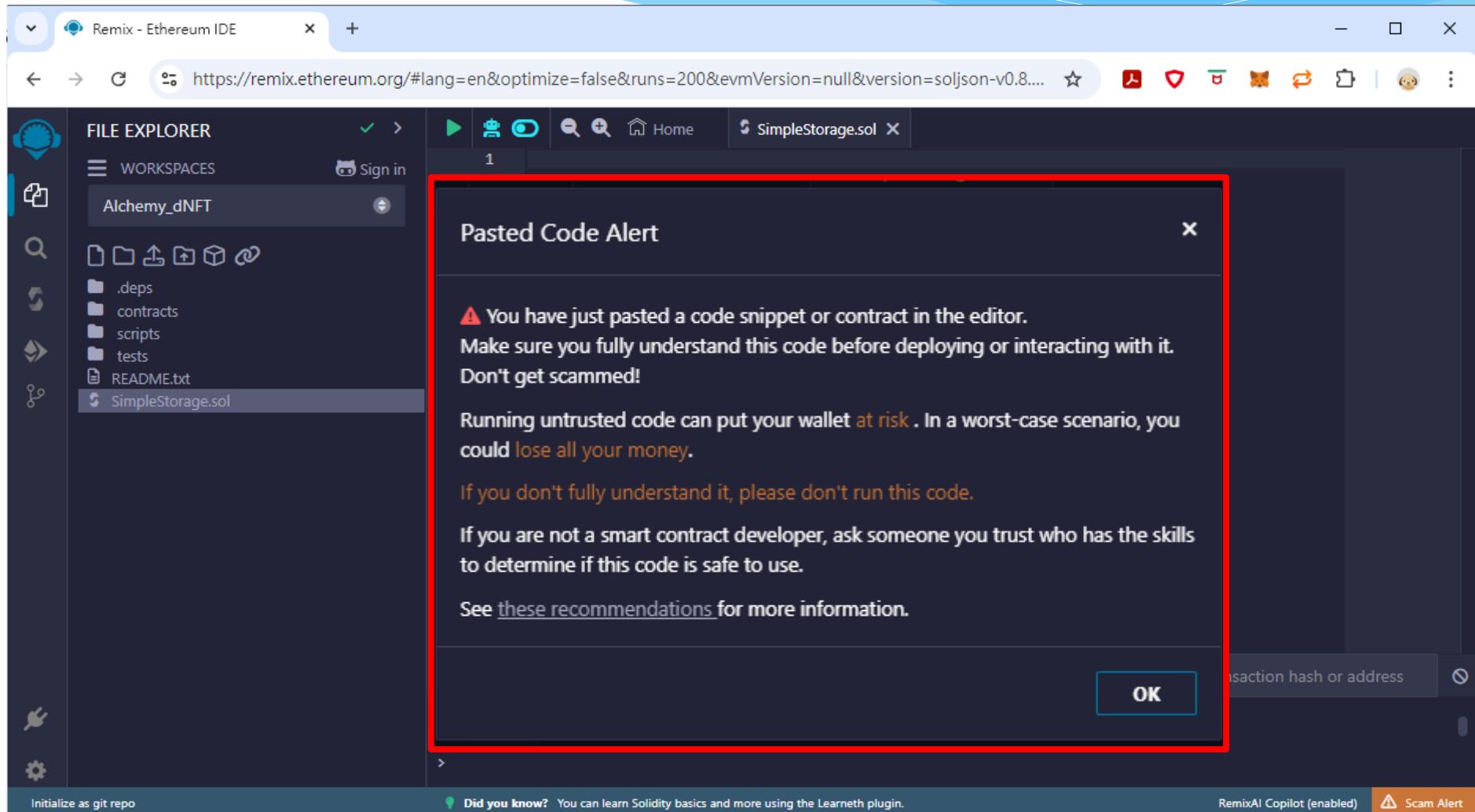
Create new file

The screenshot shows the Remix Ethereum IDE interface. On the left, the **FILE EXPLORER** panel displays a workspace named **Alchemy_dNFT**. A red box highlights the **New File** icon (a document with a plus sign) in the sidebar of the file explorer. The central area features the **REMIX** logo and the tagline *The Native IDE for Web3 Development.* Below it, there are several buttons: **Start Coding**, **ZK Semaphore**, **ERC20**, **Uniswap V4 Hooks**, **NFT / ERC721**, and **MultiSig**. To the right, the **Featured** section highlights the **v0.54.0 RELEASE HIGHLIGHTS**, which include: Login to GitHub from 'File Explorer', Added Cookbook workspace templates, and Added 'sendRawTransaction' API to remix-simulator. There is also a **What's New** section with a cartoon character and a **Read More** button. At the bottom, there are sections for **Recent Workspaces**, **Accessible Libraries** (listing `web3.js` and `ethers.js`), and **Featured Plugins**. The footer includes links for **Initialize as git repo**, **Did you know?**, **RemixAI Copilot (enabled)**, and **Scam Alert**.

SimpleStorage.sol



SimpleStorage.sol



SimpleStorage.sol

The screenshot shows the Ethereum IDE interface with the file `SimpleStorage.sol` open. The code is highlighted with a red box.

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

The code defines a Solidity contract named `SimpleStorage`. It contains two functions: `set` and `get`. The `set` function takes a `uint` parameter `x` and stores it in the `storedData` variable. The `get` function returns the current value of `storedData`.

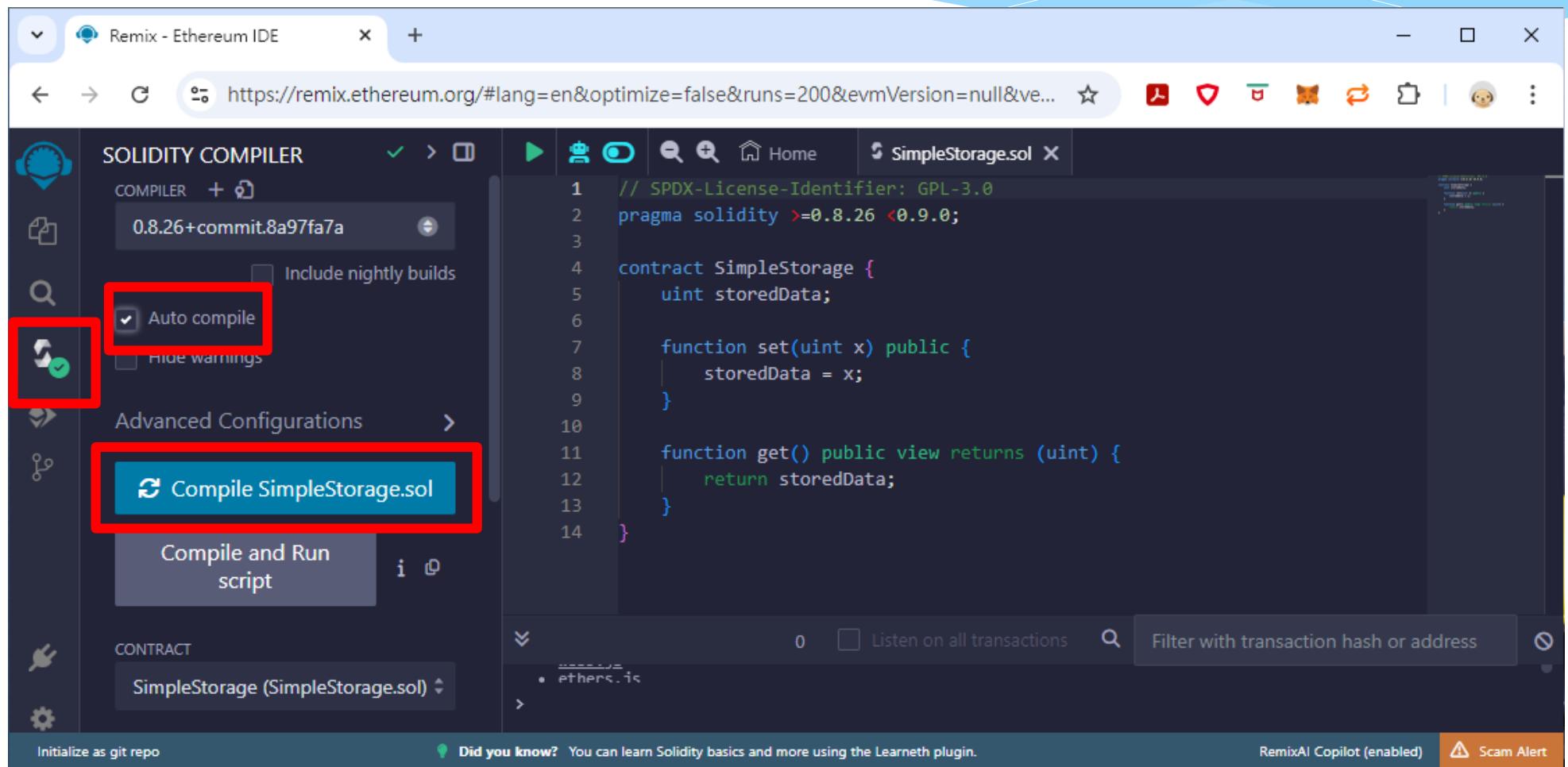
Initialize as git repo

Did you know? You can learn Solidity basics and more using the Learneth plugin.

RemixAI Copilot (enabled)

Scam Alert

Compile Smart Contract



Deploy and Run Transaction

The screenshot shows the Remix Ethereum IDE interface and a separate MetaMask window.

Remix Ethereum IDE:

- Deploy & Run Transactions:** The "ENVIRONMENT" dropdown is set to "Injected Provider - MetaMask" (highlighted with a red box).
 - ACCOUNT:** A dropdown menu is open, showing "Custom (17000) network".
 - GAS LIMIT:** Set to "Estimated Gas".
 - VALUE:** Set to 0 Wei.
- Contract:** The contract selected is "SimpleStorage - SimpleStorage.sol".
 - evm version:** Set to "cancun".
 - Deploy:** A button at the bottom of the sidebar (highlighted with a red box).

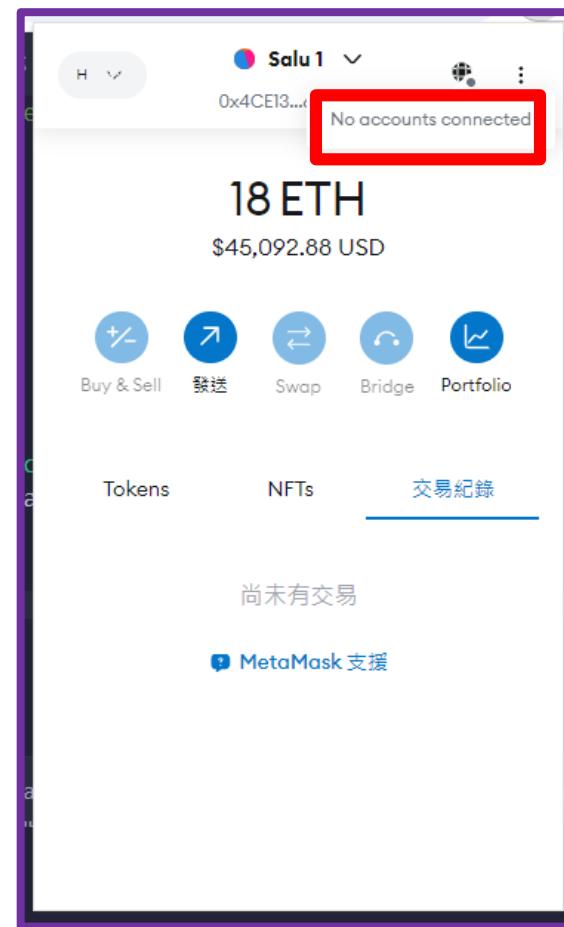
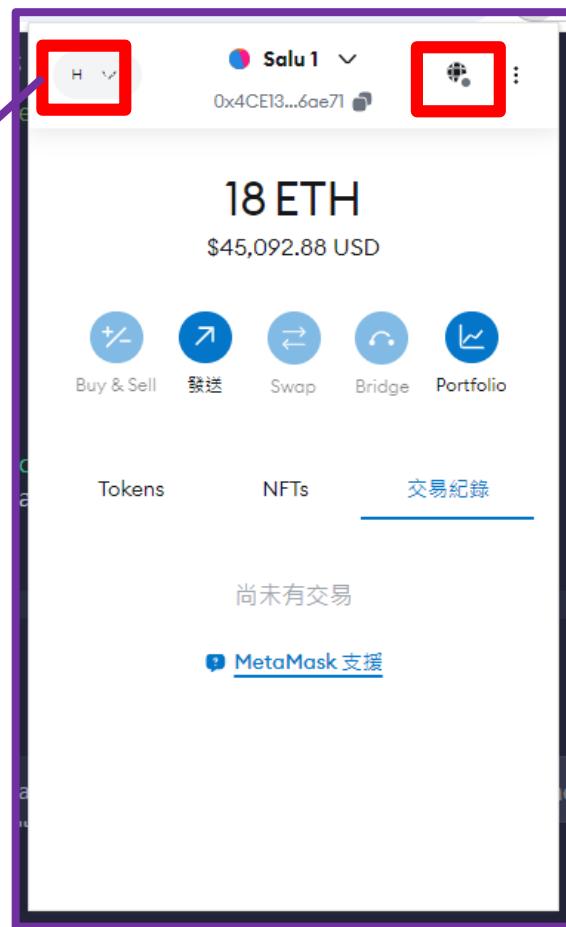
MetaMask:

- A splash screen with an orange dog icon and the text "歡迎回來! 去中心化網路世界等待著您".
- A password input field with placeholder "密碼" and a redacted password.
- A blue "解鎖" (Unlock) button.
- A "Forgot password?" link.
- A "需要幫助? 聯繫 MetaMask 支援" link.

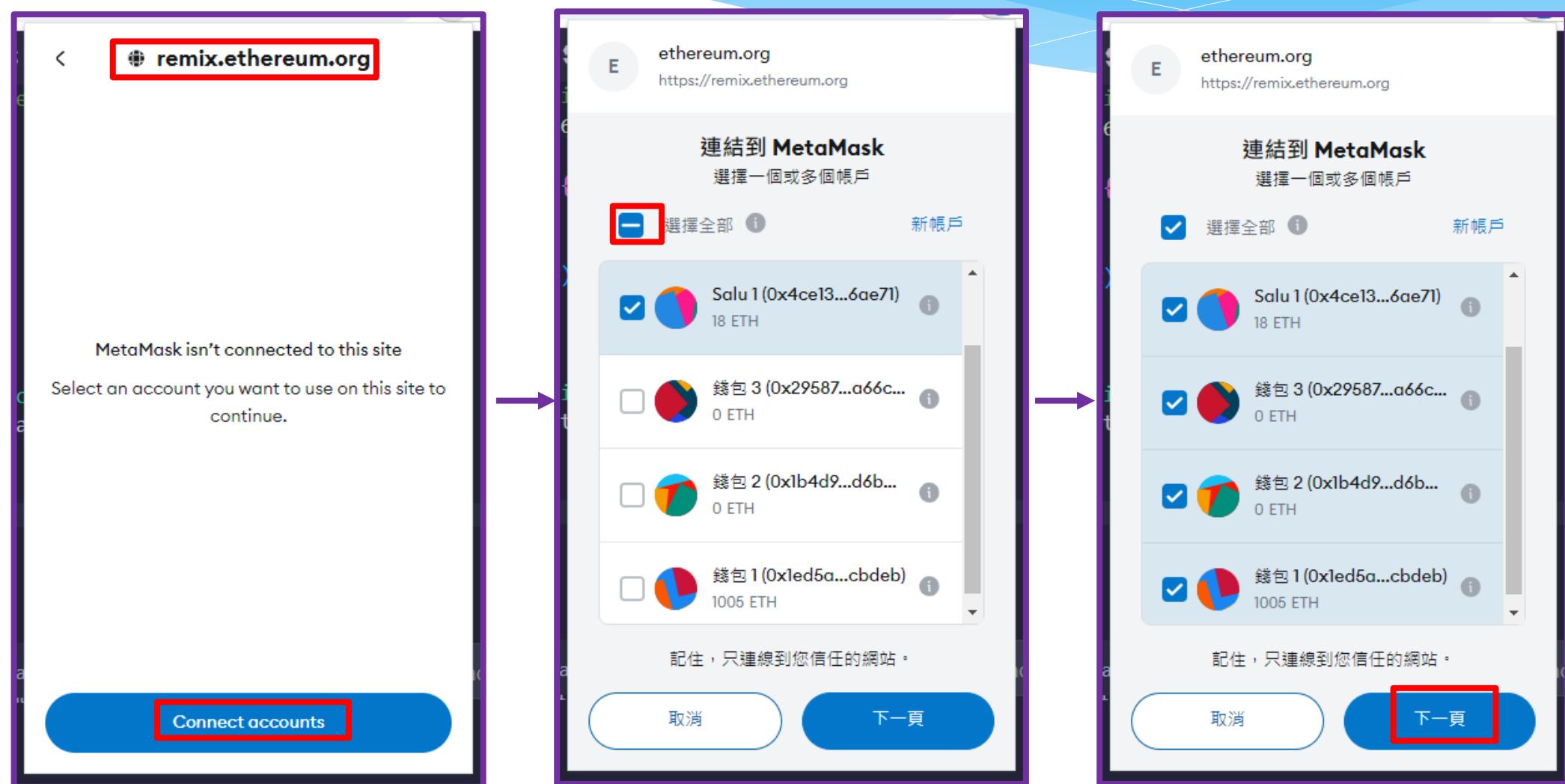
At the bottom of the Remix interface, there is a note: "Did you know? To learn new contract patterns and prototype, you can activate and try the cookbook plugin!"

Connect to Holesky testnet

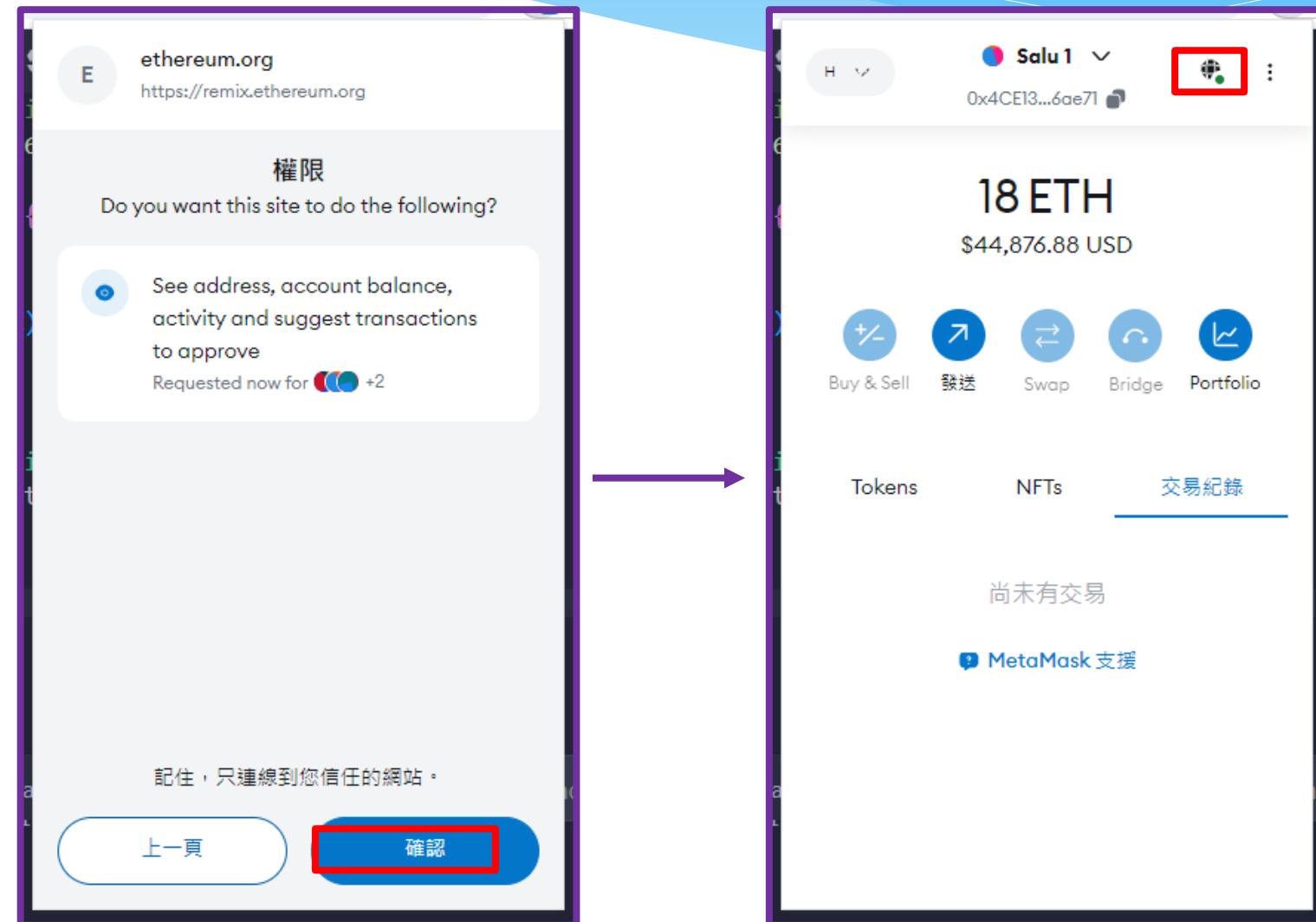
Holesky testnet



Connect to remix.ethereum.org



Connect to remix.ethereum.org



Deploy and Run Transaction

The screenshot shows the Ethereum IDE interface with two main windows.

Left Window (Remix - Ethereum IDE):

- DEPLOY & RUN TRANSACTIONS:**
 - ENVIRONMENT:** Injected Provider - MetaMask (highlighted with a red box).
 - ACCOUNT:** 0x4CE13...6ae71 (18 ether) (highlighted with a red box).
 - GAS LIMIT:** Estimated Gas (radio button selected).
 - VALUE:** 0 Wei.
 - CONTRACT:** SimpleStorage - SimpleStorage.sol (highlighted with a red box).
 - Deploy:** A large orange button at the bottom left.
- Code Editor:** Solidity code for SimpleStorage contract.

Right Window (MetaMask Wallet):

- Address:** 0x4CE13...6ae71 (highlighted with a red box).
- Balances:** 18 ETH (\$44,876.88 USD).
- Actions:** Buy & Sell, 發送 (Send), Swap, Bridge, Portfolio.
- Tab:** 交易紀錄 (Transactions) is selected.
- Text:** 尚未有交易 (No transactions yet).
- Support:** MetaMask 支援 (MetaMask Support).

Deploy and Run Transaction

The screenshot shows the Ethereum IDE interface with the following details:

- DEPLOY & RUN TRANSACTIONS** sidebar:
 - ENVIRONMENT: Injected Provider - MetaMask
 - ACCOUNT: 0x4CE...6ae71 (18 ether)
 - GAS LIMIT: Estimated Gas (radio button selected), Custom value: 3000000
 - VALUE: 0 Wei
- Contract Code:**

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```
- Deployment Panel (right side):**
 - Contract Name: Holesky
 - Account: Salu1
 - 部署合約 (Deploy Contract) button
 - Estimated fee: \$0.24 (0.0000964 ETH)
 - Market -60 sec (Market price)
 - 總量 (Total amount): \$0.24 (0.0000964 ETH)
 - Amount + gas fee: 0.0000964 ETH
 - Max amount: 0.0000964 ETH
 - 自訂 NONCE (Custom NONCE) input field: 0
 - Alert buttons: 拒絕 (Reject) and 確認 (Confirm) (both highlighted with red boxes)

Deploy and Run Transaction

The image shows two overlapping windows. On the left is the Ethereum IDE (Remix) interface. The central area displays the following Solidity code:

```
// SPDX-License-Identifier: pragma solidity >=0.8.26 <0.9.0; contract SimpleStorage { uint storedData; function set(uint x) public { storedData = x; } function get() public view returns(uint) { return storedData; } }
```

The right window is a MetaMask wallet interface. It shows the user's balance: 17.9999 ETH (\$44,929.20 USD). Below the balance are five buttons: Buy & Sell, Swap, Bridge, and Portfolio. At the bottom, there are tabs for Tokens, NFTs, and 交易紀錄 (Transactions). A red box highlights a transaction in the history list, which is also highlighted by a red arrow pointing from the text "click" below it. The transaction details are as follows:

Oct 7, 2024
部署合約
已確認
-0 ETH
-\$0.00 USD
MetaMask 支援

Below the transaction list, the text "click" is written in red.

IDE Left Panel Labels:
DEPLOY & RUN TRANSACTIONS
ENVIRONMENT: Injected Provider - MetaMask
ACCOUNT: 0x4CE...6ae71 (18 ether)
GAS LIMIT: Estimated Gas (radio button selected)
Custom: 3000000
VALUE: 0 Wei
CONTRACT: SimpleStorage - SimpleStorage.sol
evm version: canary
Deploy

IDE Bottom Bar Labels:
Initialize as git repo
Did you know? To learn new contract patterns and prototype, you can activate and try the cookbooks

Deploy and Run Transaction

The screenshot shows the Ethereum IDE (Remix) interface. On the left, the sidebar includes icons for file operations, environment, account selection (set to 'Injected Provider - MetaMask' and 'Custom (17000) network'), gas limit (set to 'Estimated Gas' at 3000000), value (set to 0 Wei), and contract selection ('SimpleStorage - SimpleStorage.sol'). The main area displays Solidity code for a 'SimpleStorage' contract with two functions: 'set' and 'get'. To the right, a modal window titled '部署合約' (Deploy Contract) provides details about the transaction. A red box highlights the 'View on block explorer' button, which is located next to the status '已確認' (Confirmed). The transaction details include the source account (0x4CE...6ae71), destination account (0x4CE13...6...), nonce (0), gas limit (123005), gas price (123005 GWEI), base fee (0.000000009 GWEI), priority fee (0.783743356 GWEI), total gas fee (0.0000096 ETH / \$0.24 USD), and max gas price (0.000000001 ETH / \$0.00 USD).

DEPLOY & RUN TRANSACTIONS

ENVIRONMENT
Injected Provider - MetaMask
Custom (17000) network

ACCOUNT
0x4CE...6ae71 (18 ether)

GAS LIMIT
Estimated Gas
Custom 3000000

VALUE
0 Wei

CONTRACT
SimpleStorage - SimpleStorage.sol
evm version: cancan
Deploy

Did you know? To learn new contract patterns and prototype, you can activate and try the cookbook pattern.

// SPDX-License-Identifier: GPL-3.0-or-later
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
 uint storedData;

 function set(uint x) public {
 storedData = x;
 }

 function get() public view returns (uint) {
 return storedData;
 }
}

部署合約

Status
已確認

View on block explorer
複製交易 ID

來源帳戶
0x4CE13...6...
建立新合約

目的帳戶

交易

Nonce	0
數量	-0 ETH
Gas 上限 (單位)	123005
Gas 用量 (單位)	123005
Base fee (GWEI)	0.000000009
Priority fee (GWEI)	0.783743356
Total gas fee	0.0000096 ETH \$0.24 USD
Max fee per gas	0.000000001 ETH \$0.00 USD

Did you know? To learn new contract patterns and prototype, you can activate and try the cookbook pattern.

View Transaction on Holesky

The screenshot shows a web browser window displaying the Etherscan interface for a Holesky Testnet transaction. The URL in the address bar is holesky.etherscan.io/tx/0xd46082347b2c7ac9b2ee82deddfab1951f0128eb16a2ecfd14af780e14a5a037. The page title is "Holesky Transaction Hash (Txh)".

The transaction details are as follows:

- Transaction Hash: `0xd46082347b2c7ac9b2ee82deddfab1951f0128eb16a2ecfd14af780e14a5a037`
- Status: Success
- Block: 2483246 (39 Block Confirmations)
- Timestamp: 8 mins ago (Oct-07-2024 04:01:24 AM UTC)
- Transaction Action: Call `0x60806040` Method by `0x4CE135aB...64E06ae71`
- From: `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71`
- To: `[0x73a61e22f57d9534110d76e8e92657750345bfc5 Created]` (This entry is highlighted with a red box.)

Smart Contract Information

The screenshot shows the Etherscan interface for the smart contract at address `0x73a61e22f57d9534110d76e8e92657750345bfc5`. The page is titled "Contract Address 0x73a61e22f57d9534110d76e8e92657750345bfc5". The main content area includes sections for Overview, More Info, and Multichain Info. The "Transactions" tab is selected, showing one transaction: `0xd46082347b...` from `0x4CE135aB...64E06ae71` to the contract at `0x73a61e22f57d9534110d76e8e92657750345bfc5` for a fee of `0.0000964` ETH. The "More Info" section highlights the "CONTRACT CREATOR" as `0x4CE135aB...64E06ae71`.

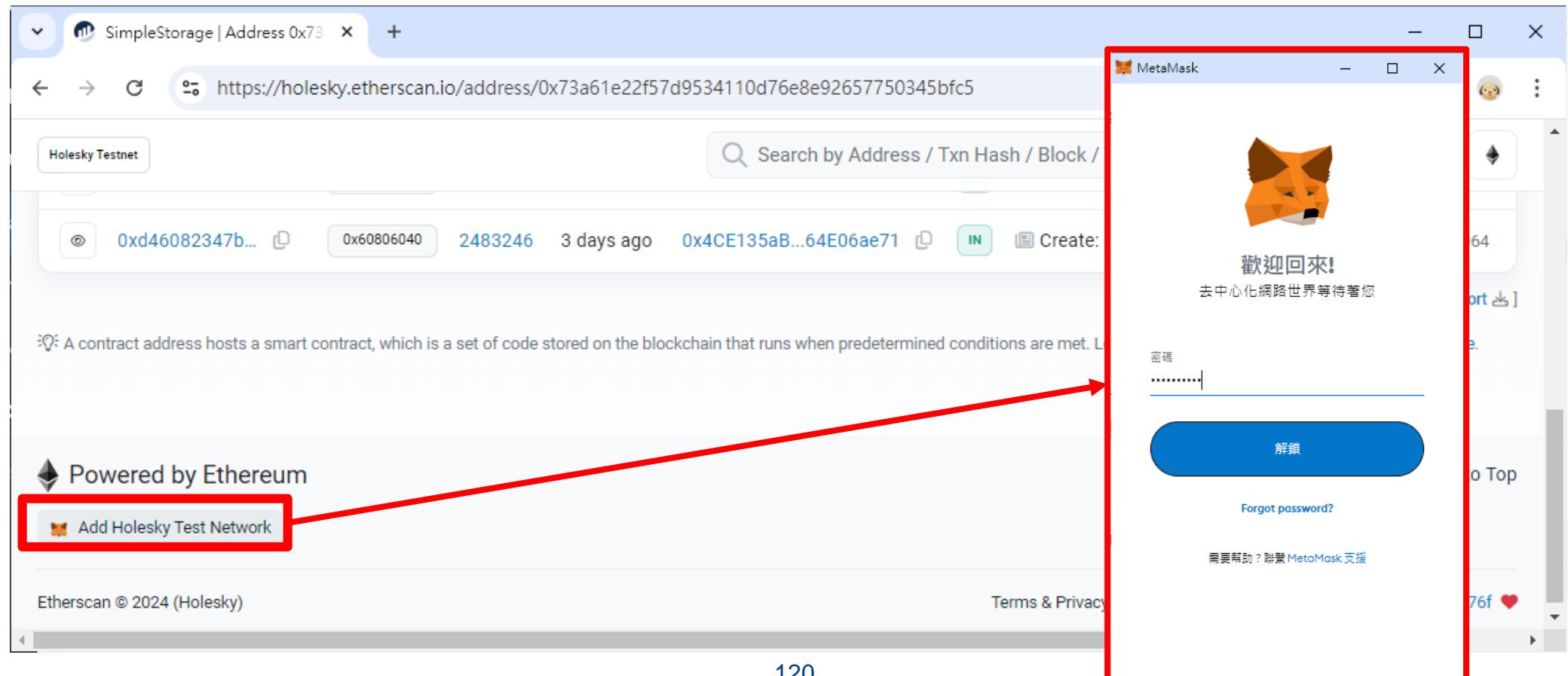
Contract `0x73A61e22f57D9534110D76E8E92657750345bfC5`

ETH BALANCE
0 ETH

CONTRACT CREATOR
`0x4CE135aB...64E06ae71` at txn `0xd46082347b...`

Transaction Hash	Method	Block	Age	From	To	Amount	Txn Fee
<code>0xd46082347b...</code>		2483246	10 mins ago	<code>0x4CE135aB...64E06ae71</code>	Contract Creation	0 ETH	0.0000964

Add Holesky Test Network



SimpleStorage | Address 0x73...

https://holesky.etherscan.io/address/0x73a61e22f57d9534110d76e8e92657750345bfc5

Holesky Testnet

Search by Address / Txn Hash / Block /

0xd46082347b... 0x60806040 2483246 3 days ago 0x4CE135aB...64E06ae71 IN Create:

A contract address hosts a smart contract, which is a set of code stored on the blockchain that runs when predetermined conditions are met. Learn more

Powered by Ethereum

Add Holesky Test Network

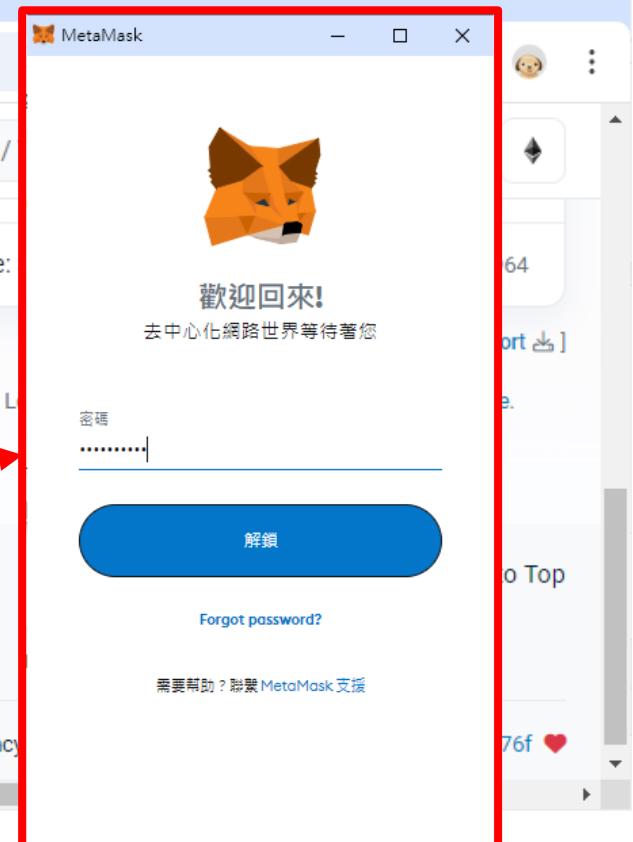
Etherscan © 2024 (Holesky)

Terms & Privacy

120

https://holesky.etherscan.io/address/0x73a61e22f57d9534110d76e8e92657750345bfc5

MetaMask



歡迎回來!

去中心化網路世界等待著您

密碼

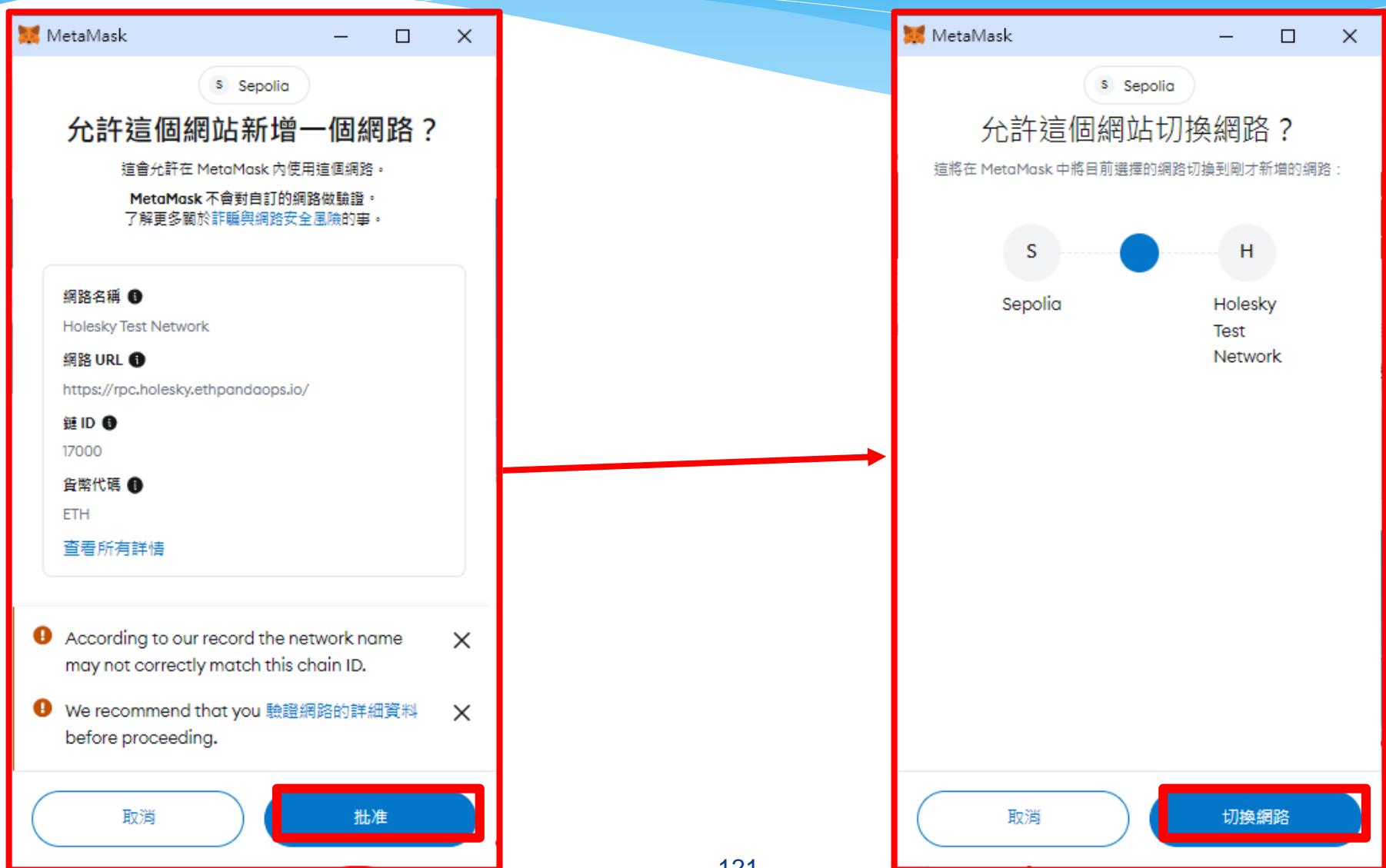
解鎖

Forgot password?

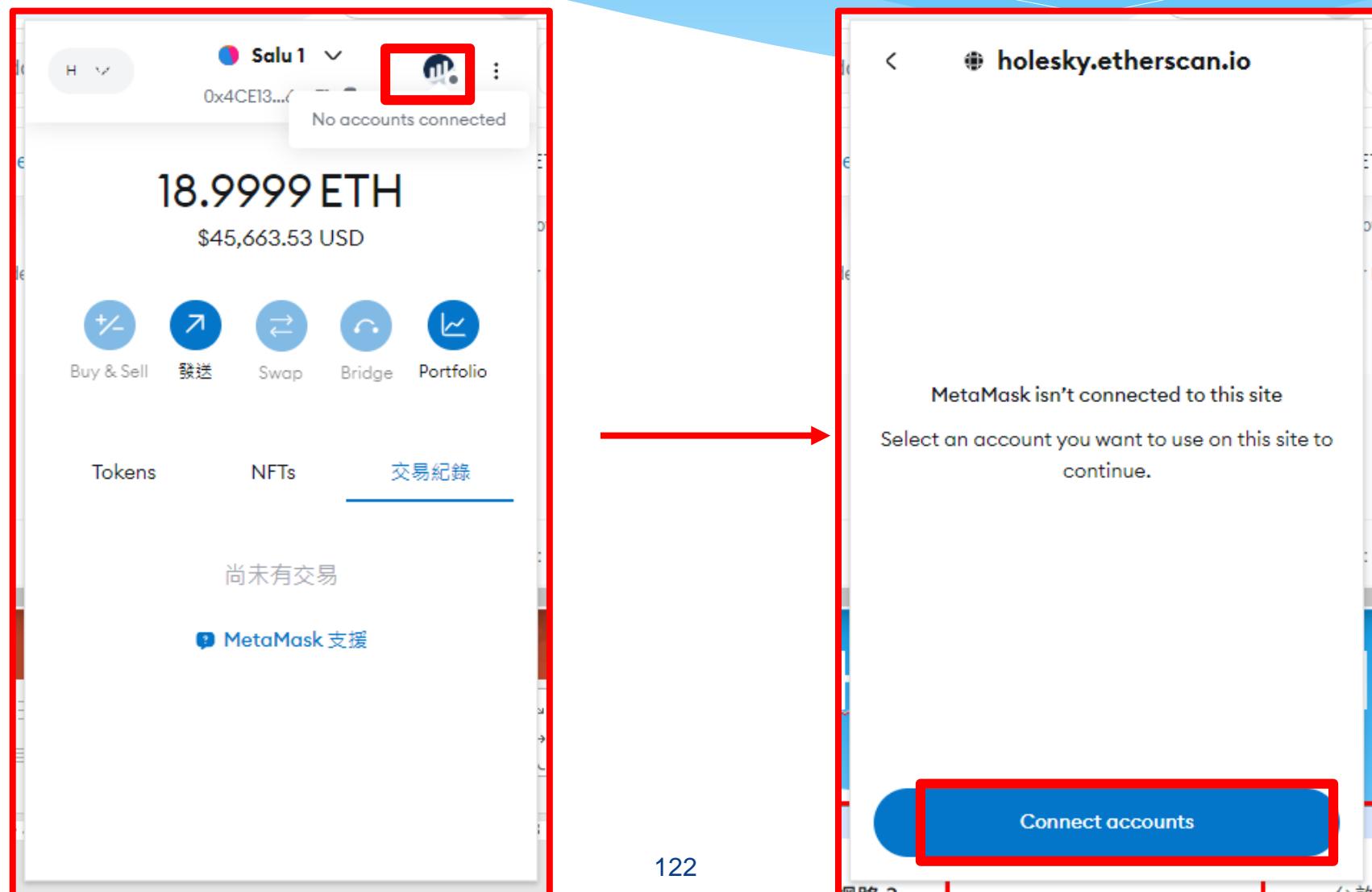
需要幫助？聯繫 MetaMask 支援

76f ❤️

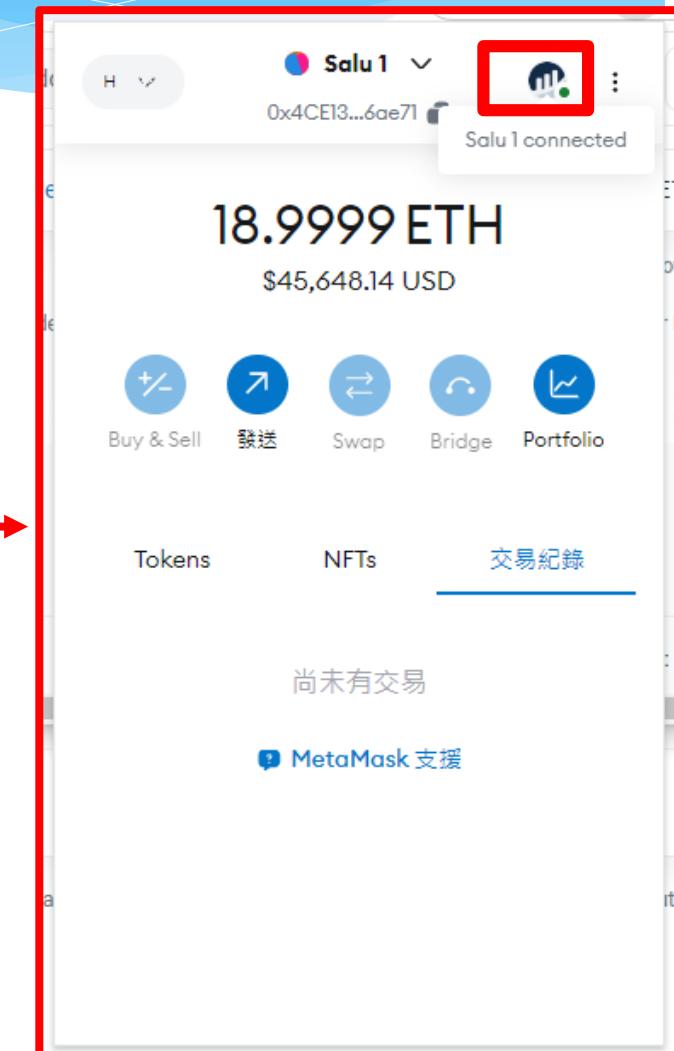
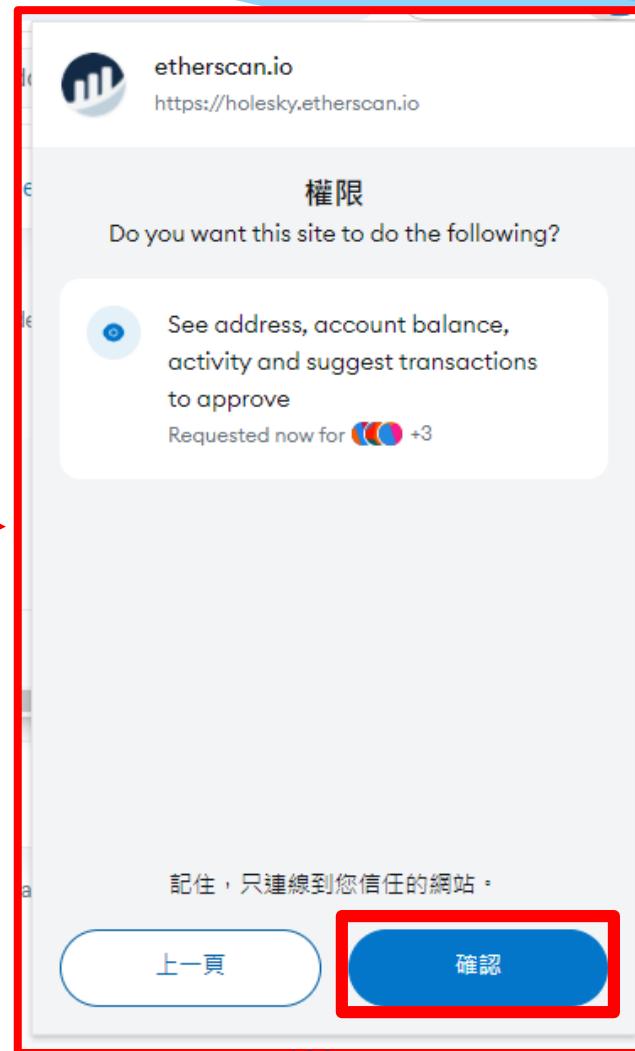
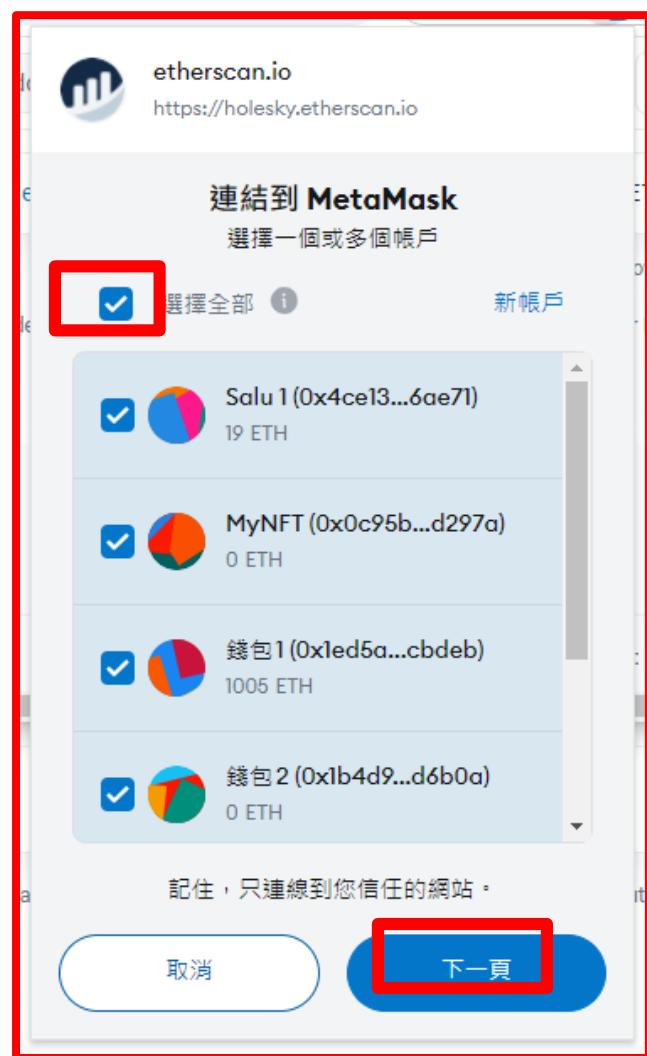
Add Holesky Test Network



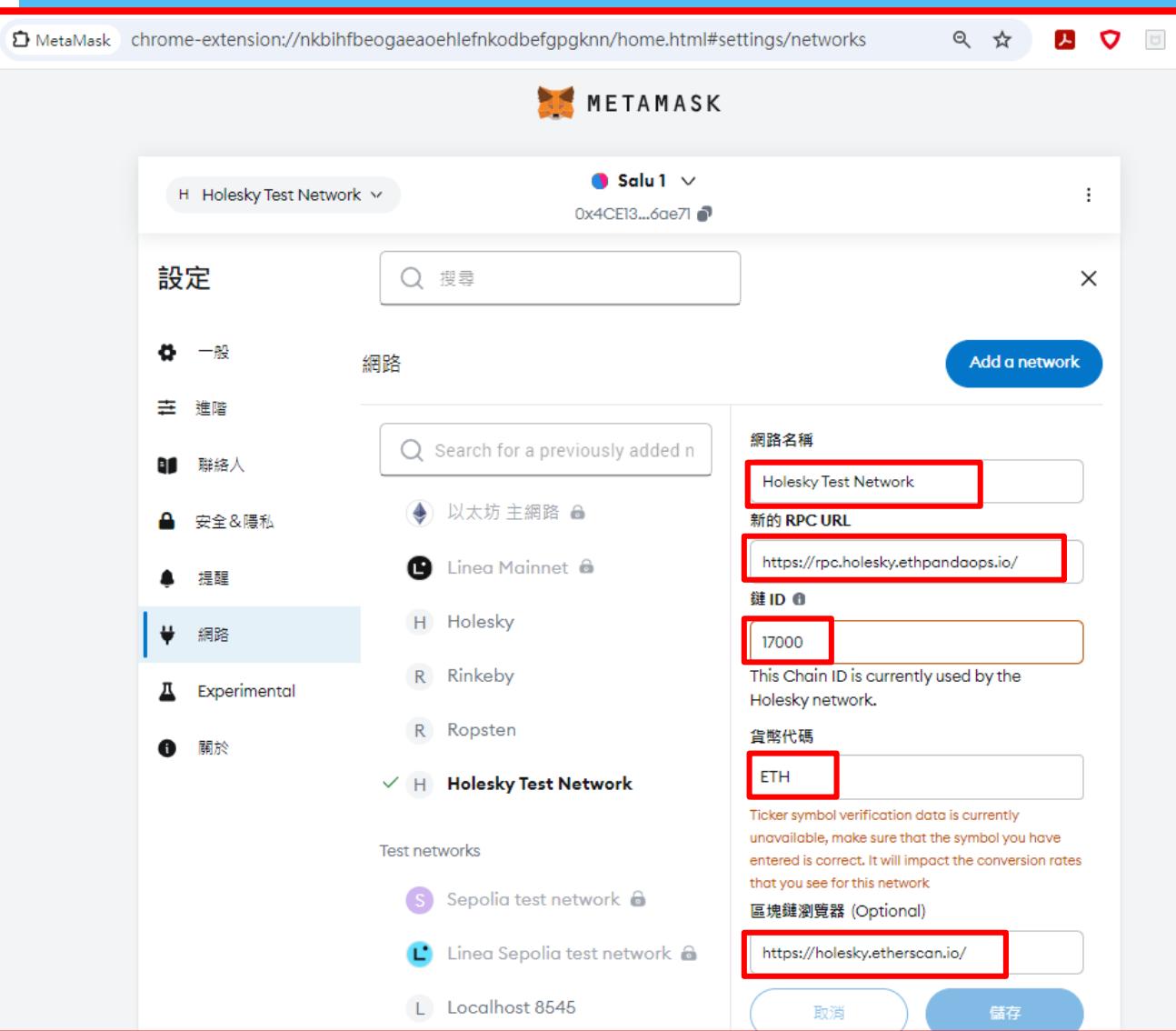
Add Holesky Test Network



Add Holesky Test Network



Add Holesky Test Network



網路名稱 : Holesky Test Network

新的 RPC URL : <https://rpc.holesky.ethpandaops.io/>

鏈 ID : 17000

貨幣代碼 : ETH

區塊鏈瀏覽器 : <https://holesky.etherscan.io/>

View & Publish

Contract Address 0x73a61e22f57d9534110d76e8e92657750345bfc5 #code

Holesky Testnet

Search by Address / Txn Hash / Block / Token

Contract 0x73A61e22f57D9534110D76E8E92657750345bfC5

Overview

ETH BALANCE 0 ETH

More Info

CONTRACT CREATOR 0x4CE135aB...64E06ae71 at txn 0xd46082347b...

Multichain Info N/A

Transactions Token Transfers (ERC-20) Contract Events

Are you the contract creator? [Verify and Publish](#) your contract source code today!

Note: We also found another contract with matching byte codes

Decompile Bytecode Switch to Opcodes View Similar Contracts

```
0x608060405234801561000f575f80fd5b5060043610610034575f3560e01c806360fe47b1146100385780636d4ce63c14610054575b5f80fd5b610052600480360381019061004d91906100ba565b610072565b005b61005c61007b565b60405161006991906100f4565b60405180910390f35b805f8190555050565b5f8054905090565b5f80fd5b5f819050919050565b61009981610087565b81146100a3575f80fd5b50565b5f813590506100b481610090565b92915050565b5f602082840312156100cf576100ce610083565b5b5f6100dc848285016100a6565b91505092915050565b6100ee8161008765b82525050565b5f6020820190506101075f8301846100e5565b9291505056fea2646970667358221220e3dccc8baafef98cffcc8cccf8e3a1869a13008c4d4711d524c2bb3d43be8102964736f6c634300081a0033
```

View & Publish Contract Source Code

The screenshot shows a web browser window titled "Verify & Publish Contract Sou". The URL in the address bar is holesky.etherscan.io/verifyContract?a=0x73a61e22f57d9534110d76e8e92657750345bfc5. The page is titled "Verify & Publish Contract Source Code" and explains that source code verification provides transparency for users interacting with smart contracts. It shows two steps: "Enter Contract Details" (selected) and "Verify & Publish".

Enter Contract Details

Please enter the Contract Address you would like to verify
0x73a61e22f57d9534110d76e8e92657750345bfc5

Please select Compiler Type
Solidity (Single file)

Please select Compiler Version
v0.8.26+commit.8a97fa7a

Uncheck to show all nightly commits

Please select Open Source License Type ⓘ
5) GNU General Public License v3.0 (GNU GPLv3)

I agree to the terms of service

Continue **Reset**

View & Publish Contract Source Code

The screenshot shows a web browser window titled "Holesky Solidity Contract Sou..." with the URL "holesky.etherscan.io/verifyContract-solc?a=0x73a61e22f57d9534110d76e8e92657750345bfc5&c=...". The page is titled "Verify & Publish Contract Source Code". It explains that source code verification provides transparency by matching uploaded code with blockchain data. It also notes a simple and structured interface for verifying contracts. A progress bar indicates step 1 "Enter Contract Details" is complete and step 2 "Verify & Publish" is in progress. The "Upload Contract Source Code" section contains instructions for REMIX compilation and programmatic verification, along with a red box highlighting the "Contract Address: 0x73a61e22f57d9534110d76e8e92657750345bfc5". Below this, compiler details are listed: "Compiler Type: SINGLE FILE / CONCATENATED METHOD" and "Compiler Version: v0.8.26+commit.8a97fa7a". At the bottom, there's a text input field for "Enter the Solidity Contract Code below *".

Holesky Solidity Contract Sou...

holesky.etherscan.io/verifyContract-solc?a=0x73a61e22f57d9534110d76e8e92657750345bfc5&c=...

Holesky Testnet

Search by Address / Txn Hash / Block / Token

Verify & Publish Contract Source Code

Source code verification provides transparency for users interacting with smart contracts. By uploading the source code, Etherscan will match the compiled code with that on the blockchain. [Read more](#).

A simple and structured interface for verifying smart contracts that fit in a single file.

1 Enter Contract Details — 2 Verify & Publish

Upload Contract Source Code

1. If the contract compiles correctly at REMIX, it should also compile correctly here.
2. We have limited support for verifying contracts created by another contract and there is a timeout of up to 45 seconds for each contract compiled.
3. For programmatic contract verification, check out the [Contract API Endpoint](#).

Contract Address:
0x73a61e22f57d9534110d76e8e92657750345bfc5

Compiler Type:
SINGLE FILE / CONCATENATED METHOD

Compiler Version:
v0.8.26+commit.8a97fa7a

Enter the Solidity Contract Code below *

Fetch from Gist

View & Publish Contract Source Code

The screenshot shows a web browser window for the Holesky Testnet on etherscan.io. The URL in the address bar is `holesky.etherscan.io/verifyContract-solc?a=0x73a61e22f57d9534110d76e8e92657750345bfc5&c=...`. The page displays a form for entering Solidity contract code. A red box highlights the code input area, which contains the following Solidity code:

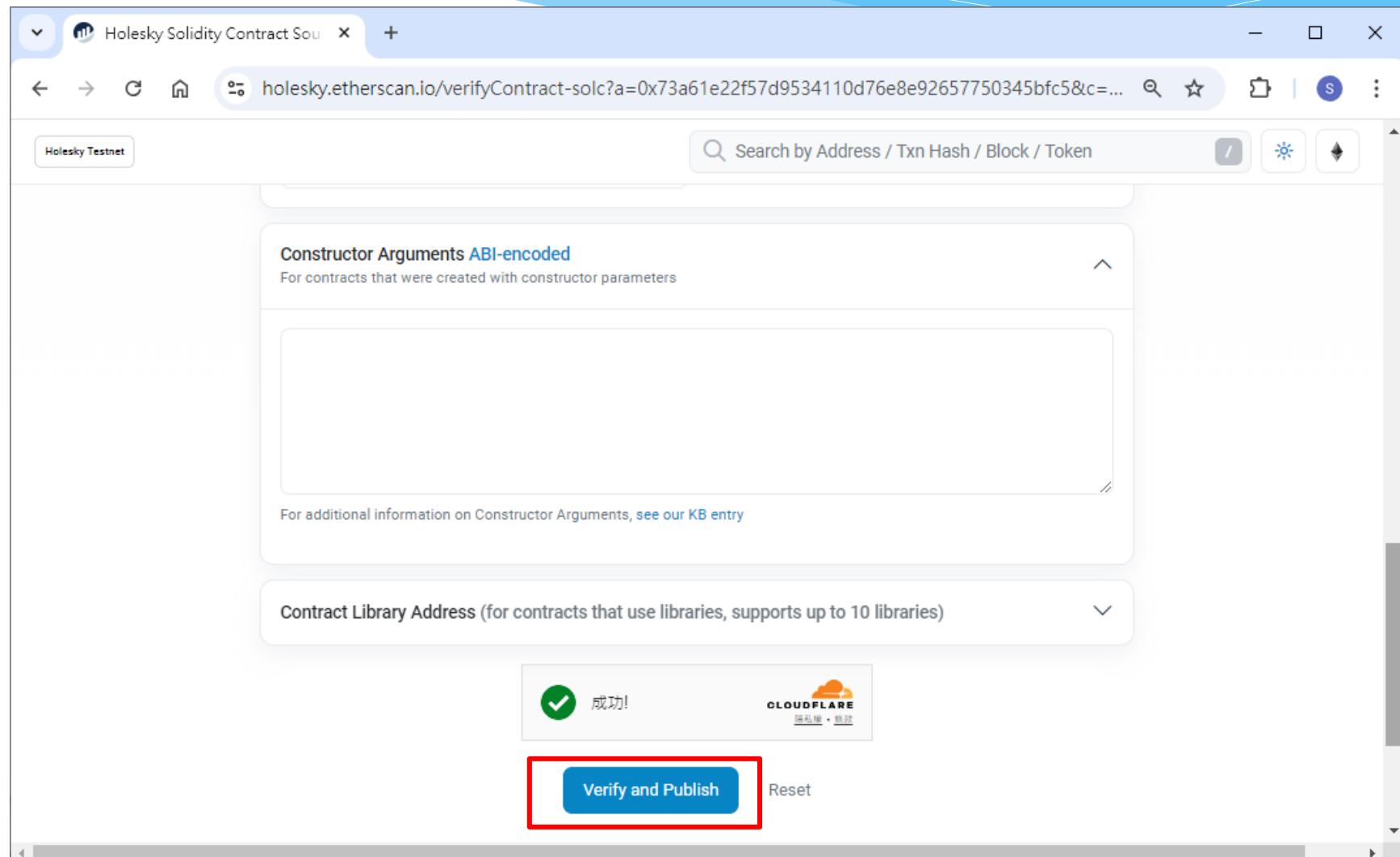
```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }
}
```

Below the code input, there is an "Advanced Configuration" section with three dropdowns: "Optimization" (set to "No"), "Runs (Optimizer)" (set to "200"), and "EVM Version to target" (set to "default (compiler defaults)"). A red box highlights the "License Type" section, which shows a dropdown menu with the option "5) GNU General Public License v3.0 (GNU GPLv3)".

View & Publish Contract Source Code



View & Publish Contract Source Code

The screenshot shows a web browser window for the Holesky Solidity Contract Sou... page on etherscan.io. The URL in the address bar is holesky.etherscan.io/verifyContract-solc?a=0x73a61e22f57d9534110d76e8e92657750345bfc5&c=... . The page title is "Verify & Publish Contract Source Code". The Etherscan logo is at the top left, and the navigation menu includes Home, Blockchain, Tokens, NFTs, and More.

Verify & Publish Contract Source Code

Source code verification provides transparency for users interacting with smart contracts. By uploading the source code, Etherscan will match the compiled code with that on the blockchain. [Read more](#).

A simple and structured interface for verifying smart contracts that fit in a single file.

1 Enter Contract Details — 2 Verify & Publish

Successfully generated Bytecode and ABI for Contract Address
[0x73a61e22f57d9534110d76e8e92657750345bfc5]

! Learn how to verify your contract on multiple blockchains with a single API key [here](#).

A red box highlights the contract address [0x73a61e22f57d9534110d76e8e92657750345bfc5], and a red arrow points to it with the text "click".

View Contract Source Code

The screenshot shows the Etherscan interface for the contract `SimpleStorage` at address `0x73a61e22f57d9534110d76e8e92657750345bfc5`. The browser URL is highlighted with a red box. The page displays the verified contract source code in Solidity, which is also highlighted with a red box. The code is as follows:

```
1 /**
2  *Submitted for verification at Etherscan.io on 2024-10-07
3 */
4
5 // SPDX-License-Identifier: GPL-3.0
6 pragma solidity >=0.8.26 <0.9.0;
7
8 contract simpleStorage {
9     uint storedData;
10
11     function set(uint x) public {
12         storedData = x;
13     }
14
15     function get() public view returns (uint) {
16         return storedData;
17     }
18 }
```

The interface includes tabs for `Code`, `Read Contract`, and `Write Contract`. It also shows optimization settings (`No with 200 runs`) and other settings (`default evmVersion, GNU GPLv3 license`). Below the code editor, there is a section for the `Contract ABI` with JSON data.

View & Publish Contract Source Code

The screenshot shows the Etherscan interface for the SimpleStorage contract at address 0x73a61e22f57d9534110d76e8e92657750345bfc5. The 'Contract' tab is selected. A red box highlights the 'Code' button in the navigation bar below the tabs. Another red box highlights the 'Contract' tab itself. The page displays the verified contract source code, compiler version, optimization settings, and other details.

SimpleStorage | Address 0x73a61e22f57d9534110d76e8e92657750345bfc5 #code

0 ETH

0x4CE135aB...64E06ae71 at txn 0xd46082347...

Transactions Token Transfers (ERC-20) Contract Events

Code Read Contract Write Contract

Contract Source Code Verified (Exact Match)

Contract Name: SimpleStorage Optimization Enabled: No with 200 runs

Compiler Version: v0.8.26+commit.8a97fa7a Other Settings: default evmVersion, GNU GPLv3 license

Contract Source Code (Solidity)

b IDE Outline More Options

Write Contract-Connected Web3

The screenshot shows the Etherscan interface for the address `0x73a61e22f57d9534110d76e8e92657750345bfc5`. The browser title bar reads "SimpleStorage | Address 0x73". The URL in the address bar is `holesky.etherscan.io/address/0x73a61e22f57d9534110d76e8e92657750345bfc5#writeContract`. The page header includes a "Holesky Testnet" button, a search bar, and various navigation icons.

The main content area has tabs for "Transactions", "Token Transfers (ERC-20)", "Contract" (which is selected and highlighted in blue), and "Events". Below these tabs, there are buttons for "Code", "Read Contract", "Write Contract" (which is also highlighted with a red box), and "Connect to Web3" (also highlighted with a red box). A message indicates "0 ETH" and a transaction hash `0x4CE135aB...64E06ae71` at block `0xd46082347...`.

The "Write Contract" section contains a single entry: "1. set (0x60fe47b1)". At the bottom right of this section are three small icons: a magnifying glass, a gear, and a right-pointing arrow.

Connect to Web3

The screenshot shows a web browser window displaying the Etherscan interface for the address 0x73a61e22f57d9534110d76e8e92657750345bfc5. The URL in the address bar is holesky.etherscan.io/address/0x73a61e22f57d9534110d76e8e92657750345bfc5#writeContract. The page is titled "SimpleStorage | Address 0x73...".

The interface includes a sidebar with "Holesky Testnet" and a balance of "0 ETH". The main content area has tabs for "Transactions", "Token Transfers (ERC-20)", and "Contract". The "Contract" tab is active, showing sub-options "Code", "Read Contract", and "Write Contract". A red circle highlights the "Write Contract" button.

A modal window is centered on the screen, titled "holesky.etherscan.io 顯示". It contains a note: "Please take note that this is a beta version feature and is provided on an "as is" and "as available" basis. Etherscan does not give any warranties and will not be liable for any loss, direct or indirect through continued use of this feature." At the bottom of the modal are two buttons: a blue "確定" button with a red border and a light blue "取消" button.

On the right side of the interface, there is a sidebar with a "lock / Token" section and icons for search, star, and refresh.

Connect to Web3

The screenshot shows a web browser window displaying the Etherscan interface for the address `0x73a61e22f57d9534110d76e8e92657750345bfc5`. The main page shows the user has 0 ETH in their Holesky Testnet account. A modal window titled "Connect a Wallet" is open, listing three options: MetaMask (selected and highlighted with a red box), WalletConnect, and Coinbase Wallet.

MetaMask

Popular

WalletConnect

Coinbase Wallet

SimpleStorage | Address 0x73a61e22f57d9534110d76e8e92657750345bfc5

holesky.etherscan.io/address/0x73a61e22f57d9534110d76e8e92657750345bfc5#writeContract

Transactions Token Transfers (ERC-20) Contract

Code Read Contract Write Contract

Connect to Web3

1. set (0x60fe47b1)

[Expand all] [Reset]

Connected-Web3

etherscan.io
https://holesky.etherscan.io

連結到 MetaMask
選擇一個或多個帳戶

選擇全部 i 新帳戶

Salu 1 (0x4ce13...6ae71)
18 ETH

錢包 3 (0x29587...a66c...)
0 ETH

錢包 2 (0xb4d9...d6b...)
0 ETH

錢包 1 (0x1ed5a...cbdeb)
1005 ETH

記住，只連線到您信任的網站。

取消 下一页

etherscan.io
https://holesky.etherscan.io

權限
Do you want this site to do the following?

See address, account balance, activity and suggest transactions to approve
Requested now for

記住，只連線到您信任的網站。

上一页 確認

136

Write contract-set function

SimpleStorage | Address 0x73

holesky.etherscan.io/address/0x73a61e22f57d9534110d76e8e926577503

Holesky Testnet

Search by Address / Txn Hash

0 ETH

Transactions Token Transfers (ERC-20) Contract Events

Code Read Contract Write Contract

Connected - Web3 [0x4CE1...ae71]

1. set (0x60fe47b1)

x (uint256) +

8888

Write

Estimated fee \$0.06
0.00002377 ETH
Market -60 sec Max fee: 0.00002377 ETH

總量 \$0.06 0.00002377 ETH
Amount + gas fee Max amount: 0.00002377 ETH

自訂 NONCE 1

拒絕 確認

View your transaction

The screenshot shows a web browser window for the Holesky Testnet on Etherscan. The address is 0x73a61e22f57d9534110d76e8e92657750345bfc5. The tab title is "SimpleStorage | Address 0x73...". The URL in the address bar is "holesky.etherscan.io/address/0x73a61e22f57d9534110d76e8e92657750345bfc5#writeContract". The page displays a list of transactions, with one specific transaction highlighted: 0x4CE135aB...64E06ae71 at txn 0xd46... The transaction details show 0 ETH sent. Below the transaction list, there are tabs for "Transactions", "Token Transfers (ERC-20)", "Contract", and "Events", with "Contract" being the active tab. Under the "Contract" tab, there are buttons for "Code", "Read Contract", and "Write Contract", with "Write Contract" being the active button. A red box highlights the "View your transaction" button. At the bottom left, there are "Write" and "View your transaction" buttons. At the bottom right, there are "[Expand all]" and "[Reset]" links.

View your transaction

The screenshot shows a web browser displaying the Etherscan.io transaction details page for a Holesky Testnet transaction. The transaction hash is `0xd375a6948376f58cf0fd25c0a5d6409f16260eaf9d4a78fdcf5d8450bb7d7588`. The status is listed as "Success" with a green checkmark icon, which is highlighted with a red box. The transaction was included in block `2483453` with 32 confirmations. It occurred 7 minutes ago at `Oct-07-2024 04:46:24 AM UTC`. The transaction action involved calling a function on contract `0x4CE135aB...64E06ae71` from address `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71` to address `0x73A61e22f57D9534110D76E8E92657750345bfC5`. The "Call" button is also highlighted with a red box.

[This is a Holesky **Testnet** transaction only]

② Transaction Hash: `0xd375a6948376f58cf0fd25c0a5d6409f16260eaf9d4a78fdcf5d8450bb7d7588`

② Status: Success

② Block: `2483453` 32 Block Confirmations

② Timestamp: 7 mins ago (Oct-07-2024 04:46:24 AM UTC)

④ Transaction Action: Call Set Function by `0x4CE135aB...64E06ae71` on `0x73A61e22...50345bfC5`

② From: `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71`

② To: `0x73A61e22f57D9534110D76E8E92657750345bfC5`

View your transaction-More Detail

The screenshot shows a web browser displaying the transaction details for a Holesky Testnet transaction. The transaction hash is `0xd375a6948376f58cf0fd25c0a5d6409f16260eaf9d4a78fdcf5d8450bb7d7588`. The status is marked as **Success** with **2483453** block confirmations. The transaction was performed 3 days ago (Oct-07-2024 04:46:24 AM UTC) from address `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71` to address `0x73A61e22f57D9534110D76E8E92657750345bfC5`. The value transferred was 0 ETH (\$0.00), and the transaction fee was 0.000023768163187552 ETH (\$0.00). The gas price was 0.543794344 Gwei (0.000000000543794344 ETH). A red box highlights the transaction hash, and another red box highlights the "More Details" button at the bottom.

[This is a Holesky Testnet transaction only]

② Transaction Hash: `0xd375a6948376f58cf0fd25c0a5d6409f16260eaf9d4a78fdcf5d8450bb7d7588`

② Status: **Success**

② Block: **2483453** 22483 Block Confirmations

② Timestamp: 3 days ago (Oct-07-2024 04:46:24 AM UTC)

⚡ Transaction Action: Call Function by `0x4CE135aB...64E06ae71` on `0x73A61e22...50345bfC5`

② From: `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71`

② To: `0x73A61e22f57D9534110D76E8E92657750345bfC5`

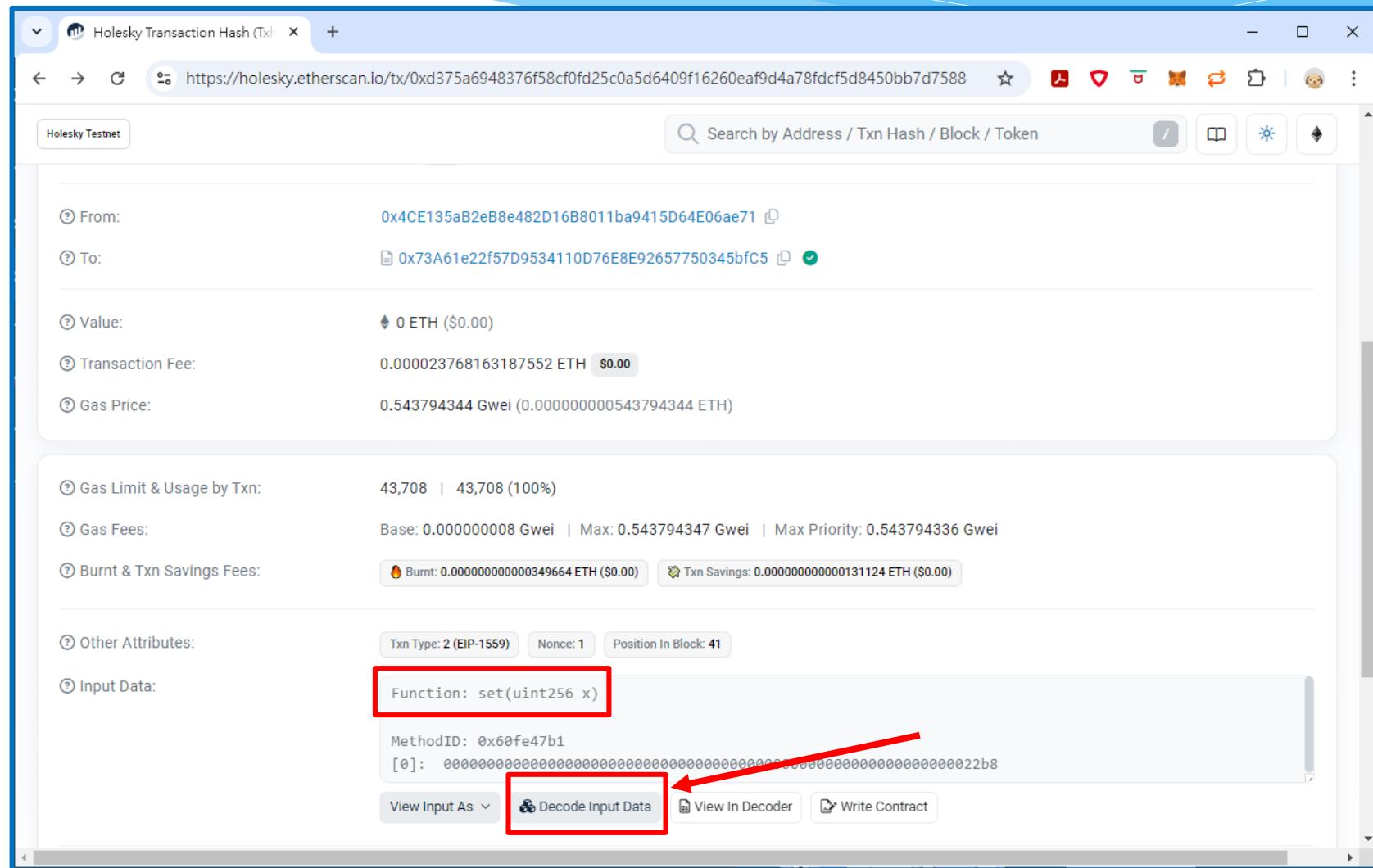
② Value: 0 ETH (\$0.00)

② Transaction Fee: 0.000023768163187552 ETH \$0.00

② Gas Price: 0.543794344 Gwei (0.000000000543794344 ETH)

More Details: + Click to show more

View your transaction-Input Data



View your transaction-Decode Input Data

The screenshot shows a browser window displaying the Holesky Transaction Hash (Txn) details on etherscan.io. The URL in the address bar is <https://holesky.etherscan.io/tx/0xd375a6948376f58cf0fd25c0a5d6409f16260eaf9d4a78fdcf5d8450bb7d7588>.

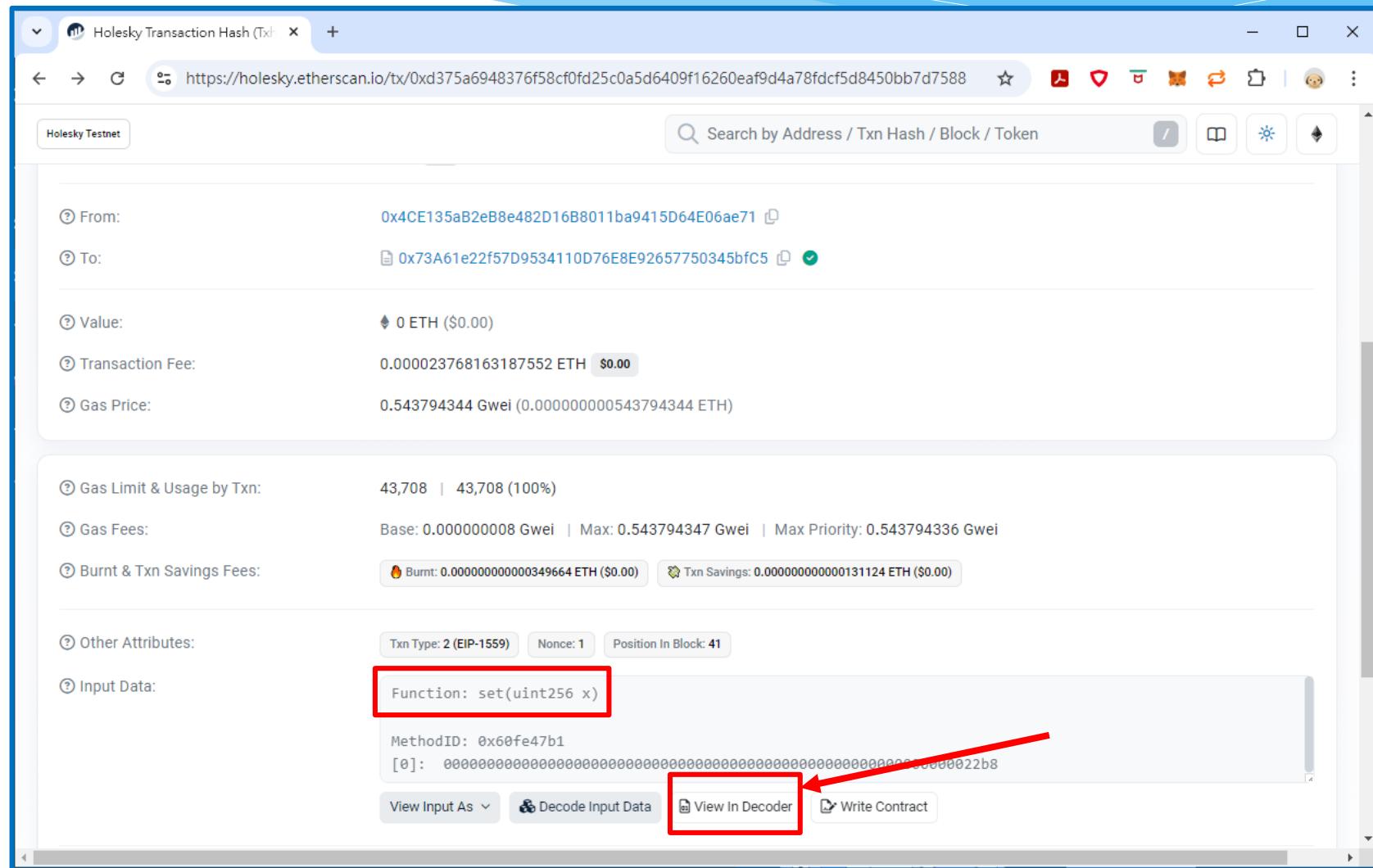
The page displays various transaction details:

- Gas Price: 0.543794344 Gwei (0.000000000543794344 ETH)
- Gas Limit & Usage by Txn: 43,708 | 43,708 (100%)
- Gas Fees: Base: 0.00000008 Gwei | Max: 0.543794347 Gwei | Max Priority: 0.543794336 Gwei
- Burnt & Txn Savings Fees:
 - Burnt: 0.00000000000349664 ETH (\$0.00)
 - Txn Savings: 0.000000000000131124 ETH (\$0.00)
- Other Attributes:
 - Txn Type: 2 (EIP-1559)
 - Nonce: 1
 - Position In Block: 41
- Input Data:

#	Name	Type	Data
0	x	uint256	8888

A red arrow points to the 'Data' column of the first row in the 'Input Data' table, highlighting the value '8888'.

View your transaction-Input Data



View your transaction-Input Data Decoder

The screenshot shows a browser window for the Etherscan Input Data Decoder. The URL in the address bar is <https://holesky.etherscan.io/inputdatadecoder?tx=0xd375a6948376f58cf0fd25c0a5d6409f16260eaf9d4a78fdf5d8450bb7d7588>. The page title is "Input Data Decoder | Etherscan". The main content area is titled "Input Data Decoder Beta" and says "Interpret and analyze data sent to Ethereum smart contracts." It has a "Choose option:" section with radio buttons for "from Transaction" (selected), "from Address", "from ABI", and "without ABI". Below that is a "Txn Hash" input field containing the value "0xd375a6948376f58cf0fd25c0a5d6409f16260eaf9d4a78fdf5d8450bb7d7588", which is highlighted with a red box. There are "Decode" and "Reset" buttons. The "Decoded Results:" section shows a "Function" button with a "set" dropdown, also highlighted with a red box. Below it is a "uint256" input field with the value "8888", which is also highlighted with a red box and has a red arrow pointing to it from the bottom left.

Connect to Web3

The screenshot shows the Etherscan interface for the SimpleStorage contract at address 0x73a61e22f57d9534110d76e8e92657750345bfc5. The 'Contract' tab is selected. A red box highlights the 'Read Contract' button. Another red box highlights the 'Connect to Web3' button. A callout box with a purple border and a red arrow points from the 'Connect to Web3' button to the callout. The callout contains the following text:

holesky.etherscan.io 顯示
Please take note that this is a beta version feature and is provided on an "as is" and "as available" basis. Etherscan does not give any warranties and will not be liable for any loss, direct or indirect through continued use of this feature.

At the bottom right of the callout are two buttons: a blue '確定' (Confirm) button and a light blue '取消' (Cancel) button.

Below the callout, there are links for '[Expand all]' and '[Reset]'. At the very bottom of the page, there is a footer with the number 145 and the URL <https://holesky.etherscan.io/address/0x73a61e22f57d9534110d76e8e92657750345bfc5#readContract>.

Connect a Wallet-MetaMask

The screenshot shows a web browser window with the URL holesky.etherscan.io/address/0x73a61e22f57d9534110d76e8e92657750345bfc5#readContract. A modal window titled "Connect a Wallet" is displayed, containing the following information:

- A note: "Connecting wallet for read function is optional, useful if you want to call certain functions or simply use your wallet's node."
- Three options:
 - MetaMask** (highlighted with a red box)
 - WalletConnect
 - Coinbase Wallet

At the bottom right of the modal, there are "[Expand all]" and "[Reset]" buttons.

Connected Web3 using account

The screenshot illustrates the integration of a Web3 wallet (MetaMask) with an Ethereum blockchain explorer (Etherscan). On the left, the Etherscan interface shows a deployed contract at address 0x73a61e22f57d9534110d76e8e92657750345bfc5. The 'Contract' tab is active, and the 'Read Contract' button is highlighted. A red box and arrow point to the 'Connected - Web3 [0x4CE1...ae71]' status message in the bottom-left corner of the browser window. On the right, the MetaMask wallet interface is displayed, showing a balance of 17.9999 ETH (equivalent to \$44,762.28 USD). The 'Transactions' tab is selected, showing two recent events: 'Set' (confirmed) and '部署合約' (Deployed Contract) (confirmed), both with a value of -0 ETH and -\$0.00 USD.

Date	Event	Value	USD Value
Oct 7, 2024	Set	-0 ETH	-\$0.00 USD
Oct 7, 2024	部署合約	-0 ETH	-\$0.00 USD

Read Contract-get function

SimpleStorage | Address 0x73a61e22f57d9534110d76e8e92657750345bfc5

holesky.etherscan.io/address/0x73a61e22f57d9534110d76e8e92657750345bfc5#readContract

0 ETH

0x4CE135aB...64E06ae71 at txn 0xd4608...

Transactions Token Transfers (ERC-20) Contract Events

Code Read Contract Write Contract

Connected - Web3 [0x4CE1...ae71]

Read Contract Information

1. get

8888 uint256

click

[Expand all] [Reset]

set Contract using Remix

The screenshot shows the Ethereum Remix IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' sidebar is open, showing the deployed contract address `0x73A61e22f57D9534110D` (circled 1). Below it, the 'Deployed/Unpinned Contracts' section lists the deployed contract `SIMPLESTORAGE AT 0X73A...` (circled 2). At the bottom, there are two buttons: `set` (circled 4) and `get`. The `set` button has the value `6666` in its input field (circled 3).

The main workspace displays the Solidity code for the `SimpleStorage` contract:

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

To the right, a detailed transaction dialog is shown for the `set` transaction. It includes the recipient address `0x73A61...5bfC5`, the function name `SET`, and the estimated fee information:

- Estimated fee: `$0.03` (`0.00001017 ETH`)
- Market -60 sec: `Max fee: 0.00001017 ETH`
- Amount + gas fee: `$0.03 0.00001017 ETH` (`Max amount: 0.00001017 ETH`)

The dialog also features a `NONCE` input field with the value `2` (circled 5), and two buttons at the bottom: `拒絕` and `確認`.

View transaction on etherscan

The screenshot shows a mobile application interface for managing Ethereum assets. At the top, it displays a balance of **17.9999 ETH** (equivalent to **\$44,740.48 USD**). Below the balance are five navigation buttons: **Buy & Sell**, **發送** (Send), **Swap**, **Bridge**, and **Portfolio**. Underneath these buttons are three tabs: **Tokens**, **NFTs**, and **交易紀錄** (Transactions), with **交易紀錄** being the active tab. A date filter shows **Oct 7, 2024**. The transaction history lists three entries:

操作	狀態	金額
Set	已確認	-0 ETH -\$0.00 USD
Set	已確認	-0 ETH -\$0.00 USD
部署合約	已確認	-0 ETH -\$0.00 USD

A red box highlights the first transaction entry, and a red circle with the number **1** is placed over the "Set" button in that row.

This screenshot shows a detailed view of a specific transaction on Etherscan. The transaction is identified by the label **Set** and has a status of **已確認** (Confirmed). A red box highlights the **View on block explorer** button, which is circled with the number **2**.

Status: 已確認

交易:

欄位	值
Nonce	2
數量	-0 ETH
Gas 上限 (單位)	26951
Gas 用量 (單位)	26608
Base fee (GWEI)	0.000000009
Priority fee (GWEI)	0.37730754
Total gas fee	0.00001 ETH \$0.02 USD
Max fee per gas	<0.000001 ETH \$0.00 USD

View transaction on etherscan

The screenshot shows a web browser window with the title "Holesky Transaction Hash (Txn)" and the URL "holesky.etherscan.io/tx/0xb9456c6a9fcd4225676f0efab6ab64e8eb93cc47b5ff807d88ba738cd93d287f". The page is titled "Transaction Details" and displays various transaction details. A red box highlights the URL in the address bar and the "Success" status indicator for the transaction.

This is a Holesky Testnet transaction only

Transaction Hash: 0xb9456c6a9fcd4225676f0efab6ab64e8eb93cc47b5ff807d88ba738cd93d287f

Status: Success

Block: 2483649 | 39 Block Confirmations

Timestamp: 8 mins ago (Oct-07-2024 05:28:12 AM UTC)

Transaction Action: Call Function by 0x4CE135aB...64E06ae71 on 0x73A61e22...50345bfC5

From: 0x4CE135aB2e88e482D16B8011ba9415D64E06ae71

To: 0x73A61e22f57D9534110D76E8E92657750345bfC5

Value: 0 ETH (\$0.00)

Transaction Fee: 0.000010039399263792 ETH (\$0.00)

Gas Price: 0.377307549 Gwei (0.000000000377307549 ETH)

get Contract using Remix

The screenshot shows the Ethereum IDE interface in Remix. On the left, the sidebar displays "DEPLOY & RUN TRANSACTIONS" with a section for "Transactions recorded" and "Pinned Contracts". Below that is a "Deployed/Unpinned Contracts" section showing a deployed contract named "SIMPLESTORAGE AT 0X73A...E" with a balance of 0 ETH. On the right, the main workspace shows the Solidity code for the "SimpleStorage" contract:

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.8.26 <0.9.0;

contract SimpleStorage {
    uint storedData;

    function set(uint x) public {
        storedData = x;
    }

    function get() public view returns (uint) {
        return storedData;
    }
}
```

Below the code, the transaction history shows a recent call to the "get" function:

- ① A red circle highlights the "get" button in the "Low level interactions" panel.
- ② A red rectangle highlights the transaction log entry: "[block:2483608 txIndex:1] from: 0x4f1...13309 to: 0x6376d142582f2ec76d6bf3d47020e0a163c51874 0x637...51874 value: 0 wei data: 0xfa5...4b07d logs: 0 hash: 0xa59...15f31 transact to SimpleStorage.set pending ... call to simplestorage.get".
- ③ A red rectangle highlights the value "0: uint256: 6666" in the "Low level interactions" panel.

At the bottom, there are status messages: "Did you know? To learn new contract patterns and prototype, you can activate and try the cookbook plugin!", "RemixAI Copilot (enabled)", and "Scam Alert".



References

References

- MetaMask

<https://github.com/MetaMask/metamask-extension>

- Remix

<https://github.com/ethereum/browser-solidity>

- Go Ethereum

<https://github.com/ethereum/go-ethereum>

- Solc (Solidity compiler)

<https://github.com/ethereum/solidity>

- Mist (Ethereum Wallet)

<https://github.com/ethereum/mist>

- Truffle

<https://github.com/trufflesuite/truffle>

References

- Ethereum EVM
<https://github.com/takenobu-hs>
- Stack Exchange: Ethereum block architecture
<https://ethereum.stackexchange.com/questions/268/ethereum-block-architecture/6413>
- Diving Into The Ethereum VM
<https://blog.qtum.org/diving-into-the-ethereum-vm-6e8d5d2f3c30>
- Awesome Ethereum Virtual Machine
<https://github.com/pirapira/awesome-ethereum-virtual-machine>

References

- Ethereum

- <https://github.com/ethereum/>

- Solidity Documentation

- <https://solidity.readthedocs.io>

- <https://docs.soliditylang.org/en/v0.8.28/>

- Web3 JavaScript app API for 0.2x.x

- <https://github.com/ethereum/wiki/wiki/JavaScript-API>

- Ledger

- <https://hyperledger-fabric.readthedocs.io/en/release-2.2/ledger/ledger.html>