

Blockchain

Create and Deploy an ERC20 Token

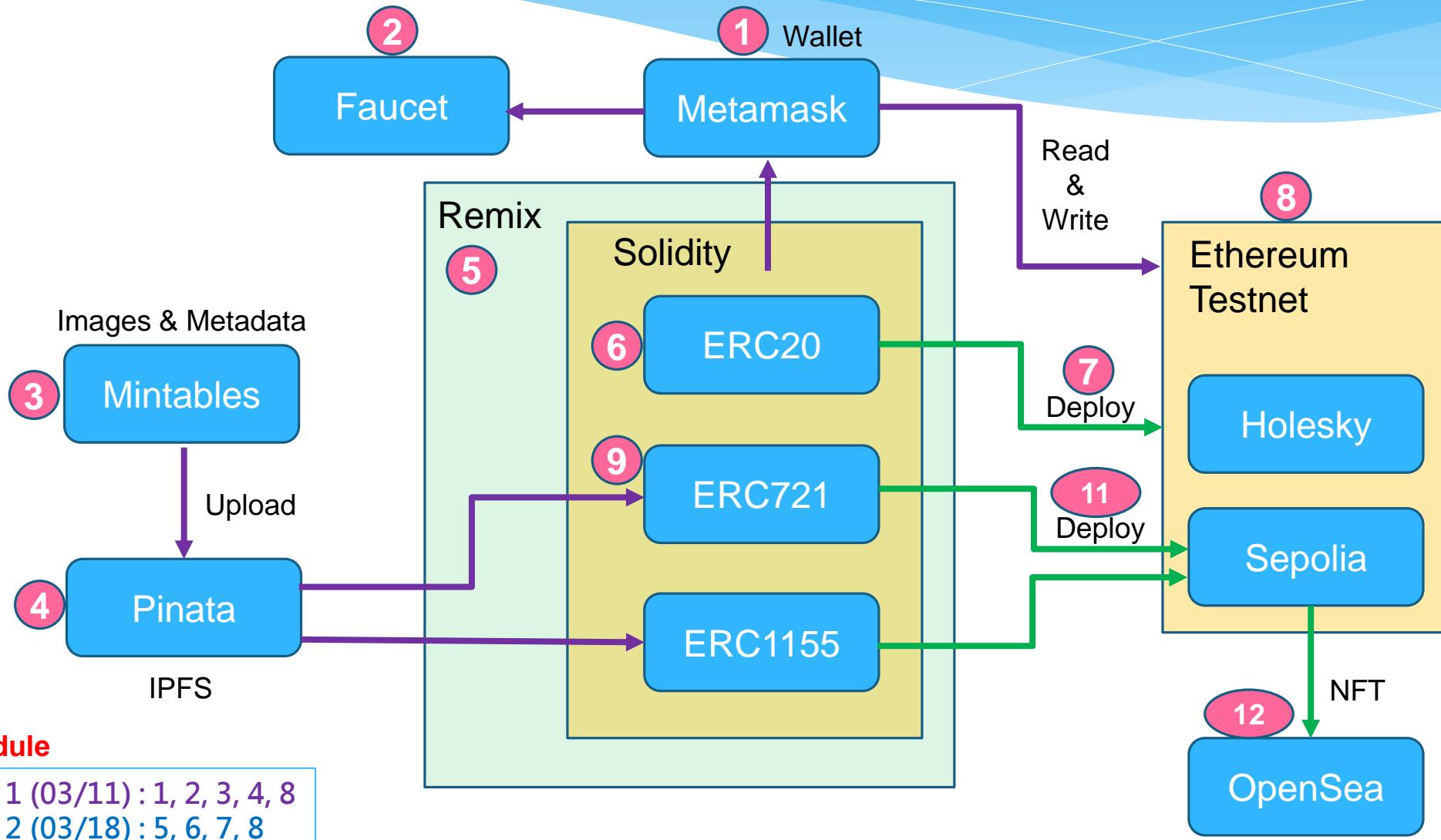
2024

目錄

- What is an ERC-20 Token
- Install the Metamask extension
- Create an ERC-20 Token
- Deploy an ERC-20 Token
- Import Token
- Transfer Token
- Modify and Extend Token
- Security and Access control
- Ether Token



Architecture Flow



What is an ERC-20 Token?

- 在ERC-20出現之前，每個要產生新代幣(Token)的人都必須重新撰寫程式碼，這代表著所有代幣都彼此不同。
 - 開發人員想要使用某一個代幣，他們必須了解該代幣的整個智能合約程式碼，原因就在於缺少建立新代幣的任何特定結構或文件指引。
 - 對於錢包和交易平台來說相當困難，因在任何應用程式上添加不同類型的代幣，需要開發人員仔細檢查每個代幣的程式碼並理解它，才能夠在其平台上處理這些代幣。
- 目前錢包和交易所使用 ERC-20 標準，係將各種標準化代幣整合到其平台上，並促進 ERC-20 代幣與其他代幣之間的互動可輕鬆地進行交換。

What is an ERC-20 Token?

- ERC-20 的ERC代表以太坊(Ethereum)請求意見，而20則是提案時的識別號碼。
- ERC-20 主旨在改善以太坊網路。
- ERC-20 已是最重要的ERC之一。它已成為在以太坊區塊鏈網路上編寫智能合約的技術標準，並用於實作建立新代幣(Token)。
- ERC-20 是包含所有在以太坊的代幣，都必須遵循的一組標準規則。
- ERC-20 將代幣定義為基於區塊鏈的、可以發送/接收且具有價值的資產。
- ERC-20 在許多方面與比特幣和萊特幣相似。然而，最顯著的區別是，ERC-20 不是在自己的區塊鏈網路上運行，而是在以太坊的區塊鏈網路上運行，並使用 **Gas** 作為交易費用。

Ethereum Improvement Proposals

The screenshot shows a web browser window displaying the Ethereum Improvement Proposals (EIPs) website at eips.ethereum.org/all. The page title is "Ethereum Improvement Proposals". Below the title, there is a navigation bar with links for All, Core, Networking, Interface, ERC, Meta, and Informational categories. The main content area is divided into two sections: "All" and "Living". The "All" section contains a table with two rows, and the "Living" section contains a table with six rows. A red box highlights the last row in the "Living" section, which corresponds to the EIP listed in the "All" section.

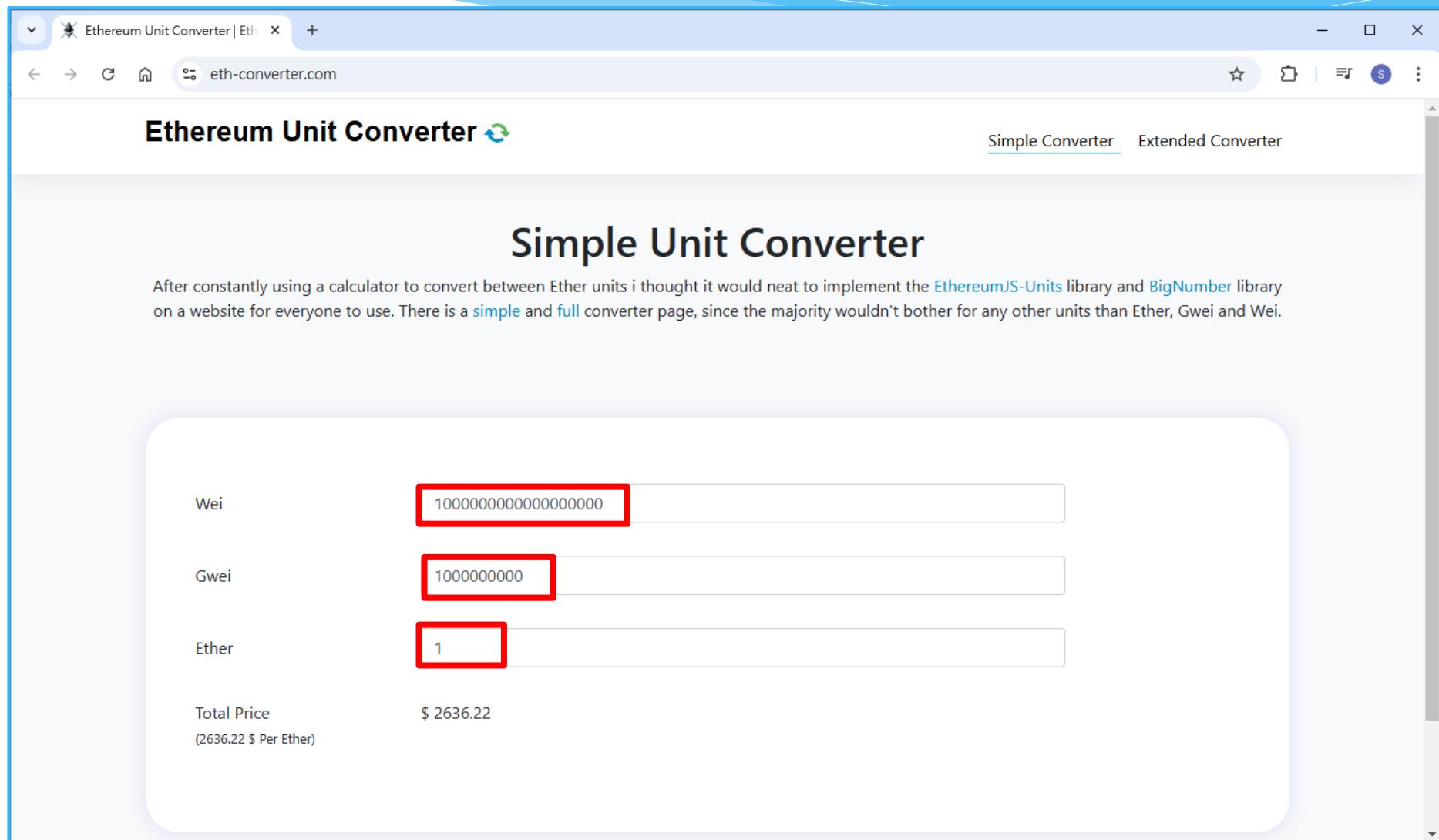
Number	Title	Author
1	EIP Purpose and Guidelines	Martin Becze <mb@ethereum.org>, Hudson Jameson <hudson@ethereum.org>, et al.
5069	EIP Editor Handbook	Pooja Ranjan (@poojaranjan), Gavin John (@Pandapip1), Sam Wilson (@SamWilson), et al.

Number	Title	Author
2	Homestead Hard-fork Changes	Vitalik Buterin (@vbuterin)
4	EIP Classification	Joseph Chow (@ethers)
5	Gas Usage for `RETURN` and `CALL*`	Christian Reitwiessner <c@ethdev.com>
6	Renaming SUICIDE opcode	Hudson Jameson <hudson@hudsonjameson.com>
7	DELEGATECALL	Vitalik Buterin (@vbuterin)
8	devp2p Forward Compatibility Requirements for Homestead	Felix Lange <felix@ethdev.com>
20	Token Standard	Fabian Vogelsteller <fabian@ethereum.org>, Vitalik Buterin <vitalik.buterin@ethereum.org>

Ethereum Gas Fee

Unit	Alternative Name	Wei Value	Gwei Value	Ether Value
Wei	Wei	1 Wei	10^{-9} Gwei	10^{-18} ETH
Kwei	Babbage	10^3 Wei	10^{-6} Gwei	10^{-15} ETH
Mwei	Lovelace	10^6 Wei	10^{-3} Gwei	10^{-12} ETH
Gwei	Shannon	10^9 Wei	1 Gwei	10^{-9} ETH
Twei	Szabo	10^{12} Wei	10^3 Gwei	10^{-6} ETH
Pwei	Finney	10^{15} Wei	10^6 Gwei	10^{-3} ETH
Ether	—	10^{18} Wei	10^9 Gwei	1 ETH

Ethereum Unit Converter



Ethereum Unit Converter

Extended Unit Converter

After constantly using a calculator to convert between Ether units I thought it would neat to implement the [EthereumJS-Units](#) library and [BigNumber](#) library on a website for everyone to use. There is a [simple](#) and [full](#) converter page, since the majority wouldn't bother for any other units than Ether, Gwei and Wei.

Wei	10000000000000000000
(factor: 0)	
Kwei / Babbage / Femtoether	1000000000000000000
(factor: 3)	
Mwei / Lovelace / Picoether	10000000000000
(factor: 6)	
Gwei / Shannon / Nanoether / Nano	1000000000
(factor: 9)	
Szabo / Microether/ Micro	1000000
(factor: 12)	
Finney / Milliether / Milli	1000
(factor: 15)	
Ether	1
(factor: 18)	
Kether / Grand	0.001
(factor: 21)	
Mether	0.000001
(factor: 24)	
Gether	0.000000001
(factor: 27)	
Tether	0.000000000001
(factor: 30)	
Total Price	\$ 2633.18
(2633.18 \$ Per Ether)	

Key Points about ERC-20

- **標準化功能**：這表示它們具有通用的規則和功能清單。這包括如何轉移代幣、如何批准交易、用戶如何存取有關代幣的數據以及代幣的總供應量。
- **智能合約和 DeFi (Decentralized Finance)**：使用智能合約可以實現複雜金融作業的自動化和執行。這對 DeFi 平台至關重要，因為這些代幣可以代表各種金融工具，例如貸款或流動性池中的股權。
- **互通性**：由於代幣遵循相同的標準，因此它們可以輕鬆互換，並且可以與以太坊網路上其他符合 ERC-20 的代幣和應用程式無縫協作。這種標準化簡化了建立新代幣的過程，並使它們立即與現有錢包、交易所和其他服務相容。
- **使用案例**：ERC-20 代幣可以代表廣泛的資產或公用事業。例如，貸款抵押品、流動性挖礦中的生息資產以及在去中心化自治組織（DAO）中授予投票權的治理代幣。
- **可轉讓性和交易**：這些代幣可以作為付款從一個帳戶轉移到另一個帳戶，類似於比特幣等加密貨幣，並且可以在各種加密貨幣交易所進行交易。

ERC-20 - 6 Mandatory Functions

- **totalSupply**: A method that defines the total supply of your tokens; when this limit is reached, the smart contract will refuse to create new tokens.
- **balanceOf**: A method that returns the number of tokens a wallet address has.
- **transfer**: A method that takes a certain amount of tokens from the total supply and gives it to a user.
- **transferFrom**: Another type of transfer method that is used to transfer tokens between users.
- **approve**: This method verifies whether a smart contract is allowed to allocate a certain amount of tokens to a user, considering the total supply.
- **allowance**: This method is exactly the same as the approved method except that it checks if one user has enough balance to send a certain amount of tokens to another.

ERC-20 - 2 Events

- **Transfer:** Triggered when tokens are transferred between accounts.
- **Approval:** Triggered when a token owner approves another account to spend tokens on their behalf.

ERC-20 - 3 Optional Functions

- **name**: A method that returns the name of the token.
- **symbol**: A method that returns the symbol of the token.
- **decimals**: A method that returns the number of decimals the token uses. It is used to define the smallest unit of the token. For example, if an ERC-20 token has a decimals value of 6, this means that the token can be divided up to six decimal places.

ERC-20 - 3 cryptocurrency supply

- **Fixed Supply:** The total supply of the token is issued on deployment and sent to the deploying wallet address..
- **Uncapped Lazy Supply:** Tokens aren't minted and issued in a unique batch on deployment, **but in small quantities and sent to the specified wallet(s)**, consequently to one or more actions.
- **Capped Lazy Supply:** Like in the **Uncapped Lazy Supply** mechanism, the **tokens are issued in small quantities**, with the only difference that here the max-supply is decided beforehand and hardcoded in the smart contract, or passed on deployment.

ERC-20 – Fixed supply

- Let's say we want a token with a fixed supply of 1000 SaluTokens initially allocated to the wallet that deploys the contract, to do so we simply called the `_mint()` private function inside the Token Constructor:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 1000 * (10 ** 18));
    }
}
```

Knowing that the constructor gets called only once, on deployment, this smart contract not only transferred 1000 tokens (the amount has 18 decimal places) to our wallet but made it also impossible to mint any new token, not having an exposed `_mint()` function in its code.

ERC-20 – Uncapped Lazy supply

- To achieve this we can simply move the mint function from our constructor to a new public function that can be called under different circumstances:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {}

    function issueToken() public {
        _mint(msg.sender, 1000 * (10 ** 18));
    }
}
```

→

```
function issueToken(address receiver, uint256 amount) public {
    _mint(receiver, amount);
}
```

ERC-20 – Capped Lazy supply

- To add a **max supply** to our Token, we can use the ERC20Capped OpenZeppelin library.
- ERC20Capped is a sub-class of ERC20Mintable that in turn inherits from the standard ERC20 contract. This allows us to substitute the ERC20.sol library in our code, with ERC20Capped, while maintaining methods such as `_mint()` and `balanceOf()`.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/extensions/ERC20Capped.sol";

contract SaluToken is ERC20Capped {
    constructor(uint256 cap) ERC20("SaluToken", "ST") ERC20Capped(cap) {}

    function issueToken() public {
        _mint(msg.sender, 1000 * (10 ** 18));
    }
}
```

We'll also need to specify a **cap value** by passing a uint256 argument to our Token constructor and feeding it into the **ERC20Capped constructor**.

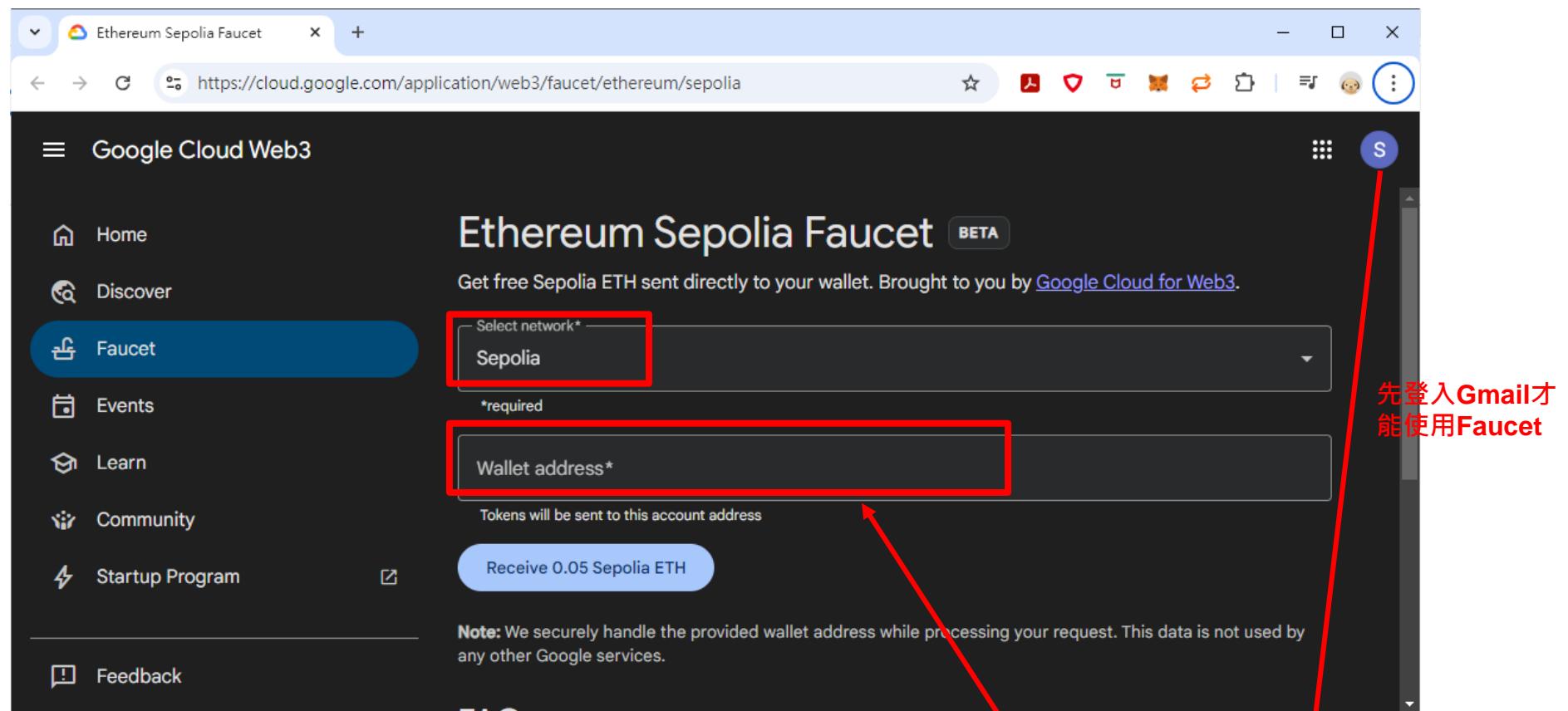
Install the Metamask extension

■ 請參考Metamask安裝文件，並安裝。



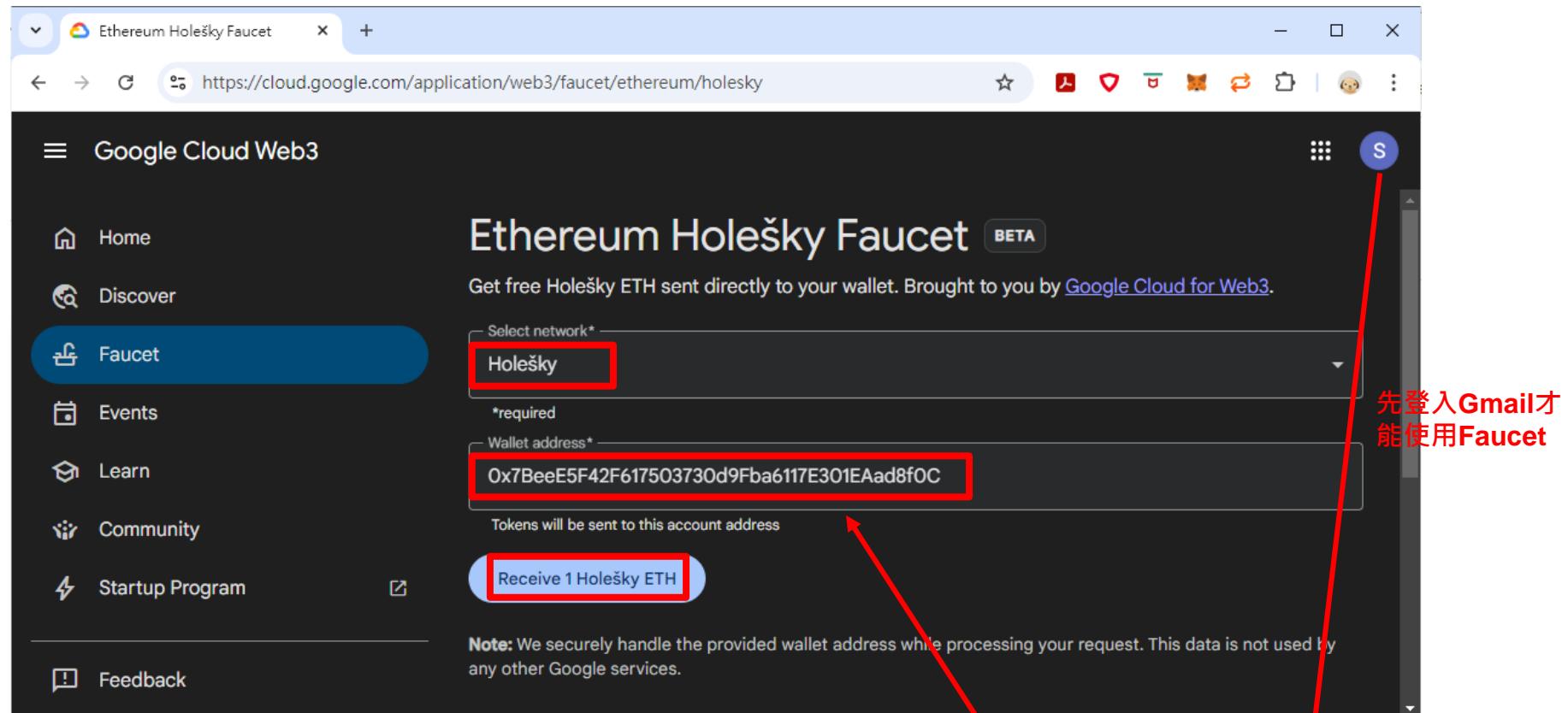
Sepolia - Faucets

■ <https://cloud.google.com/application/web3/faucet/ethereum/sepolia>



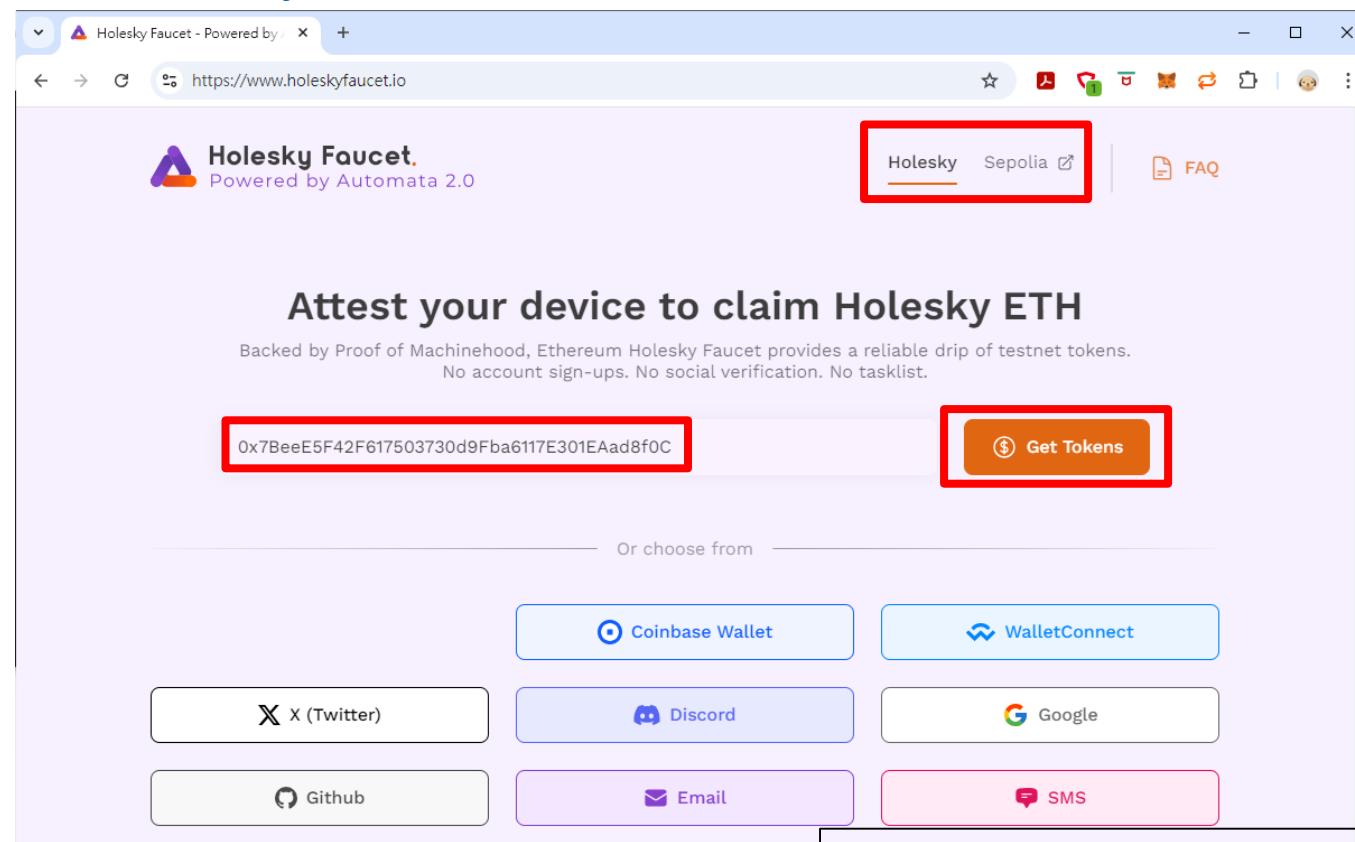
Holesky - Faucets

■ <https://cloud.google.com/application/web3/faucet/ethereum/holesky>



Holesky & Sepolia - Faucets

■ <https://www.holeskyfaucet.io/>



1. 此功能僅限win10 & Win11
2. 每個對外ip一天只能給一個Wallet address使用



Create an ERC-20 Token

Create an ERC-20 Token

- For ease and security, we'll use the OpenZeppelin ERC-20 contract to create our token. With OpenZeppelin, we don't need to write the whole ERC-20 interface. Instead, we can import the library contract and use its functions.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 100000000 * (10 ** uint256(decimals())));
    }
}
```

SaluToken.sol

=**100,000,000 * (10¹⁸)**

18

Means the total supply will be $100,000,000 \times 10^{18}$ tokens. We use 10^n because **Solidity doesn't (fully) support decimal numbers**, only integers. In wallets, dApps, etc. this number is converted back to the actual balance with decimals.
To specify 2.5 tokens, for example, you would use 2.5×10^{18} ($= 2,500,000,000,000,000,000$) instead. You can reverse it by multiplying with 10^{-18} instead.

See Full
ERC-20
code

Create an ERC-20 Token

- The **SPDX-License-Identifier** comment specifies the license under which the contract is released.
- The **pragma** directive states the compiler version to use.
- The **ERC20** contract from OpenZeppelin is imported and used as a base.
- **SaluToken** is the name of your contract, and it extends the ERC20 contract.
- The **constructor** function initializes your token with a name ("SaluToken") and a symbol ("ST").
- The **_mint** function in the constructor mints an initial supply of tokens. In this example, **100000000** tokens are minted and assigned to the address that deploys the contract. The number of tokens is adjusted by the **decimals** value, which defaults to **18** in the OpenZeppelin implementation.

Create an ERC-20 Token

- 假設在ERC-20中，我們宣告一變數`_totalSupply`為`uint256`.

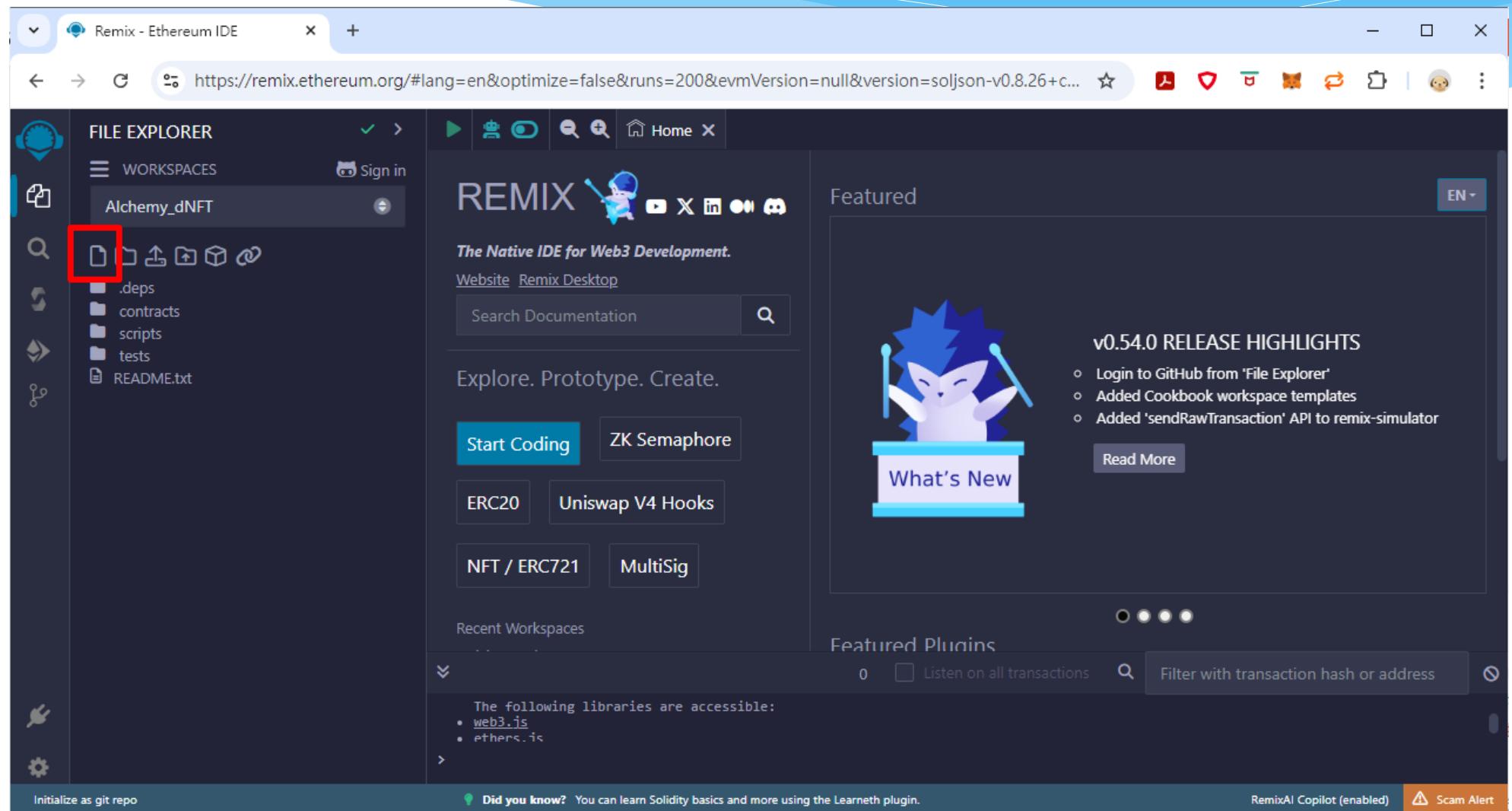
```
uint256 private _totalSupply;
```

- 理論上最大的token數量可以達到 $2^{256}-1$. 超過這個數字Solidity 0.8會阻止並產生**overflow**。而且只有在解釋token這個數字時，小數位數才重要。
- 我們曉得 2^{256} 將有 $\log(2^{256}) + 1$ 位數，即有 78位數。這意味著如果小數點後為 0，那麼最大的數字是以下數字：

1	2	3	4	5	6	7
123,456,789,012,345,678,901,234,567,890,123,456,789,012,345,678,901,234,567,890,123,456,789,012,345,678						
115,792,089,237,316,195,423,570,985,008,687,907,853,269,984,665,640,564,039,457,584,007,913,129,639,936 - 1						

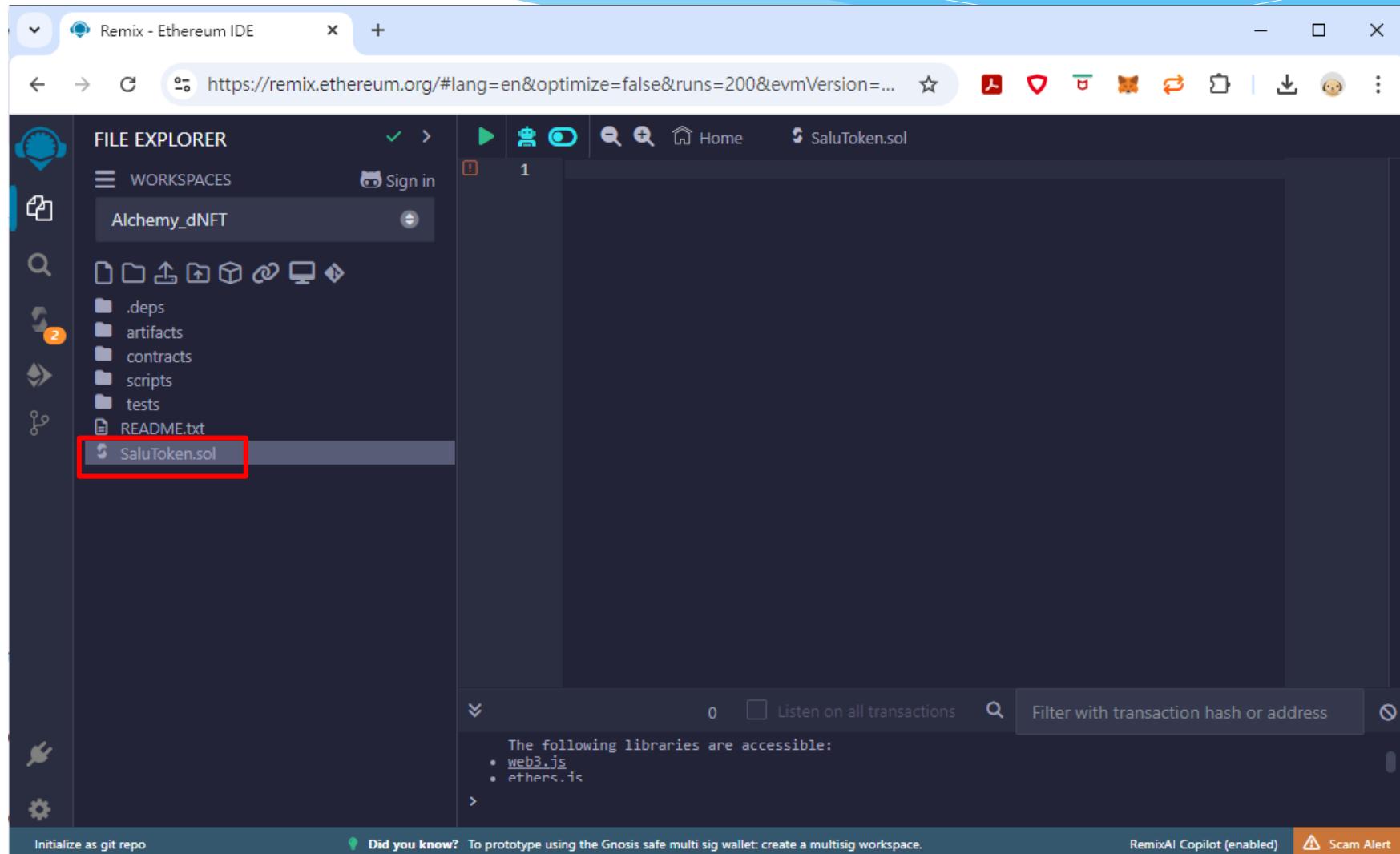
Case 1

Create an ERC-20 Token

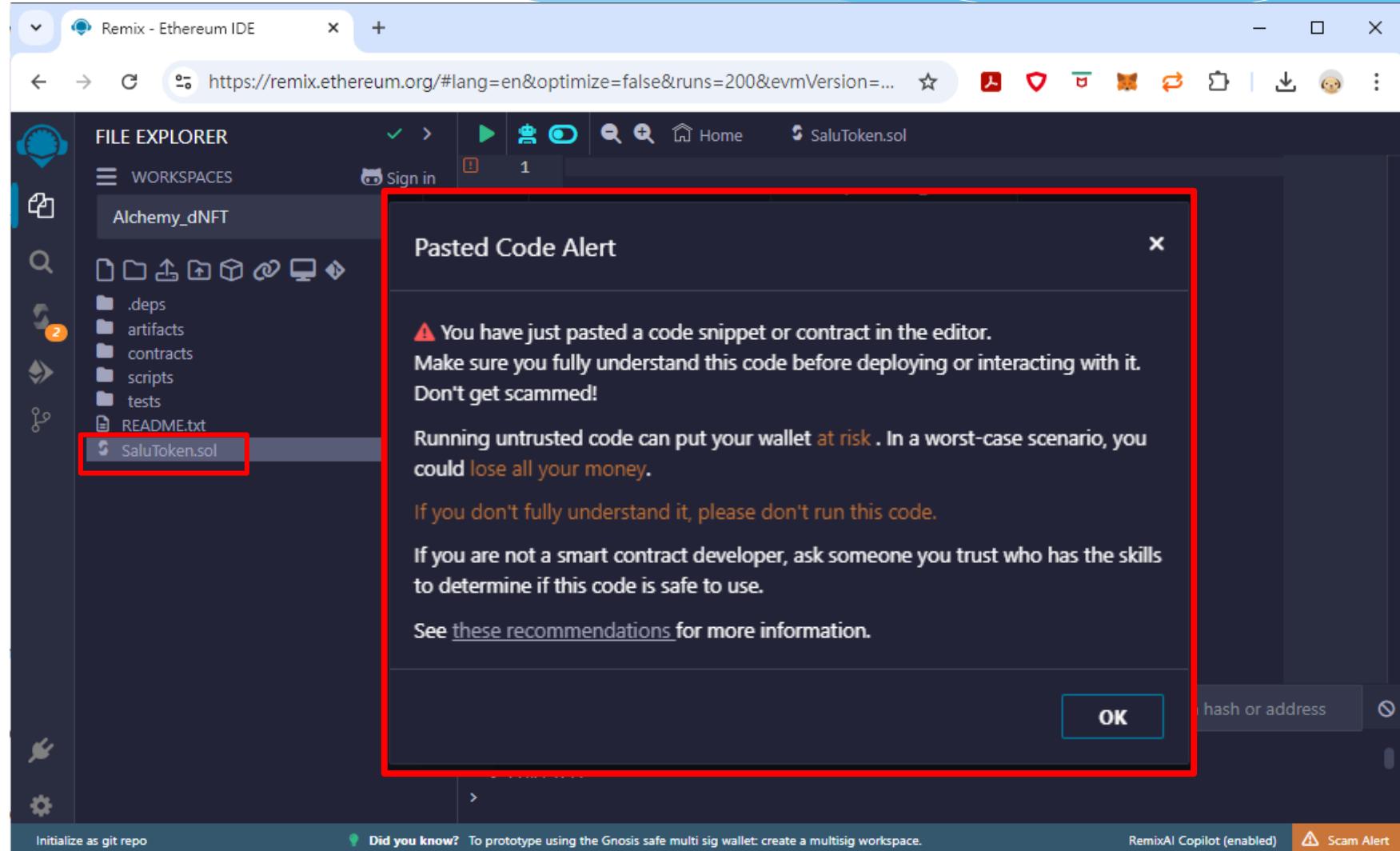


The screenshot shows the Remix Ethereum IDE interface. On the left, the **FILE EXPLORER** panel displays a workspace named **Alchemy_dNFT**. A red box highlights the copy icon (a clipboard with a plus sign) in the toolbar of this panel. The main workspace area features the **REMIX** logo and the tagline "The Native IDE for Web3 Development". It includes sections for "Start Coding" (with buttons for **ERC20**, **ZK Semaphore**, **Uniswap V4 Hooks**, **NFT / ERC721**, and **MultiSig**), "Recent Workspaces", and a "Featured" section with a "What's New" update for v0.54.0. The bottom of the interface includes a footer with links for **Learneth**, **RemixAI Copilot**, and **Scam Alert**.

SaluToken.sol



SaluToken.sol



SaluToken.sol

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

- Toolbar:** Includes standard browser controls (Back, Forward, Refresh), a search bar with the URL <https://remix.ethereum.org/#lang=en&optimize=false&runs=200&evmVersion=null>, and various tool icons.
- File Explorer:** Shows the project structure:
 - WORKSPACES: Alchemy_dNFT
 - folders: .deps, artifacts, contracts, scripts, tests
 - files: README.txt, SaluToken.sol
- Code Editor:** Displays the Solidity code for `SaluToken.sol`. The code defines a new ERC20 token named "SaluToken".

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 100000000 * (10 ** uint256(decimals())));
    }
}
```

A red box highlights the entire code block.
- Bottom Panel:** Includes a transaction history section with items like `ethers.js` and `sol-gpt <your Solidity question here>`, a command input field, and status indicators for RemixAI Copilot (enabled) and Scam Alert.

Compile Smart Contract

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

- SOLIDITY COMPILER:** Version 0.8.26+commit.8a97fa7a.
- Auto compile:** Checked (highlighted with a red box).
- Compile SaluToken.sol:** Button highlighted with a red box.
- Contract:** SaluToken (SaluToken.sol) selected.
- Remix Analysis:** Run Remix Analysis button.
- Compiler Output:** Shows the Solidity code for SaluToken.sol.
- Bottom Bar:** Initialize as git repo, Did you know? (Gnosis safe multi sig wallet), RemixAI Copilot (enabled), Scam Alert.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 100000000 * (10 ** uint256(decimals())));
    }
}
```

Deploy an ERC-20 Token

The screenshot shows the Remix Ethereum IDE interface with the following components:

- Left Sidebar:** Contains icons for Deploy & Run Transactions, Environment, Account, Gas Limit (radio buttons for Estimated Gas and Custom), Value (0 Wei), and CONTRACT (SaluToken - SaluToken.sol). The CONTRACT icon is highlighted with a red box.
- Middle Panel:** Displays the Solidity code for the SaluToken contract:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts/token/ERC20/extensions/ERC20Burnable.sol";
import "@openzeppelin/contracts/access/Ownable.sol";

contract SaluToken is ERC20, ERC20Burnable, Ownable {
    constructor() ERC20("SaluToken", "SLU") {
        _mint(msg.sender, 100000000 * 10 ** 18);
    }

    function mint(address to, uint256 amount) public onlyOwner {
        _mint(to, amount);
    }

    function burn(uint256 amount) public {
        _burn(msg.sender, amount);
    }
}
```
- Right Panel:** Shows the MetaMask extension interface with a lock icon, a password field containing five dots, and a blue "Unlock" button. It also includes links for "Forgot password?", "需要帮助？聯繫 MetaMask 支援", and "Scam Alert".
- Header:** Shows the title "Remix - Ethereum IDE" and the URL "https://remix.ethereum.org/#lang=en&optimize=false&runs=200&evmVersion=nul...".

Deploy and Run Transaction

The screenshot shows the Remix Ethereum IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' sidebar is visible, with the 'Injected Provider - MetaMask' section highlighted by a red box. The 'ACCOUNT' dropdown shows 'Sepolia (11155111) network'. Below it, the 'GAS LIMIT' is set to 'Estimated Gas' with a value of '3000000'. The 'CONTRACT' dropdown is also highlighted by a red box and shows 'SaluToken - SaluToken.sol'. At the bottom of the sidebar, there are buttons for 'Deploy' and 'Publish to IPFS'. The main area displays the Solidity code for the SaluToken contract:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts/token/ERC20/extensions/ERC20Burnable.sol";
import "@openzeppelin/contracts/access/Ownable.sol";

contract SaluToken is ERC20, ERC20Burnable, Ownable {
    constructor() ERC20("SaluToken", "ST") {}
    _mint(msg.sender, 100000000 * (10**18));
}
```

Below the code, there's a note about accessible libraries: `web3.js`, `ethers.js`, and `sol-gpt <your Solidity question here>`. A prompt at the bottom says 'Type the library name to see available commands.'

To the right of the IDE, a MetaMask wallet interface is shown. It displays a balance of '1.5297 SepoliaETH' and a 'Portfolio' section. Below this are buttons for 'Buy & Sell', 'Swap', 'Bridge', '發送' (Send), and '接收' (Receive). The '交易紀錄' (Transactions) tab is selected, showing two entries from October 4, 2024:

Date	Action	Status	Amount
Oct 4, 2024	Set	已確認	-0 SepoliaETH -0 SepoliaETH
Oct 3, 2024	部署合約	已確認	-0 SepoliaETH -0 SepoliaETH

Deploy and Run Transaction

The screenshot shows the Remix Ethereum IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' sidebar is visible, with the 'Injected Provider - MetaMask' section highlighted. A red box surrounds the 'Sepolia (11155111) network' dropdown and the account address '0x4CE...6ae71 (1.52968398290716)'. Below these, the 'Deploy' button is also highlighted with a red box. The main code editor displays the Solidity contract code for SaluToken.

Contract Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts/access/Ownable.sol";

contract SaluToken is ERC20, Ownable {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 100000000 * (10**18));
    }
}
```

The right side of the interface shows a wallet summary for 'Salu 1' with the address '0x4CE13...6ae71'. It displays a balance of '1.5297 SepoliaETH'. Below this, a transaction history table lists two entries:

Date	Action	Status	Value
Oct 4, 2024	Set	已確認	-0 SepoliaETH -0 SepoliaETH
Oct 3, 2024	部署合約	已確認	-0 SepoliaETH -0 SepoliaETH

Deploy and Run Transaction

The screenshot shows the Remix Ethereum IDE and the MetaMask extension integrated in a browser window.

Remix Ethereum IDE:

- DEPLOY & RUN TRANSACTIONS:**
 - ENVIRONMENT:** Sepolia (11155111) network
 - ACCOUNT:** 0x4CE...6ae71 (1.52968398290716)
 - GAS LIMIT:** Estimated Gas (3000000)
 - VALUE:** 0 Wei
- CONTRACT:** SaluToken - SaluToken.sol (evm version: cancun)
- Buttons:** Deploy, Publish to IPFS

Code Editor: Solidity code for SaluToken contract:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 100000000 * (10 ** 18));
    }
}
```

MetaMask Extension:

- Network:** Sepolia (highlighted with a red box)
- Accounts:** Salu1 (highlighted with a red box)
- Buttons:** 建立新合約 (Create New Contract) (highlighted with a red box)
- Information:**
 - 部署合約 (Deploy Contract)
 - 詳情 (Details), HEX (Hexadecimal)
 - Estimated changes: No changes predicted for your wallet
 - Estimated fee: 0.01244785 SepoliaETH (Market ~60 sec), Maxfee: 0.01630247 SepoliaETH
- Buttons at the bottom:** 拒絕 (Reject), 確認 (Confirm) (highlighted with a red box)

Bottom Bar: Initialize as git repo, Did you know? To prototype using the Gnosis safe multi sig wallet: create a multisig workspace.

Deploy and Run Transaction

The screenshot shows two windows side-by-side. On the left is the Remix Ethereum IDE. In the center-left panel, the code for `SaluToken.sol` is displayed:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "SALU") {
        _mint(msg.sender, 100000000);
    }
}
```

The right panel of the IDE shows deployment details: "Deployed at 0x4b37bb9f122192ce09e19f53e022960f5b4f4c22e65fbe8f6dee114ca09dd9ec". A red arrow points from this address to the Etherscan link in the bottom status bar.

On the right is a MetaMask wallet interface. It displays a balance of **1.5181 SepoliaETH**. Below the balance are buttons for **Buy & Sell**, **Swap**, **Bridge**, **發送** (Send), and **接收** (Receive). At the bottom, under the **交易紀錄** tab, there are two entries:

- Oct 17, 2024**: 部署合約 已確認 (Deployment confirmed)
- Oct 4, 2024**: Set 已確認 (Set confirmed)

The URL in the bottom status bar is <https://sepolia.etherscan.io/tx/0x4b37bb9f122192ce09e19f53e022960f5b4f4c22e65fbe8f6dee114ca09dd9ec>.

Deploy and Run Transaction

The screenshot shows two windows side-by-side. On the left is the Remix Ethereum IDE. In the center-left panel, the code for `SaluToken.sol` is displayed:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "SALU") {
        _mint(msg.sender, 100000000);
    }
}
```

The right panel of the IDE shows deployment details: "Deployed at 0x4b37bb9f122192ce09e19f53e022960f5b4f4c22e65fbe8f6dee114ca09dd9ec". A red arrow points from this address to the Etherscan transaction link at the bottom of the page.

On the right is a MetaMask wallet interface for account `0x4CE13...6ae71`. It displays a balance of **1.5181 SepoliaETH** and a transaction history tab labeled **交易紀錄**.

Date	Type	Status
Oct 17, 2024	部署合約	已確認
Oct 4, 2024	Set	已確認

Deploy and Run Transaction

The screenshot shows the Remix Ethereum IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' sidebar is visible, showing the environment as 'Injected Provider - MetaMask' connected to the Sepolia network, account 0x4CE...6ae71, and a gas limit of 3,000,000. The central code editor displays the Solidity contract SaluToken.sol:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "SALU") {
        _mint(msg.sender, 100000000 * (10 ** 18));
    }
}
```

Below the code, a transaction receipt is shown with a green checkmark, indicating success. The transaction details include:

- Type: creation of SaluToken pending...
- block: 6892662
- txIndex: 54
- from: 0x4ce...6ae71
- value: 0 wei
- data: 0x608...a0033
- logs: 1 hash: 0x4b37bb9f122192ce09e19f53e022960f5b4f4c22e65fbe8f6dee114ca09dd9ec
- status: 0
- Nonce: 0
- Listen on all transactions: checked

A red box highlights the 'view on etherscan' link at the bottom of the transaction details.

On the right, a modal window titled '部署合約' (Deploy Contract) displays the deployment status and transaction details. A red box highlights the 'View on block explorer' button. The transaction details are as follows:

項目	內容
Status	已確認
來源帳戶	0x4CE13...6...
目的帳戶	建立新合約
交易	
Nonce	5
數量	-0 SepoliaETH
Gas 上限 (單位)	956440
Gas 用量 (單位)	947751
Base fee (GWEI)	10.753596893
Priority fee (GWEI)	1.5
Total gas fee	0.011613 SepoliaETH
Max fee per gas	0.000000017 SepoliaETH
總量	0.01161336 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/0x4b37bb9f122192ce09e19f53e022960f5b4f4c22e65fbe8f6dee114ca09dd9ec>.

The transaction details are as follows:

- Transaction Hash: 0x4b37bb9f122192ce09e19f53e022960f5b4f4c22e65fbe8f6dee114ca09dd9ec
- Status: Success (highlighted with a red box)
- Block: 6892662 (46 Block Confirmations)
- Timestamp: 11 mins ago (Oct-17-2024 02:08:00 PM UTC)
- Transaction Action: Call 0x60806040 Method by 0x4CE135aB...64E06ae71
- From: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71
- Interacted With (To): [0x225b8f7ec3f056cf68fd5e9fef82584e85a660a5 Created] (highlighted with a red box)

Smart Contract Information

Contract Address 0x225b8f7ec3f056cf68fd5e9fef82584e85a660a5

Search by Address / Txn Hash / Block / Token

Contract 0x225b8F7ec3F056CF68FD5E9FEf82584E85A660A5

More Info

CONTRACT CREATOR
0x4CE135aB...64E06ae71 at txn 0x4b37bb9f12...

TOKEN TRACKER
SaluToken (ST)

Transactions

Latest 1 from a total of 1 transactions

Transaction Hash	Method	Block	Age	From	To	Amount	Txn Fee
0x4b37bb9f12...	0x60806040	6892662	15 mins ago	0x4CE135aB...64E06ae71	IN Contract Creation	0 ETH	0.01161335

Smart Contract Information

The screenshot shows a browser window displaying the SaluToken (ST) Token Tracker on the Sepolia Testnet. The URL is <https://sepolia.etherscan.io/token/0x225b8f7ec3f056cf68fd5e9fef82584e85a660a5>.

Token Details:

- MAX TOTAL SUPPLY:** 100,000,000 ST
- HOLDERS:** 1
- TOTAL TRANSFERS:** 1

Market Information:

- ONCHAIN MARKET CAP:** \$0.00
- CIRCULATING SUPPLY MARKET CAP:** -

Other Info:

- TOKEN CONTRACT (WITH 18 DECIMALS):** 0x225b8f7ec3f056cf68fd5e9fef82584e85a660a5

Transactions:

A total of 1 transaction found.

Transaction Hash	Method	Block	Age	From	To	Amount
0x4b37bb9f122...	0x60806040	6892662	1 hr ago	0x00000000...000000000	0x4CE135aB...64E06ae71	100,000,000

Deploy and Run Transaction

The screenshot shows the Remix Ethereum IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' sidebar is visible, showing deployment settings: Value set to 0 Wei, Contract selected as 'SaluToken - SaluToken.sol', EVM version set to 'cancun', and a Deploy button highlighted with a red box. Below this, sections for 'Transactions recorded' and 'Deployed Contracts' are shown, with the 'Deployed Contracts' section listing 'ALUTOKEN AT 0x225...660A5 (BLOCKCHAIN)'.

The main workspace displays the Solidity code for the 'SaluToken' contract:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }
}
```

At the bottom of the workspace, a transaction receipt is displayed:

Type the library name to see available commands.
creation of SaluToken pending...

[view on etherscan](#)

[block:6892662 txIndex:54] from: 0x4ce...6ae71 to: SaluToken.(constructor)
value: 0 wei data: 0x608...a0033 logs: 1 hash: 0xb96...41c23

Buttons for 'Debug' and a dropdown menu are also present at the bottom right.

Deploy and Run Transaction

The screenshot shows the Remix Ethereum IDE interface. On the left, the sidebar displays the "DEPLOY & RUN TRANSACTIONS" section, which lists the "Deployed Contracts" as "SALUTOKEN AT 0x225...660A5 (BLOCKCHAIN)". Below this, a list of functions is shown, each with a description and a copy icon:

- approve address spender, uint256 value
- transfer address to, uint256 value
- transferFrom address from, address to, uint256 value
- allowance address owner, address spender
- balanceOf address account
- decimals
- name
- symbol
- totalSupply

A red box highlights the first three functions: approve, transfer, and transferFrom.

The main workspace shows the Solidity code for the SaluToken contract:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }
}
```

At the bottom, the transaction history shows a pending transaction for the constructor:

[block:6892662 txIndex:54] from: 0x4ce...6ae71 to: SaluToken.(constructor) value: 0 wei data: 0x608...a0033 logs: 1 hash: 0xb96...41c23

Call totalSupply()

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

- Deploy & Run Transactions:** Shows a deployed contract named "SALUTOKEN AT 0x225...660A5 (BLOCKCHAIN)".
- Balance:** 0 ETH.
- Low level interactions:** A red box highlights the "totalSupply" button, which is currently selected. Below it, the output shows: "0: uint256: 10000000000000000000000000000000".
- Contract Code:** SaluToken.sol

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }
}
```

- Transactions:** A red box highlights a transaction log: "[block:6892662 txIndex:54] from: 0x4ce...6ae71 to: SaluToken.(constructor) value: 0 wei data: 0x608...a033 logs: 1 hash: 0xb96...41c23". Below it, another log shows a call to "totalSupply": "call to SaluToken.totalSupply".
- Logs:** A red box highlights a log entry: "CALL [call] from: 0x4CE135aB2e88e482D1688011ba9415D64E06ae71 to: SaluToken.totalSupply() data: 0x181...60ddd".

Call symbol()

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

- Contract Name:** SaluToken.sol
- Contract Source Code:**

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }
}
```
- Deploy & Run Transactions:** Shows a deployed contract named SALUTOKEN AT 0X225...660A5 (BLOCKCHAIN) with a balance of 0 ETH. It lists several functions:
 - approve: address spender, uint256 value
 - transfer: address to, uint256 value
 - transferFrom: address from, address to, uint256 value
 - allowance: address owner, address spender
 - balanceOf: address account
 - decimals
 - name
 - symbol** (highlighted with a red box)
 - totalSupply
- Logs:** Displays two transaction logs, both highlighted with a red box:
 - [call] from: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 to: SaluToken.totalSupply() data: 0x181...60ddd
 - [call] from: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 to: SaluToken.symbol() data: 0x95d...89b41

Call name()

The screenshot shows the Remix Ethereum IDE interface. On the left, the sidebar has icons for deploying contracts, running transactions, and interacting with deployed contracts. The main area is divided into three sections: 'DEPLOY & RUN TRANSACTIONS' (left), 'Contract Source Code' (center), and 'Transactions' (right).

Deploy & Run Transactions: Shows a deployed contract named 'SALUTOKEN AT 0x225...660A5 (BLOCKCHAIN)'. It displays the balance as '0 ETH' and lists several functions: approve, transfer, transferFrom, allowance, balanceOf, decimals, name, symbol, and totalSupply. The 'name' button is highlighted with a red box.

Contract Source Code: The code for the SaluToken contract is shown in Solidity:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }
}
```

Transactions: The transaction history shows two calls to the 'name' function:

- [call] from: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 to: SaluToken.name()
data: 0x95d...89b41
- [call] from: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 to: SaluToken.name()
data: 0x06f...dde03

At the bottom, there are buttons for 'Initialize as git repo', 'Did you know? To prototype using the Gnosis safe multi sig wallet: create a multisig workspace.', 'RemixAI Copilot (enabled)', and 'Scam Alert'.

Call decimals()

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

- Top Bar:** Shows the title "Remix - Ethereum IDE" and a URL bar with the address "https://remix.ethereum.org/#lang=en&optimize=false&runs=200&evmVersion=null&version=soljs...".
- Left Sidebar:** Contains icons for deploying contracts, running transactions, and interacting with the blockchain.
- Middle Left Panel:** Titled "DEPLOY & RUN TRANSACTIONS" and "Deployed Contracts 1". It shows a single deployed contract entry: "SALUTOKEN AT 0X225...660A5 (BLOCKCHAIN)". Below it is a table of functions with their descriptions and parameter types:

approve	address spender, uint256 value
transfer	address to, uint256 value
transferFrom	address from, address to, uint256 value
allowance	address owner, address spender
balanceOf	address account
decimals	0: uint8: 18
name	0: string: SaluToken
symbol	0: string: ST
totalSupply	
- Right Panel:** Titled "SaluToken.sol X". It displays the Solidity code for the SaluToken contract:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }
}
```
- Bottom Panel:** Shows transaction logs. One log is highlighted with a red box:

```
call [call] from: 0x4CE135aB2eB8e482D1688011ba9415D64E06ae71 to: SaluToken.name()
data: 0x313...ce567
```

View & Publish

Contract Address 0x225b8f7e

https://sepolia.etherscan.io/address/0x225b8f7ec3f056cf68fd5e9fef82584e85a660a5#code

Sepolia Testnet

Search by Address / Txn Hash / Block / Token

Overview

ETH BALANCE

0 ETH

More Info

CONTRACT CREATOR

0x4CE135aB...64E06ae71 at txn 0xb37bb9f12...

TOKEN TRACKER

SaluToken (ST)

Multichain Info

N/A

Transactions Token Transfers (ERC-20) **Contract** Events

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

Decompile Bytecode Switch to Opcodes View Similar Contracts

```
0x608060405234801561000f575f80fd5b5060043610610091575f3560e01c8063313ce56711610064578063313ce5671461013157806370a082311461014f57806395d89b411461017f578063a9059cbb1461019d578063dd62ed3e146101cd57610091565b806306fdde0314610095578063095ea7b3146100b357806318160ddd146100e357806323b872dd14610101575b5f80fd5b61009d6101fd565b6040516100aa9190610a5a565b60405180910390f35b6100cd60048036038101906100c89190610b0b565b61028d565b6040516100da9190610b63565b60405180910390f35b6100eb6102af565b6040516100f89190610b8b565b60405180910390f35b61011b60048036038101906101169190610ba4565b6102b8565b6040516101289190610b63565b60405180910390f35b6101396102e6565b6040516101469190610c0f565b60405180910390f35b61016960048036038101906101649190610c28565b6102ee565b6040516101769190610b8b565b60405180910390f35b610187610333565b6040516101949190610a5a565b60405180910390f35b6101b760048036038101906101b29190610b0b565b6103c3565b6040516101c49190610b63565b60405180910390f35b6101e760048036038101906101e29190610c53565b6103e5565b6040516101f49190610b8b565b60405180910390f35b60606003805461020c90610cbe565b80601
```

View & Publish Contract Source Code

The screenshot shows a web browser window titled "Verify & Publish Contract Sou" with the URL <https://sepolia.etherscan.io/verifyContract?a=0x225b8f7ec3f056cf68fd5e9fef82584e85a660a5>. 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 a two-step process: "Enter Contract Details" (Step 1) and "Verify & Publish" (Step 2). The "Enter Contract Details" step is active, showing fields for Contract Address (0x225b8f7ec3f056cf68fd5e9fef82584e85a660a5), Compiler Type (Solidity (Single file)), Compiler Version (v0.8.26+commit.8a97fa7a), Open Source License Type (3) MIT License (MIT), and a checkbox for agreeing to terms of service. The "Continue" button is highlighted with a red box.

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](#).

1 Enter Contract Details — 2 Verify & Publish

Please enter the Contract Address you would like to verify
0x225b8f7ec3f056cf68fd5e9fef82584e85a660a5

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 ⓘ
3) MIT License (MIT)

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=0x225b8f7ec3f056cf68fd5e9fef82584e85a660a5&c=v0.8.2...>. The page is titled "Verify & Publish Contract Source Code". It explains that source code verification provides transparency and matches uploaded code with the blockchain. It's a simple interface for verifying smart contracts. The process is divided into two steps: "Enter Contract Details" (step 1) and "Verify & Publish" (step 2). The "Verify & Publish" step is currently active. A red box highlights the "Contract Address" field, which contains the value `0x225b8f7ec3f056cf68fd5e9fef82584e85a660a5`. Below it, the "Compiler Type" is listed as "SINGLE FILE / CONCATENATED METHOD" and the "Compiler Version" as "v0.8.26+commit.8a97fa7a".

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: `0x225b8f7ec3f056cf68fd5e9fef82584e85a660a5`

Compiler Type: SINGLE FILE / CONCATENATED METHOD

Compiler Version: v0.8.26+commit.8a97fa7a

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=0x225b8f7ec3f056cf68fd5e9fef82584e85a660a5&c=v0.8.2...>. The page displays a Solidity contract code for "SaluToken" based on the ERC20 standard. A red box highlights the license header and the constructor logic. A red arrow points from a callout box on the right to the highlighted code area. The callout box contains the text: "請把剛才的檔案 SaluToken.sol 貼上來。". Below the code editor, there are sections for "Advanced Configuration" and "License Type". The "License Type" dropdown is set to "3) MIT License (MIT)".

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 100000000 * (10 ** uint256(decimals())));
    }
}
```

請把剛才的檔案
SaluToken.sol
貼上來。

Advanced Configuration

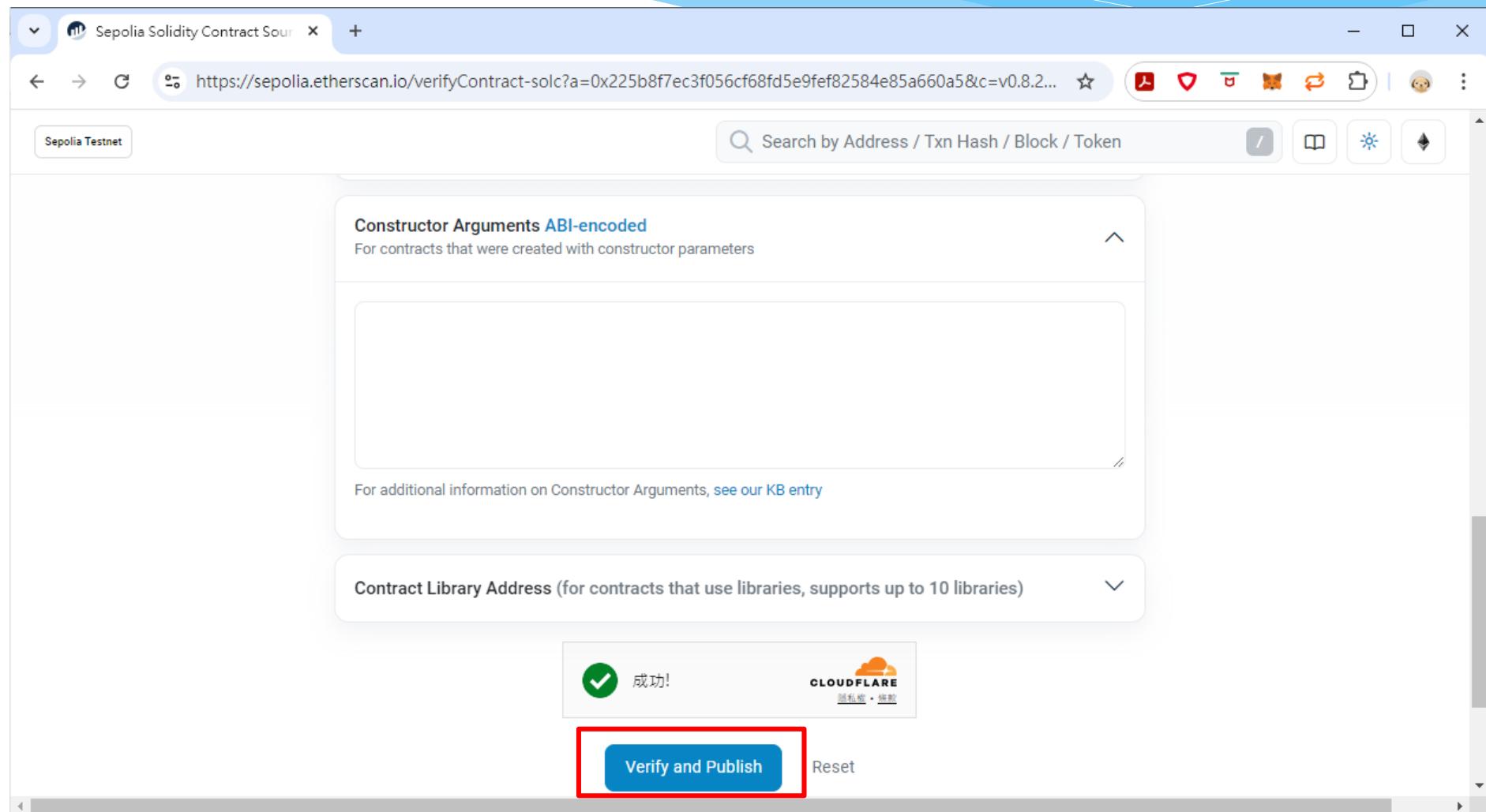
Optimization ⓘ Runs (Optimizer) ⓘ EVM Version to target ⓘ

No 200 default (compiler defaults)

License Type ⓘ

3) MIT License (MIT)

View & Publish Contract Source Code



View & Publish Contract Source Code

Sepolia Solidity Contract Sourc... <https://sepolia.etherscan.io/verifyContract-solc?a=0x225b8f7ec3f056cf68fd5e9fef82584e85a660a5&c=v0.8.26%2bcommit.8a97fa7a...>

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

① Learn about contract verification [troubleshooting](#) and [common errors](#).

Compiler Output

Compiler Warning(s):

```
ParserError: Source "@openzeppelin/contracts/token/ERC20/ERC20.sol" not found: File import callback not supported
--> myc:4:1:
 |
4 | import "@openzeppelin/contracts/token/ERC20/ERC20.sol"
 | ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

1. Case 1因為有import OpenZeppelin的檔案，如果沒有一併把相關檔案上傳，則做Verify時會有問題。
2. 使用以下Case 2的Flatten方式可以簡化import程序，並使Verify正常無誤。

Case 2

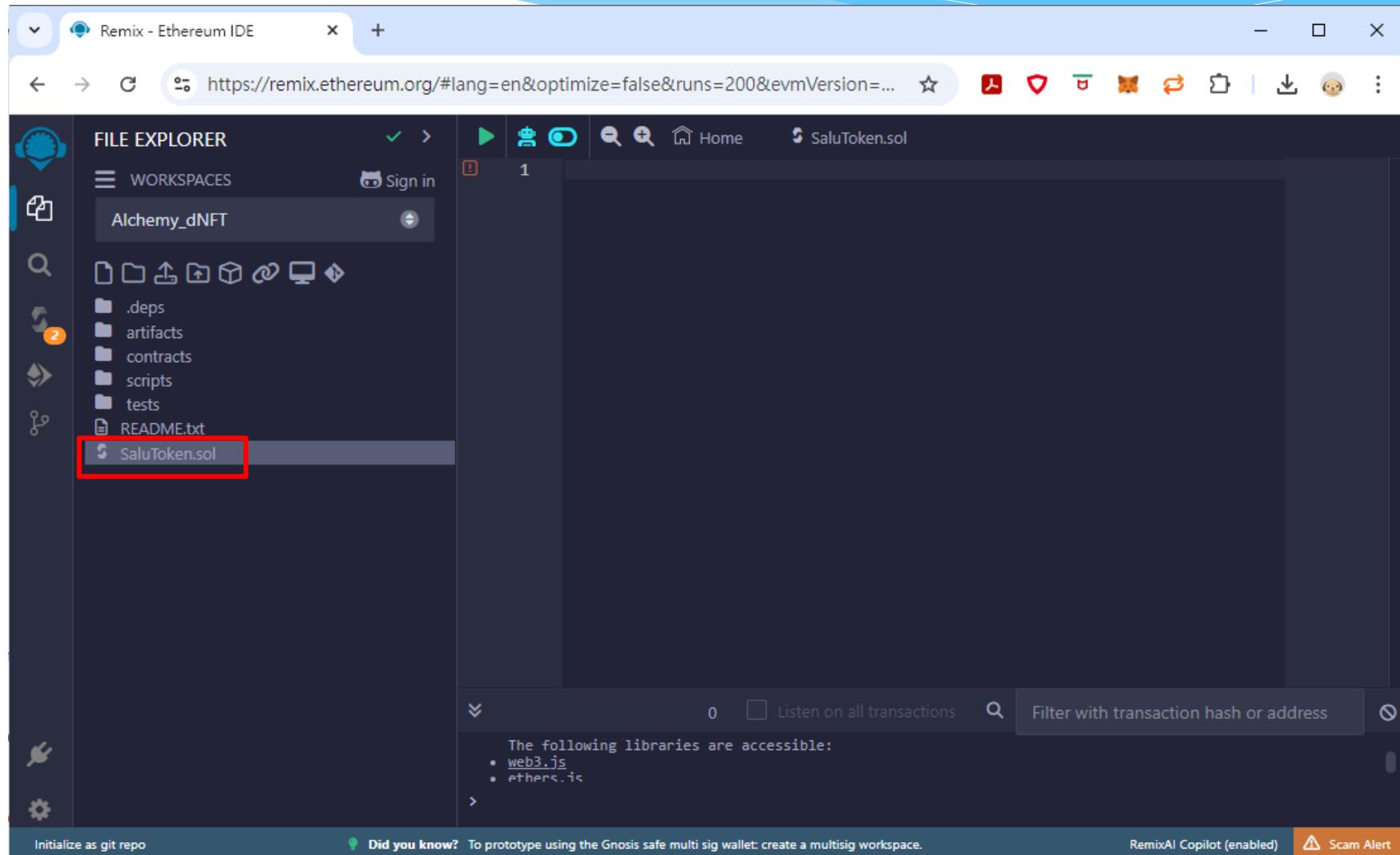
Create an ERC-20 Token

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 folder icon in the file list, which contains subfolders like `.deps`, `contracts`, `scripts`, `tests`, and a file `README.txt`. The main central area features the **REMIX** logo and the tagline *The Native IDE for Web3 Development.* Below it, there's a search bar for documentation and several buttons for **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
- Added 'sendRawTransaction' API to remix-simulator

 There's also a **What's New** button and a **Read More** link. At the bottom, there's a **Recent Workspaces** section, a note about accessible libraries (`web3.js`, `ethers.js`), and a **Featured Plugins** section. The footer includes links for **Initialize as git repo**, **Did you know?** (Solidity basics), **RemixAI Copilot (enabled)**, and **Scam Alert**.

SaluToken.sol



SaluToken.sol

The screenshot shows the Remix Ethereum IDE interface. On the left is the File Explorer sidebar with a dark theme. A red box highlights the 'SaluToken.sol' file in the contracts folder. In the center, a modal window titled 'Pasted Code Alert' contains a warning message. The message reads:

⚠ 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.

The 'OK' button at the bottom right of the modal is highlighted with a blue border. The background of the IDE shows other workspace files like '.deps', 'artifacts', 'contracts', 'scripts', 'tests', and 'README.txt'. The top bar shows the URL 'https://remix.ethereum.org/#lang=en&optimize=false&runs=200&evmVersion=...', and the bottom bar includes status messages like 'Did you know?', 'RemixAI Copilot (enabled)', and 'Scam Alert'.

SaluToken.sol

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

- Toolbar:** Includes standard browser controls (Back, Forward, Refresh), a search bar with the URL <https://remix.ethereum.org/#lang=en&optimize=false&runs=200&evmVersion=null>, and various tool icons.
- File Explorer:** Shows the workspace structure:
 - WORKSPACES: Alchemy_dNFT
 - Folders: .deps, artifacts, contracts, scripts, tests
 - Files: README.txt, SaluToken.sol
- Code Editor:** Displays the Solidity code for `SaluToken.sol`. The code defines a new ERC20 token named "SaluToken".

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 100000000 * (10 ** uint256(decimals())));
    }
}
```

A red box highlights the entire code block.
- Bottom Panel:** Includes a transaction history section with items like `ethers.js` and `sol-gpt <your Solidity question here>`, a command input field, and status indicators for RemixAI Copilot (enabled) and Scam Alert.

Flatten SaluToken.sol

The screenshot shows the Remix Ethereum IDE interface. The left sidebar contains a 'FILE EXPLORER' with a 'WORKSPACES' section showing 'Alchemy_dNFT'. The main workspace shows a Solidity file named 'SaluToken.sol' with the following code:

```
// SPDX-License-Identifier: MIT
import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }
}
```

A red box highlights the SPDX license declaration at the top of the code. A blue circle labeled '1' is positioned over the 'SaluToken.sol' tab in the file list. A blue circle labeled '2' is positioned over the code area. A red box labeled 'Copy' is positioned over the code area, indicating where to copy the flattened contract.

Flatten SaluToken.sol

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

- FILE EXPLORER:** Displays the workspace "Alchemy_dNFT" with files like .deps, artifacts, SaluToken_metadata.json, SaluToken.json, contracts, scripts, tests, and README.txt.
- Code Editor:** The file `SaluToken.sol` is open, containing the following Solidity code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

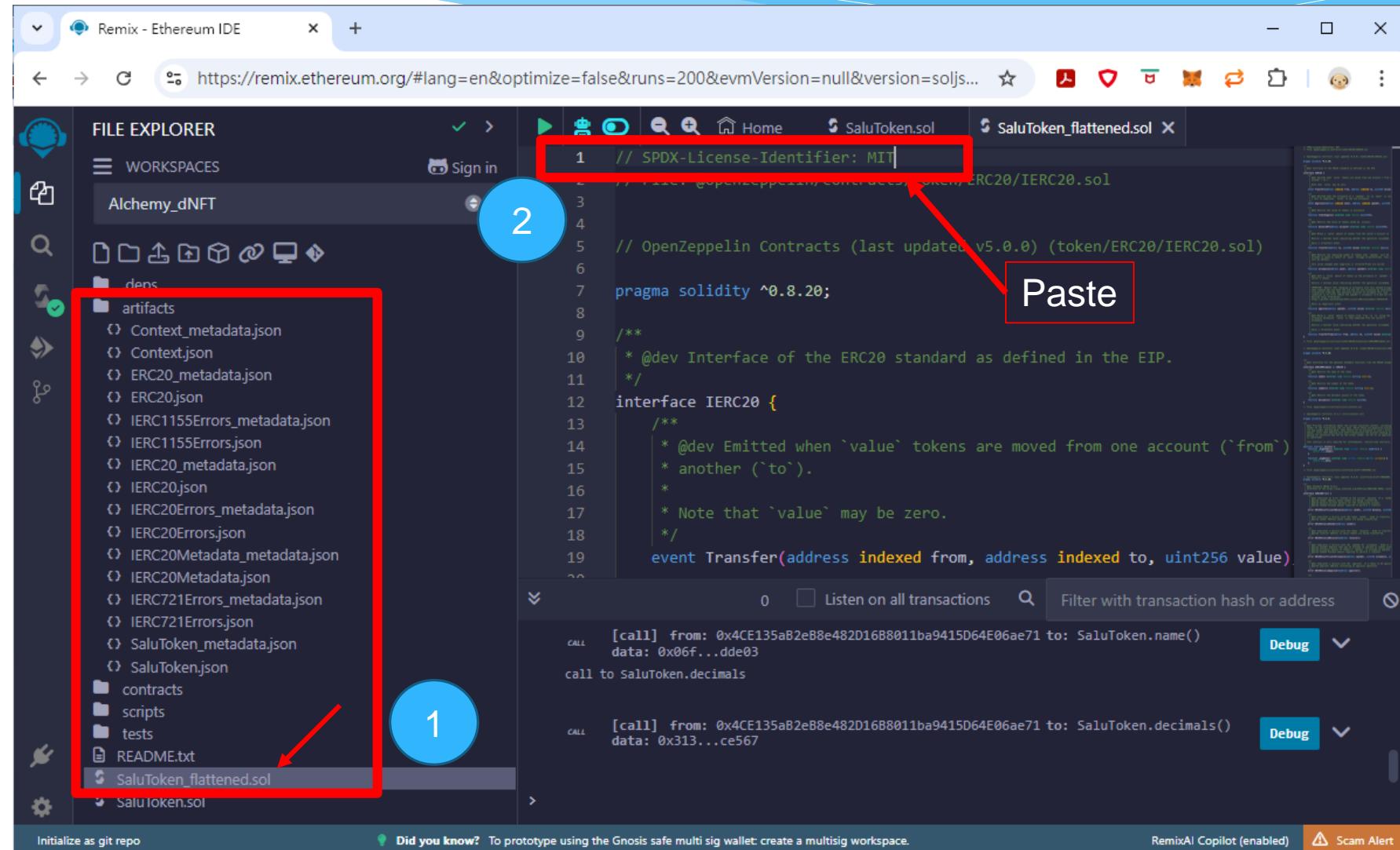
contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }
}
```
- Context Menu:** A context menu is open over the `SaluToken.sol` file in the file explorer, with option `Flatten` highlighted.
- Annotations:** Two numbered circles highlight the steps: 1 points to the `SaluToken.sol` file in the file explorer, and 2 points to the `Flatten` option in the context menu.
- Bottom Bar:** Includes links for "Initialize as git repo", "Did you know?", "RemixAI Copilot (enabled)", and "Scam Alert".

Flatten SaluToken.sol

The screenshot shows the Remix Ethereum IDE interface. The left sidebar contains a 'FILE EXPLORER' with a 'WORKSPACES' section showing 'Alchemy_dNFT'. Below it are 'artifacts' (Context_metadata.json, Context.json, ERC20_metadata.json, ERC20.json, IERC1155Errors_metadata.json, IERC1155Errors.json, IERC20_metadata.json, IERC20.json, IERC20Errors_metadata.json, IERC20Errors.json, IERC20Metadata_metadata.json, IERC20Metadata.json, IERC721Errors_metadata.json, IERC721Errors.json, SaluToken_metadata.json, SaluToken.json, contracts, scripts, tests, README.txt, SaluToken_flattened.sol) and a 'contracts' section with SaluToken.sol. A red box highlights the 'artifacts' section, and a blue circle labeled '1' points to SaluToken_flattened.sol. The main code editor shows SaluToken.sol being flattened into SaluToken_flattened.sol. A red box highlights the flattened code, and a blue circle labeled '2' points to the flattened file. A red arrow points from the flattened code back to the 'artifacts' section. A message 'Missing SPDX-License' is displayed in a red box. The bottom status bar includes 'Initialize as git repo', 'Did you know? To prototype using the Gnosis safe multi sig wallet: create a multisig workspace.', 'RemixAI Copilot (enabled)', and 'Scam Alert'.

```
// OpenZeppelin Contracts (last updated v5.0.0) (contracts/math/Math.sol)  
// This file is part of the OpenZeppelin Contracts package. The source code  
// is available at https://github.com/OpenZeppelin/openzeppelin-contracts.  
// SPDX-License-Identifier: MIT  
// SPDX-FileHash: 0x0000000000000000000000000000000000000000000000000000000000000000  
pragma solidity ^0.8.20;  
  
/**  
 * @dev Interface of the ERC20 standard as defined in the EIP.  
 */  
interface IERC20 {  
    /**  
     * @dev Emitted when `value` tokens are moved from one account (`from`) to  
     * another (`to`).  
     *  
     * Note that `value` may be zero.  
     */  
    event Transfer(address indexed from, address indexed to, uint256 value);  
}
```

Flatten SaluToken.sol



Compile Smart Contract

The screenshot shows the Ethereum IDE interface with the following steps highlighted:

- 1** Click the **Compile** icon (a blue gear with a green checkmark) in the sidebar.
- 2** Check the **Auto compile** checkbox.
- 3** Select the **Context (SaluToken_flattened.sol)** from the **CONTRACT** dropdown.
- 4** Click the **Compile SaluToken_flattened.sol** button.

The code editor on the right displays the Solidity source code for `SaluToken.sol` and its flattened version `SaluToken_flattened.sol`. The flattened code includes the OpenZeppelin ERC20 interface and standard.

```
// SPDX-License-Identifier: MIT
// File: @openzeppelin/contracts/token/ERC20/IERC20.sol

// OpenZeppelin Contracts (last updated v5.0.0) (token/ERC20/IERC20.sol)

pragma solidity ^0.8.20;

/**
 * @dev Interface of the ERC20 standard as defined in the EIP.
 */
interface IERC20 {
    /**
     * @dev Returns the amount of tokens in existence.
     */
    function totalSupply() external view returns (uint256);

    /**
     * @dev Returns the token symbol.
     */
    function symbol() external view returns (string memory);

    /**
     * @dev Returns the token name.
     */
    function name() external view returns (string memory);

    /**
     * @dev Returns the decimal places.
     */
    function decimals() external view returns (uint8);

    /**
     * @dev Moves `amount` tokens from the caller's account to `recipient`.
     *
     * Requirements:
     *
     * - the caller must have at least `amount` tokens.
     */
    function transfer(address recipient, uint256 amount) external returns (bool);

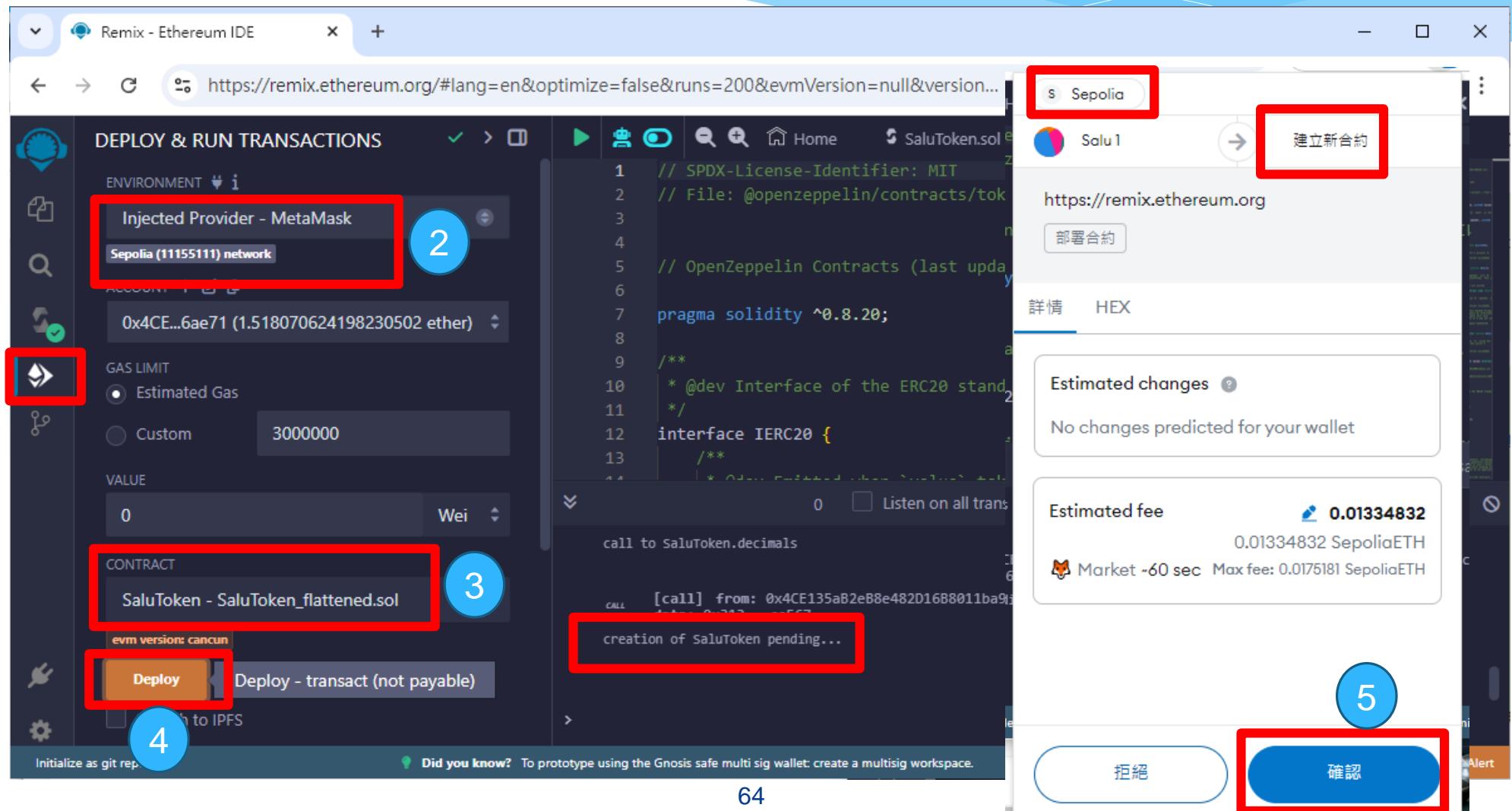
    /**
     * @dev Returns true if the caller has `amount` tokens allowed to be spent by the spender.
     */
    function allowance(address owner, address spender) external view returns (uint256);

    /**
     * @dev Sets `amount` as the allowance of `spender` over the caller's tokens.
     *
     * Requirements:
     *
     * - if the call succeeds, the spender's allowance for the caller will be reset to `0`.
     */
    function approve(address spender, uint256 amount) external returns (bool);

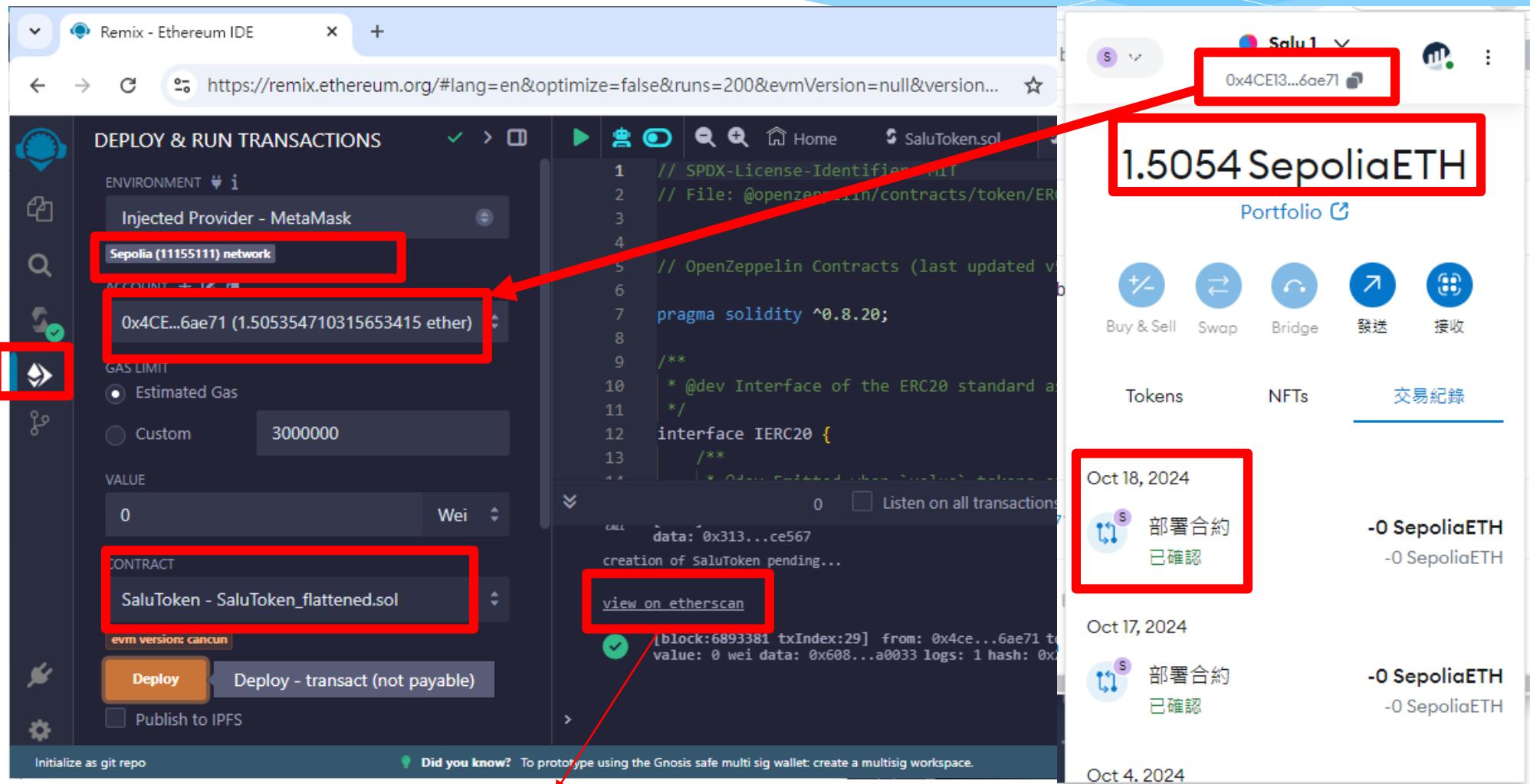
    /**
     * @dev Moves `amount` tokens from `sender` to `recipient`. The total balance of the
     * contract is unchanged.
     *
     * Requirements:
     *
     * - `sender` cannot be the zero address.
     * - `sender` must have a balance of at least `amount`.
     * - the caller must approve this spending beforehand.
     */
    function transferFrom(address sender, address recipient, uint256 amount) external returns (bool);
}
```

The bottom status bar shows: Initialize as git repo, Did you know? To prototype using the Gnosis safe multi sig wallet: create a multisig workspace., RemixAI Copilot (enabled), and Scam Alert.

Deploy an ERC-20 Token



Deploy and Run Transaction



Deploy and Run Transaction

The screenshot shows the Remix Ethereum IDE interface. 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 (1.505354710315653415 ether)'. The gas limit is set to 'Estimated Gas' with a value of 3000000. The contract selected is 'SaluToken - SaluToken_flattened.sol'. At the bottom, there are buttons for 'Deploy' (highlighted in orange), 'Deploy - transact (not payable)', and 'Publish to IPFS'. A red arrow points from the 'Deploy' button to a red box around the 'view on etherscan' link in the transaction output area.

On the right, a modal window titled '部署合約' (Deploy Contract) is displayed. It shows the transaction details:

Status	已確認
來源帳戶	0x4CE13...6ae71
目的帳戶	建立新合約
交易	
Nonce	6
數量	-0 SepoliaETH
Gas 上限 (單位)	956440
Gas 用量 (單位)	947751
Base fee (GWEI)	11.916935337
Priority fee (GWEI)	1.5
Total gas fee	0.012716 SepoliaETH
Max fee per gas	0.000000018 SepoliaETH
總量	0.01271591 SepoliaETH

A red box highlights the 'View on block explorer' button, which is also connected by a red arrow from the 'view on etherscan' link in the main interface. The transaction hash shown in the modal is 0x6893381.

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/0x4e8b9e8c2bd4e366b01b6706b8adb514abf6bf6d40be807bf84fd2633bc6b6f8>.

The transaction details are as follows:

- Transaction Hash: [0x4e8b9e8c2bd4e366b01b6706b8adb514abf6bf6d40be807bf84fd2633bc6b6f8](#)
- Status: Success (highlighted with a red box)
- Block: [6893381](#) | 69 Block Confirmations
- Timestamp: [15 mins ago \(Oct-17-2024 05:03:00 PM UTC\)](#)
- Transaction Action: Call [0x60806040](#) Method by [0x4CE135aB...64E06ae71](#)
- From: [0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71](#)
- Interacted With (To): [\[0xe7ffc96d36d74d6e81f2513b21669893a913205b Created\]](#) (highlighted with a red box)

Smart Contract Information

The screenshot shows a browser window displaying the Sepolia Testnet version of Etherscan for the smart contract at address `0xe7ffc96d36d74d6e81f2513b21669893a913205b`. The page is titled "Contract Address 0xe7ffc96d36d74d6e81f2513b21669893a913205b".

Contract Details: The contract address is highlighted with a red box. Below it, the "More Info" section contains two boxes: "CONTRACT CREATOR" (with address `0x4CE135aB...64E06ae71`) and "TOKEN TRACKER" (with icon and text "SaluToken (ST)"). A red arrow points from the "TOKEN TRACKER" box to the "Transactions" section.

Transactions: The "Transactions" tab is selected. It shows one transaction: `0x4e8b9e8c2b...` (Method: `0x60806040`, Block: `6893381`, Age: 17 mins ago, From: `0x4CE135aB...64E06ae71`, To: `IN`, Amount: 0 ETH, Txn Fee: 0.01271591).

Page Footer: The URL `https://sepolia.etherscan.io/address/0xe7ffc96d36d74d6e81f2513b21669893a913205b` is displayed at the bottom.

Smart Contract Information

SaluToken (ST) Token Tracker | <https://sepolia.etherscan.io/token/0xe7ffc96d36d74d6e81f2513b21669893a913205b>

Search by Address / Txn Hash / Block / Token

Token SaluToken (ST)

ERC-20

Overview

MAX TOTAL SUPPLY
100,000,000 ST

HOLDERS
1

TOTAL TRANSFERS
1

Market

ONCHAIN MARKET CAP
\$0.00

CIRCULATING SUPPLY MARKET CAP
-

Other Info

TOKEN CONTRACT (WITH 18 DECIMALS)
[0xe7ffc96d36d74d6e81f2513b21669893a91...](#)

Transfers Holders Contract

A total of 1 transaction found

Transaction Hash	Method	Block	Age	From	To	Amount
0x4e8b9e8c2b...	0x60806040	6893381	20 mins ago	0x00000000...0000000000	0x4CE135aB...64E06ae71	100,000,000

Deploy and Run Transaction

The screenshot shows the Remix Ethereum IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' sidebar is visible, showing deployment settings: Value set to 0 Wei, Contract selected as 'SaluToken - SaluToken.sol', EVM version set to 'cancun', and a Deploy button highlighted with a red box. Below this, sections for 'Transactions recorded' and 'Deployed Contracts' are shown, with the 'Deployed Contracts' section listing 'ALUTOKEN AT 0x225...660A5 (BLOCKCHAIN)'.

The main workspace displays the Solidity code for the 'SaluToken' contract:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }
}
```

At the bottom of the workspace, a transaction receipt is displayed:

Type the library name to see available commands.
creation of SaluToken pending...

[view on etherscan](#)

[block:6892662 txIndex:54] from: 0x4ce...6ae71 to: SaluToken.(constructor)
value: 0 wei data: 0x608...a0033 logs: 1 hash: 0xb96...41c23

Buttons for 'Debug' and a dropdown menu are also present at the bottom right.

Deploy and Run Transaction

The screenshot shows the Remix Ethereum IDE interface. On the left, the sidebar displays the "DEPLOY & RUN TRANSACTIONS" section, which lists the "Deployed Contracts" and provides a list of functions for the "SALUTOKEN AT 0x225...660A5 (BLOCKCHAIN)" contract. A red box highlights the first five functions: approve, transfer, transferFrom, allowance, and balanceOf. The main panel shows the Solidity code for the SaluToken contract, which is an ERC20 token with a constructor that mints 10,000,000 tokens to the msg.sender.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }
}
```

The bottom right corner of the interface shows a transaction log entry:

- [block:6892662 txIndex:54] from: 0x4ce...6ae71 to: SaluToken.(constructor)
- value: 0 wei
- data: 0x608...a0033
- logs: 1 hash: 0xb96...41c23

Call totalSupply()

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

- Deploy & Run Transactions:** Shows a deployed contract named "SALUTOKEN AT 0x225...660A5 (BLOCKCHAIN)".
- Balance:** 0 ETH.
- Low level interactions:** A red box highlights the "totalSupply" button, which is currently selected. Below it, the output shows: "0: uint256: 10000000000000000000000000000000".
- Contract Code:** SaluToken.sol

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }
}
```

- Transactions:** A red box highlights a transaction log: "[block:6892662 txIndex:54] from: 0x4ce...6ae71 to: SaluToken.(constructor) value: 0 wei data: 0x608...a033 logs: 1 hash: 0xb96...41c23". Below it, another log shows a call to "totalSupply": "call to SaluToken.totalSupply".
- Logs:** A red box highlights a log entry: "CALL [call] from: 0x4CE135aB2e88e482D1688011ba9415D64E06ae71 to: SaluToken.totalSupply() data: 0x181...60ddd".

Call symbol()

The screenshot shows the Remix Ethereum IDE interface. On the left, the "DEPLOY & RUN TRANSACTIONS" sidebar lists deployed contracts, with "SALUTOKEN AT 0X225...660A5 (BLOCKCHAIN)" expanded to show its balance (0 ETH) and various interaction buttons: approve, transfer, transferFrom, allowance, balanceOf, decimals, name, symbol, and totalSupply. The "symbol" button is highlighted with a red box. On the right, the code editor displays the Solidity source code for the SaluToken contract:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }
}
```

Below the code editor, the transaction history shows two calls to the SaluToken contract:

- [call] from: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 to: SaluToken.totalSupply() data: 0x181...60ddd
- [call] from: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 to: SaluToken.symbol() data: 0x95d...89b41

The transaction for calling `symbol()` is also highlighted with a red box.

Call name()

The screenshot shows the Remix Ethereum IDE interface. On the left, the sidebar has icons for deploying contracts, running transactions, and interacting with deployed contracts. The main area is split into two panes: 'DEPLOY & RUN TRANSACTIONS' on the left and the Solidity code editor on the right.

Deployed Contracts: SALUTOKEN AT 0x225...660A5 (BLOCKCHAIN)

Balance: 0 ETH

Contract Functions (available via dropdown menu):

- approve
- transfer
- transferFrom
- allowance
- balanceOf
- decimals
- name** (highlighted with a red box)
- symbol
- totalSupply

Solidity Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }
}
```

Transactions:

- [call] from: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 to: SaluToken.symbol() data: 0x95d...89b41
- call to SaluToken.name()** (highlighted with a red box)
- [call] from: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 to: SaluToken.name() data: 0x06f...dde03

Bottom Bar:

- Initialize as git repo
- Did you know? To prototype using the Gnosis safe multi sig wallet: create a multisig workspace.
- RemixAI Copilot (enabled)
- Scam Alert

Call decimals()

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

- Top Bar:** Shows the title "Remix - Ethereum IDE" and a URL bar with the address "https://remix.ethereum.org/#lang=en&optimize=false&runs=200&evmVersion=null&version=soljs...".
- Left Sidebar:** Contains icons for deploying contracts, running transactions, and interacting with the blockchain.
- Middle Left Panel:** Titled "DEPLOY & RUN TRANSACTIONS" and "Deployed Contracts 1". It lists "SALUTOKEN AT 0X225...660A5 (BLOCKCHAIN)" with a balance of 0 ETH. Below it are buttons for "approve", "transfer", "transferFrom", "allowance", "balanceOf", and "decimals". The "decimals" button is highlighted with a red box.
- Middle Right Panel:** Titled "SaluToken.sol X". It displays the Solidity code for the SaluToken contract, which inherits from ERC20 and has a constructor that mints 10,000,000 tokens.
- Bottom Panel:** Shows transaction history with two entries. The first entry is a call to "SaluToken.name()". The second entry is a call to "SaluToken.decimals()", which is also highlighted with a red box. Both entries have a "Debug" button next to them.

View & Publish

The screenshot shows a web browser window displaying the Etherscan.io contract view for the address `0xe7ffc96d36d74d6e81f2513b21669893a913205b`. The page is titled "Contract" and includes sections for Overview, More Info, and Multichain Info. The "Contract" tab is selected in the navigation bar. A red box highlights the "Verify and Publish" button, which is located below the "Contract" tab. Another red box highlights the "Verify and Publish" text in a callout bubble.

Contract Address: 0xe7ffc96d36d74d6e81f2513b21669893a913205b

Search by Address / Txn Hash / Block / Token

Contract 0xE7FFC96d36d74D6E81F2513b21669893a913205B

Overview

ETH BALANCE
0 ETH

More Info

CONTRACT CREATOR
0x4CE135aB...64E06ae71 at txn 0x4e8b9e8c2b...

TOKEN TRACKER
SaluToken (ST)

Multichain Info
N/A

Transactions Token Transfers (ERC-20) **Contract** Events

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

Decompile Bytecode Switch to Opcodes View Similar Contracts

```
0x608060405234801561000f575f80fd5b5060043610610091575f3560e01c8063313ce56711610064578063313ce5671461013157806370a082311461014f57806395d89b411461017f578063a9059cbb1461019d578063dd62ed3e146101cd57610091565b806306fdde0314610095578063095ea7b3146100b357806318160ddd146100e357806323b872dd14610101575b5f80fd5b61009d6101fd565h6040516100aa9190610a5a565h60405180910390f35h6100rd60048036038101906100c89190610h0h565h61028d565h6040516100da9190610h63565h60405180910390f3
```

View & Publish Contract Source Code

The screenshot shows a web browser window titled "Verify & Publish Contract Sou" with the URL <https://sepolia.etherscan.io/verifyContract?a=0xe7ffc96d36d74d6e81f2513b21669893a913205b>. The page is titled "Verify & Publish Contract Source Code". It has two main steps: 1. Enter Contract Details and 2. Verify & Publish. Step 1 is active. A red box highlights the contract address input field containing "0xe7ffc96d36d74d6e81f2513b21669893a913205b". Another red box highlights the "Solidity (Single file)" dropdown menu. A third red box highlights the "v0.8.26+commit.8a97fa7a" dropdown menu. A fourth red box highlights the "3) MIT License (MIT)" dropdown menu. A fifth red box highlights the "Continue" button at the bottom.

1 Enter Contract Details — 2 Verify & Publish

Please enter the Contract Address you would like to verify
0xe7ffc96d36d74d6e81f2513b21669893a913205b

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 ⓘ
3) MIT License (MIT)

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

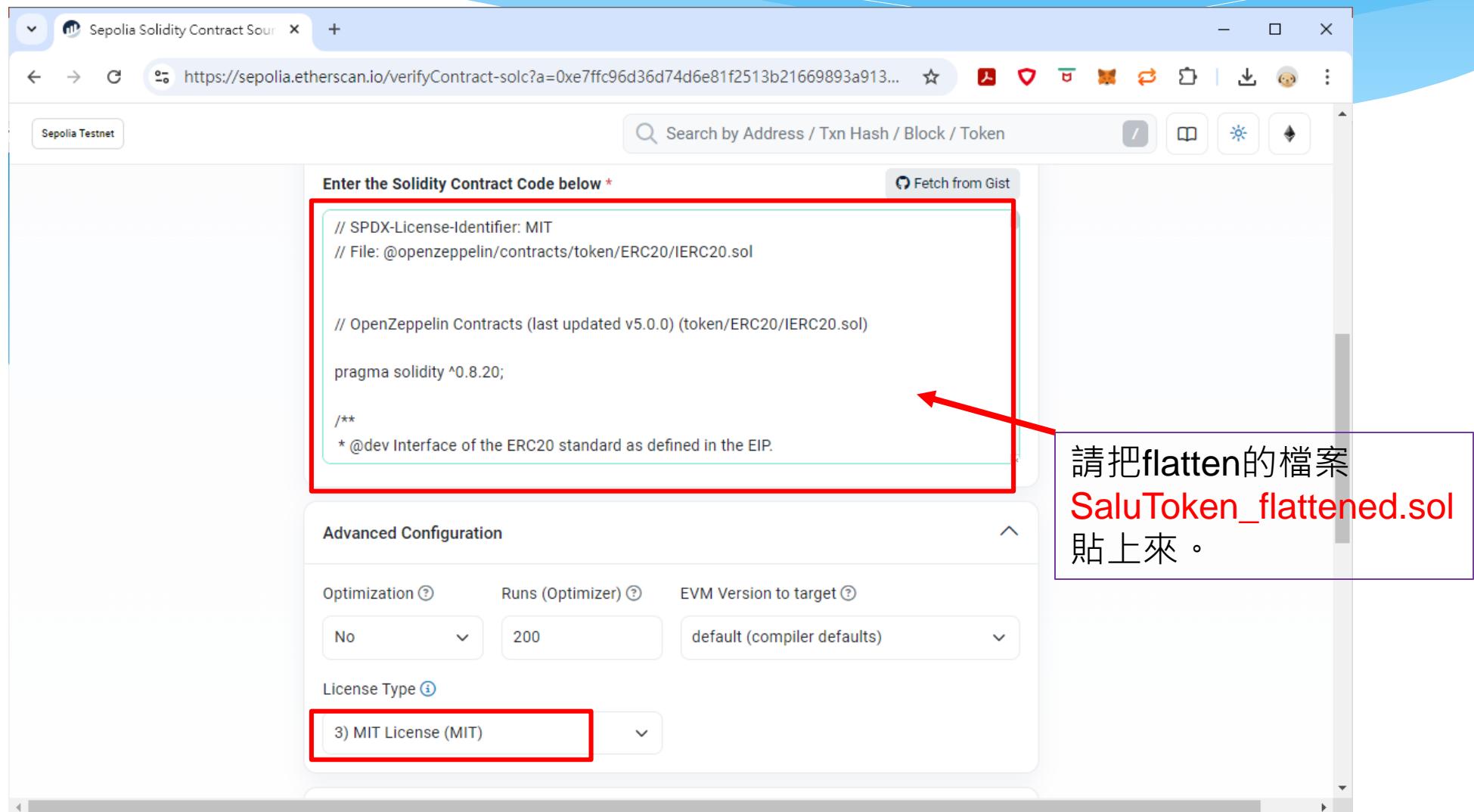
Continue Reset

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=0xe7ffc96d36d74d6e81f2513b21669893a913205b>. The page is titled "Verify & Publish Contract Source Code". It explains that source code verification provides transparency by matching uploaded code with the blockchain. A simple interface is provided for contracts fitting in a single file. The process is divided into two steps: "Enter Contract Details" (step 1) and "Verify & Publish" (step 2). The "Enter Contract Details" step is active, showing a "Upload Contract Source Code" section. It contains three numbered instructions: 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](#). Below these instructions, a form is shown with the following fields:

Contract Address:	0xe7ffc96d36d74d6e81f2513b21669893a913205b
Compiler Type:	SINGLE FILE / CONCATENATED METHOD
Compiler Version:	v0.8.26+commit.8a97fa7a

View & Publish Contract Source Code



Enter the Solidity Contract Code below *

```
// SPDX-License-Identifier: MIT
// File: @openzeppelin/contracts/token/ERC20/IERC20.sol

// OpenZeppelin Contracts (last updated v5.0.0) (token/ERC20/IERC20.sol)

pragma solidity ^0.8.20;

/**
 * @dev Interface of the ERC20 standard as defined in the EIP.

```

Fetch from Gist

Advanced Configuration

Optimization ⓘ Runs (Optimizer) ⓘ EVM Version to target ⓘ

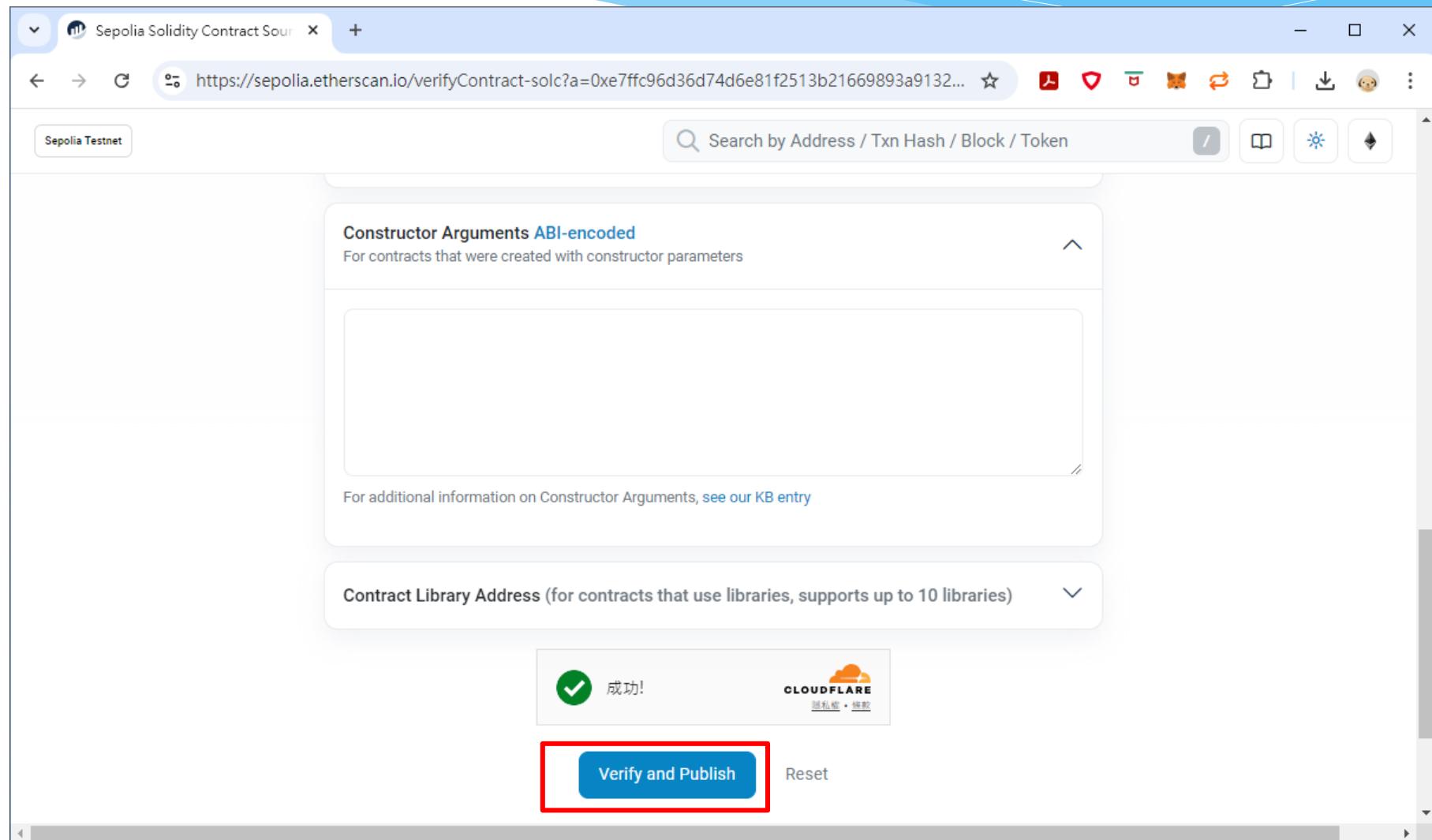
No 200 default (compiler defaults)

License Type ⓘ

3) MIT License (MIT)

請把flatten的檔案
SaluToken_flattened.sol
貼上來。

View & Publish Contract Source Code



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=0xe7ffc96d36d74d6e81f2513b21669893a913205b>. The page is titled "Verify & Publish Contract Source Code". It explains that source code verification provides transparency by matching uploaded code with the blockchain. A red box highlights a success message: "Successfully generated Bytecode and ABI for Contract Address [0xe7ffc96d36d74d6e81f2513b21669893a913205b]". A red arrow points to this message with the text "click". Below the message, there's a link to learn about verifying contracts on multiple blockchains. The "Code Reader" section contains a "Prompt" for reviewing the smart contract's source code.

1 Enter Contract Details — 2 Verify & Publish

Successfully generated Bytecode and ABI for Contract Address
[0xe7ffc96d36d74d6e81f2513b21669893a913205b]

① Learn how to verify your contract on multiple blockchains with a single API key [here](#).

Code Reader ②

Prompt:

You are a conscientious blockchain security auditor. Review this smart contract's source code to suggest best practices it could follow and share any security concerns with the code.

View Contract Source Code

The screenshot shows the Etherscan IDE interface for the SaluToken contract. The browser address bar displays the URL: <https://sepolia.etherscan.io/address/0xe7ffc96d36d74d6e81f2513b21669893a913205b#code>. A red box highlights this URL. Below the address bar, there are tabs for 'Code' (which is selected), 'Read Contract', and 'Write Contract'. A search bar at the top right allows searching by Address / Txn Hash / Block / Token. A message 'Contract Source Code Verified (Exact Match)' is displayed in a red-bordered box. The contract details section shows 'Contract Name: SaluToken', 'Compiler Version: v0.8.26+commit.8a97fa7a', 'Optimization Enabled: No with 200 runs', and 'Other Settings: default evmVersion, MIT license'. The 'Contract Source Code (Solidity)' section contains the following code, also highlighted with a red box:

```
5 // SPDX-License-Identifier: MIT
6 // File: @openzeppelin/contracts/token/ERC20/IERC20.sol
7
8
9 // OpenZeppelin Contracts (last updated v5.0.0) (token/ERC20/IERC20.sol)
10
11 pragma solidity ^0.8.20;
12
13 /**
14  * @dev Interface of the ERC20 standard as defined in the EIP.
15  */
16 interface IERC20 {
17 /**
18  * @dev Emitted when `value` tokens are moved from one account (`from`) to
19  * another (`to`).
20  *
21  * Note that `value` may be zero.
22  */
23 event Transfer(address indexed from, address indexed to, uint256 value);
```

View & Publish Contract Source Code

The screenshot shows a web browser window displaying the Etherscan interface for a contract at address 0xe7ffc96d36d74d6e81f2513b21669893a913205b. The page is titled "SaluToken | Address 0xe7ffc96d36d74d6e81f2513b21669893a913205b". The URL in the address bar is <https://sepolia.etherscan.io/address/0xe7ffc96d36d74d6e81f2513b21669893a913205b#code>. The browser toolbar includes icons for back, forward, search, and refresh.

The main content area displays the "Source Code" tab for the contract. The "More Info" section shows the contract creator as 0x4CE135aB...64E06ae71, created at tx 0x4e8b9e8c2b..., and the token tracker as SaluToken (ST). The "Multichain Info" section indicates N/A. Below the tabs, a red box highlights the "Contract" tab, which is currently selected. Another red box highlights the "Code" button in the "Actions" row, along with "Read Contract" and "Write Contract". A "Search Source Code" input field is also present.

At the bottom, it is noted that the "Contract Source Code Verified (Exact Match)". The contract name is SaluToken, optimization is disabled, and it has been run 200 times.

Connect to Web3

The screenshot shows a browser window displaying the etherscan.io interface for the address 0xe7ffc96d36d74d6e81f2513b21669893a913205b. The 'Contract' tab is selected. A red box highlights the 'Read Contract' button. Another red box highlights the 'Connect to Web3' button. A third red box highlights a modal window titled 'sepolia.etherscan.io 顯示'. The modal contains a note about the beta status of the feature and two buttons: '確定' (Confirm) and '取消' (Cancel). A red arrow points from the 'Connect to Web3' button to the 'sepolia.etherscan.io 顯示' modal.

sepolia.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.

確定 取消

Connect a Wallet-MetaMask

The screenshot shows a web browser window displaying the Etherscan interface for a SaluToken on the Sepolia Testnet. The URL in the address bar is <https://sepolia.etherscan.io/address/0xe7ffc96d36d74d6e81f2513b21669893a913205b#readContract>. A modal window titled "Connect a Wallet" is open, containing instructions and three wallet connection options: MetaMask (selected and highlighted with a red box), WalletConnect, and Coinbase Wallet.

MetaMask

Popular

WalletConnect

Coinbase Wallet

Connected Web3 using account

The image displays two side-by-side screenshots illustrating the connection of a Web3 account.

Left Screenshot (Etherscan):

- The title bar shows "SaluToken | Address 0xe7ffc96...".
- The URL in the address bar is "https://sepolia.etherscan.io/address/0xe7ffc96d36d74d6e81f2513b21669893a913205b#readContract".
- The "Contract" tab is selected.
- A red box highlights the "Connected - Web3 [0x4ce1...ae71]" status message.
- The sidebar shows "Sepolia Testnet" and a list of contract functions: allowance, balanceOf, decimals, name, symbol, and totalSupply.

Right Screenshot (Wallet Interface):

- The title bar shows "Salu1" with the account address "0x4CE13...ae71".
- The main display shows "1.5054 SepoliaETH".
- The "Portfolio" section includes "Buy & Sell", "Swap", "Bridge", "發送" (Send), and "接收" (Receive) buttons.
- The "交易紀錄" (Transactions) section lists two entries:
 - Oct 18, 2024: 部署合約 (Deploy Contract) 已確認 (Confirmed) -0 SepoliaETH
 - Oct 17, 2024: 部署合約 (Deploy Contract) 已確認 (Confirmed) -0 SepoliaETH

Read Contract - balanceOf

The screenshot shows the Etherscan interface for the SaluToken contract (Address 0xe7ffc96d36d74d6e81f2513b21669893a913205b) on the Sepolia Testnet. The 'Contract' tab is selected. In the 'Read Contract' section, the 'balanceOf' method is chosen. The 'account (address)' field contains the value '0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71'. The 'Query' button is highlighted with a red box. The response is shown in a box: '[balanceOf(address) method Response]' followed by '» uint256 : 10000000000000000000000000000000'.

Read Contract-decimals

The screenshot shows a browser window for the Sepolia Testnet version of etherscan.io. The address being analyzed is 0xe7ffc96d36d74d6e81f2513b21669893a913205b. The main content area displays a list of contract functions:

- 1. allowance
- 2. balanceOf
- 3. decimals
- 4. name
- 5. symbol
- 6. totalSupply

The "decimals" function is expanded, showing its return value as "18 uint8". This value is highlighted with a red rectangular box. The interface includes standard browser navigation buttons, a search bar, and various toolbars at the top.

Read Contract-name

SaluToken | Address 0xe7ffc96d36d74d6e81f2513b21669893a913205b

https://sepolia.etherscan.io/address/0xe7ffc96d36d74d6e81f2513b21669893a913205b#readContract

Sepolia Testnet

Connected - Web3 [0x4ce1...ae71]

Read Contract Information [Expand all] [Reset]

- 1. allowance
- 2. balanceOf
- 3. decimals
- 4. name
- 5. symbol
- 6. totalSupply

SaluToken string

Read Contract-symbol

The screenshot shows the Etherscan interface for the Sepolia Testnet. The address being analyzed is 0xe7ffc96d36d74d6e81f2513b21669893a913205b. The 'Read Contract Information' section is displayed, listing various contract functions. The 'symbol' function is highlighted with a red box around its value, 'ST string'. Other listed functions include allowance, balanceOf, decimals, name, and totalSupply.

Function	Value	Actions
1. allowance		🔗 ⚙️ →
2. balanceOf		🔗 ⚙️ →
3. decimals		🔗 ⚙️ →
4. name		🔗 ⚙️ →
5. symbol	ST string	🔗 ⚙️ ↓
6. totalSupply		🔗 ⚙️ →

Read Contract-totalSupply

List Contract using Remix

The screenshot shows the Ethereum IDE interface in Remix. On the left, the sidebar displays "DEPLOY & RUN TRANSACTIONS" with fields for "VALUE" (0 Wei), "CONTRACT" (SaluToken - SaluToken_flattened.sol), and a "Deploy" button. A red box highlights the "Deploy" button, and a circled '1' is next to the "Deployed Contracts" section, which lists "SALUTOKEN AT 0xE7F...3205B (BLOCKCHAIN)". The main panel shows the Solidity code for SaluToken.sol, which is a flattened version of the OpenZeppelin ERC20 standard. Below the code, the transaction history shows a pending creation of the SaluToken contract with a value of 0 wei. The bottom status bar includes links for "Initialize as git repo", "Did you know?", "RemixAI Copilot (enabled)", and "Scam Alert".

```
// SPDX-License-Identifier: MIT
// File: @openzeppelin/contracts/token/ERC20/IERC20.sol
// OpenZeppelin Contracts (last updated v5.0.0) (token/ERC20/IERC20.sol)
pragma solidity ^0.8.20;

/**
 * @dev Interface of the ERC20 standard as defined in the EIP.
 */
interface IERC20 {
    /**
     * @dev Emitted when `value` tokens are moved from one account (`from`) to
     * another (`to`).
     */
}
```

Transactions recorded: 2 i

Deployed Contracts: 1

SALUTOKEN AT 0xE7F...3205B (BLOCKCHAIN)

Did you know? To prototype using the Gnosis safe multi sig wallet: create a multisig workspace.

RemixAI Copilot (enabled)

Scam Alert

List Contract using Remix

The screenshot shows the Ethereum IDE interface in Remix. On the left, there's a sidebar with various icons. The main area is titled "DEPLOY & RUN TRANSACTIONS". Under "Deployed Contracts", it lists "SALUTOKEN AT 0xE7F...3205B (BLOCKCHAIN)" with a balance of "0 ETH". A red box highlights a list of functions: "approve", "transfer", "transferFrom", "allowance", "balanceOf", "decimals", "name", "symbol", and "totalSupply". To the right, the code editor shows the Solidity source code for "SaluToken.sol" and its flattened version "SaluToken_flattened.sol". The code is based on the OpenZeppelin ERC20 standard. At the bottom, there are logs of transactions, including one for the constructor deployment.

```
// SPDX-License-Identifier: MIT
// File: @openzeppelin/contracts/token/ERC20/IERC20.sol
pragma solidity ^0.8.20;

/**
 * @dev Interface of the ERC20 standard as defined in the EIP.
 */
interface IERC20 {
    /**
     * @dev Emitted when `value` tokens are moved from one account (`from`) to
     * another (`to`).
     */
    event Transfer(address indexed from, address indexed to, uint256 value);
    event Approval(address indexed owner, address indexed spender, uint256 value);
    function totalSupply() external view returns (uint256);
    function balanceOf(address account) external view returns (uint256);
    function allowance(address owner, address spender) external view returns (uint256);
    function transfer(address to, uint256 value) external returns (bool);
    function approve(address spender, uint256 value) external returns (bool);
    function transferFrom(address from, address to, uint256 value) external returns (bool);
}
```

balanceOf function

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

- Deploy & Run Transactions:** Shows a deployed contract named "SALUTOKEN AT 0xE7F...3205B (BLOCKCHAIN)".
 - Balance:** 0 ETH
 - Low level interactions:** A list of functions:
 - approve
 - transfer
 - transferFrom
 - allowance
 - balanceOf** (highlighted with a red box) - Address: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71, Value: 0: uint256: 10000000000000000000000000000000
 - decimals
 - name
 - symbol
 - totalSupply
- Code Editor:** Displays the Solidity code for the SaluToken contract.

```
// SPDX-License-Identifier: MIT
// File: @openzeppelin/contracts/token/ERC20/IERC20.sol
// OpenZeppelin Contracts (last updated v5.0.0) (token/ERC20/IERC20.sol)
pragma solidity ^0.8.20;

/**
 * @dev Interface of the ERC20 standard as defined in the EIP.
 */
interface IERC20 {
    /**
     * @dev Emitted when `value` tokens are moved from one account (`from`) to
     * another (`to`).
     */
    event Transfer(address indexed from, address indexed to, uint256 value);
    ...
}
```
- Logs:** Shows two transaction logs:
 - [block:6893381 txIndex:29] from: 0x4ce...6ae71 to: SaluToken.(constructor) value: 0 wei
 - call to SaluToken.balanceof
 - [call] from: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 to: SaluToken.balanceOf(address)

decimals function

The screenshot shows the Ethereum IDE interface with the SaluToken contract deployed. The left sidebar displays the deployed contracts, including SALUTOKEN AT 0xE7F...3205B (BLOCKCHAIN) with a balance of 0 ETH. The main pane shows the SaluToken.sol source code with two functions highlighted:

```
100
101 /**
102 * @dev Returns the symbol of the token.
103 */
104 function symbol() external view returns (string memory);
105
106 /**
107 * @dev Returns the decimals places of the token.
108 */
109 function decimals() external view returns (uint8);
110 }
111
112 // File: @openzeppelin/contracts/utils/Context.sol
113
114
```

The "decimals" button in the Deploy & Run Transactions sidebar is highlighted with a red box. Its value is shown as 0: uint8: 18. The bottom section shows transaction logs for calls to the balanceOf and decimals functions.

name function

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

- Deployed Contracts:** SALUTOKEN AT 0xE7F...3205B (BLOCKCHAIN)
- Balance:** 0 ETH
- Available Functions (highlighted in red box):**
 - approve
 - transfer
 - transferFrom
 - allowance
 - balanceOf
 - decimals
 - name**
 - symbol
 - totalSupply
- Contract Code:**

```
100
101    /**
102     * @dev Returns the symbol of the token.
103     */
104    function symbol() external view returns (string memory);
105
106    /**
107     * @dev Returns the decimals places of the token.
108     */
109    function decimals() external view returns (uint8);
110
111    // File: @openzeppelin/contracts/utils/Context.sol
112
113
114
```
- Transaction History:**
 - CALL [call] from: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 to: SaluToken.decimals()
data: 0x313...ce567
call to SaluToken.name
 - CALL [call] from: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 to: SaluToken.name()
data: 0x06f...dde03
call to SaluToken.name

symbol function

The screenshot shows the Remix Ethereum IDE interface. On the left, the "DEPLOY & RUN TRANSACTIONS" sidebar displays several functions with their return values:

- transferFrom: address from, address to, uint256 value
- allowance: address owner, address spender
- balanceOf: 0x4CE135aB2eB8e482D16B8011ba9415D
- decimals: 0: uint8: 18
- name: 0: string: SaluToken
- symbol**: 0: string: ST (highlighted with a red box)
- totalSupply: 0: uint256: 1000

The main code editor window contains the Solidity code for the `SaluToken.sol` contract, which includes the `symbol()` and `decimals()` functions.

```
100 /**
101 * @dev Returns the symbol of the token.
102 */
103
104 function symbol() external view returns (string memory);
105
106 /**
107 * @dev Returns the decimals places of the token.
108 */
109 function decimals() external view returns (uint8);
110
111 // File: @openzeppelin/contracts/utils/Context.sol
112
113
114
```

The bottom transaction history shows two calls to the `symbol()` function:

- call [call] from: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 to: SaluToken.symbol()
data: 0x06f...dde03
- call to SaluToken.symbol
- call [call] from: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 to: SaluToken.symbol()
data: 0x95d...89b41

totalSupply function

The screenshot shows the Ethereum IDE interface with the SaluToken contract loaded. The left sidebar displays deployment and transaction details, including the address 0x4CE135aB2eB8e482D1688011ba9415D. The main code editor shows the Solidity code for the SaluToken contract, which includes a `totalSupply()` function. This function is highlighted with a red box. Below the code, the transaction history shows two calls to `totalSupply()`, both originating from the same address and pointing to the `SaluToken.totalSupply()` function. The bottom status bar indicates "RemixAI Copilot (enabled)" and "Scam Alert".

```
100 /**
101 * @dev Returns the symbol of the token.
102 */
103 function symbol() external view returns (string memory);
104 /**
105 * @dev Returns the decimals places of the token.
106 */
107 function decimals() external view returns (uint8);
108 }
109 // File: @openzeppelin/contracts/utils/Context.sol
110
111
112
113
114
```

DEPLOY & RUN TRANSACTIONS

- transferFrom address from, address to, uint256 value
- allowance address owner, address spender
- balanceOf 0x4CE135aB2eB8e482D1688011ba9415D
 - 0: uint256: 10000000000000000000000000000000
- decimals
 - 0: uint8: 18
- name
 - 0: string: SaluToken
- symbol
 - 0: string: ST
- totalSupply** 0: uint256: 10000000000000000000000000000000
- totalSupply - call

Low level interactions

CALLDATA

Transact

Did you know? To prototype using the Gnosis safe multi sig wallet: create a multisig workspace.

0 Listen on all transactions Filter with transaction hash or address

CALL [call] from: 0x4CE135aB2eB8e482D1688011ba9415D64E06ae71 to: SaluToken.symbol() data: 0x95d...89b41 call to SaluToken.totalSupply

CALL [call] from: 0x4CE135aB2eB8e482D1688011ba9415D64E06ae71 to: SaluToken.totalSupply() data: 0x181...60ddd

Initialize as git repo

RemixAI Copilot (enabled)

Scam Alert

Import Tokens

Copy Contract address

A screenshot of an Etherscan contract page for the SaluToken (ST) token on the Sepolia Testnet. The URL in the browser is <https://sepolia.etherscan.io/address/0xe7ffc96d36d74d6e81f2513b21669893a913205b>. A red arrow points to the copy icon (a clipboard with a plus sign) located next to the contract address `0xE7FFC96d36d74D6E81F2513b21669893a913205B`. Two red boxes highlight the "CONTRACT CREATOR" section, which shows the address `0x4CE135aB...64E06ae71` and the transaction hash `0x4e8b9e8c2...`, and the "TOKEN TRACKER" section, which shows the SaluToken (ST) logo and name.

Contract `0xE7FFC96d36d74D6E81F2513b21669893a913205B`

CONTRACT CREATOR
`0x4CE135aB...64E06ae71` at txn `0x4e8b9e8c2...`

TOKEN TRACKER
SaluToken (ST)

Import Tokens

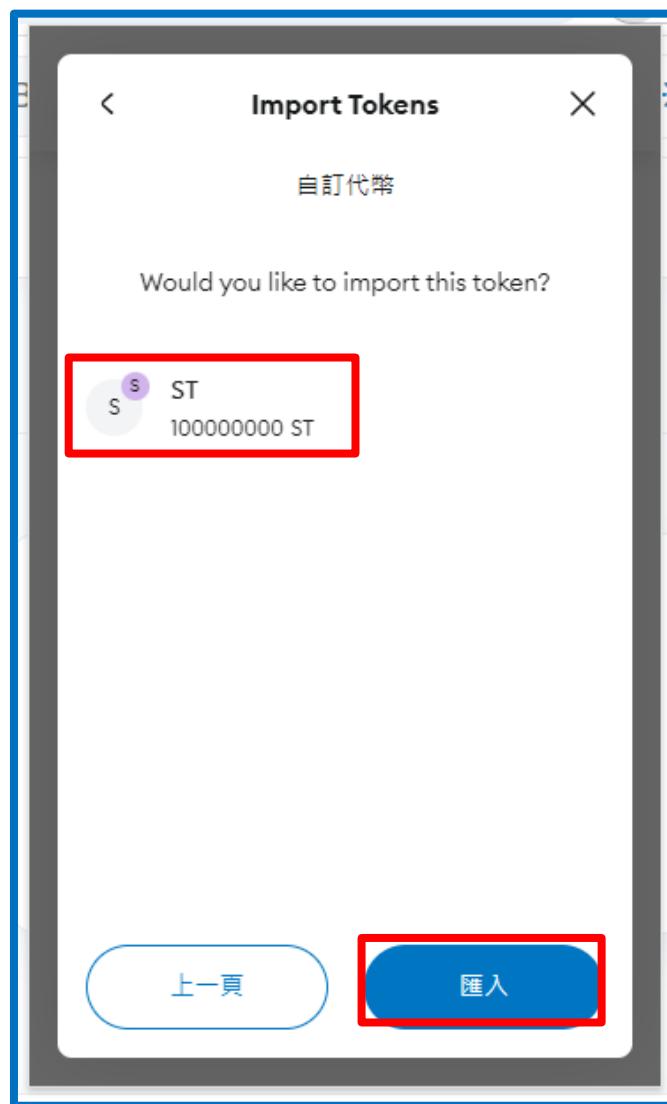
The image consists of three panels illustrating the process of importing tokens into a wallet.

Left Panel: Shows the main wallet interface. At the top, it displays the account name "Salu 1" and the address "0x4CE13...6ae71". Below this, the balance is shown as "1.6139 SepoliaETH" and "\$4,208.17 USD". There are several buttons for "Buy & Sell", "Swap", "Bridge", "發送" (Send), and "接收" (Receive). A blue button labeled "+ Import Tokens" is highlighted with a red box. Other tabs like "Tokens", "NFTs", and "交易紀錄" (Transactions) are visible. At the bottom, there's a "Refresh list" button and a "MetaMask 支援" (MetaMask Support) link.

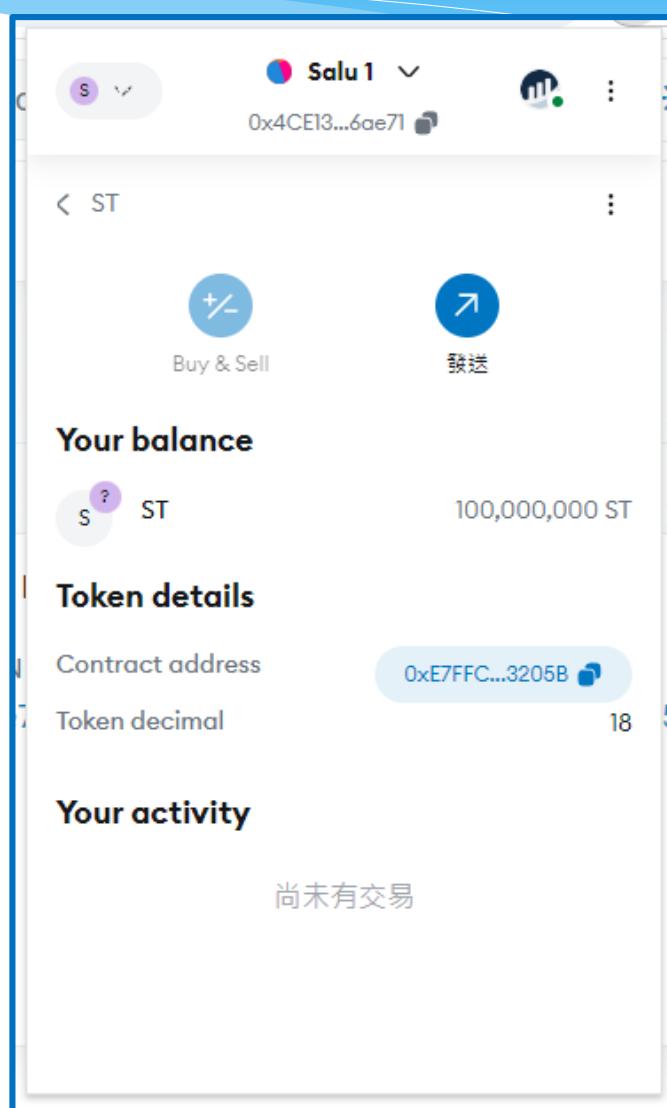
Middle Panel: Shows the "Import Tokens" dialog box. It starts with a message: "Token detection is not available on this network yet. Please import token manually and make sure you trust it. Learn about [scams and security risks](#)". Below this is a text input field labeled "代幣合約位址" (Token Contract Address) with a red box around it. A red arrow points from the word "paste" to the right side of this input field. At the bottom is a blue "下一步" (Next) button.

Right Panel: Shows the completed token import form. The "代幣合約位址" field now contains the value "0xE7FFC96d36d74D6E81F2513b21669". The "代幣代號" (Token Symbol) field contains "ST". The "小數點位數" (Decimals) field contains "18". A red box highlights the blue "下一步" (Next) button at the bottom.

Import Tokens



Token details



Transfer Token

Transfer Token

The image displays two screenshots of a mobile application interface for token transfer, likely MetaMask or a similar wallet extension.

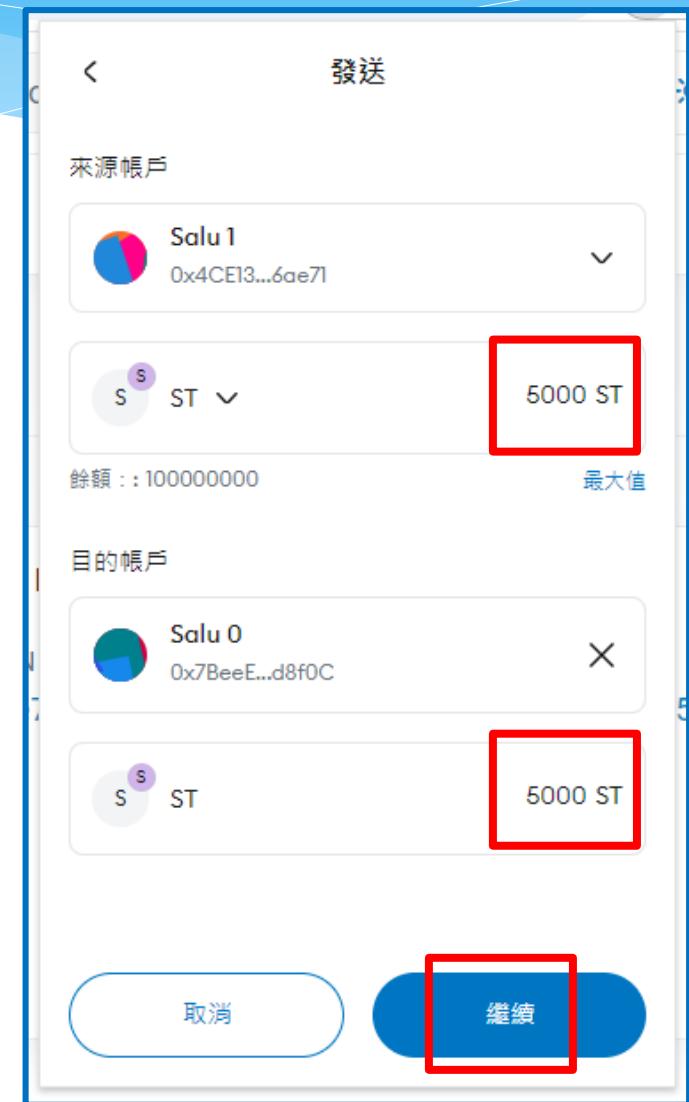
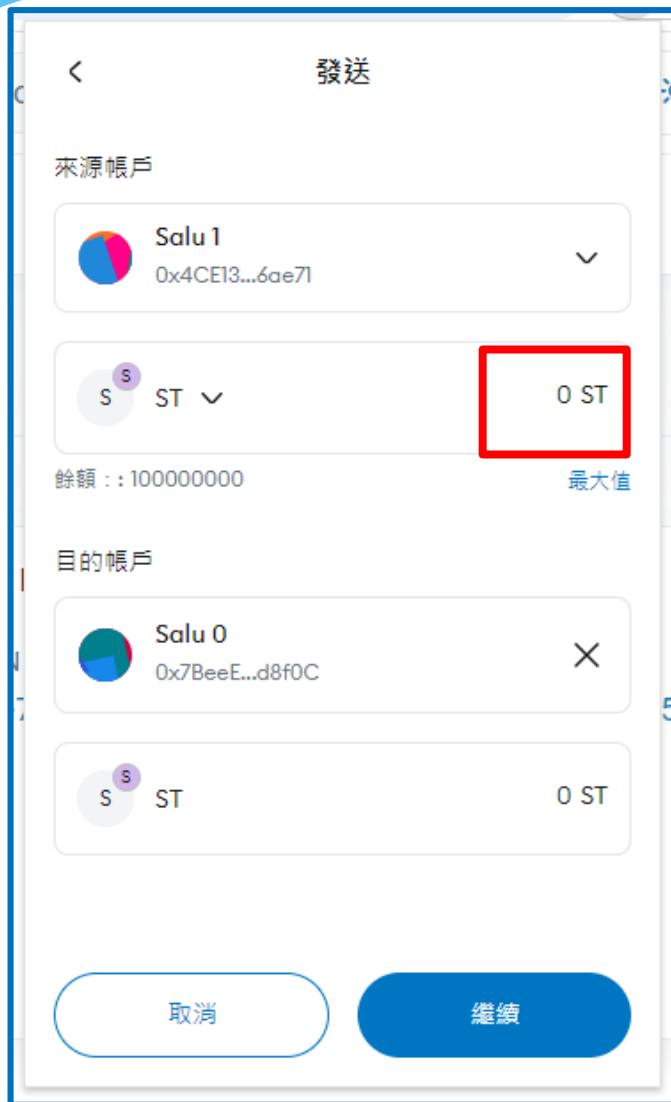
Screenshot 1: Home Screen (Left)

- User profile: Salu 1 (0x4CE13...6ae71)
- Balance: 100,000,000 ST
- Token details:
 - Contract address: 0xE7FFC...3205B
 - Token decimal: 18
- Activity: 尚未有交易 (No transactions yet)
- Action button: 發送 (Send) highlighted with a red box.

Screenshot 2: Transfer Confirmation (Right)

- Recipient field: Enter public address (0x) or domain name
- Account selection:
 - Your accounts
 - Salu 0 (0x7BeeE...d8f0C) selected and highlighted with a red box.
 - Salu 1 (0x4CE13...6ae71) Imported
 - 錢包 3 (0x29587...a66cB) Imported
 - 錢包 2
 - 聯絡人 (Contacts)
- Buttons at the bottom: 取消 (Cancel) and 繼續 (Continue)

Transfer Token



Transfer Token

The image consists of three side-by-side screenshots illustrating the token transfer process between two wallets, Salu 1 and Salu 0.

Left Screenshot (Transfer Step):

- Shows the transfer amount as 5000 ST.
- A red box highlights the amount "5000 ST" and the status "尚未有匯率比較值" (No exchange rate comparison value available).
- The "確認" (Confirm) button at the bottom is also highlighted with a red box.

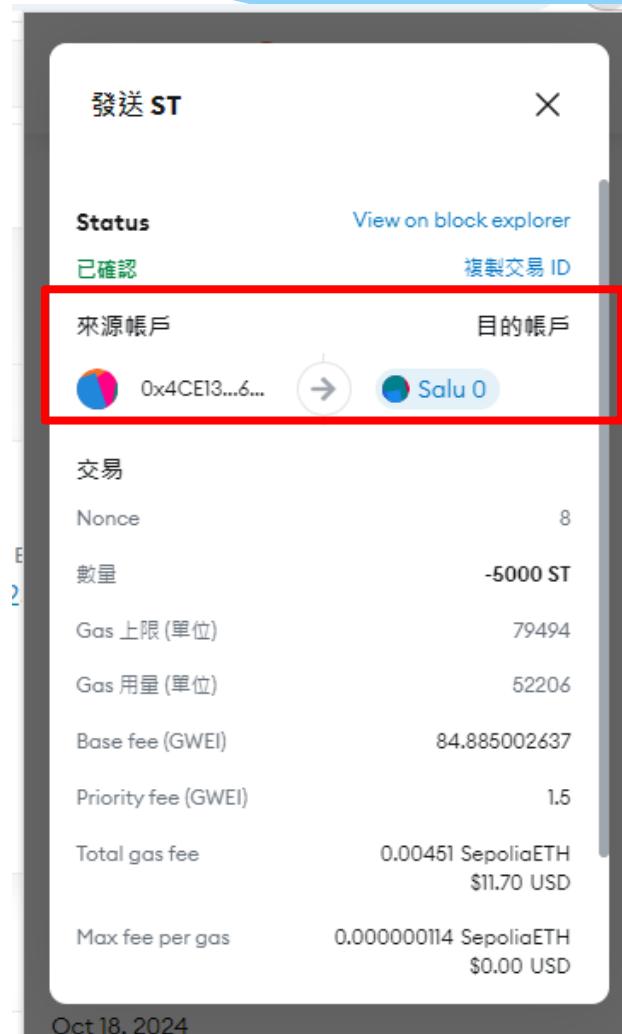
Middle Screenshot (Pending Step):

- Shows the wallet balance as 1.6139 SepoliaETH.
- Shows a transaction history entry for "Oct 23, 2024" with the action "發送 ST" (Send ST) and amount "-5000 ST". This entry is highlighted with a red box.
- Shows a "等待處理" (Waiting for processing) status next to the transaction.
- Shows "取消" (Cancel) and "加速" (Priority) buttons below the transaction.

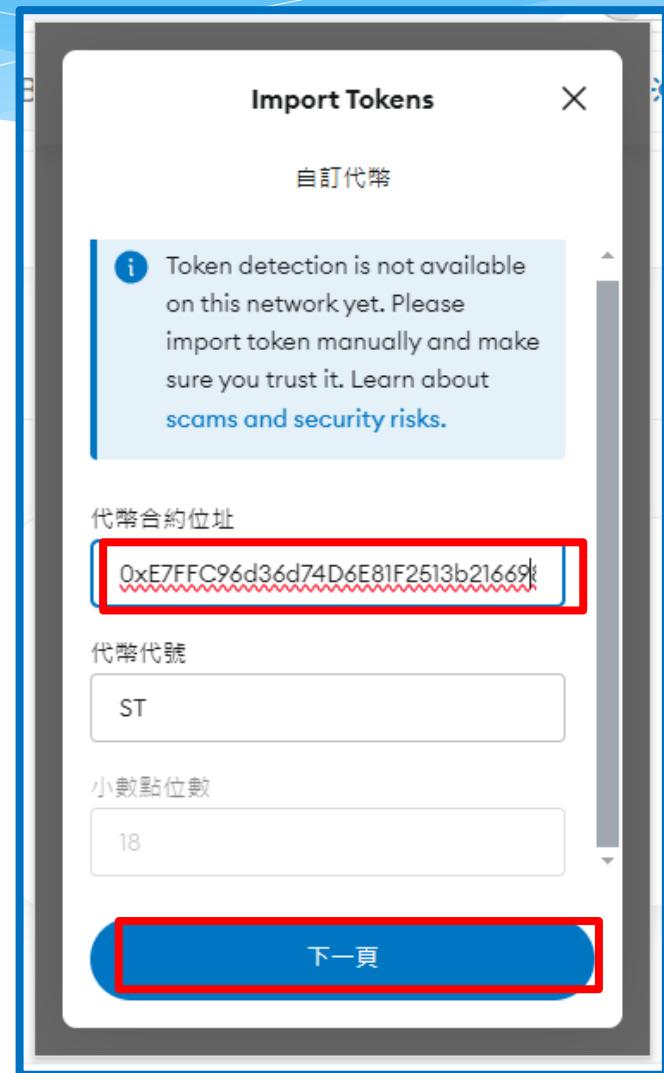
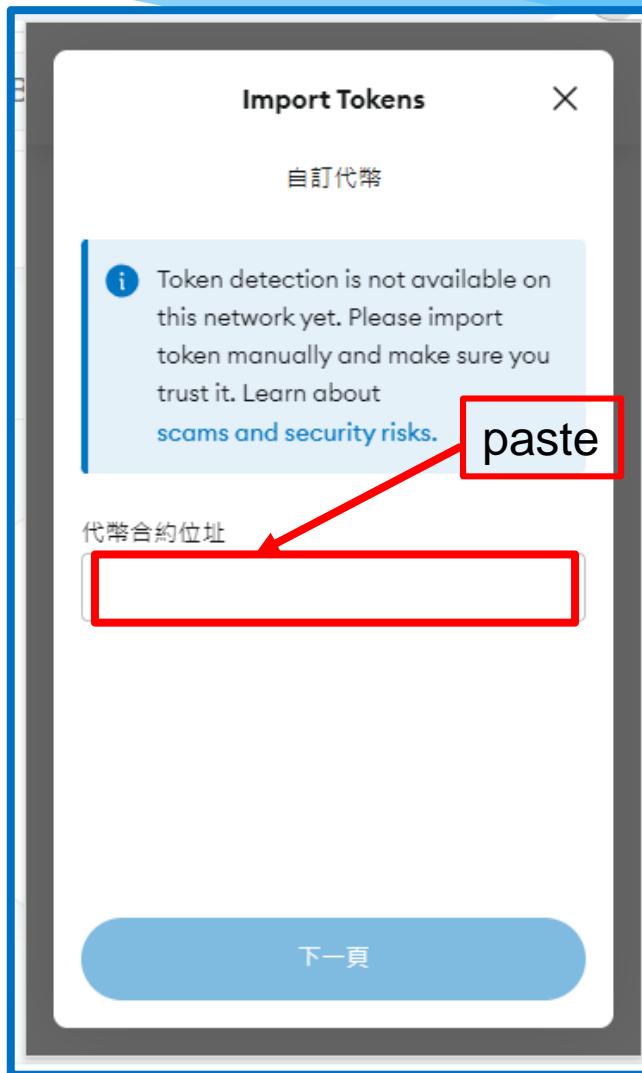
Right Screenshot (Completed Step):

- Shows the wallet balance as 1.6094 SepoliaETH.
- Shows a transaction history entry for "Oct 23, 2024" with the action "發送 ST" (Send ST) and amount "-5000 ST". This entry is highlighted with a red box.
- Shows a green "已確認" (Confirmed) status next to the transaction.
- Shows a transaction history entry for "Oct 21, 2024" with the action "接收" (Receive) and amount "0.05 SepoliaETH".
- Shows a green "已確認" (Confirmed) status next to the receive transaction.

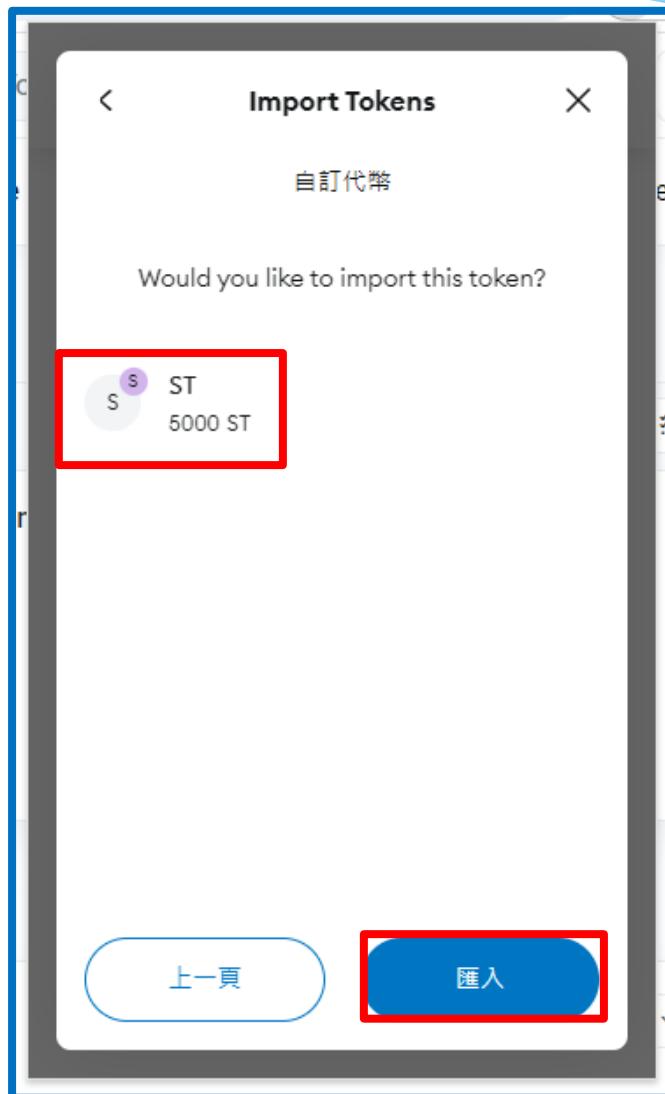
Transfer Token



Import Tokens



Import Tokens



Token Overview

The screenshot shows the Etherscan.io interface for the SaluToken contract. Key elements highlighted with red boxes include:

- Contract Address:** 0xE7FFC96d36d74d6e81f2513b21669893a913205b
- ETH Balance:** 0 ETH
- More Info:** CONTRACT CREATOR (0x4CE135aB...), TOKEN TRACKER (SaluToken (ST))
- To Address:** 0xE7FFC96d36d74d6e81f2513b21669893a913205b
- Method:** Transfer

Below the table, the URL of the page is displayed: <https://sepolia.etherscan.io/address/0xe7ffc96d36d74d6e81f2513b21669893a913205b>

Token Overview

The screenshot shows the Etherscan Token Tracker for SaluToken (ST) on the Sepolia Testnet. The token is identified as an ERC-20.

Token Details:

- MAX TOTAL SUPPLY:** 100,000,000 ST
- HOLDERS:** 2
- TOTAL TRANSFERS:** 2

Market Information:

- ONCHAIN MARKET CAP: \$0.00
- CIRCULATING SUPPLY MARKET CAP: -

Other Info:

TOKEN CONTRACT (WITH 18 DECIMALS)
0xe7ffc96d36d74d6e81f2513b21669893a913205b

Transactions:

A total of 2 transactions found

Method	To	Amount
Transfer	0x7BeeE5F4...1EAad8f0C	5,000
	0x4CE135aB...64E06ae71	100,000,000

Modify and Extend Token

Buy function

- Allow users to mint new tokens by depositing some ether.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    uint256 price = 0.0000001 ether; // price of 1 token in ether

    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 100000000 * (10 ** uint256(decimals())));
    }

    function buy() external payable {
        require(msg.value > 0, "You must send some ether");
        _mint(msg.sender, msg.value * 10 ** decimals() / price);
    }
}
```

Burn function

■ Allowing users to burn their tokens.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    uint256 price = 0.0000001 ether; // price of 1 token in ether

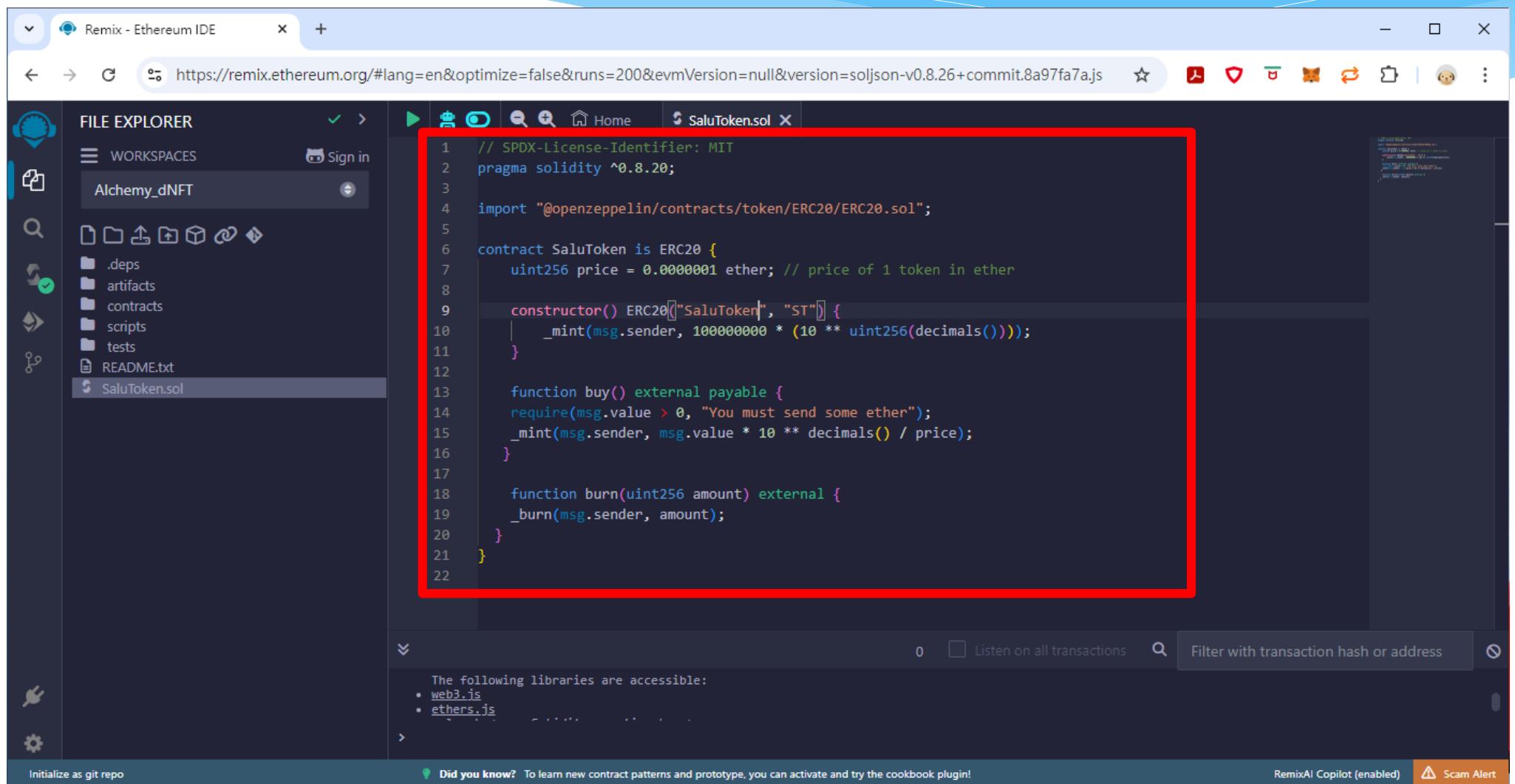
    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }

    function buy() external payable {
        require(msg.value > 0, "You must send some ether");
        _mint(msg.sender, msg.value * 10 ** decimals() / price);
    }

    function burn(uint256 amount) external {
        _burn(msg.sender, amount);
    }
}
```

1. Minting and burnings are treated as transfers, so they should emit a **Transfer** event.
2. When minting new tokens, **_from** is 0x0.
3. When burning, **_to** is 0x0.

Modify SaluToken.sol



The screenshot shows the Remix Ethereum IDE interface. The left sidebar contains a 'FILE EXPLORER' with a 'WORKSPACES' section showing 'Alchemy_dNFT' and a file list including '.deps', 'artifacts', 'contracts', 'scripts', 'tests', 'README.txt', and 'SaluToken.sol'. The central area displays the Solidity code for 'SaluToken.sol'. A red box highlights the constructor and buy/burn functions. The bottom right corner of the code editor has a small preview window showing the deployed contract's interface. The bottom navigation bar includes links for 'Initialize as git repo', 'Did you know?', 'RemixAI Copilot (enabled)', and 'Scam Alert'.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    uint256 price = 0.000001 ether; // price of 1 token in ether

    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }

    function buy() external payable {
        require(msg.value > 0, "You must send some ether");
        _mint(msg.sender, msg.value * 10 ** decimals() / price);
    }

    function burn(uint256 amount) external {
        _burn(msg.sender, amount);
    }
}
```

Copy SPDX-License

The screenshot shows the Remix Ethereum IDE interface. A blue circle labeled '1' highlights the 'FILE EXPLORER' sidebar on the left, which lists a workspace named 'Alchemy_dNFT' containing files like '.deps', 'artifacts', 'contracts', 'scripts', 'tests', and 'README.txt'. A red box surrounds the file 'SaluToken.sol' in the list. A blue circle labeled '2' highlights the code editor area. A red box surrounds the first two lines of the Solidity contract code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;
```

A red arrow points from the 'Copy' button in the top right corner of the code editor towards the highlighted code. The bottom of the interface shows accessible libraries: 'web3.js' and 'ethers.js'.

Flatten SaluToken.sol

The screenshot shows the Remix Ethereum IDE interface. On the left, the FILE EXPLORER sidebar lists WORKSPACES, including 'Alchemy_dNFT'. A file named 'SaluToken.sol' is selected, highlighted with a red box and circled with a blue number 1. A context menu is open over this file, with the 'Flatten' option highlighted and circled with a blue number 2. The main workspace displays the Solidity code for the 'SaluToken' contract, which inherits from 'ERC20' and includes functions for buying and burning tokens.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract SaluToken is ERC20 {
    uint256 price = 0.000001 ether; // price of 1 token in ether

    constructor() ERC20("SaluToken", "ST") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }

    function buy() external payable {
        require(msg.value > 0, "You must send some ether");
        _mint(msg.sender, msg.value * 10 ** decimals() / price);
    }

    function burn(uint256 amount) external {
        _burn(msg.sender, amount);
    }
}
```

SaluToken_flattened.sol

The screenshot shows the Remix Ethereum IDE interface. The left sidebar displays a file tree under the 'FILE EXPLORER' tab, showing a workspace named 'Alchemy_dNFT' containing files like .deps, artifacts, contracts, scripts, tests, README.txt, and SaluToken_flattened.sol. A red box highlights the 'artifacts' folder, and a blue circle labeled '1' points to the 'SaluToken_flattened.sol' file in the tree. The main editor area shows the Solidity code for the flattened token contract, which includes an interface for the ERC20 standard and an event for transfers. A red box highlights the warning 'Missing SPDX-License' in the status bar at the bottom right. The bottom navigation bar includes links for 'Initialize as git repo', 'Did you know?', 'RemixAI Copilot (enabled)', and 'Scam Alert'.

```
// OpenZeppelin Contracts (last updated v5.0.0) (contracts/math/Math.sol)  
pragma solidity ^0.8.20;  
  
/**  
 * @dev Interface of the ERC20 standard as defined in the EIP.  
 */  
interface IERC20 {  
    /**  
     * @dev Emitted when `value` tokens are moved from one account (`from`) to  
     * another (`to`).  
     *  
     * Note that `value` may be zero.  
     */  
    event Transfer(address indexed from, address indexed to, uint256 value);  
}
```

SaluToken_flattened.sol

The screenshot shows the Remix Ethereum IDE interface. The left sidebar has a red box around the 'artifacts' folder, with a blue circle labeled '1' and a red arrow pointing to the 'SaluToken_flattened.sol' file in the bottom list. The main code editor window has a red box around the first line of code, with a blue circle labeled '2' and a red arrow pointing to a 'Paste' button. The code editor contains the following Solidity code:

```
// SPDX-License-Identifier: MIT
// OpenZeppelin Contracts (last updated v5.0.0) (token/ERC20/IERC20.sol)
pragma solidity ^0.8.20;

/**
 * @dev Interface of the ERC20 standard as defined in the EIP.
 */
interface IERC20 {
    /**
     * @dev Emitted when `value` tokens are moved from one account (`from`) to
     * another (`to`).
     *
     * Note that `value` may be zero.
     */
    event Transfer(address indexed from, address indexed to, uint256 value);
}
```

At the bottom of the interface, there are two transaction logs:

- CALL [call] from: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 to: SaluToken.name() data: 0x06f...dde03 call to SaluToken.decimals
- CALL [call] from: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 to: SaluToken.decimals() data: 0x313...ce567

At the very bottom, there are links for 'Initialize as git repo', 'Did you know?', 'RemixAI Copilot (enabled)', and 'Scam Alert'.

Compile Smart Contract

The screenshot shows the Remix Ethereum IDE interface with several UI elements highlighted by red and blue boxes and numbered 1 through 4.

- 1**: A red box highlights the **Compile** icon in the sidebar.
- 2**: A red box highlights the **Compiler** dropdown menu in the sidebar, which is set to **0.8.26+commit.8a97fa7a**. A blue circle labeled **2** points to this dropdown.
- 3**: A red box highlights the **Contract** dropdown menu in the sidebar, which is set to **SaluToken (SaluToken_flattened.sol)**. A blue circle labeled **3** points to this dropdown.
- 4**: A red box highlights the **Compile SaluToken_flattened...** button in the sidebar. A blue circle labeled **4** points to this button.

The main workspace displays the Solidity code for the **SaluToken** contract, which is a flattened version of the **ERC20** standard. The code includes the **IERC20** interface and the **Transfer** event.

```
// SPDX-License-Identifier: MIT
// File: @openzeppelin/contracts/token/ERC20/IERC20.sol

// OpenZeppelin Contracts (last updated v5.0.0) (token/ERC20/IERC20.sol)

pragma solidity ^0.8.20;

/**
 * @dev Interface of the ERC20 standard as defined in the EIP.
 */
interface IERC20 {
    /**
     * @dev Emitted when `value` tokens are moved from one account (`from`) to
     * another (`to`).
     *
     * Note that `value` may be zero.
     */
    event Transfer(address indexed from, address indexed to, uint256 value);
}

/*
 * @dev Emitted when the allowance of a `spender` for an `owner` is set by
 * a call to `approve`. This allows reentrancy protection.
 */
event Approval(address indexed owner, address indexed spender, uint256 value);
```

At the bottom of the interface, there is a note about accessible libraries:

- [web3.js](#)
- [ethers.js](#)

Other footer elements include "Did you know?", "Initialize as git repo", "RemixAI Copilot (enabled)", and "Scam Alert".

Deploy an ERC-20 Token

The screenshot shows the Remix Ethereum IDE interface with several numbered callouts indicating steps in the deployment process:

- 1**: A red box highlights the **Deploy** icon in the sidebar.
- 2**: A red box highlights the **Injected Provider - MetaMask** dropdown, which is set to **Sepolia (11155111) network**.
- 3**: A red box highlights the **CONTRACT** dropdown, which is set to **SaluToken - SaluToken_flattened.sol**.
- 4**: A red box highlights the **Deploy** button at the bottom of the sidebar.
- 5**: A red box highlights the **確認** (Confirm) button in the deployment confirmation dialog.

The central workspace displays the Solidity code for the `SaluToken` contract, which is a flattened version of the OpenZeppelin ERC20 standard. The code includes imports for `IERC20`, `SafeMath`, and `ERC20Detailed`. The `Deploy & Run Transactions` panel shows the environment set to Sepolia, the account set to `0x4CE...6ae71` with 1.518070624198230502 ether, and a gas limit of 3000000 Wei. The deployment status shows "creation of SaluToken pending...".

Deploy and Run Transaction

The screenshot shows the process of deploying a smart contract (SaluToken) using the Ethereum IDE (Remix) and interacting with it via a wallet (Salu1).

Remix - Ethereum IDE (Left):

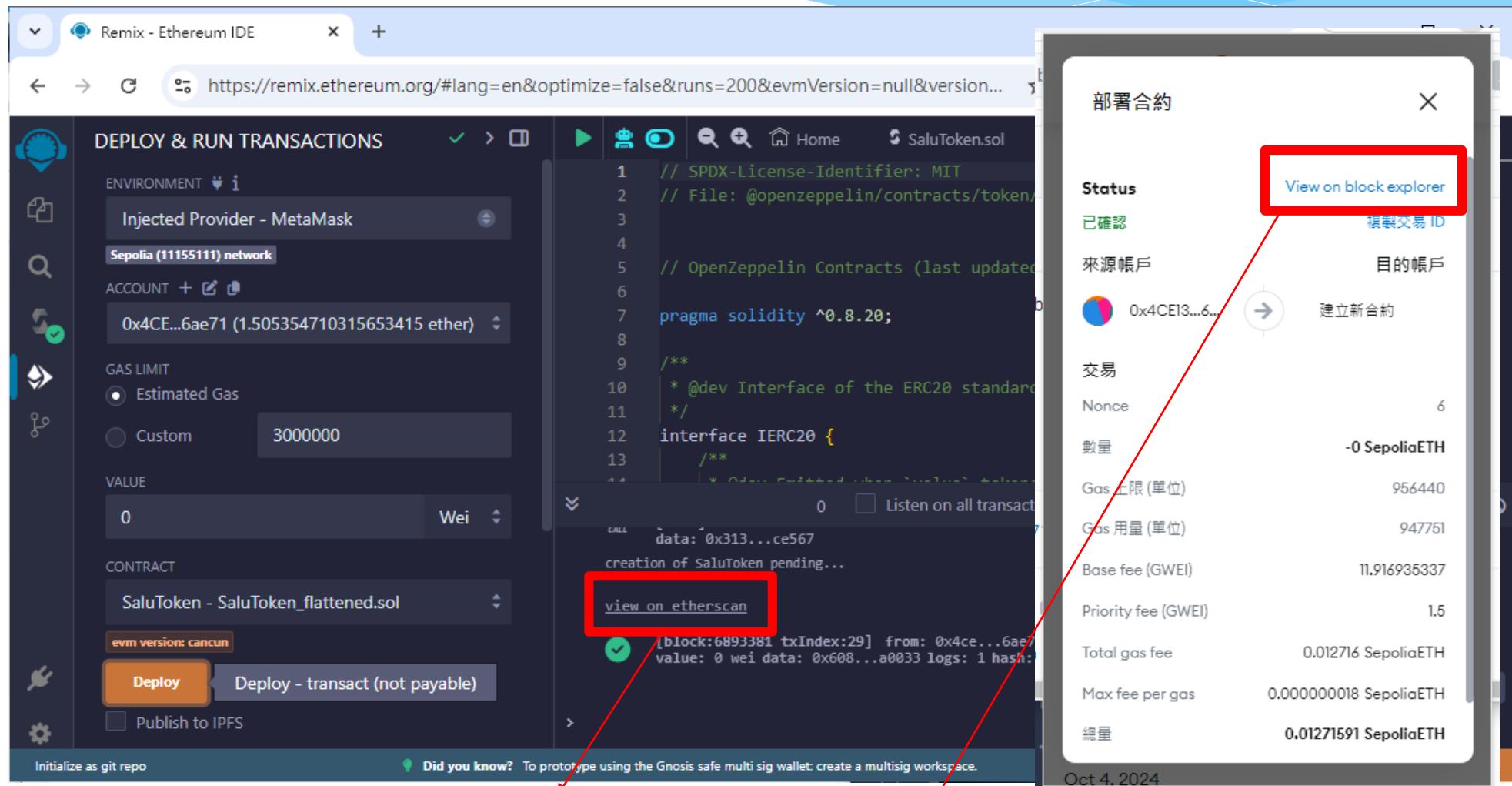
- ENVIRONMENT:** Sepolia (11155111) network
- ACCOUNT:** 0x4CE13...6ae71 (1.505354710315653415 ether)
- GAS LIMIT:** Estimated Gas (3000000 Wei)
- CONTRACT:** SaluToken - SaluToken_flattened.sol
- Deploy** button

Salu1 Wallet (Right):

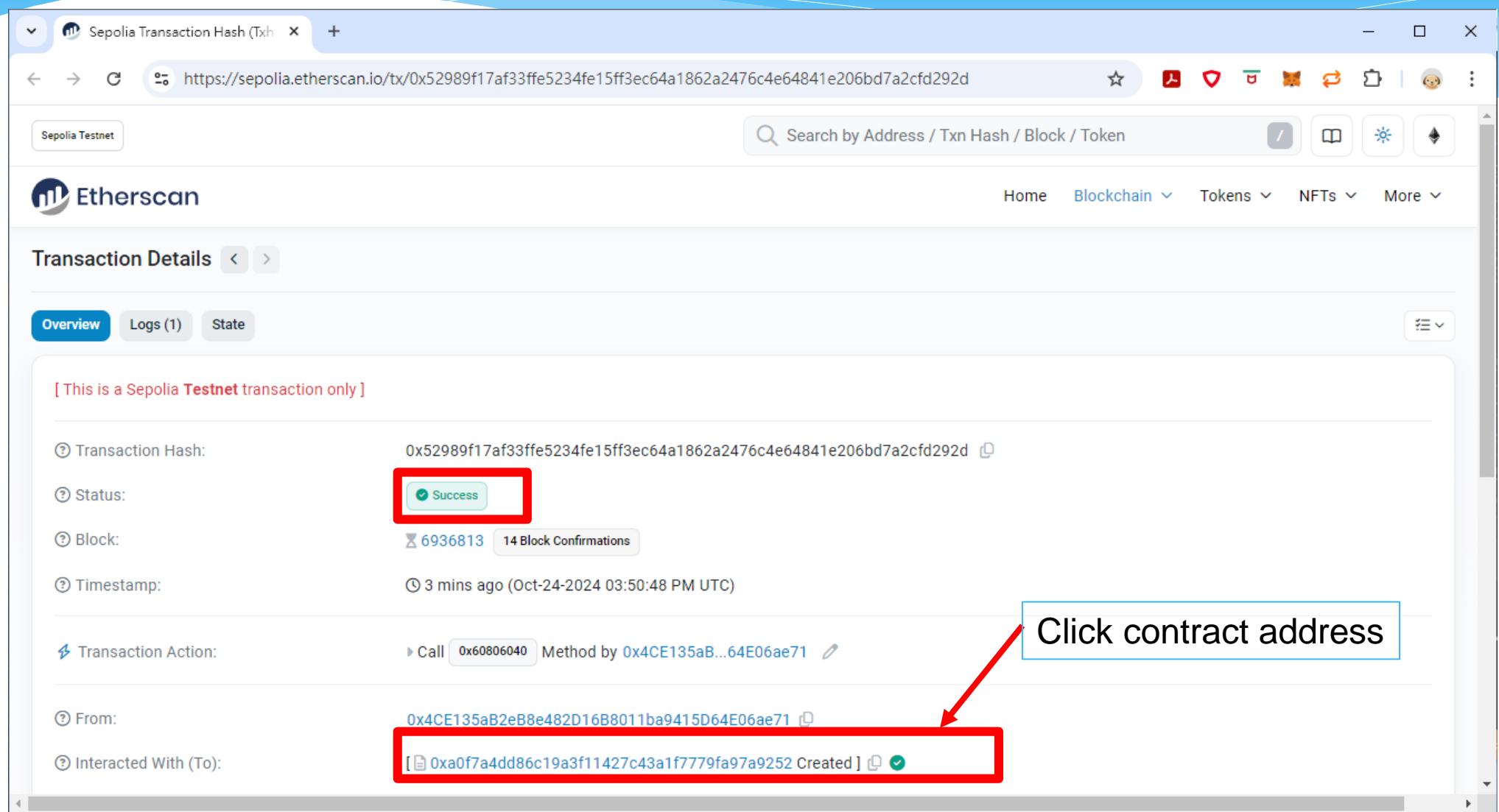
- Address:** 0x4CE13...6ae71
- Balances:** 1.6032 SepoliaETH
- Transactions:**
 - Oct 24, 2024: 部署合約 已確認 (Deployment confirmed)
 - Oct 18, 2024: 部署合約 已確認 (Deployment confirmed)
 - Oct 17, 2024: 部署合約 已確認 (Deployment confirmed)

A red arrow points from the transaction hash in the Remix IDE's transaction history to the corresponding deployment confirmation in the Salu1 wallet.

Deploy and Run Transaction



View Transaction on Sepolia



The screenshot shows a web browser displaying the Etherscan Sepolia Testnet transaction details for the hash `0x52989f17af33ffe5234fe15ff3ec64a1862a2476c4e64841e206bd7a2cf292d`. The transaction was successful, having 14 block confirmations at the time of capture. A red box highlights the status button, and another red box highlights the contract address in the 'Interacted With (To)' field. A blue callout box with the text 'Click contract address' points to the highlighted contract address.

[This is a Sepolia Testnet transaction only]

Transaction Hash: `0x52989f17af33ffe5234fe15ff3ec64a1862a2476c4e64841e206bd7a2cf292d`

Status: Success

Block: 6936813 | 14 Block Confirmations

Timestamp: 3 mins ago (Oct-24-2024 03:50:48 PM UTC)

Transaction Action: Call `0x60806040` Method by `0x4CE135aB...64E06ae71`

From: `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71`

Interacted With (To): `[0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252 Created]`

Click contract address

Smart Contract Information

The screenshot shows the Etherscan interface for a Sepolia Testnet contract at address `0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252`. The page displays various sections of the contract's information:

- Contract Address:** `0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252` (highlighted with a red box).
- More Info:** Contains the **CONTRACT CREATOR** (`0x4CE135aB...64E06ae71`) and the **TOKEN TRACKER** (`SaluToken (ST)`). A blue box labeled "click" points to the TOKEN TRACKER section.
- Multichain Info:** Shows "N/A".
- Transactions:** A table showing the latest transaction from the contract.

Transaction Hash	Method	Block	Age	From	To	Amount	Txn Fee
0x52989f17af3...	0x60806040	6936813	6 mins ago	0x4CE135aB...64E06ae71	Contract Creation	0 ETH	0.00612251

Smart Contract Information

Screenshot of the Etherscan Token Tracker for SaluToken (ST) on the Sepolia Testnet.

The page URL is <https://sepolia.etherscan.io/token/0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252>.

Token Details: SaluToken (ST)

ERC-20

Overview

- MAX TOTAL SUPPLY: 100,000,000 ST (highlighted with a red box)
- HOLDERS: 1
- TOTAL TRANSFERS: 1

Market

- ONCHAIN MARKET CAP: \$0.00
- CIRCULATING SUPPLY MARKET CAP: -

Other Info

- TOKEN CONTRACT (WITH 18 DECIMALS): [0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252](#)

Transfers (selected tab), **Holders**, **Contract**

A total of 1 transaction found.

Transaction Hash	Method	Block	Age	From	To	Amount
0x52989f17af3...	0x60806040	6936813	10 mins ago	0x00000000...0000000000	0x4CE135aB...64E06ae71	100,000,000

View & Publish

Contract Address 0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252 #code

Sepolia Testnet

Etherscan

Contract 0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252

Overview

ETH BALANCE
0 ETH

More Info

CONTRACT CREATOR
0x4CE135aB...64E06ae71 at txn 0x52989f17af3...

TOKEN TRACKER
SaluToken (ST)

Multichain Info
N/A

Transactions Token Transfers (ERC-20) Contract Events

Are you the contract creator? Verify and Publish your contract source code today!

Decompile Bytecode Switch to Opcodes View Similar Contracts

View & Publish Contract Source Code

The screenshot shows a web browser window titled "Verify & Publish Contract Sou" with the URL <https://sepolia.etherscan.io/verifyContract?a=0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252>. The page is titled "Verify & Publish Contract Source Code" and explains that source code verification provides transparency by matching uploaded code with the blockchain. It shows two steps: "Enter Contract Details" (selected) and "Verify & Publish". The "Enter Contract Details" step includes fields for Contract Address (0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252), Compiler Type (Solidity (Single file)), Compiler Version (v0.8.26+commit.8a97fa7a), Open Source License Type (3) MIT License (MIT), and checkboxes for "Uncheck to show all nightly commits" and "I agree to the terms of service". The "Continue" button at the bottom is highlighted with a red box.

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.](#)

1 Enter Contract Details — 2 Verify & Publish

Please enter the Contract Address you would like to verify
0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252

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 [\(i\)](#)
3) MIT License (MIT)

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

[Continue](#) [Reset](#)

View & Publish Contract Source Code

The screenshot shows a web browser window for the Sepolia Testnet on the Etherscan website. The URL in the address bar is <https://sepolia.etherscan.io/verifyContract-solc?a=0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252&c=v0.8.26%2bcommi...>. The page title is "Sepolia Solidity Contract Sour". The main heading is "Verify & Publish Contract Source Code". Below it, a subtext explains: "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." A "Read more." link is present. A note states: "A simple and structured interface for verifying smart contracts that fit in a single file." The interface has two steps: "1 Enter Contract Details" and "2 Verify & Publish". The "Enter Contract Details" step is active, showing a form titled "Upload Contract Source Code". It contains three numbered instructions: 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](#). A red box highlights the "Contract Address" field, which contains the value "0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252". Below it, the "Compiler Type" is listed as "SINGLE FILE / CONCATENATED METHOD" and the "Compiler Version" as "v0.8.26+commit.8a97fa7a".

Sepolia Solidity Contract Sour

https://sepolia.etherscan.io/verifyContract-solc?a=0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252&c=v0.8.26%2bcommi...

Sepolia Testnet

Etherscan

Search by Address / Txn Hash / Block / Token

Home Blockchain Tokens NFTs 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

Upload Contract Source Code

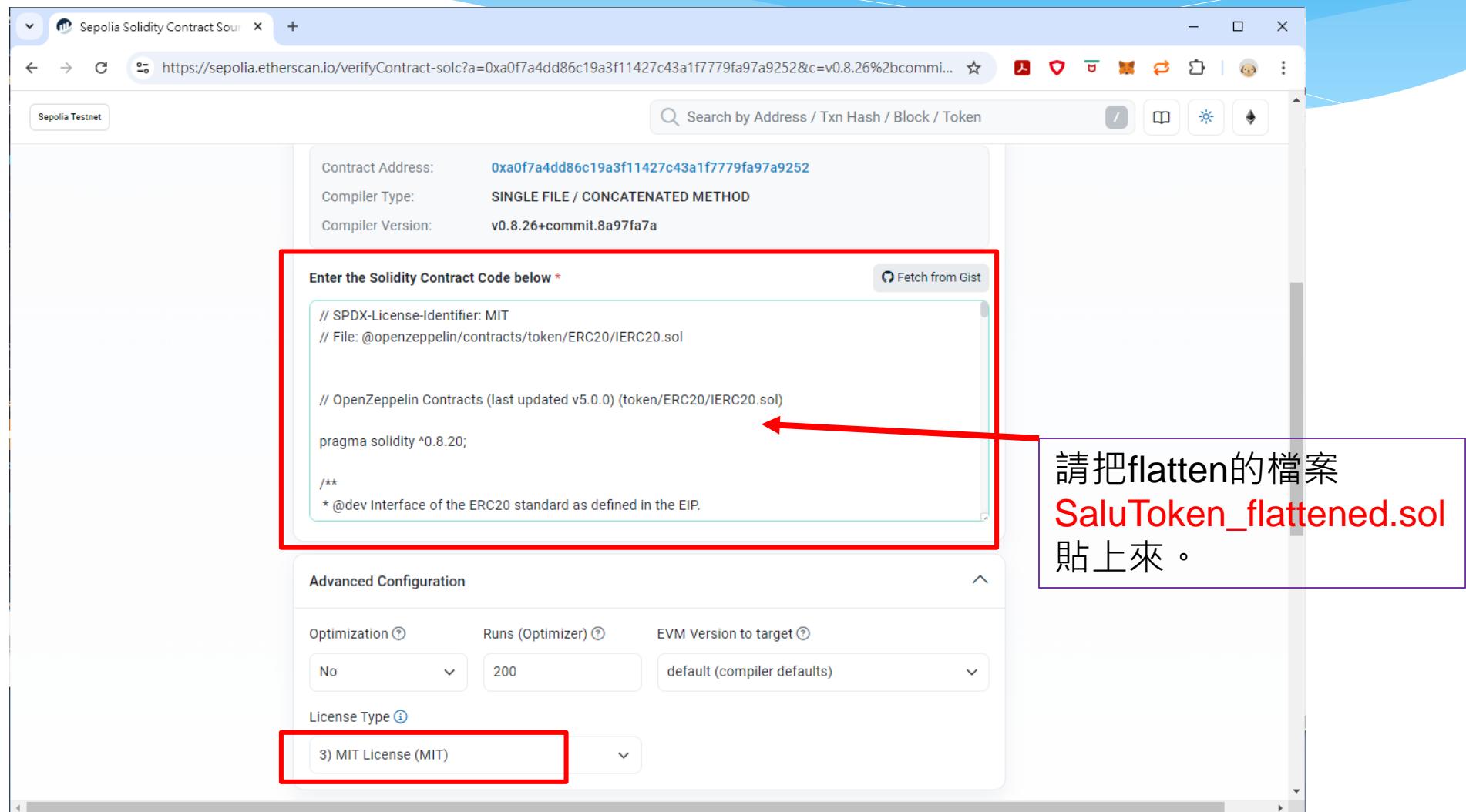
- If the contract compiles correctly at [REMIX](#), it should also compile correctly here.
- We have limited support for verifying contracts created by another contract and there is a timeout of up to 45 seconds for each contract compiled.
- For programmatic contract verification, check out the [Contract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

Contract Address: **0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252**

Compiler Type: SINGLE FILE / CONCATENATED METHOD

Compiler Version: v0.8.26+commit.8a97fa7a

View & Publish Contract Source Code



The screenshot shows a browser window titled "Sepolia Solidity Contract Sour" with the URL <https://sepolia.etherscan.io/verifyContract-solc?a=0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252&c=v0.8.26%2bcommit.8a97fa7a>. The page displays contract details: Contract Address: 0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252, Compiler Type: SINGLE FILE / CONCATENATED METHOD, and Compiler Version: v0.8.26+commit.8a97fa7a. A red box highlights the "Enter the Solidity Contract Code below" input field, which contains the following code:

```
// SPDX-License-Identifier: MIT
// File: @openzeppelin/contracts/token/ERC20/IERC20.sol

// OpenZeppelin Contracts (last updated v5.0.0) (token/ERC20/IERC20.sol)

pragma solidity ^0.8.20;

/***
 * @dev Interface of the ERC20 standard as defined in the EIP.
 */


```

A red arrow points from the text "請把flatten的檔案 SaluToken_flattened.sol 貼上來。" to the input field. Below the input field, another red box highlights the "3) MIT License (MIT)" dropdown menu under "License Type".

請把flatten的檔案
SaluToken_flattened.sol
貼上來。

Advanced Configuration

Optimization: No

Runs (Optimizer): 200

EVM Version to target: default (compiler defaults)

License Type: 3) MIT License (MIT)

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?". The page displays a "Constructor Arguments ABI-encoded" section for contracts created with constructor parameters. Below this is a "Contract Library Address" section for contracts using libraries. At the bottom, there is a success message from Cloudflare with a green checkmark icon and the text "成功!". A red box highlights the "Verify and Publish" button, which is blue with white text. To its right is a "Reset" button.

Constructor Arguments ABI-encoded
For contracts that were created with constructor parameters

Contract Library Address (for contracts that use libraries, supports up to 10 libraries)

成功!

CLOUDFLARE

Verify and Publish

Reset

View & Publish Contract Source Code

Sepolia Solidity Contract Sourc... <https://sepolia.etherscan.io/verifyContract-solc?a=0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252&c=v0.8.26%2bcommi...>

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

✓ Successfully generated Bytecode and ABI for Contract Address
[0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252]

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

Code Reader [?](#)

Prompt:

You are a conscientious blockchain security auditor. Review this smart contract's source code to suggest best practices it could follow and share any security concerns with the code.

click

View Contract Source Code

The screenshot shows a web browser window displaying the Etherscan interface for the SaluToken contract. The URL in the address bar is highlighted with a red box and points to <https://sepolia.etherscan.io/address/0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252#code>. The browser's title bar shows "SaluToken | Address 0xa0f7a4".

The page header includes tabs for "Code", "Read Contract", and "Write Contract", with "Code" being the active tab. It also features a search bar for "Search by Address / Txn Hash / Block / Token" and a "Search Source Code" button.

Contract details shown:

- Contract Name:** SaluToken (highlighted with a red box)
- Compiler Version:** v0.8.26+commit.8a97fa7a
- Optimization Enabled:** No with 200 runs
- Other Settings:** default evmVersion, MIT license

The "Contract Source Code (Solidity)" section displays the following Solidity code, which is also highlighted with a red box:

```
1 /**
2  *Submitted for verification at Etherscan.io on 2024-10-24
3 */
4
5 // SPDX-License-Identifier: MIT
6 // File: @openzeppelin/contracts/token/ERC20/IERC20.sol
7
8
9 // OpenZeppelin Contracts (last updated v5.0.0) (token/ERC20/IERC20.sol)
10
11 pragma solidity ^0.8.20;
12
13 /**
14  * @dev Interface of the ERC20 standard as defined in the EIP.
15  */
16 interface IERC20 {
17     /**
18      * @dev Emitted when `value` tokens are moved from one account (`from`) to
19      * another (`to`).
20  }
```

View & Publish Contract Source Code

The screenshot shows the Etherscan.io interface for the contract address 0xa0f7a4. The 'Contract' tab is selected, indicated by a red box. Below it, three buttons are highlighted with a red box: 'Code', 'Read Contract', and 'Write Contract'. The 'Code' button is currently active.

Overview
ETH BALANCE
0 ETH

More Info
CONTRACT CREATOR
0x4CE135aB...64E06ae71 at txn 0x52989f17af3...
TOKEN TRACKER
SaluToken (ST)

Multichain Info
N/A

Transactions **Token Transfers (ERC-20)** **Contract** **Events**

Code **Read Contract** **Write Contract**

Contract Source Code Verified (Exact Match)

Contract Name: SaluToken Optimization Enabled: No with 200 runs

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

Contract Source Code (Solidity)

IDE Outline More Options



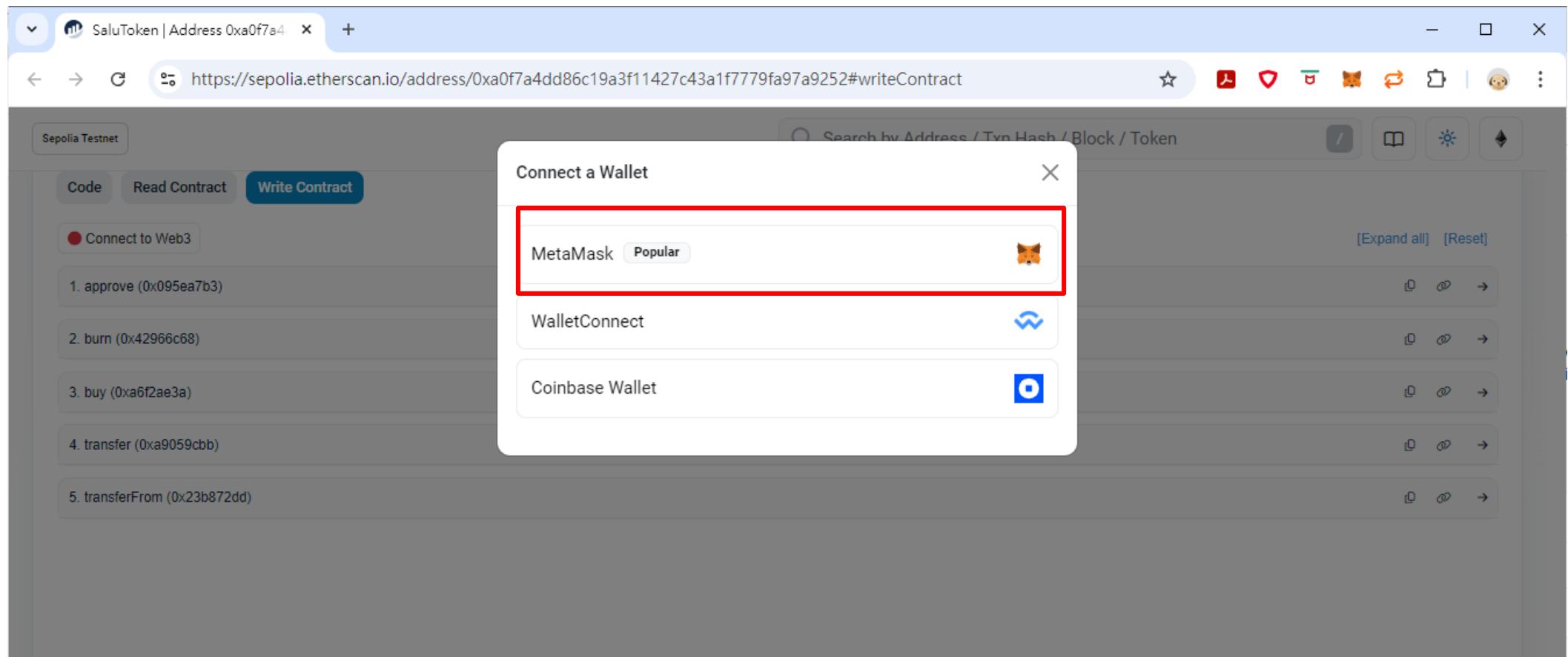
Buy Token

Connect to Web3

The screenshot shows a web browser window with the following details:

- Title Bar:** SaluToken | Address 0xa0f7a4
- URL:** https://sepolia.etherscan.io/address/0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252#writeContract
- Header:** Sepolia Testnet
- Buttons:** Code, Read Contract, Write Contract (highlighted with a red box)
- Text:** "Connect to Web3" (highlighted with a red box) and "sepolia.etherscan.io 顯示"
- List:** 1. approve (0x095ea7b3), 2. burn (0x42966c68), 3. buy (0xa6f2ae3a), 4. transfer (0xa9059cbb), 5. transferFrom (0x23b872dd)
- Message:** 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.
- Buttons:** 確定 (highlighted with a red box) and 取消

Connect a Wallet-MetaMask



The screenshot shows a web browser window with the URL <https://sepolia.etherscan.io/address/0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252#writeContract>. The page is titled "SaluToken | Address 0xa0f7a4". The "Write Contract" tab is active. A modal window titled "Connect a Wallet" is displayed, listing three options: "MetaMask" (Popular), "WalletConnect", and "Coinbase Wallet". The "MetaMask" option is highlighted with a red box. The background shows a list of transactions with their hex codes and details.

Connected Web3 using account

The image shows a screenshot of a web browser and a wallet application side-by-side, illustrating the connection between them.

Web Browser (Left): The browser window displays the Sepolia Testnet version of Etherscan for the SaluToken contract at address 0xa0f7a4. The "Write Contract" tab is selected. A red box highlights the status bar message "Connected - Web3 [0x4ce1...ae71]".

Wallet Application (Right): The wallet interface shows a balance of 1.6032 SepoliaETH. It features a "Portfolio" section with icons for Buy & Sell, Swap, Bridge, Send, and Receive. Below this, a "Transactions" section lists two entries:

- Oct 24, 2024: 部署合約 已確認 -0 SepoliaETH -0 SepoliaETH
- Oct 18, 2024: 部署合約 已確認 -0 SepoliaETH -0 SepoliaETH

A red arrow points from the "Connected - Web3" message in the browser to the wallet's account address "0x4CE13...ae71" in the top right corner.

Write Contract - Buy

此處的buy指的是付款0.1 Sepolia ETH買Token.

計算後可買1,000,000個token

140
<https://sepolia.etherscan.io/address/0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252#writeContract>

Write Contract - Buy

The image shows a blockchain wallet interface with two main panels. The left panel displays a summary of SepoliaETH balance (1.503 SepoliaETH) and various transaction options like Buy & Sell, Swap, Bridge, Send, and Receive. It also lists recent transactions, with one specific 'Buy' transaction from Oct 25, 2024, highlighted with a red box. This transaction is shown in more detail in the right panel.

Left Panel (Summary):

- Salu 1 (Account)
- 0x4CE13...6ae71 (Address)
- 1.503 SepoliaETH
- Portfolio (Link)
- Buy & Sell, Swap, Bridge, 發送, 接收
- Tokens, NFTs, 交易紀錄 (selected)
- Oct 25, 2024: Buy (已確認)
- Oct 24, 2024: 部署合約 (已確認)
- Oct 18, 2024: (empty)

Right Panel (Detailed View):

Buy

Status	View on block explorer
已確認	複製交易 ID
來源帳戶	目的帳戶
0x4CE13...6ae71	0xa0F7a.....
交易	
Nonce	10
數量	-0.1 SepoliaETH
Gas 上限 (單位)	55699
Gas 用量 (單位)	36769
Base fee (GWEI)	4.193161116
Priority fee (GWEI)	1.5
Total gas fee	0.0000209 SepoliaETH
Max fee per gas	0.000000008 SepoliaETH
總量	0.10020933 SepoliaETH

Oct 18, 2024

Write Contract - Buy

The screenshot shows a web browser window displaying the Sepolia Testnet transaction details on Etherscan. The transaction hash is `0xf137f509fcd21037fe7121f7b529394c95a597b292df2716e7e77373ef512aa4`. The status is listed as "Success". The transaction was included in block `6937090` with 24 confirmations. It was timestamped 4 minutes ago on Oct-24-2024 at 04:49:24 PM UTC. The transaction action was a call to the `Buy` function of contract `0x4CE135aB...64E06ae71` on address `0xa0F7a4DD...FA97A9252`. The transaction originated from address `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71` and was sent to address `0xa0F7a4DD86c19a3F11427C43a1f7779FA97A9252`.

Sepolia Transaction Hash (Txn Hash)

https://sepolia.etherscan.io/tx/0xf137f509fcd21037fe7121f7b529394c95a597b292df2716e7e77373ef512aa4

Etherscan

Transaction Details

Overview Logs (1) State

[This is a Sepolia Testnet transaction only]

② Transaction Hash: `0xf137f509fcd21037fe7121f7b529394c95a597b292df2716e7e77373ef512aa4`

② Status: Success

② Block: `6937090` 24 Block Confirmations

② Timestamp: 4 mins ago (Oct-24-2024 04:49:24 PM UTC)

④ Transaction Action: Call `Buy` Function by `0x4CE135aB...64E06ae71` on `0xa0F7a4DD...FA97A9252`

② From: `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71`

② To: `0xa0F7a4DD86c19a3F11427C43a1f7779FA97A9252`

Write Contract - Buy

The screenshot shows the Sepolia Testnet version of the Etherscan interface for the SaluToken contract at address 0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252. A red box highlights the 'Buy' method in the transaction table.

Contract Overview:

- ETH BALANCE:** 0.1 ETH
- CONTRACT CREATOR:** 0x4CE135aB...64E06ae71 at tx 0x52989f17af3...
- TOKEN TRACKER:** SaluToken (ST)

Transactions:

Transaction Hash	Method	Block	Age	From	To	Amount	Txn Fee
0xf137f509fc...	Buy	6937090	7 mins ago	0x4CE135aB...64E06ae71	0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252	0.1 ETH	0.00020933
0x52989f17af3...	0x60806040	6936813	1 hr ago	0x4CE135aB...64E06ae71	Create: SaluToken	0 ETH	0.00612251

Write Contract - Buy

The screenshot shows a browser window for the Sepolia Testnet Token Tracker. The URL is <https://sepolia.etherscan.io/token/0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252>. The page displays token statistics: MAX TOTAL SUPPLY (101,000,000 ST), ONCHAIN MARKET CAP (\$0.00), and TOKEN CONTRACT (WITH 18 DECIMALS) (0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252). It also shows HOLDER COUNT (1) and TOTAL TRANSFERS (2). A red box highlights the MAX TOTAL SUPPLY. A red callout box with the text "此次花費0.1 Sepolia ETH 計算後可買1,000,000個token" points to the transaction details table. The table lists two transactions: one from 0xf137f509fc... to 0x4CE135aB... with an amount of 1,000,000, and another from 0x52989f17af3... to 0x4CE135aB... with an amount of 100,000,000. Red boxes highlight the "Buy" method for the first transaction and the amount 1,000,000.

MAX TOTAL SUPPLY
101,000,000 ST

ONCHAIN MARKET CAP
\$0.00

TOKEN CONTRACT (WITH 18 DECIMALS)
0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252

HOLDERS
1

TOTAL TRANSFERS
2

此次花費0.1 Sepolia ETH
計算後可買1,000,000個token

Transaction Hash	Method	Block	Age	From	To	Amount
0xf137f509fc...	Buy	6937090	8 mins ago	0x00000000...00000000	0x4CE135aB...64E06ae71	1,000,000
0x52989f17af3...	0x60806040	6936813	1 hr ago	0x00000000...00000000	0x4CE135aB...64E06ae71	100,000,000



Burn Token

Connect to Web3

Sepolia Testnet

Code Read Contract Write Contract

Connect to Web3

1. approve (0x095ea7b3)
2. burn (0x42966c68)
3. buy (0xa6f2ae3a)
4. transfer (0xa9059cbb)
5. transferFrom (0x23b872dd)

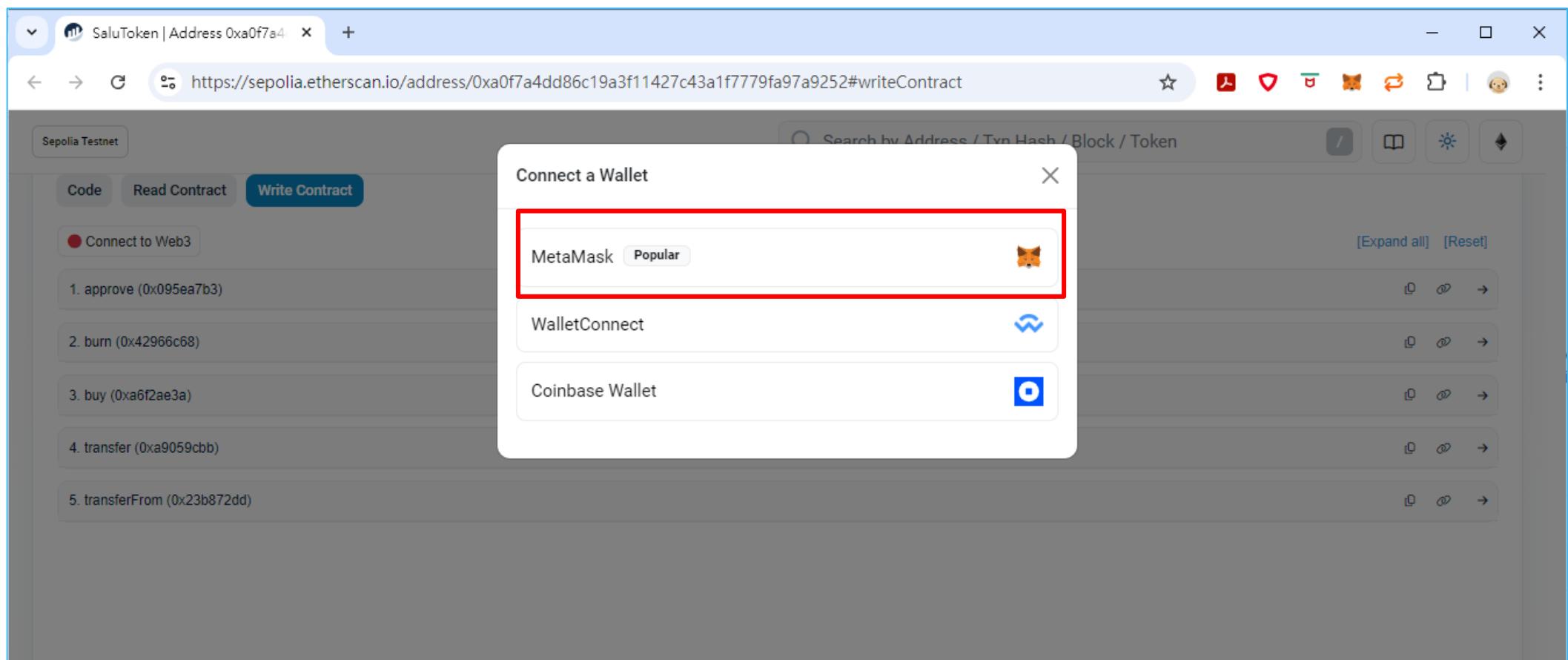
sepolia.etherscan.io 顯示
Please take note that this is a beta version feature and is provided on an "as is" and "as available" basis. Etherscan does not give any warranties and will not be liable for any loss, direct or indirect through continued use of this feature.

[Expand all] [Reset]

確定 取消

https://sepolia.etherscan.io/address/0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252#writeContract

Connect a Wallet-MetaMask



The screenshot shows a web browser window with the URL <https://sepolia.etherscan.io/address/0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252#writeContract>. The page is titled "SaluToken | Address 0xa0f7a4". The "Write Contract" tab is active. A modal window titled "Connect a Wallet" is displayed, listing three options: "MetaMask" (Popular), "WalletConnect", and "Coinbase Wallet". The "MetaMask" option is highlighted with a red box.

Connected Web3 using account

SaluToken | Address 0xa0f7a4

https://sepolia.etherscan.io/address/0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252#writeContract

Sepolia Testnet

Code Read Contract Write Contract

Connected - Web3 [0x4ce1...ae71]

1. approve (0x095ea7b3)
2. burn (0x42966c68)
3. buy (0xa6f2ae3a)
4. transfer (0xa9059ccb)
5. transferFrom (0x23b872dd)

Search by Address

1.6032 SepoliaETH

Portfolio

Buy & Sell Swap Bridge 發送 接收

Tokens NFTs 交易紀錄

Oct 24, 2024 部署合約 已確認 -0 SepoliaETH -0 SepoliaETH

Oct 18, 2024 部署合約 已確認 -0 SepoliaETH -0 SepoliaETH

Oct 17, 2024 部署合約 已確認 -0 SepoliaETH -0 SepoliaETH

0x4CE13...ae71

Write Contract - burn

The screenshot shows two windows side-by-side. On the left is the 'SaluToken (ST) Token Tracker' on the Sepolia Testnet. It displays the total supply of 101,000,000 ST tokens, which is highlighted with a red box. Below it, there are statistics for Holders (1) and Total Transfers (2). The 'Contract' tab is selected, showing a list of actions: 1. approve (0x095ea7b3) and 2. burn (0x42966c68). The 'burn' action has an 'amount (uint256)' field containing '2500', also highlighted with a red box. A note below says 'Please enter a positive integer'. At the bottom is a 'Write' button. On the right is the 'MetaMask' wallet interface. It shows the account '0xa0f7a...A9252' and the transaction '0xa0f7a...A9252 : BURN'. It provides estimated changes ('You send - <0.000001 0xa0f7a...A9252'), an estimated fee ('0.0001704 SepoliaETH'), and a maximum fee ('Max fee: 0.00020297 SepoliaETH'). The '確認' (Confirm) button is highlighted with a red box.

Write Contract - Burn

1.5029 SepoliaETH

Portfolio [View on block explorer](#)

Buy & Sell Swap Bridge 發送 接收

Tokens NFTs 交易紀錄

Oct 25, 2024

Burn 已確認

-0 SepoliaETH
-0 SepoliaETH

Buy 已確認

-0.1 SepoliaETH
-0.1 SepoliaETH

Oct 24, 2024

部署合約 -0 SepoliaETH

Burn

Status: 已確認 [View on block explorer](#) | 檢視交易 ID

來源帳戶: 0x4CE13...6ae71 | 目的帳戶: 0xa0F7a....

交易:

Nonce	11
數量	-0 SepoliaETH
Gas 上限 (單位)	51573
Gas 用量 (單位)	34024
Base fee (GWEI)	1.593323466
Priority fee (GWEI)	1.5
Total gas fee	0.000105 SepoliaETH
Max fee per gas	0.000000004 SepoliaETH
總量	0.00010525 SepoliaETH

Write Contract - Burn

The screenshot shows a browser window displaying the Etherscan transaction details for a Sepolia Testnet transaction. The transaction hash is `0x3778be090488a9b1829cf514ef419d6106412d072970139a12e120479244beab`. The status is marked as "Success" with a red box highlighting it. The transaction was included in block `6937236` with 8 block confirmations. It was timestamped 1 min ago (Oct-24-2024 05:21:12 PM UTC). The transaction action was a "Burn" function call by `0x4CE135aB...64E06ae71` on contract `0xa0F7a4DD...FA97A9252`. The transaction originated from address `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71` and interacted with `0xa0F7a4DD86c19a3F11427C43a1f7779FA97A9252`.

Sepolia Transaction Hash (Txh) <https://sepolia.etherscan.io/tx/0x3778be090488a9b1829cf514ef419d6106412d072970139a12e120479244beab>

Etherscan

Transaction Details

Overview Logs (1) State

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

② Transaction Hash: `0x3778be090488a9b1829cf514ef419d6106412d072970139a12e120479244beab`

② Status: Success

② Block: `6937236` 8 Block Confirmations

② Timestamp: 1 min ago (Oct-24-2024 05:21:12 PM UTC)

② Transaction Action: Call Burn Function by `0x4CE135aB...64E06ae71` on `0xa0F7a4DD...FA97A9252`

② From: `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71`

② Interacted With (To): `0xa0F7a4DD86c19a3F11427C43a1f7779FA97A9252`

Write Contract - Burn

SaluToken | Address 0xa0f7a4

https://sepolia.etherscan.io/address/0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252

Contract 0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252

Source Code

Overview

ETH BALANCE
0.1 ETH

More Info

CONTRACT CREATOR
0x4CE135aB...64E06ae71 at txn 0x52989f17af...

TOKEN TRACKER
SaluToken (ST)

Multichain Info
N/A

Transactions Token Transfers (ERC-20) Contract Events

Latest 3 from a total of 3 transactions

Transaction Hash	Method	Block	Age	From	To	Amount	Txn Fee
0x3778be0904...	Burn	6937236	4 mins ago	0x4CE135aB...64E06ae71	IN	0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252	0 ETH 0.00010524
0xf137f509fc...	Buy	6937090	36 mins ago	0x4CE135aB...64E06ae71	IN	0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252	0.1 ETH 0.00020933
0x52989f17af3...		0x60806040	1 hr ago	0x4CE135aB...64E06ae71	IN	Create: SaluToken	0 ETH 0.00612251

Write Contract - Burn

The screenshot shows the Sepolia Testnet SaluToken (ST) Token Tracker on Etherscan. Key statistics include MAX TOTAL SUPPLY (100,999,999.99999999... ST), HOLDERS (1), and TOTAL TRANSFERS (3). The transaction history lists three entries: a Burn transaction (Method: Burn, Block: 6937236, Age: 7 mins ago, From: 0x4CE135aB...64E06ae71, To: 0x00000000...00000000, Amount: 0.0000000000000025), a Buy transaction (Method: Buy, Block: 6937090, Age: 39 mins ago, From: 0x00000000...00000000, To: 0x4CE135aB...64E06ae71, Amount: 1,000,000), and another Burn transaction (Method: 0x60806040, Block: 6936813, Age: 1 hr ago, From: 0x00000000...00000000, To: 0x4CE135aB...64E06ae71, Amount: 100,000,000).

MAX TOTAL SUPPLY
100,999,999.99999999... ST

HOLDERS
1

TOTAL TRANSFERS
3

ONCHAIN MARKET CAP
\$0.00

CIRCULATING SUPPLY MARKET CAP

TOKEN CONTRACT (WITH 18 DECIMALS)
0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252

A total of 3 transactions found

Transaction Hash	Method	Block	Age	From	To	Amount
0x3778be0904...	Burn	6937236	7 mins ago	0x4CE135aB...64E06ae71	0x00000000...00000000	0.0000000000000025
0xf137f509fc...	Buy	6937090	39 mins ago	0x00000000...00000000	0x4CE135aB...64E06ae71	1,000,000
0x52989f17af3...	0x60806040	6936813	1 hr ago	0x00000000...00000000	0x4CE135aB...64E06ae71	100,000,000

此次burn 2500

Download Page Data

Write Contract - Burn

The screenshot shows the Etherscan interface for the SaluToken (ST) token on the Sepolia Testnet. The token contract address is listed as `0xa0f7a4dd86c19a3f11427c43a1f7779fa97a9252`. The page displays three main sections: Overview, Market, and Other Info. In the Overview section, it shows 1 holder and 3 total transfers. The Market section shows an on-chain market cap of \$0.00. The Other Info section provides the token contract address and its decimal precision (18). Below these sections, a table lists three transactions. The first transaction is highlighted with a red box and labeled 'Copy address' with a red arrow pointing to the 'From' field. The second transaction is highlighted with a red box and labeled 'Paste address' with a red arrow pointing to the 'To' field. The third transaction is also highlighted with a red box and labeled 'Find' with a red arrow pointing to the search bar.

Transaction Hash	Method	Block	Age	From	To	Amount
0x3778be0904...	Burn	6937236	3 days ago	0x4CE135aB...64E06ae71	0x00000000...00000000	0.0000000000000025
0xf137f509fc...	Buy	6937090	3 days ago	0x00000000...00000000	0x4CE135aB...64E06ae71	1,000,000
0x52989f17af3...	0x60806040	6936813	3 days ago	0x00000000...00000000	0x4CE135aB...64E06ae71	100,000,000

Write Contract - Burn

The screenshot shows the Etherscan interface for the SaluToken (ST) contract. Key elements highlighted with red boxes and annotations are:

- Token SaluToken (ST)**: The token name is highlighted.
- MAX TOTAL SUPPLY**: 100,999,999,9999999... ST
- FILTERED BY TOKEN HOLDER**: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71
- BALANCE**: 1234567890123456
100,999,999,99999999999975 ST
- Contract Address**: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71
- Amount**: 123456789012345678
0.0000000000000025 → X 10¹⁸ = 2500
- Text Annotation**: 此次burn 2500

Below the token details, a table lists three transactions:

Transaction Hash	Method	Block	Age	From	To	Amount
0x3778be0904...	Burn	6937236	3 days ago	0x4CE135aB...64E06ae71	0x00000000...00000000	123456789012345678 0.0000000000000025 → X 10 ¹⁸ = 2500
0xf137f509fc...	Buy	6937090	3 days ago	0x00000000...00000000	0x4CE135aB...64E06ae71	1,000,000
0x52989f17af3...	0x60806040	6936813	3 days ago	0x00000000...00000000	0x4CE135aB...64E06ae71	100,000,000



Security and Access Control

Security and Access Control

- Access control, or “who is allowed to do this thing”, is incredibly important when dealing with smart contracts. The access control of your contract may **govern who can mint tokens, vote on proposals, freeze transfers, and many other things**.
- It is therefore **critical** to understand how you implement it, lest someone else **steals your whole system**.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts@4.9.0/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts@4.9.0/access/Ownable.sol";

contract SaluToken is ERC20, Ownable{
    constructor() ERC20("SaluToken", "ST") {}

    function issueToken() public onlyOwner{
        _mint(msg.sender, 1000 * (10 ** 18));
    }
}
```

使用v4.9.0版本

1. The most common and basic form of access control is the concept of **ownership** where there's only an account allowed to perform sensitive tasks on a contract (the owner). This approach is perfectly reasonable for contracts that have a single administrative user.
2. OpenZeppelin provides **Ownable** for implementing ownership in contracts.

By default, the owner of an **Ownable contract** is the account that **deployed** it, which is usually exactly what you want.

onlyOwner will run every time `issueToken()` gets called, verifying if the caller is the owner of the contract.

Security and Access Control

- Access control, or “who is allowed to do this thing”, is incredibly important when dealing with smart contracts. The access control of your contract may **govern who can mint tokens, vote on proposals, freeze transfers, and many other things**.
- It is therefore **critical** to understand how you implement it, lest someone else **steals your whole system**.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts/access/Ownable.sol";

contract SaluToken is ERC20, Ownable(address(msg.sender)) {
    constructor() ERC20("SaluToken", "ST") {}

    function issueToken() public onlyOwner{
        _mint(msg.sender, 1000 * (10 ** 18));
    }
}
```

預設使用目前
最新版本v5.1.0

1. The most common and basic form of access control is the concept of **ownership** where there's only an account allowed to perform sensitive tasks on a contract (the owner). This approach is perfectly reasonable for contracts that have a single administrative user.
2. OpenZeppelin provides **Ownable** for implementing ownership in contracts.

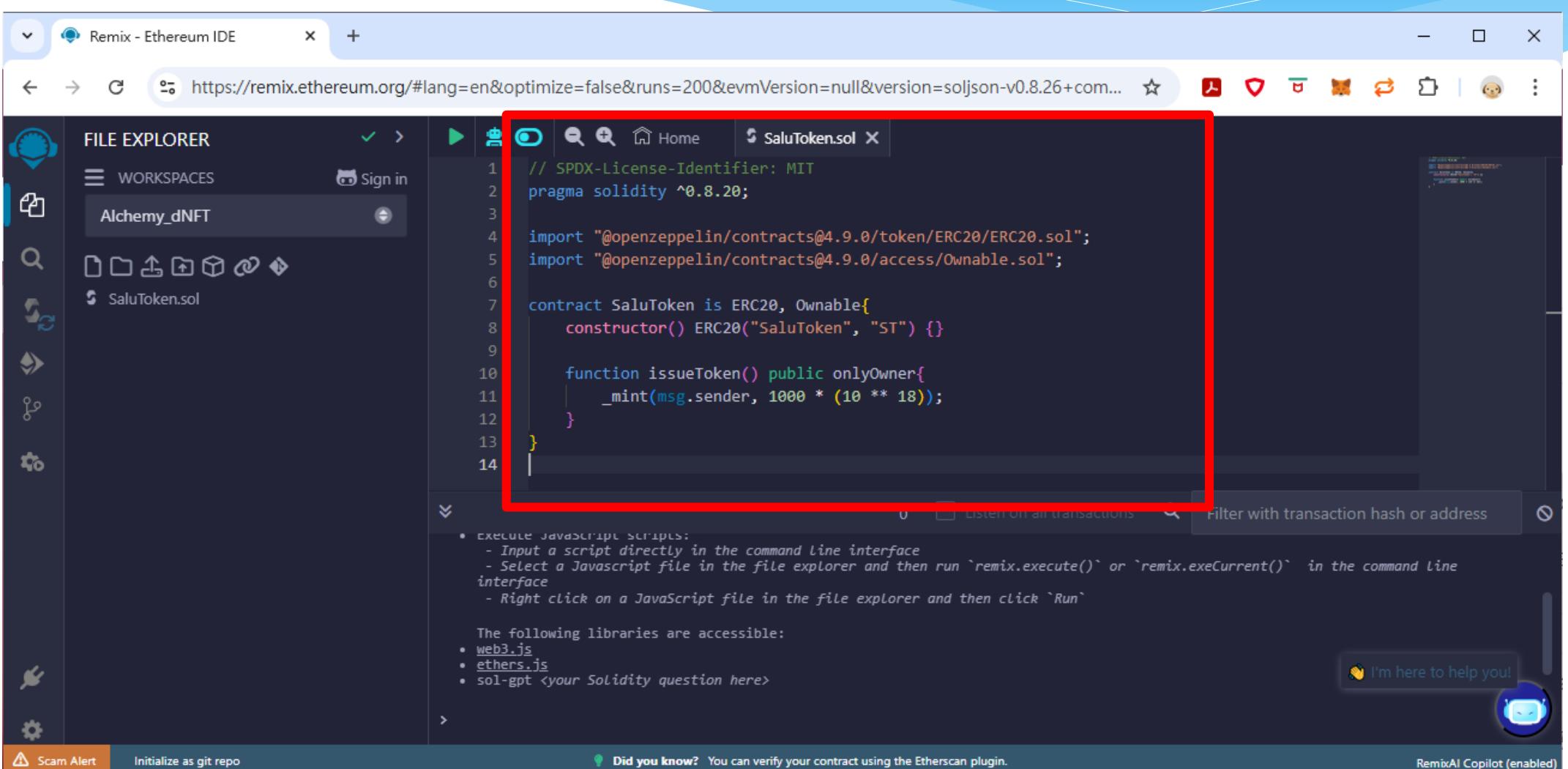
By default, the owner of an **Ownable contract** is the account that **deployed** it, which is usually exactly what you want.

onlyOwner will run every time issueToken() gets called, verifying if the caller is the owner of the contract.



Openzeppelin v4.9.0

Modify SaluToken.sol



The screenshot shows the Remix Ethereum IDE interface. The central area displays the Solidity code for the `SaluToken.sol` contract. A red box highlights the constructor and the `issueToken()` function. The code is as follows:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts@4.9.0/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts@4.9.0/access/Ownable.sol";

contract SaluToken is ERC20, Ownable{
    constructor() ERC20("SaluToken", "ST") {}

    function issueToken() public onlyOwner{
        _mint(msg.sender, 1000 * (10 ** 18));
    }
}
```

The Remix interface includes a FILE EXPLORER sidebar, a toolbar with various icons, and a bottom navigation bar with links like "Scam Alert", "Initialize as git repo", "Did you know?", and "RemixAI Copilot (enabled)". A tooltip message "I'm here to help you!" is visible in the bottom right corner.

Copy SPDX-License

The screenshot shows the Remix Ethereum IDE interface. A Solidity file named `SaluToken.sol` is open in the code editor. The file contains the following code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts@4.9.0/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts@4.9.0/access/Ownable.sol";

contract SaluToken is ERC20, Ownable{
    constructor() ERC20("SaluToken", "ST") {}

    function issueToken() public onlyOwner{
        _mint(msg.sender, 1000 * (10 ** 18));
    }
}
```

Two numbered circles indicate the steps to copy the SPDX license identifier:

- Circle 1: Surrounds the file name `SaluToken.sol` in the File Explorer sidebar.
- Circle 2: Surrounds the line `// SPDX-License-Identifier: MIT` in the code editor, which is highlighted with a red box. An arrow points from this box to a red-bordered button labeled "Copy".

The browser address bar shows the URL: <https://remix.ethereum.org/#lang=en&optimize=false&runs=200&evmVersion=null&version=v0.8.26+com...>

At the bottom, there is a "Scam Alert" icon, a "Did you know?" section, and a "RemixAI Copilot (enabled)" status indicator.

Flatten SaluToken.sol

The screenshot shows the Remix Ethereum IDE interface. A Solidity file named `SaluToken.sol` is open in the code editor. The file contains the following code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts@4.9.0/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts@4.9.0/access/Ownable.sol";

contract SaluToken is ERC20, Ownable{
    constructor() ERC20("SaluToken", "ST") {}

    function issueToken() public onlyOwner{
        _mint(msg.sender, 1000 * (10 ** 18));
    }
}
```

The `FILE EXPLORER` sidebar shows a workspace named `Alchemy_dNFT`. A context menu is open over the `SaluToken.sol` file, with the following options:

- Rename
- Delete
- Copy
- Copy name
- Copy path
- Download
- Publish file to gist
- Flatten**
- Compile
- Compile for Nahmii
- Generate UML
- Generate Docs

Two numbered circles indicate the steps: circle 1 points to the `SaluToken.sol` file in the file explorer, and circle 2 points to the `Flatten` option in the context menu. Both the file in the file explorer and the `Flatten` option are highlighted with red boxes.

At the bottom of the interface, there are several status indicators and links:

- Scam Alert
- Initialize as git repo
- Did you know? You can verify your contract using the Etherscan plugin.
- RemixAI Copilot (enabled)

SaluToken_flattened.sol

The screenshot shows the Remix Ethereum IDE interface with the following annotations:

- 1**: A red box highlights the ".deps" folder in the File Explorer, which contains the "SaluToken_flattened.sol" file.
- 2**: A blue circle highlights the "File: @openzeppelin/contracts@4.9.0/token/ERC20/IERC20.sol" import statement in the code editor. A red box and arrow point to the text "Missing SPDX-License".
- A red box highlights the "OpenZeppelin Contracts (last updated v4.9.0) (token/ERC20/IERC20.sol)" dependency information in the code editor. A red box and arrow point to the text "Import 使用v4.9.0".

Code Editor Content:

```
1 // File: @openzeppelin/contracts@4.9.0/token/ERC20/IERC20.sol
2
3 pragma solidity ^0.8.0;
4
5 /**
6  * @dev Interface of the ERC20 standard as defined in the EIP.
7  */
8 interface IERC20 {
9     /**
10      * @dev Emitted when `value` tokens are moved from one account (`from`) to
11      * another (`to`).
12      *
13      * Note that `value` cannot be zero.
14      */
15 }
```

Bottom Status Bar:

- Scam Alert
- Initialize as git repo
- Did you know? You can verify your contract using the Etherscan plugin.
- RemixAI Copilot (enabled)

SaluToken_flattened.sol

The screenshot shows the Ethereum IDE interface. On the left, the FILE EXPLORER panel displays a workspace named 'Alchemy_dNFT' with a file named 'SaluToken_flattened.sol' highlighted by a red box and circled with a blue number '1'. In the center, the code editor shows the Solidity contract 'SaluToken_flattened.sol'. A specific line of code, `// SPDX-License-Identifier: MIT`, is highlighted with a red box and circled with a blue number '2'. A red arrow points from this highlighted line to a red box containing the word 'Paste', indicating where the copied code should be pasted.

```
// SPDX-License-Identifier: MIT
// OpenZeppelin Contracts (last updated v4.9.0) (token/ERC20/IERC20.sol)
pragma solidity ^0.8.0;

/**
 * @dev Interface of the ERC20 standard as defined in the EIP.
 */
interface IERC20 {
    /**
     * @dev Emitted when `value` tokens are moved from one account (`from`) to
     * another (`to`).
     *
     * Note that `value` may be zero.
     */
    event Transfer(address indexed from, address indexed to, uint256 value);

    /**
     * @dev Returns the amount of tokens in existence.
     */
    function totalSupply() external view returns (uint256);

    /**
     * @dev Returns the token holder account balance.
     */
    function balanceOf(address account) external view returns (uint256);

    /**
     * @dev Moves `value` tokens from the caller's account to `to`.
     *
     * Returns a boolean value indicating whether the operation succeeded.
     *
     * Requirements:
     *
     * - the caller must have at least `value` tokens.
     */
    function transfer(address to, uint256 value) external returns (bool);

    /**
     * @dev Returns the remaining number of tokens the caller can move.
     */
    function allowance(address owner, address spender) external view returns (uint256);

    /**
     * @dev Sets `amount` as the allowance of `spender` over the caller's tokens.
     *
     * Returns a boolean value indicating whether the operation succeeded.
     *
     * Requirements:
     *
     * - the caller must own `tokens`.
     * - the caller cannot approve over a supply of zero.
     */
    function approve(address spender, uint256 amount) external returns (bool);

    /**
     * @dev Moves `value` tokens from `from` to `to` using the
     * allowance mechanism. `amount` is then deducted from the caller's
     * allowance.
     *
     * Returns a boolean value indicating whether the operation succeeded.
     */
    function transferFrom(address from, address to, uint256 value) external returns (bool);
}
```

Compile Smart Contract

The screenshot shows the Remix Ethereum IDE interface with several numbered steps indicating the process of compiling a smart contract:

- 1** In the bottom-left sidebar, click the **Contract** icon.
- 2** In the Solidity Compiler section, ensure the compiler version is set to **0.8.26+commit.8a97fa7a** and the **Auto compile** checkbox is checked.
- 3** In the CONTRACT section, select the contract **SaluToken (SaluToken_flattened.sol)**.
- 4** Click the **Compile SaluToken_flattened...** button.

The code editor displays the Solidity source code for the SaluToken contract, which is a flattened version of the OpenZeppelin ERC20 standard. The code includes SPDX license information, a pragma solidity statement, and the IERC20 interface definition.

At the bottom of the interface, there is a note about running a JavaScript file and a list of accessible libraries:

- Right click on a JavaScript file in the file explorer and then click 'Run'
- The following libraries are accessible:
 - [web3.js](#)
 - [ethers.js](#)
 - [sol-gpt <your Solidity question here>](#)

Other UI elements include a Scam Alert warning, an Initialize as git repo button, a Did you know? link, and a RemixAI Copilot status indicator.

Deploy an ERC-20 Token

The image shows a composite screenshot of the Ethereum development environment. On the left, the **Remix - Ethereum IDE** interface is displayed, featuring a Solidity code editor for the `SaluToken.sol` contract. The code is based on the OpenZeppelin ERC20 standard. On the right, the **MetaMask** extension window is open, showing the `Sepolia` network and a wallet account named `Salu1`. A red box highlights the `Sepolia` network selection in MetaMask.

The **DEPLOY & RUN TRANSACTIONS** section in Remix is highlighted with a red box labeled **1**. It includes fields for **ENVIRONMENT** (set to `Injected Provider - MetaMask`), **ACCOUNT** (set to `0x4CE...6ae71 (1.55806951542443)`), **GAS LIMIT** (set to `Estimated Gas`), and **VALUE** (set to `0 Wei`). A blue circle labeled **2** points to the **Deploy** button at the bottom of the sidebar.

The **CONTRACT** dropdown in Remix is highlighted with a red box labeled **3**, showing the selected file `SaluToken - SaluToken_flattened.sol`. A blue circle labeled **4** points to the `creation of SaluToken pending...` message in the terminal area.

In the MetaMask window, a blue circle labeled **5** points to the **確認** (Confirm) button, which is highlighted with a red box. Other visible elements in the MetaMask window include the **Estimated changes** and **Estimated fee** sections.

Deploy an ERC-20 Token

The image shows two side-by-side interfaces: the Remix Ethereum IDE on the left and a wallet application on the right.

Remix - Ethereum IDE (Left):

- ENVIRONMENT:** Sepolia (11155111) network
- ACCOUNT:** 0x4CE13...6ae71 (1.55406778576512)
- CONTRACT:** SaluToken - SaluToken_flattened.sol
- Deploy Buttons:** Deploy (orange), Deploy - transact (not payable)

Code Snippet:

```
// SPDX-License-Identifier: MIT
// File: @openzeppelin/contracts@4.9.0/token/ERC20/IERC20.sol
pragma solidity ^0.8.0;

interface IERC20 {
    /**
     * @dev Emitted when `value` tokens are moved from one account (`from`) to
     * another (`to`).
     */
    event Transfer(address indexed from, address indexed to, uint256 value);
}
```

Wallet Application (Right):

- Account:** Salu1 (0x4CE13...6ae71)
- Balances:** 1.5541 SepoliaETH
- Buttons:** Buy & Sell, Swap, Bridge, 發送 (Send), 接收 (Receive)
- Transactions:**

 - Nov 15, 2024: 部署合約 (Deployment confirmed)
 - Nov 1, 2024: Issue Token (confirmed)
 - Oct 31, 2024: (No visible transaction details)

Deploy an ERC-20 Token

The screenshot shows the Remix Ethereum IDE interface. 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 (1.55406778576512 Wei)'. The gas limit is set to 'Estimated Gas' (3000000). The value is set to 0 Wei. The contract being deployed is 'SaluToken - SaluToken_flattened.sol' (evm version: canary). The 'Deploy' button is highlighted.

The central code editor displays two files: `SaluToken.sol` and `SaluToken_flattened.sol`. The code is an OpenZeppelin ERC-20 implementation. A red arrow points from the 'View on block explorer' link in the deployment status box to the error message in the terminal output.

Status

- 已確認
- 來源帳戶: 0x4CE13...6ae71
- 目的帳戶: 建立新合約
- 交易
- Nonce: 14
- 數量: -0 SepoliaETH
- Gas 上限(單位): 1430028
- Gas 用量(單位): 1417610
- Base fee (GWEI): 1.322870648
- Priority fee (GWEI): 1.5
- Total gas fee: 0.004002 SepoliaETH
- Max fee per gas: 0.00000003 SepoliaETH
- 總量: 0.00400173 SepoliaETH

View on block explorer

Did you know? You can verify your contract using the Etherscan plugin.

Oct 31, 2024

View Transaction on Sepolia

Sepolia Transaction Hash (Txh) <https://sepolia.etherscan.io/tx/0x07d0687e06bb0ece638c6648f28719a90f0814e4c4ab664d924ef1fbf1584cc4>

Transaction Details

Overview Logs (1) State

[This is a Sepolia Testnet transaction only]

② Transaction Hash: 0x07d0687e06bb0ece638c6648f28719a90f0814e4c4ab664d924ef1fbf1584cc4 ⓘ

② Status: Success

② Block: 7076355 18 Block Confirmations

② Timestamp: 4 mins ago (Nov-14-2024 04:26:24 PM UTC)

④ Transaction Action: Call 0x60806040 Method by 0x4CE135aB...64E06ae71 ⓘ

② From: 0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 ⓘ

② To: [0xb57c2dad... Created] ⓘ ⓘ

Click contract address

Smart Contract Information

The screenshot shows a web browser displaying the Sepolia Testnet version of Etherscan at the URL <https://sepolia.etherscan.io/address/0xb57c2dad...54d1f0e56>. The page is titled "Contract Address 0xb57c2dac".

The main content area includes:

- Contract Address:** 0xB57c2DAdB542664D698CB60e06D03dC54D1f0e56 (highlighted with a red box).
- More Info:** CONTRACT CREATOR (0x4CE135aB...64E06ae71) and TOKEN TRACKER (SaluToken (ST)). The TOKEN TRACKER section has a blue "click" button and a red arrow pointing to it.
- Multichain Info:** N/A
- Overview:** ETH BALANCE (0 ETH)
- Transactions:** Tabbed section showing Transactions (selected), Token Transfers (ERC-20), Contract, and Events. A single transaction is listed below.
- Latest Transaction:** 0x07d0687e06... (Method: 0x60806040, Block: 7076355, Age: 9 mins ago, From: 0x4CE135aB...64E06ae71, To: Contract Creation, Amount: 0 ETH, Txn Fee: 0.00400172).

Smart Contract Information

The screenshot shows a browser window displaying the Etherscan Token Tracker for the SaluToken (ST) on the Sepolia Testnet. The URL is <https://sepolia.etherscan.io/token/0xb57c2dadb542664d698cb60e06d03dc54d1f0e56>. The page is divided into three main sections: Overview, Market, and Other Info. The Overview section highlights the MAX TOTAL SUPPLY as 0 ST. The Market section shows an on-chain market cap of \$0.00. The Other Info section provides the token contract address (0xb57c2dadb542664d698cb60e06d03dc54d1f0e56). A red box highlights the 'MAX TOTAL SUPPLY' field. Another red box highlights the transaction history table at the bottom.

SaluToken (ST) Token Tracker

https://sepolia.etherscan.io/token/0xb57c2dadb542664d698cb60e06d03dc54d1f0e56

Search by Address / Txn Hash / Block / Token

Home Blockchain Tokens NFTs More

Token SaluToken (ST)

ERC-20

Overview

MAX TOTAL SUPPLY
0 ST

HOLDERS
0

TOTAL TRANSFERS
0

Market

ONCHAIN MARKET CAP
\$0.00

CIRCULATING SUPPLY MARKET CAP
-

Other Info

TOKEN CONTRACT (WITH 18 DECIMALS)
0xb57c2dadb542664d698cb60e06d03dc54d1f0e56

Transfers Holders Contract

Transaction Hash	Method	Block	Age	From	To	Amount
[Redacted]						

View & Publish

Contract Address 0xb57c2dac

https://sepolia.etherscan.io/address/0xb57c2dadb542664d698cb60e06d03dc54d1f0e56#code

Sepolia Testnet

Etherscan

Contract 0xB57c2DAdB542664D698CB60e06D03dC54D1f0e56

Overview

ETH BALANCE
0 ETH

More Info

CONTRACT CREATOR
0x4CE135aB...64E06ae71 at txn 0x07d0687e06...

TOKEN TRACKER
SaluToken (ST)

Multichain Info
N/A

Transactions Token Transfers (ERC-20) Contract Events

Are you the contract creator? Verify and Publish your contract source code today!

Decompile Bytecode Switch to Opcodes View Similar Contracts

View & Publish Contract Source Code

The screenshot shows a web browser window titled "Verify & Publish Contract Sou" with the URL <https://sepolia.etherscan.io/verifyContract?a=0xb57c2dadb542664d698cb60e06d03dc54d1f0e56>. 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 a two-step process: "Enter Contract Details" (Step 1) and "Verify & Publish" (Step 2). The "Enter Contract Details" step is active, showing fields for Contract Address (0xb57c2dadb542664d698cb60e06d03dc54d1f0e56), Compiler Type (Solidity (Single file)), Compiler Version (v0.8.26+commit.8a97fa7a), Open Source License Type (3) MIT License (MIT), and a checkbox for agreeing to terms of service. The "Continue" button at the bottom is highlighted with a red box.

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.](#)

1 Enter Contract Details — 2 Verify & Publish

Please enter the Contract Address you would like to verify

0xb57c2dadb542664d698cb60e06d03dc54d1f0e56

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 [?](#)

3) MIT License (MIT)

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

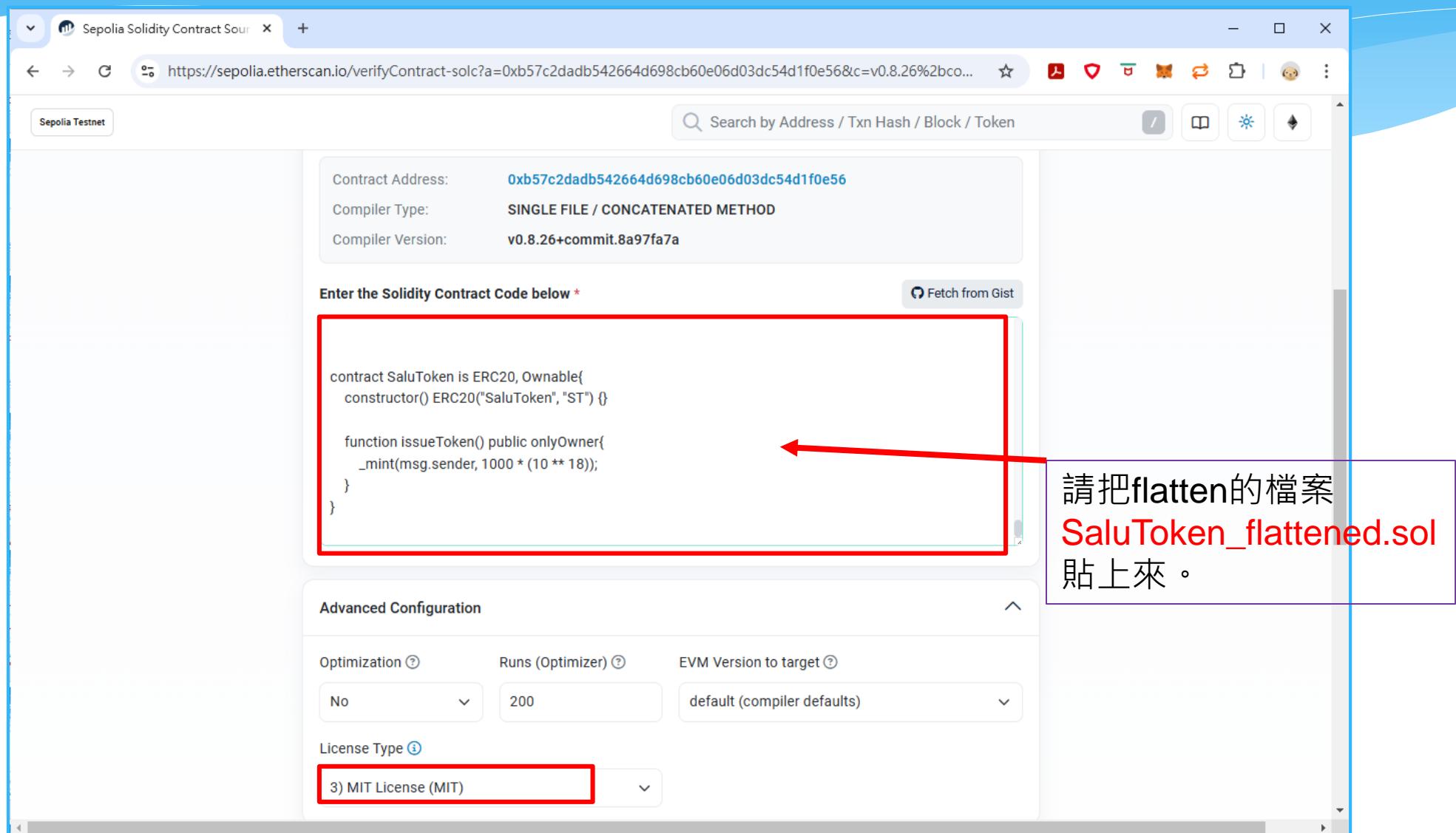
Continue Reset

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=0xb57c2dad...&c=v0.8.2...>. 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 in a single file. The process is divided into two steps: "Enter Contract Details" (step 1) and "Verify & Publish" (step 2). The "Enter Contract Details" step is currently active. A section titled "Upload Contract Source Code" lists three instructions: 1. If the contract compiles correctly at REMIX, it should also compile here. 2. Support is limited for contracts created by another contract with a 45-second timeout. 3. For programmatic verification, see the Contract API Endpoint. Below this, a summary table shows the following details:

Contract Address:	0xb57c2dad...54d1f0e56
Compiler Type:	SINGLE FILE / CONCATENATED METHOD
Compiler Version:	v0.8.26+commit.8a97fa7a

View & Publish Contract Source Code



The screenshot shows a browser window displaying the Sepolia Solidity Contract Source Code verification page at <https://sepolia.etherscan.io/verifyContract-solc?a=0xb57c2dad...>. The page includes fields for Contract Address (0xb57c2dad...), Compiler Type (SINGLE FILE / CONCATENATED METHOD), and Compiler Version (v0.8.26+commit.8a97fa7a). A large text area for pasting Solidity code is shown, with the following code highlighted:

```
contract SaluToken is ERC20, Ownable{
    constructor() ERC20("SaluToken", "ST") {}

    function issueToken() public onlyOwner{
        _mint(msg.sender, 1000 * (10 ** 18));
    }
}
```

A red box highlights the code area, and a red arrow points from the right margin towards it. A purple box contains the instruction: "請把flatten的檔案 SaluToken_flattened.sol 貼上來。". The bottom section of the page shows Advanced Configuration options for Optimization (No), Runs (Optimizer) (200), EVM Version to target (default (compiler defaults)), and License Type (3) MIT License (MIT).

請把flatten的檔案
SaluToken_flattened.sol
貼上來。

View & Publish Contract Source Code

The screenshot shows a web browser window for the Sepolia Testnet. The URL in the address bar is [https://sepolia.etherscan.io/verifyContract-solc?a=0xb57c2dadb542664d698cb60e06d03dc54d1f0e56&c=v0.8.26...](https://sepolia.etherscan.io/verifyContract-solc?a=0xb57c2dad...&c=v0.8.26...). The page displays a form for verifying a Solidity contract. At the top, there is a section for "Constructor Arguments ABI-encoded" which is currently empty. Below it, a note says "For additional information on Constructor Arguments, see our KB entry". Further down, there is a section for "Contract Library Address (for contracts that use libraries, supports up to 10 libraries)". At the bottom, a success message "成功!" is shown next to a Cloudflare logo. A red box highlights the "Verify and Publish" button.

Sepolia Solidity Contract Sour

https://sepolia.etherscan.io/verifyContract-solc?a=0xb57c2dad...&c=v0.8.26...

Sepolia Testnet

Search by Address / Txn Hash / Block / Token

Constructor Arguments **ABI-encoded**

For contracts that were created with constructor parameters

For additional information on Constructor Arguments, [see our KB entry](#)

Contract Library Address (for contracts that use libraries, supports up to 10 libraries)

成功!

CLOUDFLARE

Verify and Publish

Reset

View & Publish Contract Source Code

The screenshot shows a web browser window for the Sepolia Testnet on Etherscan. The URL is https://sepolia.etherscan.io/verifyContract-solc?a=0xb57c2dad...&c=v0.8.26... The page title is "Verify & Publish Contract Source Code". The main content area explains that source code verification provides transparency by matching uploaded code with blockchain data. It features a simple interface for verifying contracts. A red box highlights a success message: "Successfully generated Bytecode and ABI for Contract Address [0xb57c2dad...]. A red arrow points from this message to the word "click" in the text below. At the bottom, there's a note about verifying contracts on multiple blockchains with a single API key.

Sepolia Solidity Contract Sour x +

https://sepolia.etherscan.io/verifyContract-solc?a=0xb57c2dadb542664d698cb60e06d03dc54d1f0e56&c=v0.8.26...

Sepolia Testnet

Etherscan

Search by Address / Txn Hash / Block / Token

Home Blockchain Tokens NFTs 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
[0xb57c2dadb542664d698cb60e06d03dc54d1f0e56]

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

View Contract Source Code

The screenshot shows the Etherscan interface for the SaluToken contract. The URL in the browser's address bar is highlighted with a red box: <https://sepolia.etherscan.io/address/0xb57c2dad542664d698cb60e06d03dc54d1f0e56#code>. The page title is "SaluToken | Address 0xb57c2c".

The "Code" tab is selected. A message "Contract Source Code Verified (Exact Match)" is displayed, also highlighted with a red box. The contract name is "SaluToken", compiler version is "v0.8.26+commit.8a97fa7a", optimization is "No with 200 runs", and other settings are "default evmVersion, MIT license".

The "Contract Source Code (Solidity)" section shows the following code:

```
1  /**
2  *Submitted for verification at Etherscan.io on 2024-11-14
3  */
4
5 // SPDX-License-Identifier: MIT
6 // File: @openzeppelin/contracts@4.9.0/token/ERC20/IERC20.sol
7
8
9 // OpenZeppelin Contracts (last updated v4.9.0) (token/ERC20/IERC20.sol)
10
11 pragma solidity ^0.8.0;
12
13 /**
14  * @dev Interface of the ERC20 standard as defined in the EIP.
15  */
16 interface IERC20 {
17     /**
18      ...
19  }
```

View & Publish Contract Source Code

The screenshot shows the Etherscan interface for a contract named SaluToken. The URL in the browser bar is <https://sepolia.etherscan.io/address/0xb57c2dadb542664d698cb60e06d03dc54d1f0e56#code>. The page displays the contract's overview, more info, and multichain info. At the bottom, there are tabs for Transactions, Token Transfers (ERC-20), Contract (highlighted with a red box), and Events. Below these tabs, there are buttons for Code (highlighted with a red box), Read Contract, and Write Contract. A message indicates that the contract source code has been verified. The contract name is SaluToken, and optimization is enabled with 200 runs.

SaluToken | Address 0xb57c2dadb542664d698cb60e06d03dc54d1f0e56

https://sepolia.etherscan.io/address/0xb57c2dadb542664d698cb60e06d03dc54d1f0e56#code

Source Code

Contract 0xB57c2DAdB542664D698CB60e06D03dC54D1f0e56

Overview

ETH BALANCE
0 ETH

More Info

CONTRACT CREATOR
0x4CE135aB...64E06ae71 at txn 0x07d0687e06...

TOKEN TRACKER
SaluToken (ST)

Multichain Info
N/A

Transactions Token Transfers (ERC-20) Contract Events

Code Read Contract Write Contract

Contract Source Code Verified (Exact Match)

Contract Name: SaluToken Optimization Enabled: No with 200 runs

Connect to Web3

sepolia.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.

確定 取消

CONTRACT CREATOR
0x4CE135aB...64E06ae71 at txn 0x07d0687e06...

TOKEN TRACKER
SaluToken (ST)

Contract 0xB57c2DAdB542664D698CB60e06d03dC54d1f0e56

Source Code

Overview

ETH BALANCE
0 ETH

Transactions Token Transfers (ERC-20) Contract Events

Code Read Contract Write Contract

Connect to Web3

Read Contract Information

1. allowance

/ Block / Token

Multichain Info
N/A

[Expand all] [Reset]

Connect a Wallet-MetaMask

The screenshot shows a web browser window displaying the Etherscan interface for the Sepolia Testnet. The URL in the address bar is <https://sepolia.etherscan.io/address/0x4acbdad8953c8c5a533a2f8b84c710b56bd99b4a#readContract>. A modal window titled "Connect a Wallet" is open in the center. Inside the modal, there is an informational message: "Connecting wallet for read function is optional, useful if you want to call certain functions or simply use your wallet's node." Below this message, there are three options: "MetaMask" (which is highlighted with a red box), "WalletConnect", and "Coinbase Wallet". The "MetaMask" option is labeled as "Popular" and has a fox icon next to it. The background of the Etherscan interface shows tabs for "Transactions", "Token Transfers (ERC-20)", and "Contract" (which is currently selected). Under the "Contract" tab, there are buttons for "Code", "Read Contract" (which is highlighted in blue), and "Write Contract". On the left side, there are sections for "Connect to Web3" (with a red dot icon) and "Read Contract Information". Below these are three numbered items: "1. allowance", "2. balanceOf", and "3. decimals". 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 a blockchain explorer (Etherscan). A red arrow points from the "Connected Web3 [0x4ce1...ae71]" status in the Etherscan interface to the account selector in the MetaMask extension.

Etherscan Interface (Left):

- Address: SaluToken | Address 0xb57c2...
- Network: Sepolia Testnet
- Contract Tab (selected)
- Code, Read Contract, Write Contract buttons
- Status: Connected Web3 [0x4ce1...ae71]
- Read Contract Information button
- List of functions: allowance, balanceOf, decimals, name, owner

MetaMask Extension (Right):

- Account: Salu1 (0x4CE13...6ae71)
- Balances: 1.5541 SepoliaETH
- Portfolio: Portfolio
- Buttons: Buy & Sell, Swap, Bridge, 發送 (Send), 接收 (Receive)
- Transactions:

 - Nov 15, 2024: 部署合約 (Deployed Contract) -0 SepoliaETH
 - Nov 1, 2024: Issue Token -0 SepoliaETH
 - Oct 31, 2024: (No visible transaction details)

Read Contract

The screenshot shows the Etherscan interface for the SaluToken contract on the Sepolia Testnet. The 'Contract' tab is selected. The 'Read Contract' sub-tab is active. The page displays the following contract variables:

- 1. allowance
- 2. balanceOf
- 3. decimals
18 uint8
- 4. name
SaluToken string
- 5. owner
0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 address
- 6. symbol
ST string
- 7. totalSupply
0 uint256

Connected Web3 using account

The image shows two web-based interfaces for interacting with a Ethereum Sepolia Testnet account.

Etherscan Interface (Left): This interface displays transaction history for the address `0xb57c2dad...542664d698cb60e06d03dc54d1f0e56`. It includes tabs for Transactions, Token Transfers (ERC-20), Contract, and Events. The Contract tab is active, showing three recent interactions:

- 1. approve (Txn: `0x095ea7b3`)
- 2. decreaseAllowance (Txn: `0xa457c2d7`)
- 3. increaseAllowance (Txn: `0x39509351`)

A message "Connected - Web3 [0x4ce1...ae71]" is highlighted with a red box. A red arrow points from this message to the connected account in the wallet interface.

Wallet Interface (Right): This interface shows the user's portfolio on the Sepolia network. The account `0x4CE13...6ae71` is connected, as indicated by the red box. The balance is **1.5541 SepoliaETH**. The interface includes buttons for Buy & Sell, Swap, Bridge, Send (發送), and Receive (接收). The "Transactions" tab is selected, showing activity from Nov 15, 2024, through Oct 31, 2024. Key events listed include:

- Nov 15, 2024: 部署合約 已確認 (Deployed Contract, confirmed) -0 SepoliaETH, -0 SepoliaETH
- Nov 1, 2024: Issue Token 已確認 (Issued Token, confirmed) -0 SepoliaETH, -0 SepoliaETH
- Oct 31, 2024: (No visible transactions)

Write Contract - issueToken

The screenshot shows the process of writing a contract to issue tokens.

Step 1: On the left, the **Contract** tab is selected on the Etherscan page. A blue arrow points from the **Write** button (circled 1) to the **issueToken** transaction listed under "Transactions".

Step 2: A blue arrow points from the **issueToken** transaction to the MetaMask wallet interface on the right. The MetaMask window displays the transaction details:

- Estimated changes:** You receive +1,000 SepoliaETH (circled 2).
- Estimated fee:** 0.00035484 SepoliaETH
- Market:** ~60 sec Max fee: 0.00042308 SepoliaETH

The MetaMask interface includes a **拒绝** (Reject) button and a large red-bordered **確認** (Confirm) button.

Write Contract - issueToken

The screenshot shows the Etherscan.io interface for the SaluToken contract (Address: 0xb57c2dad...). The top navigation bar includes tabs for Sepolia Testnet, Overview, More Info, Multichain Info, Transactions, Token Transfers (ERC-20), Contract (selected), and Events.

More Info:

- CONTRACT CREATOR: 0x4CE135aB...64E06ae71 at txn 0x07d0687e06...
- TOKEN TRACKER: SaluToken (ST)

Contract Tab:

Connected - Web3 [0x4ce1...ae71] [Expand all] [Reset]

1. approve (0x095ea7b3)
2. decreaseAllowance (0xa457c2d7)
3. increaseAllowance (0x39509351)
4. issueToken (0xa1ee8c78)

Bottom Buttons:

- Write
- View your transaction** (button highlighted with a red box)

View Transaction

The screenshot shows a web browser displaying the Etherscan interface for a Sepolia Testnet transaction. The transaction hash is `0x89734d6e46dc840e830f5c0163040e0bf71916e38fdf53c98b65a08b2a3e845`. The status is listed as "Success". The transaction was included in block `7076615` with 5 block confirmations. It occurred 1 minute ago at Nov-14-2024 05:24:24 PM UTC. The transaction action involved a call to the `Issue Token` function on contract `0x4CE135aB...64E06ae71`, which is associated with address `0xB57c2DAd...54D1f0e56`. The transaction originated from address `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71` and interacted with address `0xB57c2DAdB542664D698CB60e06D03dC54D1f0e56`.

Sepolia Transaction Hash (Txn Hash)

<https://sepolia.etherscan.io/tx/0x89734d6e46dc840e830f5c0163040e0bf71916e38fdf53c98b65a08b2a3e845>

Search by Address / Txn Hash / Block / Token

Etherscan

Home Blockchain Tokens NFTs More

Transaction Details

Overview Logs (1) State

[This is a Sepolia Testnet transaction only]

Transaction Hash: `0x89734d6e46dc840e830f5c0163040e0bf71916e38fdf53c98b65a08b2a3e845`

Status: Success

Block: `7076615` 5 Block Confirmations

Timestamp: 1 min ago (Nov-14-2024 05:24:24 PM UTC)

Transaction Action: Call `Issue Token` Function by `0x4CE135aB...64E06ae71` on `0xB57c2DAd...54D1f0e56`

From: `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71`

Interacted With (To): `0xB57c2DAdB542664D698CB60e06D03dC54D1f0e56`

Smart Contract Information

The screenshot shows the Etherscan Token SaluToken (ST) page. The token contract address is `0xb57c2dad542664d698cb60e06d03dc54d1f0e56`. The page displays the following information:

- Token SaluToken (ST)** (highlighted by a red box)
- ERC-20**
- Overview** (highlighted by a red box):
 - MAX TOTAL SUPPLY: 1,000 ST
 - HOLDERS: 1
 - TOTAL TRANSFERS: 1
- Market**:
 - ONCHAIN MARKET CAP: \$0.00
 - CIRCULATING SUPPLY MARKET CAP: -
- Other Info**:
 - TOKEN CONTRACT (WITH 18 DECIMALS): `0xb57c2dad542664d698cb60e06d03dc54d1f0e56`

Below the overview, there is a table of transactions:

Transaction Hash	Method	Block	Age	From	To	Amount
<code>0x89734d6e46...</code>	Issue Token	7076615	3 mins ago	<code>0x00000000...00000000</code>	<code>0x4CE135aB...64E06ae71</code>	1,000

Connected Web3 using account

使用另一個account address, 不是Owner

0.42 SepoliaETH

Buy & Sell Swap Bridge 發送 接收

Tokens NFTs 交易紀錄

SepoliaETH 0.42 SepoliaETH

[Reset]

+ Import Tokens

Refresh list

MetaMask 支援

Connected - Web3 [0x7bee...8f0c]

1. approve (0x095ea7b3)

2. decreaseAllowance (0xa457c2d7)

3. increaseAllowance (0x39509351)

4. issueToken (0xa1ee8c78)

Write

Write Contract - issueToken

The screenshot shows the SaluToken (ST) Token Tracker interface on the Sepolia Testnet. The main window displays the token's overview, market information, and a list of recent transactions. A red arrow points from a blue circle labeled '1' at the bottom left to the 'Write' button in the transaction list. Another red arrow points from a blue circle labeled '2' at the bottom right to a warning message about gas estimation.

Overview

- MAX TOTAL SUPPLY: 1,000 ST
- HOLDERS: 1
- TOTAL TRANSFERS: -

Market

- ONCHAIN MARKET CAP: \$0.00
- CIRCULATING SUPPLY MARKET CAP: -

Transactions

Index	Transaction	Gas
1.	approve (0x095ea7b3)	-
2.	decreaseAllowance (0xa457c2d7)	-
3.	increaseAllowance (0x39509351)	-
4.	issueToken (0xa1ee8c78)	-

Contract

Code Read Contract Write Contract

Connected - Web3 [0x7bee...8f0c]

1. approve (0x095ea7b3)
2. decreaseAllowance (0xa457c2d7)
3. increaseAllowance (0x39509351)
4. issueToken (0xa1ee8c78)

Write

Sepolia

Salu 0 → 0xB57c2...f0e56

<https://sepolia.etherscan.io>

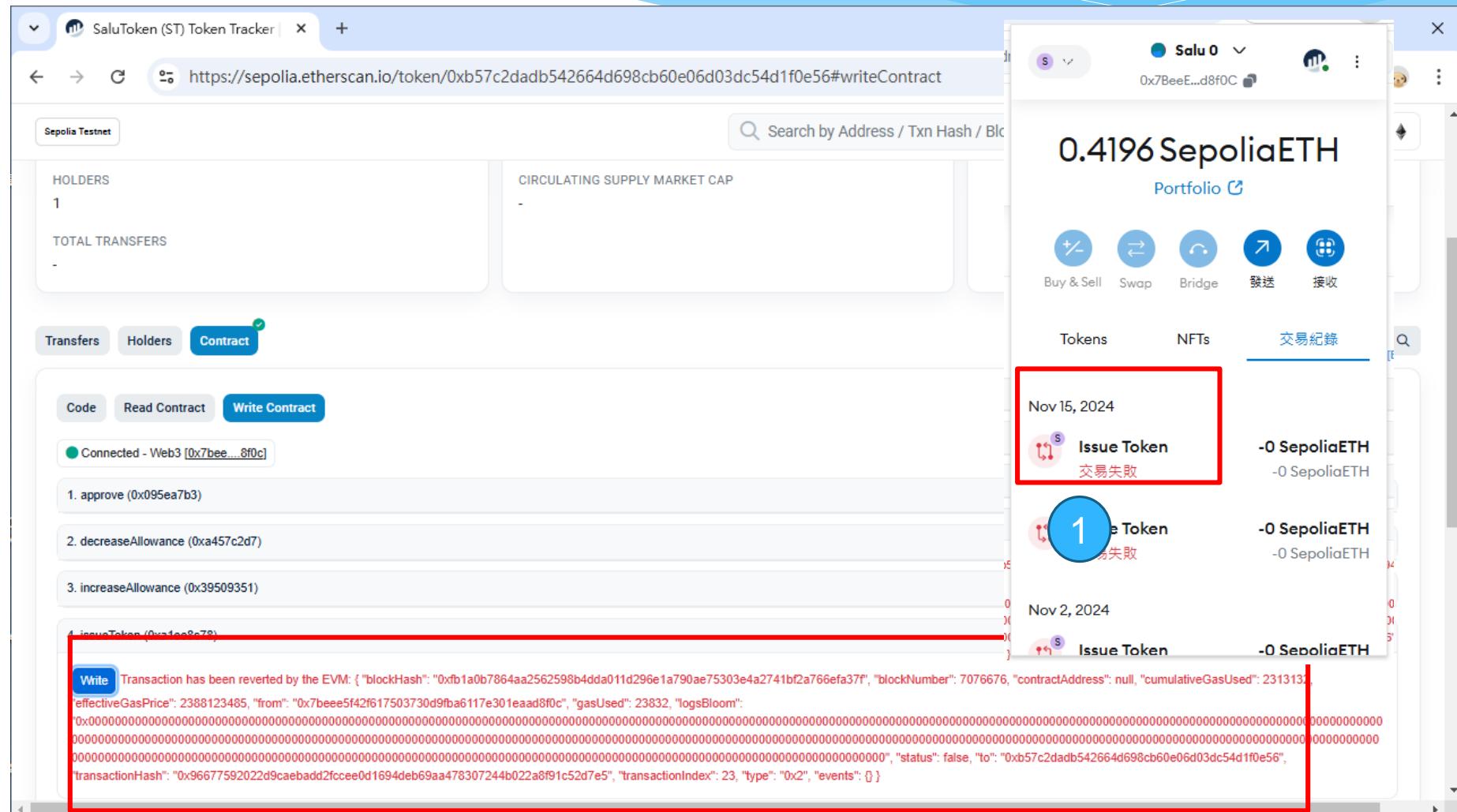
0xB57c2...f0e56 : ISSUE TOKEN

We were not able to estimate gas.
There might be an error in the
contract and this transaction may fail.
I want to proceed anyway

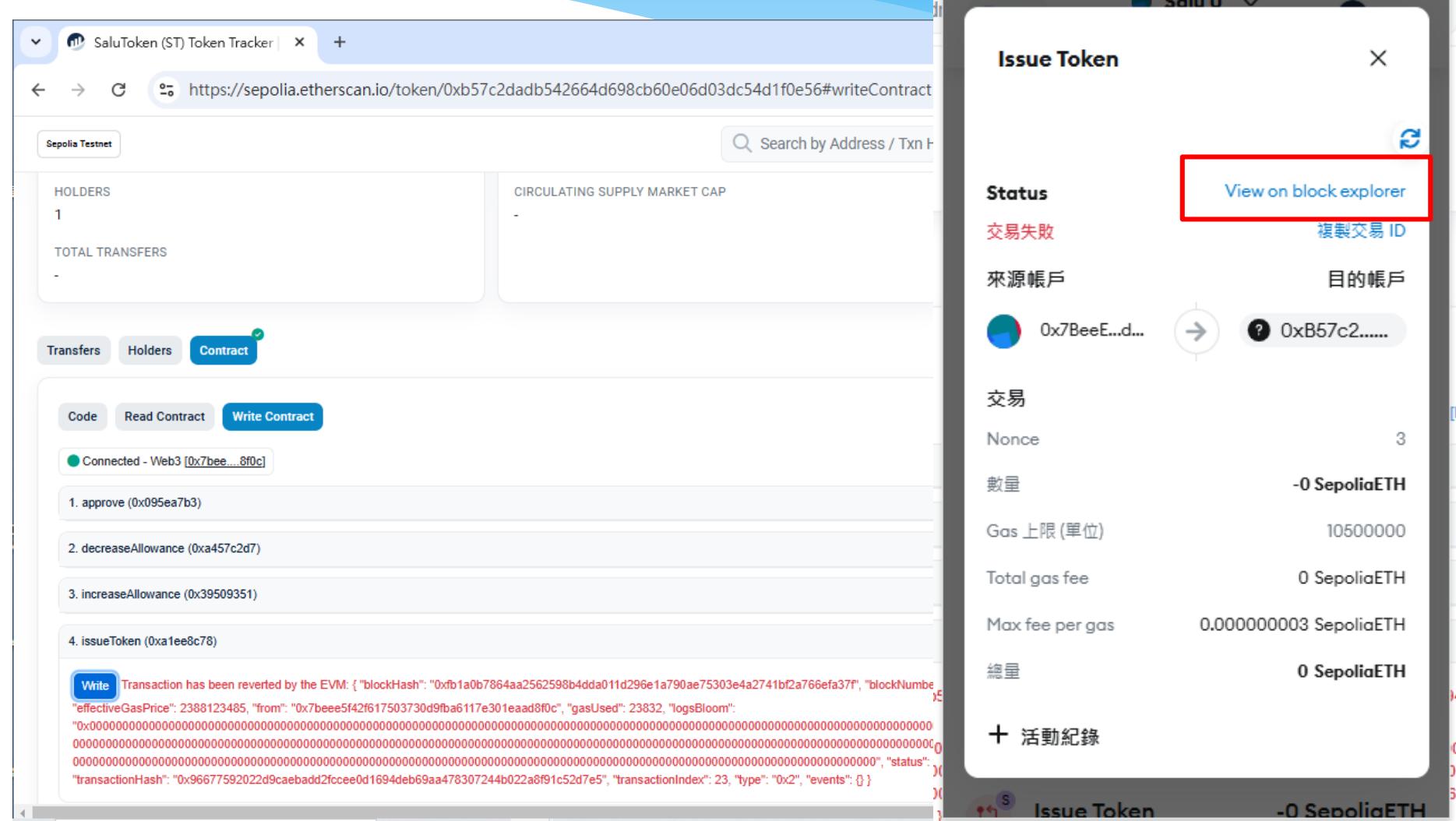
This transaction is likely to fail

總量 0.0269973 0.0269973 SepoliaETH
Amount + gas fee Max amount:
0.03093385 SepoliaETH

Write Contract - issueToken



View Transaction



View Transaction

Sepolia Transaction Hash (Txh) <https://sepolia.etherscan.io/tx/0xcb33f9e02a4fb2e7e27e887fdb2b060fb1d8591f6d2a7a9fb8107bdd15b6e35>

[This is a Sepolia Testnet transaction only]

Transaction Hash: 0xcb33f9e02a4fb2e7e27e887fdb2b060fb1d8591f6d2a7a9fb8107bdd15b6e35

Status: Fail with error 'Ownable: caller is not the owner'

Block: 7076686 | 16 Block Confirmations

Timestamp: 3 mins ago (Nov-14-2024 05:39:24 PM UTC)

Transaction Action: Call | Issue Token | Function by 0x7BeeE5F4...1EAad8f0C on 0x0

From: 0x7BeeE5F42F617503730d9Fba6117E301EAad8f0C

To: 0xB57c2DAdB542664D698CB60e06D03dC54D1f0e56

Value: 0 ETH

Warning! Error encountered during contract execution [execution reverted]

<https://github.com/OpenZeppelin/openzeppelin-contracts/blob/release-v4.9/contracts/access/Ownable.sol>

```
function owner() public view virtual returns (address) {
    return _owner;
}

/**
 * @dev Throws if the sender is not the owner.
 */
function _checkOwner() internal view virtual {
    require(owner() == msgSender(), "Ownable: caller is not the owner");
}

/**
 * @dev Leaves the contract without owner. It will not be possible to call
 * `onlyOwner` functions. Can only be called by the current owner.
 */
function renounceOwnership() external virtual override {
    require(_owner != address(0), "Ownable: no owner");
    _owner = address(0);
    emit OwnershipTransferred(_owner, address(0));
}

/**
 * NOTE: Renouncing ownership will leave the contract without an owner,
 * thereby disabling any functionality that is only available to the owner.
 */
```

Smart Contract Information

The screenshot shows the Etherscan.io interface for the SaluToken smart contract at address 0xb57c2dad...54d1f0e56. The page is titled "SaluToken | Address 0xb57c2..." and the URL is https://sepolia.etherscan.io/address/0xb57c2dad...54d1f0e56.

Contract Details: The contract address is 0xB57c2DAdB542664D698CB60e06d03dc54d1f0e56. The ETH balance is 0 ETH.

More Info: Contract Creator is 0x4CE135aB...64E06ae71, created at tx 0x07d0687e06... Token Tracker is SaluToken (ST).

Multichain Info: N/A

Transactions: The tab is selected, showing the latest 4 transactions from a total of 4. The table includes columns: Transaction Hash, Method, Block, Age, From, To, Amount, and Txn Fee.

Transaction Hash	Method	Block	Age	From	To	Amount	Txn Fee
0xcb33f9e02a4...	Issue Token	7076686	7 mins ago	0x7BeeE5F4...1EAad8f00	0xB57c2DAd...54d1f0e56	0 ETH	0.0000533
0x9667759202...	Issue Token	7076676	9 mins ago	0x7BeeE5F4...1EAad8f00	0xB57c2DAd...54d1f0e56	0 ETH	0.00005691
0x89734d6e46...	Issue Token	7076615	22 mins ago	0x4CE135aB...64E06ae71	0xB57c2DAd...54d1f0e56	0 ETH	0.00020041
0x07d0687e06...	0x60806040	7076355	1 hr ago	0x4CE135aB...64E06ae71	Create: SaluToken	0 ETH	0.00400172

A red box highlights the first two transaction rows, and another red box highlights the "To" field in the third row. A red arrow points from the text "因為不是Owner,所以產生Transaction Error" to the "To" field of the third transaction row.

因為不是Owner,所以產生Transaction Error

Smart Contract Information

The screenshot shows the Etherscan Token Tracker interface for the SaluToken (ST) on the Sepolia Testnet. The token has a maximum supply of 1,000 ST, 1 holder, and 1 total transfer. A transaction is listed at the bottom.

Token SaluToken (ST)

ERC-20

Overview

- MAX TOTAL SUPPLY
1,000 ST
- HOLDERS
1
- TOTAL TRANSFERS
1

Market

- ONCHAIN MARKET CAP ⓘ
\$0.00
- CIRCULATING SUPPLY MARKET CAP
-

Other Info

- TOKEN CONTRACT (WITH 18 DECIMALS)
[0xb57c2dad...f1f0e56](#)

Transfers **Holders** **Contract**

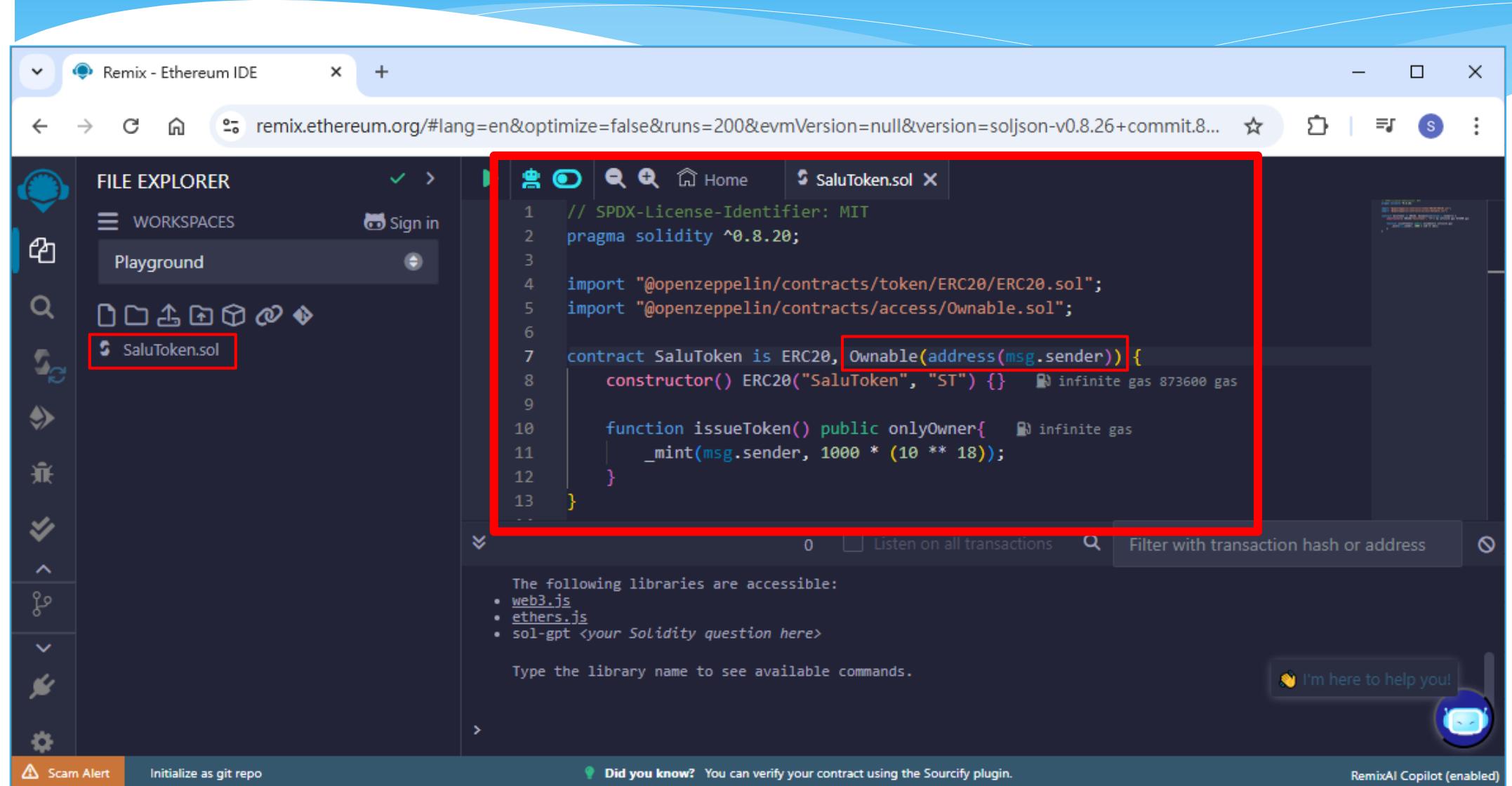
A total of 1 transaction found

Transaction Hash	Method ⓘ	Block	Age	From	To	Amount
0x89734d6e46...	Issue Token	7076615	24 mins ago	0x00000000...00000000	0x4CE135aB...64E06ae71	1,000



Openzeppelin v5.1.0

Modify SaluToken.sol



The screenshot shows the Ethereum IDE (Remix) interface. The left sidebar has icons for FILE EXPLORER, WORKSPACES (Playground selected), and various developer tools. The central workspace shows a Solidity file named `SaluToken.sol`. The code is as follows:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts/access/Ookable.sol";

contract SaluToken is ERC20, Ookable(address(msg.sender)) {
    constructor() ERC20("SaluToken", "ST") {}    infinite gas 873600 gas

    function issueToken() public onlyOwner{    infinite gas
        _mint(msg.sender, 1000 * (10 ** 18));
    }
}
```

A red box highlights the line `constructor() ERC20("SaluToken", "ST") {}`, and another red box highlights the `Ookable(address(msg.sender))` part of the `contract` declaration. At the bottom of the code editor, there's a note about accessible libraries: `web3.js`, `ethers.js`, and `sol-gpt <your Solidity question here>`. A message bar at the bottom says, "Did you know? You can verify your contract using the Sourcify plugin." A small AI icon in the bottom right corner says, "I'm here to help you!"

Copy SPDX-License

The screenshot shows the Remix Ethereum IDE interface. On the left, the FILE EXPLORER sidebar has a red box around the 'Playground' workspace, with a blue circle containing the number '1' below it. In the center, the code editor displays a Solidity contract named 'SaluToken.sol'. A red box highlights the first two lines of the code: `// SPDX-License-Identifier: MIT` and `pragma solidity ^0.8.20;`. A blue circle containing the number '2' is positioned over the highlighted code. To the right of the code editor, a 'Copy' button is also enclosed in a red box. At the bottom of the interface, there is a message from the RemixAI Copilot: "I'm here to help you!"

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts/access/Ownable.sol";

contract SaluToken is ERC20, Ownable(address(msg.sender)) {
    constructor() ERC20("SaluToken", "ST") {} // infinite gas 873600 gas

    function issueToken() public onlyOwner{ // infinite gas
        _mint(msg.sender, 1000 * (10 ** 18));
    }
}
```

The following libraries are accessible:

- [web3.js](#)
- [ethers.js](#)
- [sol-gpt <your Solidity question here>](#)

Type the library name to see available commands.

I'm here to help you!

Flatten SaluToken.sol

The screenshot shows the Remix Ethereum IDE interface. A Solidity file named `SaluToken.sol` is open in the code editor. The code defines a token contract that inherits from `ERC20` and `Ownable`, and includes a constructor and a `issueToken` function.

Two numbered steps are highlighted:

- 1**: A blue circle highlights the `SaluToken.sol` file in the File Explorer sidebar, which is also selected in the workspace dropdown.
- 2**: A blue circle highlights the `Flatten` option in the context menu that appears when right-clicking the file name in the File Explorer.

The context menu also includes other options like `Rename`, `Delete`, `Copy`, `Copy name`, `Copy path`, `Download`, `Publish file to gist`, `Compile`, `Compile for Nahmii`, `Generate UML`, and `Generate Docs`.

The bottom status bar includes links for `Scam Alert`, `Initialize as git repo`, `Did you know?` (with a note about Sourcify), and `RemixAI Copilot (enabled)`.

SaluToken_flattened.sol

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

- 1**: A red box highlights the "FILE EXPLORER" sidebar. A blue circle with the number **1** is positioned over the "SaluToken_flattened.sol" file in the sidebar.
- 2**: A red box highlights the code editor area. A blue circle with the number **2** is positioned over the first line of the code: `1 // File: @openzeppelin/contracts/token/ERC20/IERC20.sol`.
- Missing SPDX-License**: A red box with this text is placed near the first line of the code.
- Import 使用v5.1.0**: A red box with this text is placed near the line `// OpenZeppelin Contracts (last updated v5.1.0) (token/ERC20/IERC20.sol)`.

```
1 // File: @openzeppelin/contracts/token/ERC20/IERC20.sol
2 // OpenZeppelin Contracts (last updated v5.1.0) (token/ERC20/IERC20.sol)
3
4 pragma solidity ^0.8.20;
5
6 /**
7  * @dev Interface of the ERC-20 standard as defined in the ERC.
8  */
9 interface IERC20 {
10     /**
11      * @dev Emitted when `value` tokens are moved from one account (`from`) to
12      * another (`to`).
13 }
```

Other visible UI elements include the title bar "Remix - Ethereum IDE", the tabs "SaluToken.sol" and "SaluToken_flattened.sol 1", the bottom status bar with "Scam Alert", "Initialize as git repo", "Did you know?", and "RemixAI Copilot (enabled)", and a sidebar with various icons and a "Playground" workspace.

SaluToken_flattened.sol

The screenshot shows the Remix Ethereum IDE interface. On the left, the FILE EXPLORER sidebar lists two files: 'SaluToken_flattened.sol' and 'SaluToken.sol'. A red box highlights 'SaluToken_flattened.sol', and a blue circle with the number '1' is placed over it. In the main code editor area, a red box highlights the first two lines of the 'SaluToken_flattened.sol' file:

```
// SPDX-License-Identifier: MIT  
// File: @openzeppelin/contracts/token/ERC20/IERC20.sol
```

A blue circle with the number '2' is placed over the second line. A red arrow points from the text 'Paste' in a red box at the bottom right towards the highlighted code.

The code editor also displays the following content:

```
// OpenZeppelin Contracts (last updated v5.1.0) (token/ERC20/IERC20.sol)  
pragma solidity ^0.8.20;  
  
/**  
 * @dev Interface of the ERC-20 standard as defined in the ERC.  
 */  
interface IERC20 {  
    /**  
     * @dev Emitted when `value` tokens are moved from one account (`from`) to  
     * another (`to`).
```

At the bottom of the code editor, there is a message: 'The following libraries are accessible:' followed by a bulleted list: 'web3.js', 'ethers.js', and 'sol-gpt <your Solidity question here>'. Below that, it says 'Type the library name to see available commands.'

At the very bottom of the interface, there are several status bars: 'Scam Alert', 'Initialize as git repo', 'Did you know? You can verify your contract using the Sourcify plugin.', 'RemixAI Copilot (enabled)', and a 'I'm here to help you!' button with a AI icon.

Compile Smart Contract

The screenshot shows the Ethereum IDE (Remix) interface with the following steps highlighted:

- Compiler Version:** The compiler version is set to "0.8.26+commit.8a97fa7a".
- Auto Compile:** The "Auto compile" checkbox is checked.
- Contract:** The contract selected is "SaluToken (SaluToken_flattened.sol)".
- Compile Button:** The "Compile SaluToken_flattened..." button is highlighted.

The code editor displays the Solidity source code for the IERC20 interface:

```
// SPDX-License-Identifier: MIT
// File: @openzeppelin/contracts/token/ERC20/IERC20.sol

// OpenZeppelin Contracts (last updated v5.1.0) (token/ERC20/IERC20.sol)

pragma solidity ^0.8.20;

/**
 * @dev Interface of the ERC-20 standard as defined in the EIP.
 */
interface IERC20 {
```

The interface includes comments about the standard and its implementation.

At the bottom of the interface, there is a message from the AI copilot: "I'm here to help you!"

Deploy an ERC-20 Token

The image shows a composite screenshot of the Ethereum development environment. On the left, the **Remix - Ethereum IDE** window displays the Solidity code for a token contract. On the right, the **MetaMask** extension interface is visible, showing account selection and deployment options.

Remix - Ethereum IDE (Left):

- Injected Provider - MetaMask:** Shows "Sepolia (11155111) network".
- ACCOUNT:** Shows the account address **0x4CE...6ae71 (1.55386737328267)**.
- GAS LIMIT:** Set to **Estimated Gas** (radio button selected).
- Custom GAS LIMIT:** Set to **3000000**.
- VALUE:** Set to **0 Wei**.
- CONTRACT:** Shows the deployed contract **SaluToken - SaluToken_flattened.sol**.
- Deploy:** A prominent orange button.
- Scam Alert:** A yellow warning icon.

MetaMask (Right):

- Account:** Shows the account **Salu1**.
- Buttons:** Includes "建立新合約" (Create New Contract).
- Estimated changes:** Shows "No changes predicted for your wallet".
- Estimated fee:** Shows **\$5.15** for 0.00164375 SepoliaETH.
- Nonce:** Shows **16**.
- Buttons:** Includes "拒絕" (Reject) and a large red-bordered "確認" (Confirm) button.

Numbered Callouts:

- 1: Points to the MetaMask icon in the Remix sidebar.
- 2: Points to the "Injected Provider - MetaMask" dropdown in the Remix sidebar.
- 3: Points to the "Deploy" button in the Remix sidebar.
- 4: Points to the "Deploy" button in the Remix main interface.
- 5: Points to the "確認" (Confirm) button in the MetaMask interface.

Bottom Status Bar:

Did you know? You can verify your contract using the Sourcify plugin.

203

Deploy an ERC-20 Token

The image shows two screenshots illustrating the deployment of an ERC-20 token using the Ethereum IDE.

Ethereum IDE Screenshot:

- Injected Provider - MetaMask:** Shows the network selected as "Sepolia (11155111) network".
- Account Selection:** Shows the account "0x4CE...6ae71 (1.55223749867917)".
- Gas Limit:** Set to "Estimated Gas".
- Value:** Set to 0 Wei.
- Contract:** The deployed contract is "SaluToken - SaluToken_flattened.sol".
- Deploy Button:** An orange button labeled "Deploy".

Wallet Screenshot:

- Address:** 0x4CE13...6ae71
- Balances:** 1.5522 SepoliaETH
- Transactions:**
 - Nov 19, 2024:** 部署合約 已確認 (Deployment confirmed)
 - Oct 30, 2024:** 接收 已確認 (Received confirmation)
 - Transactions from Nov 19, 2024:** 接收 已確認 (Received confirmation), 0.01 SepoliaETH, 1.18 ETC
 - Transactions from Oct 30, 2024:** 接收 已確認 (Received confirmation), 0.05 SepoliaETH, 5.89 ETC

Deploy an ERC-20 Token

The screenshot shows the Remix Ethereum IDE interface. On the left, the sidebar includes icons for deploying transactions, environment (Injected Provider - MetaMask, Sepolia network), account (0x4CE...6ae71), gas limit (Estimated Gas: 3000000), value (0 Wei), and contracts (SaluToken - SaluToken_flattened.sol). The main area displays the Solidity code for the SaluToken contract, which is a flattened version of the OpenZeppelin ERC20 standard. A transaction is shown as pending, with a green checkmark and the text "creation of SaluToken pending...". Below the transaction status is a link "view on etherscan". To the right, a modal window titled "部署合約" (Deployment Contract) provides detailed information about the transaction, including the status (Confirmed), source account (0x4CE13...6ae71), target account (建立新合約), nonce (16), amount (-0 SepoliaETH), gas limits (Gas 上限: 1093150, Gas 用量: 1083384), fees (Base fee: 0.004429273 GWEI, Priority fee: 1.5 GWEI), total gas fee (0.00163 SepoliaETH / 0.19 ETC), and max fee per gas (0.000000002 SepoliaETH / 0 ETC). A red box highlights the "View on block explorer" link in the modal. Another red box highlights the "view on etherscan" link in the main transaction area. At the bottom, a footer bar includes a Scam Alert icon, Initialize as git repo, Did you know? (You can verify your contract using the Sourcify plugin), page number 205, and RemixAI Copilot (enabled).

remix.ethereum.org/#lang=en&optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.20.min.js

DEPLOY & RUN TRANSACTIONS

ENVIRONMENT
Injected Provider - MetaMask
Sepolia (11155111) network

ACCOUNT
0x4CE...6ae71 (1.55223749867917)

GAS LIMIT
Estimated Gas
Custom 3000000

VALUE
0 Wei

CONTRACT
SaluToken - SaluToken_flattened.sol
evm version: canary

Deploy
Publish to IPFS

creation of SaluToken pending...

view on etherscan

[block:7107445 txIndex:15] from: 0x4ce...6ae71 to: SaluToken data: 0x608...a0033 logs: 1 hash: 0x46b...b9ade

Status
已確認

來源帳戶
0x4CE13...6ae71

目的帳戶
建立新合約

交易
Nonce 16

數量 -0 SepoliaETH

Gas 上限 (單位) 1093150

Gas 用量 (單位) 1083384

Base fee (GWEI) 0.004429273

Priority fee (GWEI) 1.5

Total gas fee 0.00163 SepoliaETH / 0.19 ETC

Max fee per gas 0.000000002 SepoliaETH / 0 ETC

Did you know? You can verify your contract using the Sourcify plugin.

205

https://sepolia.etherscan.io/tx/0x791e18ae0dcb1005a4a261cb410b53f2ee65d057618d45b3109d26fc4a120288

RemixAI Copilot (enabled)

View Transaction on Sepolia

The screenshot shows a web browser window displaying the Etherscan Sepolia Testnet transaction details for the hash `0x791e18ae0dc...120288`. The transaction was successful, having 85 block confirmations at the time of capture.

Transaction Details:

- Transaction Hash: `0x791e18ae0dc...120288`
- Status: Success (highlighted with a red box)
- Block: `7107445` (85 Block Confirmations)
- Timestamp: 17 mins ago (Nov-19-2024 06:45:24 AM UTC)
- Transaction Action: Call `0x60806040` Method by `0x4CE135aB...64E06ae71`
- From: `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71`
- To: `[0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b Created]` (highlighted with a red box)

A red arrow points from the text "Click contract address" to the "To" field entry, indicating where to click to view the deployed contract details.

Smart Contract Information

The screenshot shows the Etherscan interface for a Sepolia Testnet contract at address `0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b`. The page is titled "Contract 0x8fDa5de52f87c6A544cA2BaB4fb75fb7a4cAD31b".

Contract Creator: `0x4CE135aB...64E06ae71` (at tx `0x791e18ae0d...`)

Token Tracker: SaluToken (ST)

Latest Transaction:

Transaction Hash	Method	Block	Age	From	To	Amount	Txn Fee
<code>0x791e18ae0d...</code>	<code>0x60806040</code>	7107445	22 mins ago	<code>0x4CE135aB...64E06ae71</code>	Contract Creation	0 ETH	0.00162987

Smart Contract Information

The screenshot shows a browser window displaying the Etherscan Token Tracker for the Sepolia Testnet. The URL in the address bar is <https://sepolia.etherscan.io/token/0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b>. The page is titled "SaluToken (ST) Token Tracker".

The main content area is divided into three sections: Overview, Market, and Other Info.

- Overview:** MAX TOTAL SUPPLY is listed as 0 ST.
- Market:** ONCHAIN MARKET CAP is \$0.00. CIRCULATING SUPPLY MARKET CAP is -.
- Other Info:** TOKEN CONTRACT (WITH 18 DECIMALS) is 0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b.

At the bottom, there is a table header for a transaction history table with columns: Transaction Hash, Method, Block, Age, From, To, and Amount.

Red boxes highlight the token name "SaluToken (ST)", the max total supply "0 ST", and the entire transaction history table header.

View & Publish

The screenshot shows a web browser displaying the Etherscan interface for a contract on the Sepolia Testnet. The URL in the address bar is <https://sepolia.etherscan.io/address/0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b#code>. The page features three main sections: Overview, More Info, and Multichain Info. Below these are tabs for Transactions, Token Transfers (ERC-20), Contract (which is highlighted with a red box), and Events. A call-to-action message encourages users to verify and publish their contract source code. At the bottom, there are links for Decompiling Bytecode, Switching to Opcodes View, and Similar Contracts, along with the contract's byte code hash.

Contract Address 0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b

sepolia.etherscan.io/address/0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b#code

Sepolia Testnet

Etherscan

Contract 0x8fDa5de52f87c6A544cA2BaB4fb75fb7a4cAD31b

Overview

ETH BALANCE
0 ETH

More Info

CONTRACT CREATOR
0x4CE135aB...64E06ae71 at txn 0x791e18ae0d...

TOKEN TRACKER
SaluToken (ST)

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

0x608060405234801561000f575f80fd5b50600436106100cd575f3560e01c8063715018a61161008a578063a1ee8c7811610064578063a1ee8c7814610201578063a9059ccb1461020b578063dd62

View & Publish Contract Source Code

The screenshot shows a web browser window titled "Verify & Publish Contract Sou" with the URL "sepolia.etherscan.io/verifyContract?a=0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b". 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". A red box highlights the contract address input field containing "0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b". Another red box highlights the "Solidity (Single file)" compiler type selection. A third red box highlights the "v0.8.26+commit.8a97fa7a" compiler version selection. A fourth red box highlights the "3) MIT License (MIT)" open source license selection. A fifth red box highlights the "Continue" button at the bottom.

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](#).

1 Enter Contract Details — 2 Verify & Publish

Please enter the Contract Address you would like to verify
0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b

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 [i](#)
3) MIT License (MIT)

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

Continue Reset

View & Publish Contract Source Code

The screenshot shows a web browser window for the Sepolia Testnet on etherscan.io. The URL in the address bar is `sepolia.etherscan.io/verifyContract-solc?a=0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b&c=v0.8.26%2bcommit.8a97fa7a&lctypte=3`. The page title is "Verify & Publish Contract Source Code". The main content area has two steps: "1 Enter Contract Details" and "2 Verify & Publish", with step 2 being active. A red box highlights the "Contract Address" field, which contains the value `0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b`. Below it, the "Compiler Type" is listed as "SINGLE FILE / CONCATENATED METHOD" and the "Compiler Version" as "v0.8.26+commit.8a97fa7a". The page also includes a search bar and navigation links for 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

Upload Contract Source Code

1. If the contract compiles correctly at [REMX](#), 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: `0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b`

Compiler Type: SINGLE FILE / CONCATENATED METHOD

Compiler Version: v0.8.26+commit.8a97fa7a

View & Publish Contract Source Code

Enter the Solidity Contract Code below *

```
// SPDX-License-Identifier: MIT
// File: @openzeppelin/contracts/token/ERC20/IERC20.sol

// OpenZeppelin Contracts (last updated v5.1.0) (token/ERC20/IERC20.sol)

pragma solidity ^0.8.20;

/** 
 * @dev Interface of the ERC-20 standard as defined in the ERC.
```

Fetch from Gist

Advanced Configuration

Optimization ⓘ Runs (Optimizer) ⓘ EVM Version to target ⓘ

No 200 default (compiler defaults)

License Type ⓘ

3) MIT License (MIT)

請把flatten的檔案
SaluToken_flattened.sol
貼上來。

View & Publish Contract Source Code

Sepolia Solidity Contract Sour x +

sepolia.etherscan.io/verifyContract-solc?a=0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b&c=v0.8.26%2bcommit.8a97fa7a&lctype=3

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

Constructor Arguments ABI-encoded
For contracts that were created with constructor parameters

For additional information on Constructor Arguments, [see our KB entry](#)

Contract Library Address (for contracts that use libraries, supports up to 10 libraries) ▾

成功! CLOUDFLARE

Verify and Publish Reset

View & Publish Contract Source Code

The screenshot shows a web browser window for the Sepolia Testnet on Etherscan. The URL in the address bar is `sepolia.etherscan.io/verifyContract-solc?a=0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b&c=v0.8.26%2bcommit.8a97fa7a&lctype=3`. The page title is "Verify & Publish Contract Source Code". The main content area has two steps: "1 Enter Contract Details" and "2 Verify & Publish". Step 1 is completed with a green success message: "Successfully generated Bytecode and ABI for Contract Address [0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b]". Step 2 is labeled "click". Below the steps, there's a note: "Learn how to verify your contract on multiple blockchains with a single API key [here](#)". At the bottom, there's a "Code Reader" button.

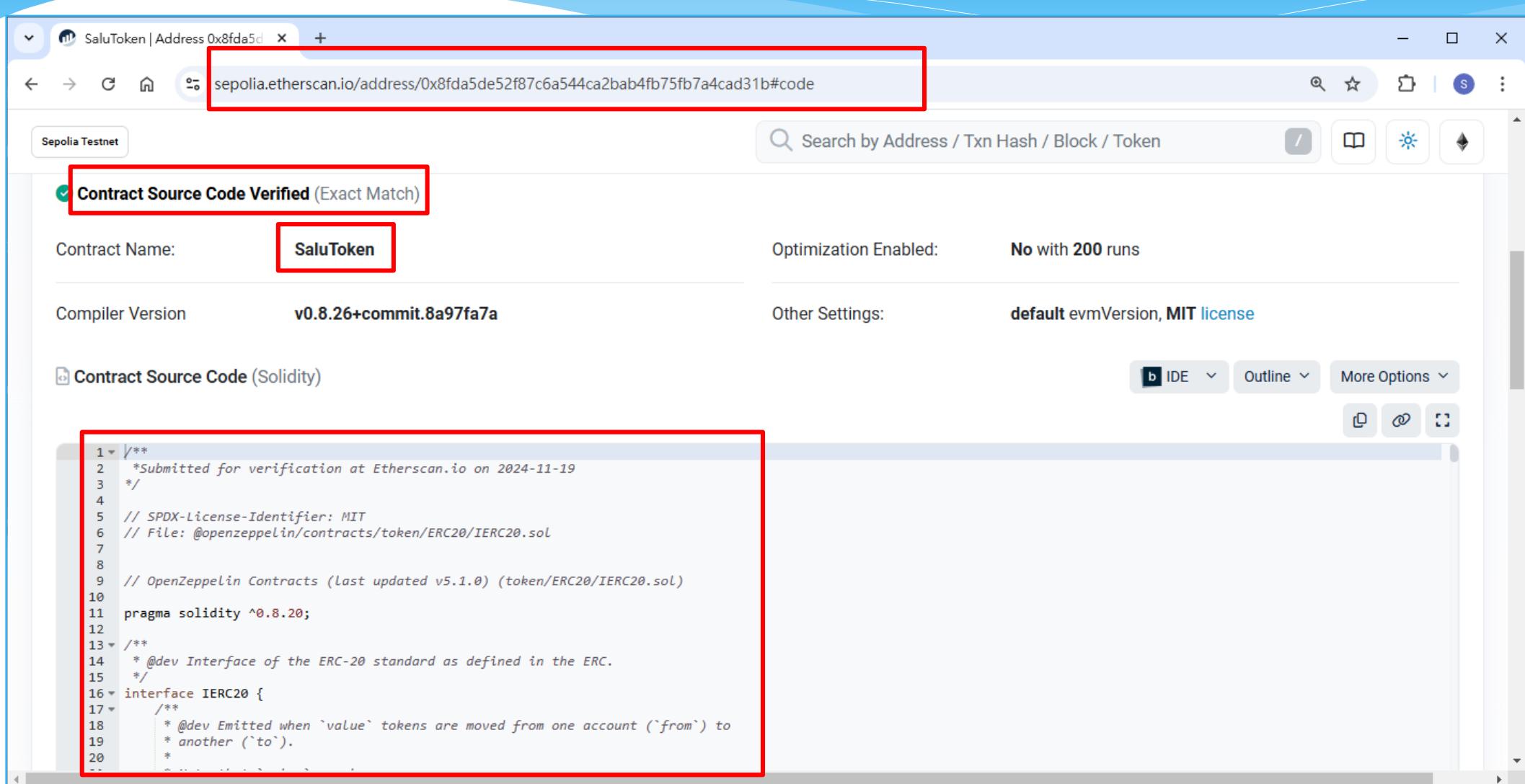
1 Enter Contract Details — 2 Verify & Publish

Successfully generated Bytecode and ABI for Contract Address
[0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b]

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

Code Reader

View Contract Source Code



View & Publish Contract Source Code

The screenshot shows the Etherscan interface for a contract at address 0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b. The browser title bar reads "SaluToken | Address 0x8fda5d". The URL in the address bar is "sepolia.etherscan.io/address/0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b#code". The page header includes a "Sepolia Testnet" button, a search bar, and various navigation icons.

Contract Details: The contract is identified by its address and name: "Contract 0x8fDa5de52f87c6A544cA2BaB4fb75fb7a4cAD31b".

Source Code Tab: The "Source Code" tab is selected, indicated by a green border. Below it, there are three tabs: "Transactions", "Token Transfers (ERC-20)", and "Contract". The "Contract" tab is also highlighted with a red border.

Contract Functions: Below the tabs, three buttons are shown: "Code" (highlighted with a red border), "Read Contract", and "Write Contract".

Contract Status: A green checkmark icon indicates "Contract Source Code Verified (Exact Match)".

Contract Metadata: The contract name is "SaluToken", compiler version is "v0.8.26+commit.8a97fa7a", optimization is "Enabled" with 200 runs, and other settings include "default evmVersion, MIT license".

More Info: The contract creator is listed as "0x4CE135aB...64E06ae71" with a transaction link to "0x791e18ae0d...". The token tracker is "SaluToken (ST)".

Multichain Info: The status is "N/A".

Search: A search bar at the bottom right allows searching for source code.

Connect to Web3

The screenshot shows the Sepolia Testnet version of the Etherscan interface for the SaluToken contract at address 0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b. The 'Contract' tab is selected. A red box highlights the 'Read Contract' button in the 'Code' section. A red arrow points from this button to a modal window titled 'sepolia.etherscan.io 顯示'. The modal contains a note about the beta status of the feature and two buttons: '確定' (Confirm) and '取消' (Cancel), with '確定' also highlighted by a red box.

sepolia.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.

確定 取消

ETH BALANCE
0 ETH

Transactions Token Transfers (ERC-20) Contract Events

Code Read Contract Write Contract

Connect to Web3

Read Contract Information

1. allowance

2. balanceOf

3. decimals

[Expand all] [Reset]

Connect a Wallet-MetaMask

The screenshot shows a web browser window displaying the Etherscan interface for a SaluToken contract on the Sepolia Testnet. The URL in the address bar is <https://sepolia.etherscan.io/address/0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b#readContract>. A modal window titled "Connect a Wallet" is overlaid on the page. The modal contains an informational message and three wallet connection options: MetaMask (selected and highlighted with a red box), WalletConnect, and Coinbase Wallet. The background of the Etherscan interface shows the contract's balance (0 ETH) and various interaction buttons like "Read Contract" and "Write Contract".

Connected Web3 using account

The screenshot illustrates the integration of a Ethereum wallet with an Etherscan interface. A red arrow points from the 'Connected Web3' status in the Etherscan sidebar to the wallet's address in the Etherscan header.

Etherscan Interface (Left):

- Address: SaluToken | Address 0x8fda5d
- Network: Sepolia Testnet
- ETH BALANCE: 0 ETH
- TRANSACTIONS: [Link]
- TOKEN TRANSFERS (ERC-20): [Link]
- Contract: [Link] (Selected)
- Events: [Link]
- Code: [Link]
- Read Contract: [Link] (Connected: Web3 [0x4ce1...ae71])
- Write Contract: [Link]
- Read Contract Information: [Link]
- FUNCTIONS: 1. allowance, 2. balanceOf, 3. decimals

Wallet Interface (Right):

- Address: 0x4CE135aB...64E06ae71
- Balance: 1.5522 SepoliaETH
- Portfolio: [Link]
- Buttons: Buy & Sell, Swap, Bridge, 發送 (Send), 接收 (Receive)
- Tokens: [Link]
- NFTs: [Link]
- Transactions: [Link] (Selected)
- Nov 19, 2024: 部署合約 已確認 -0 SepoliaETH -0 ETC
- Oct 30, 2024: 接收 已確認 0.01 SepoliaETH 1.18 ETC
- Oct 30, 2024: 接收 已確認 0.05 SepoliaETH 5.91 ETC

Read Contract

Sepolia Testnet

Connected - Web3 [0x4ce1...ae71]

Read Contract Information

[Expand all] [Reset]

- 1. allowance
- 2. balanceOf
- 3. decimals
18 uint8
- 4. name
- 5. owner
0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71 address
- 6. symbol
ST string
- 7. totalSupply
0 uint256

Connected Web3 using account

The screenshot shows a web browser window with the URL <https://sepolia.etherscan.io/address/0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b#writeContract>. The browser title bar says "SaluToken | Address 0x8fda5d". The page displays information about a contract on the Sepolia Testnet, specifically the SaluToken (ST) contract at address 0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b. The "Contract" tab is selected. The "More Info" section shows the CONTRACT CREATOR as 0x4CE135aB...64E06ae71, created at tx 0x791e18ae0d..., and the TOKEN TRACKER as SaluToken (ST). Below this, there are tabs for "Transactions", "Token Transfers (ERC-20)", "Contract" (which is active), and "Events". A red arrow points from the "Connected - Web3 [0x4ce1...ae71]" button in the "Contract" tab area to the "Salu1" account in the right-hand sidebar. The sidebar also shows a balance of 1.5522 SepoliaETH and links for "Buy & Sell", "Swap", "Bridge", "發送" (Send), and "接收" (Receive). The "Tokens" tab is selected, showing a history of transactions:

Date	Action	Value	Gas
Nov 19, 2024	部署合約 已確認	-0 SepoliaETH -0 ETC	0.00
Oct 30, 2024	接收 已確認	0.01 SepoliaETH 1.19 ETC	0.00
	接收 已確認	0.05 SepoliaETH 5.93 ETC	0.00

Write Contract - issueToken

1

2

https://sepolia.etherscan.io/address/0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b#writeContract

Write Contract - issueToken

The screenshot shows the Etherscan interface for the SaluToken contract on the Sepolia Testnet. The URL is <https://sepolia.etherscan.io/address/0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b#writeContract>.

Overview: ETH BALANCE: 0 ETH

More Info: CONTRACT CREATOR: 0x4CE135aB...64E06ae71 at txn 0x791e18ae0d...
TOKEN TRACKER: SaluToken (ST)

Multichain Info: N/A

Transactions: Contract (selected), Events

Code: Connected - Web3 [0x4ce1...ae71]

Write Contract:

1. approve (0x095ea7b3)
2. issueToken (0xa1ee8c78)

Buttons: Write (disabled), View your transaction (highlighted with a red box)

View Transaction

The screenshot shows the Etherscan Transaction Details page for a Sepolia Testnet transaction. The transaction hash is `0xd163637dc2a001e162a7eb467272457193299f08b1e6731d1fb2454e5f32dd9b`. The status is marked as "Success". The transaction was included in block `7107826` with 9 block confirmations. It occurred 2 mins ago (Nov-19-2024 08:04:48 AM UTC). The transaction action was an "Issue Token" call to contract `0x4CE135aB...64E06ae71` on address `0x8fDa5de5...7a4cAD31b`. The transaction originated from address `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71` and interacted with address `0x8fDa5de52f87c6A544cA2BaB4fb75fb7a4cAD31b`.

1 `0x8fDa5de52f87c6A544cA2BaB4fb75fb7a4cAD31b`

224
<https://sepolia.etherscan.io/tx/0xd163637dc2a001e162a7eb467272457193299f08b1e6731d1fb2454e5f32dd9b>

Smart Contract Information

The screenshot shows the Etherscan interface for the SaluToken (ST) token on the Sepolia Testnet. The token has a total supply of 1,000 ST, 1 holder, and 1 total transfer. A transaction from address 0x00000000...00000000 to address 0x4CE135aB...64E06ae71 is highlighted, showing an amount of 1,000.

Token SaluToken (ST)

ERC-20

Overview

- MAX TOTAL SUPPLY: 1,000 ST
- HOLDERS: 1
- TOTAL TRANSFERS: 1

Market

- ONCHAIN MARKET CAP: \$0.00
- CIRCULATING SUPPLY MARKET CAP: -

Other Info

- TOKEN CONTRACT (WITH 18 DECIMALS): [0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b](#)

Transfers **Holders** **Contract**

A total of 1 transaction found

Transaction Hash	Method	Block	Age	From	To	Amount
0xd163637dc2...	Issue Token	7107826	4 mins ago	0x00000000...00000000	0x4CE135aB...64E06ae71	1,000

Smart Contract Information

The screenshot shows the Etherscan.io interface for a smart contract on the Sepolia Testnet. The address is 0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b. The page is titled "Contract" with the address. Key sections include:

- Overview:** ETH BALANCE: 0 ETH.
- More Info:** CONTRACT CREATOR: 0xCE135aB...64E06ae71 (at tx 0x791e18ae0d...), TOKEN TRACKER: SaluToken (ST).
- Multichain Info:** N/A

Below these are tabs for Transactions, Token Transfers (ERC-20), Contract (selected), and Events. The Transaction section shows the latest transaction:

Transaction Hash	Method	Block	Age	From	To	Amount	Txn Fee
0xd163637dc2...	Issue Token	7107826	3 mins ago	0xCE135aB...64E06ae71	0x8fda5de5...7a4cad31b	0 ETH	0.00014119

Connected Web3 using account

使用另一個account address, 不是Owner

Connected - Web3 [0x7bee...8f0c]

Contract

0x7BeeE...d8f0C

0.42 SepoliaETH

Write Contract - issueToken

The image shows a composite screenshot illustrating the process of writing a smart contract on the Sepolia testnet.

Left Panel (Browser View):

- Address:** SaluToken | Address 0x8fd...
URL: sepolia.etherscan.io/address/0x8fd...#writeContract
- Overview:** ETH BALANCE: 0 ETH
- More Info:**
 - CONTRACT CREATOR: 0x4CE135aB...64E06ae71 at txn 0x791e18ae0d...
 - TOKEN TRACKER: SaluToken (ST)
- Contract Tab:** Write Contract (highlighted with a red box)
- Actions:** 1. approve (0x095ea7b3)
2. issueToken (0xa1ee8c78)
- Bottom Left (Blue Circle):** A blue circle labeled "1" with a red arrow pointing to the "Write" button in the Actions section.

Right Panel (MetaMask Extension):

- MetaMask UI:** Shows the address 0x8fd...AD31b and a warning message: "We were not able to estimate gas. There might be an error in the contract and this transaction may fail." with a "I want to proceed anyway" button.
- Bottom Right (Blue Circle):** A blue circle labeled "2" with a red arrow pointing to the "I want to proceed anyway" button.
- Bottom Right (Blue Circle):** A blue circle labeled "3" with a red arrow pointing to the warning message "This transaction is likely to fail".
- Bottom Right (Blue Circle):** A blue circle labeled "4" with a red arrow pointing to the bottom right corner of the MetaMask interface.
- Bottom Right (Blue Buttons):** 拒絕 (Reject) and 確認 (Confirm) buttons.

Write Contract - issueToken

The image shows a composite screenshot illustrating the process of writing a smart contract on the Sepolia testnet.

Left Panel (Browser View): A screenshot of a browser window showing the Etherscan interface for the SaluToken contract (Address: 0x8fd...). The "Contract" tab is selected. Under "Code", the "Connected - Web3" status is shown. Below it, two transaction steps are listed:

1. approve (0x095ea7b3)
2. issueToken (0xa1ee8c78)

A red box highlights the "issueToken" button, which is circled with a blue number 1.

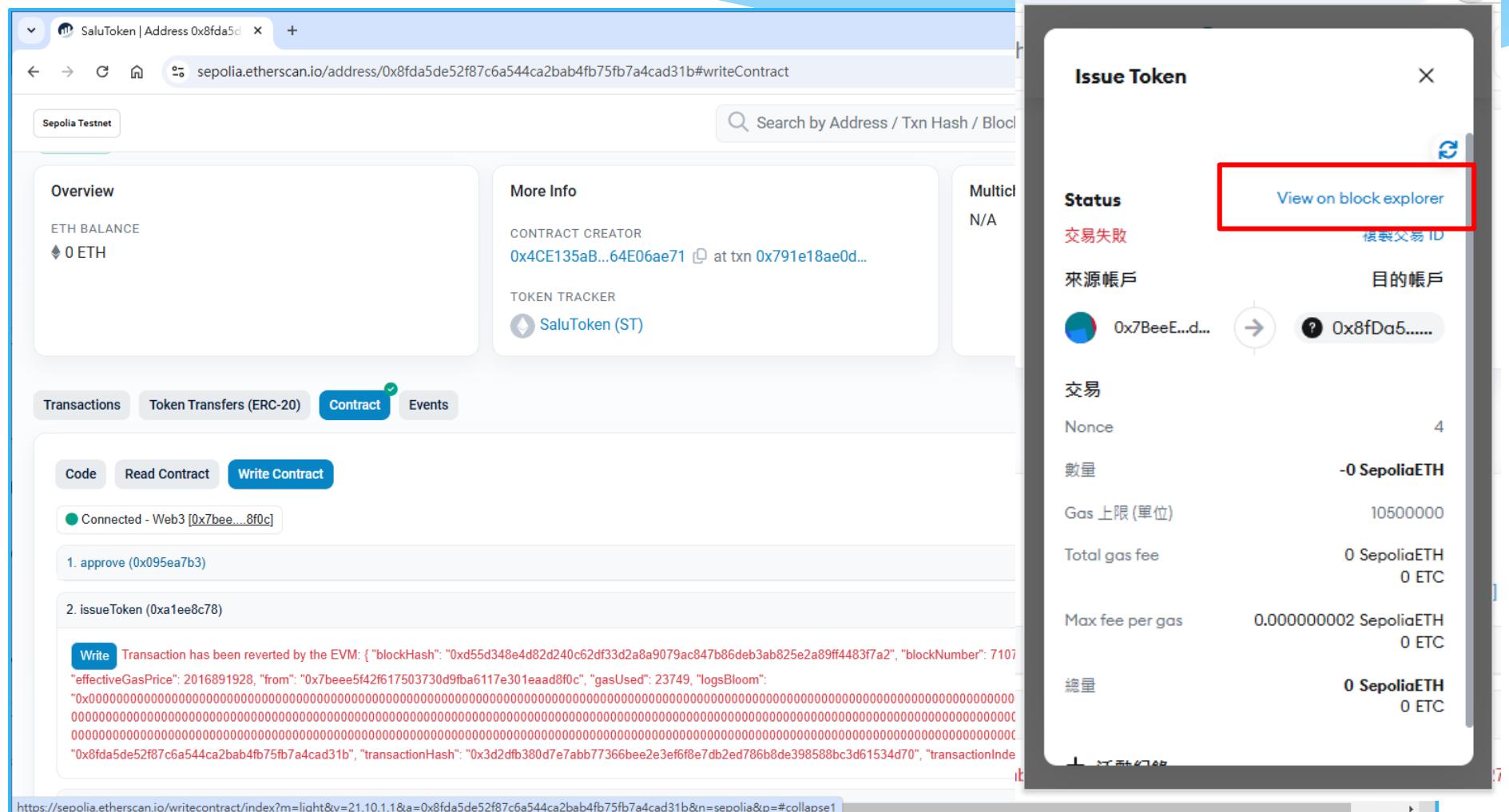
Right Panel (MetaMask): A screenshot of the MetaMask extension interface. The address bar shows "0x8fDa5...AD31b : ISSUE TOKEN". A red box highlights this address, and a blue arrow points from the browser's "issueToken" button to this area, labeled with a blue number 2.

The MetaMask interface displays the following information:

- Estimated changes:** A warning message: "We were not able to estimate gas. There might be an error in the contract and this transaction may fail." This message is highlighted with a red box.
- Estimated fee:** 2.55 SepoliaETH (Market -60 sec, Max fee: 0.02404102 SepoliaETH)
- 總量:** 2.55 0.02148959 SepoliaETH (Amount + gas fee, Max amount: 0.02404102 SepoliaETH)
- 自訂 NONCE:** Input field containing the value 4.
- Buttons:** 拒絕 (Reject) and 確認 (Confirm), the latter being highlighted with a red box and circled with a blue number 3.

Write Contract - issueToken

View Transaction



View Transaction

The screenshot shows a transaction details page on Etherscan for a Sepolia Testnet transaction. The transaction hash is 0x3d2dfb380d7e7abb77366bee2e3ef6f8e7db2ed786b8de398588bc3d61534d70. The transaction failed with a custom error 'OwnableUnauthorizedAccount' because the sender was not the owner. The transaction action was a call to the 'Issue Token' function of contract 0x7BeeE5F4...1EA. The transaction was successful with 17 block confirmations and occurred 4 minutes ago on Nov-19-2024 at 08:24:48 AM UTC. The from address is 0x7BeeE5F42F617503730d9Fba6117E301EAdd and the to address is 0x8fDa5de52f87c6A544cA2BaB4fb75fb7a4ca. A warning message indicates an execution revert.

The right side of the screenshot shows a GitHub browser tab displaying the OpenZeppelin contracts repository. The specific file 'Ownable.sol' is highlighted with a red box, and a red arrow points from the error message in the transaction details to the corresponding revert statement in the code:

```
function _checkOwner() internal view virtual {
    if (owner() != _msgSender()) {
        revert OwnableUnauthorizedAccount(_msgSender());
    }
}
```

Smart Contract Information

The screenshot shows the Etherscan interface for a smart contract at address 0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b. The page is titled 'SaluToken | Address 0x8fda5d'. The 'Contract' section is highlighted with a red box. Below it, the 'ETH BALANCE' box (containing '0 ETH') is also highlighted with a red box. The 'Transactions' tab is selected, showing three recent transactions. The first transaction, with Hash 0xd2dfb380d7..., is highlighted with a red box. The 'From' field of this transaction (containing '0x7BeeE5F4...1EAad8f00') is also highlighted with a red box. A red arrow points from the text '因為不是Owner,所以產生Transaction Error' to this 'From' field.

Contract 0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b

ETH BALANCE
0 ETH

因為不是Owner,所以產生Transaction Error

Transaction Hash	Method	Block	Age	From	To	Amount	Txn Fee
0xd2dfb380d7...	Issue Token	7107918	19 mins ago	0x7BeeE5F4...1EAad8f00	0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b	0 ETH	0.00004789
0xd163637dc2...	Issue Token	7107826	39 mins ago	0x4CE135aB...64E06ae71	0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b	0 ETH	0.00014119
0x791e18ae0d...		0x60806040	1 hr ago	0x4CE135aB...64E06ae71	Create: SaluToken	0 ETH	0.00162987

Smart Contract Information

The screenshot shows the Etherscan Token SaluToken (ST) page. Key highlighted sections include:

- Token SaluToken (ST)** (ERC-20)
- MAX TOTAL SUPPLY**: 1.000 ST
- HOLDERS**: 1
- TOTAL TRANSFERS**: 1
- ONCHAIN MARKET CAP**: \$0.00
- CIRCULATING SUPPLY MARKET CAP**: -
- TOKEN CONTRACT (WITH 18 DECIMALS)**: 0x8fda5de52f87c6a544ca2bab4fb75fb7a4cad31b

Below the table, a transaction is highlighted:

Transaction Hash	Method	Block	Age	From	To	Amount
0xd163637dc2...	Issue Token	7107826	41 mins ago	0x00000000...00000000	0x4CE135aB...64E06ae71	1,000



Case 3 – ETH

Ethereum genesis block

- The ICO carried out by ethereum.org starting from 2014/07/22 to 2014/09/02.
- The launch of the Ethereum mainnet occurred on 2015/07/30.
- Ethereum Genesis block reference
 - <https://github.com/ethereum/ethereumj/blob/develop/ethereumj-core/src/main/resources/genesis/frontier.json>
 - <https://github.com/odaceo/ethereum-genesis-block/blob/master/mainnet.json>
- Ethereum Contract Address
 - <https://etherscan.io/address/0xd76b5c2a23ef78368d8e34288b5b65d616b746ae>
 - 此smart contract沒有mint function，而是把token在一開始時放在Genesis block內給各個相關或有幫助的account address。
 - 原本是PoW機制，現已改成為PoS機制。

Creating The Genesis Block

- Every blockchain starts with a genesis block. When Geth is run with default settings for the first time, it commits the Mainnet genesis to the database. For a private network, it is generally preferable to use a different genesis block. The genesis block is configured using a **genesis.json** file whose path must be provided to Geth on start-up. When creating a genesis block, a few initial parameters for the private blockchain must be defined.
- <https://geth.ethereum.org/docs/fundamentals/private-network>

```
## to create a blockchain node  
$ geth init --datadir data genesis.json
```

```
## When Geth is started using --datadir data the genesis block defined in genesis.json will be used  
$ geth --datadir data --networkid 12345
```

Ethereum Genesis block

The screenshot shows a GitHub repository page for the Ethereum project. The URL in the browser's address bar is highlighted with a red box and points to github.com/ethereum/ethereumj/blob/develop/ethereumj-core/src/main/resources/genesis/frontier.json. The repository is public and has 82 issues and 10 pull requests. The 'Code' tab is selected. The left sidebar shows a list of files, with 'frontier.json' highlighted by a red box. The main content area displays the JSON code for the Genesis block. A red arrow points from the line "balance": "13370000000000000000000000000000" to a red box containing the text "express in WEI (10¹⁸ WEI = 1 ETH)".

```
1  {
2    "alloc": {
3      "3282791d6fd713f1e94f4bfd565eaa78b3a0599d": {
4        "balance": "13370000000000000000000000000000" ← express in WEI (1018 WEI = 1 ETH)
5      },
6      "17961d633bcf20a7b029a7d94b7df4da2ec5427f": {
7        "balance": "22942700000000000000000000000000"
8      },
9      "493a67fe23decc63b10dda75f3287695a81bd5ab": {
10        "balance": "88000000000000000000000000000000"
11      }
12    }
13  }
```

Ethereum Genesis block

The screenshot illustrates the connection between the Ethereum Genesis block and its initial state in the Ethereumj core codebase.

Left Tab (etherscan.io):

- Address: 0x3282791d6fd713f1e94f4bfd565eaa78b3a0599d (highlighted with a red box)
- ETH Price: \$3,111.31 (-0.29%)
- Gas: 6.679 Gwei
- Overview: ETH BALANCE 0 ETH, ETH VALUE \$0.00, TOKEN HOLDINGS \$58.87 (42 Tokens)
- More Info: PRIVATE NAME TAGS, TRANSACTIONS SENT (Latest: 1579 days ago), FUNDED BY 0x0F4B92E1...9f0d97b68
- Transactions: Latest 3 from a total of 3 transactions. The first transaction is a Transfer from GENESIS to 0x3282791d...8B3A0599d with an amount of 1,337 ETH.

Right Tab (GitHub repository):

- Repository: ethereum / ethereumj (Public)
- Code tab is selected.
- File: frontier.json (highlighted with a red box)
- Code content:

```
1  {
2     "alloc": {
3         "3282791d6fd713f1e94f4bfd565eaa78b3a0599d": {
4             "balance": "13370000000000000000000000000000"
5         },
6         "17961d633bcf20a7b02a7d94b7df4da2ec5427f": {
7             "balance": "22942700000000000000000000000000"
8         }
9     }
10 }
```

Ethereum Genesis block

The screenshot shows a GitHub repository page for 'odaceo / ethereum-genesis-block'. The URL in the address bar is highlighted with a red box: `github.com/odaceo/ethereum-genesis-block/blob/master/mainnet.json`. The repository is public. The left sidebar shows files: LICENSE, README.md, gorli.json, mainnet.json (which is selected and highlighted with a red box), rinkeby.json, and testnet.json. The right panel displays the content of the mainnet.json file. The file contains JSON configuration for the Ethereum blockchain, including parameters like chainId, homesteadBlock, daoForkBlock, and various EIP-related values.

```
1  {
2      "config": {
3          "chainId": 1,
4          "homesteadBlock": 1150000,
5          "daoForkBlock": 1920000,
6          "daoForkSupport": true,
7          "eip150Block": 2463000,
8          "eip150Hash": "0x2086799aeebeae135c246c65021c82b4e15a2c451340993aacfd2751886514f0",
9          "eip155Block": 2675000,
10         "eip158Block": 2675000,
11         "byzantiumBlock": 4370000,
12         "constantinopleBlock": 7280000,
13         "petersburgBlock": 7280000,
14         "istanbulBlock": 9069000,
15         "muirGlacierBlock": 9200000,
16         "ethash": {}
17     },
18     "nonce": "0x42",
19     "timestamp": "0x0",
20     "extraData": "0x11bbe8db4e347b4e8c937c1c8370e4b5ed33adb3db69cbdb7a38e1e50b1b82fa",
```

Ethereum Genesis block

The Block 0 of Ethereum

A screenshot of a web browser showing the details of Ethereum Block 0 on the Etherscan website. The page title is "Ethereum Blocks #0 | Etherscan". The main content area is titled "Block #0". A red box highlights the "Block Height" field, which shows "0 < Finalized". Another red box highlights the "Transactions" section, which states "8893 transactions and 0 contract internal transaction in this block". A red arrow points from the "Transactions" section towards the timestamp. The timestamp is listed as "3400 days ago (Jul 30 2015 03:26:13 PM +UTC)". Below these, there is a list of other block statistics.

Block Height	Finalized
0 <	Finalized
Timestamp	3400 days ago (Jul 30 2015 03:26:13 PM +UTC)
Transactions	8893 transactions and 0 contract internal transaction in this block
Mined by	0x000 (Null: 0x000...000) in 15 secs
Block Reward	5 ETH
Uncles Reward	0
Difficulty	17,179,869,184
Total Difficulty	17,179,869,184
Size	540 bytes
Gas Used	0(0.00%)
Gas Limit	5,000
Extra Data	◆◆◆N4{N◆◆ ◆p◆◆3◆◆◆i◆◆z8◆◆◆ (Hex:0x11bbe8db4e347b4e8c937c1c8370e4b5ed33adb3db69c7db7a38e1e50b1b82fa)
Ether Price	N/A

The transactions present in the genesis block

Ethereum Transactions Inform x +

etherscan.io/txs?block=0

ETH Price: \$3,110.30 (-0.71%) Gas: 7.941 Gwei

Search by Address / Txn Hash / Block / Token / Domain Name

For Block 0

Pro Tips: See the first address & transaction that funded an Externally Owned Account with ETH.

A total of 8,893 transactions found

Transaction Hash	Method	Block	Age	From	To	Amount	Gas Fee
GENESIS_756f4...	-	0	3400 days ago	GENESIS	0x756F45E3...C4db0b5a0	200 ETH	0
GENESIS_f42f9...	-	0	3400 days ago	GENESIS	0xf42f9052...49eee0F21	197 ETH	0
GENESIS_2489...	-	0	3400 days ago	GENESIS	0x2489ac12...691e9798E	1,000 ETH	0
GENESIS_ddf58...	-	0	3400 days ago	GENESIS	0xDDF5810a...B320F24Ac	17,900 ETH	0
GENESIS_c951...	-	0	3400 days ago	GENESIS	0xC951900c...071Dbc36A	327.6 ETH	0
GENESIS_6806...	-	0	3400 days ago	GENESIS	0x68064083...F6c43CDcA	1,730 ETH	0
GENESIS_9d0f3...	-	0	3400 days ago	GENESIS	0x9D0F347e...61eCf334b	4,000 ETH	0
GENESIS_9328...	-	0	3400 days ago	GENESIS	0x9328D55c...76ee840A3	4,000 ETH	0
GENESIS_7e7f1...	-	0	3400 days ago	GENESIS	0x7E7F18a0...434a25Ef7	66.85 ETH	0
GENESIS_3c86...	-	0	3400 days ago	GENESIS	0x3C869c09...b76231ff2	1,000 ETH	0

All these transactions are generated from a **genesis address** to **ETH addresses (0x...)** by transferring a certain amount of **ETH**.

1 2 3

<https://etherscan.io/txs?block=0>

Ethereum ICO page

1. The ICO carried out by ethereum.org starting from 2014/07/22 to 2014/09/02.
2. The public offering lasted 42 days and totaled the collection of 31591 Bitcoin, equal to approximately 18.4 million US dollars, or 60,102,216 ETH.

The screenshot shows the Ethereum ICO landing page. At the top right is a stylized blue and white geometric logo with the word "ethereum" below it. Below the logo is a message stating "the ethereum ether sale has now concluded." At the bottom left, there is a timer indicating "ether sale ends in:" with values 0 Days, 0 Hours, 0 Minutes, and 0 Seconds. In the center, the text "Estimated ether purchased:" is followed by a large red-bordered box containing the number "60,102,216 ETH". At the bottom right, there is another timer for "time remaining at current price:" with values 0 Days, 0 Hours, 0 Minutes, and 0 Seconds.

the ethereum ether sale has now concluded.

ether sale ends in:

0 Days 0 Hours 0 Minutes 0 Seconds

Estimated ether purchased:

60,102,216 ETH

time remaining at current price:

0 Days 0 Hours 0 Minutes 0 Seconds

Account Address-1

Address 0x000d836201318ec6899a67540690382780743280

Etherscan

Address 0x000d836201318Ec6899a67540690382780743280

Pro Tips: See the first address & transaction that funded an Externally Owned Account with ETH.

Overview

ETH BALANCE: 200 ETH

ETH VALUE: \$615,571.11 (@ \$3,077.86/ETH)

TOKEN HOLDINGS: \$27.18 (25 Tokens)

More Info

PRIVATE NAME TAGS: + Add

TRANSACTIONS SENT: Latest: N/A First: N/A

FUNDED BY: N/A

Multichain Info: \$615,598.29 (Multichain Portfolio)
No addresses found

Transactions

Latest 1 from a total of 1 transactions

Transaction Hash	Method	Block	Age	From	To	Amount	Txn Fee
GENESIS_000d836201...	-	0	3395 days ago	GENESIS	0x000D8362...780743280	200 ETH	0

Advanced Filter

Download Page Data

Download CSV Export

Genesis block

245

<https://etherscan.io/address/0x000d836201318ec6899a67540690382780743280>

Account Address-2

The screenshot shows the Etherscan.io address details page for the account `0x001762430ea9c3a26e5749afdb70da5f78ddbb8c`. The address is highlighted with a red box. The page displays various sections: Overview (ETH BALANCE: 0 ETH, ETH VALUE: \$0.00, TOKEN HOLDINGS: \$4.61), More Info (PRIVATE NAME TAGS, TRANSACTIONS SENT, FUNDED BY N/A), and Multichain Info (\$4.6 (Multichain Portfolio), No addresses found). An advertisement for Stake.com is visible on the right. Below these sections, a table lists the latest 5 transactions. A red box highlights the first transaction, which is a transfer from the Genesis block (`0x00176243...F78DDBB8C`) to the account (`0x8B20cFd2...E6780da5`). The transaction hash is `0xb5d544f451a...`, Method is Transfer, Block is 4282660, Age is 2615 days ago, Amount is 194.898236 ETH, and Txn Fee is 0.000441.

Transaction Hash	Method	Block	Age	From	To	Amount	Txn Fee
<code>0xb5d544f451a...</code>	Transfer	4282660	2615 days ago	<code>0x00176243...F78DDBB8C</code>	<code>0x8B20cFd2...E6780da5</code>	194.898236 ETH	0.000441
<code>0xde2b59b680...</code>	Transfer	4282656	2615 days ago	<code>0x00176243...F78DDBB8C</code>	<code>0x34dEC6db...073A53465</code>	4.9 ETH	0.000441
<code>0x22d3ba9f69d...</code>	Transfer	4282603	2615 days ago	<code>0x00176243...F78DDBB8C</code>	<code>0x34dEC6db...073A53465</code>	0.1 ETH	0.000441
<code>0x7cae1dcfa6c...</code>	Transfer	4282600	2615 days ago	<code>0x00176243...F78DDBB8C</code>	<code>0x22fb1C1a...f15037601</code>	0.1 ETH	0.000441
<code>GENESIS_001762430e...</code>	-	0	3395 days ago	GENESIS	<code>0x00176243...F78DDBB8C</code>	200 ETH	0

<https://etherscan.io/address/0x001762430ea9c3a26e5749afdb70da5f78ddbb8c>

Account Address-3

The screenshot shows the Etherscan.io interface for the address `0x001d14804b399c6ef80e64576f657660804fec0b`. The address is highlighted with a red box in the browser's address bar and in the main search bar. The page displays the following information:

- Overview:** ETH BALANCE: 0 ETH, ETH VALUE: \$0.00, TOKEN HOLDINGS: \$0.00 (3 Tokens).
- More Info:** PRIVATE NAME TAGS (+ Add), TRANSACTIONS SENT: Latest: 3108 days ago, First: 3355 days ago, FUNDED BY: N/A.
- Multichain Info:** \$0 (Multichain Portfolio), No addresses found.

The **Transactions** tab is selected, showing the following table:

Transaction Hash	Method	Block	Age	From	To	Amount	Txn Fee
0xe1822bdf66f...	Transfer	1501459	3108 days ago	0x001D1480...0804feC0b	OUT 0xcFAb9aC2...F72Fb5902	0.07425932 ETH	0.00042
0x6e443af86a8...	Transfer*	202249	3355 days ago	0x001D1480...0804feC0b	OUT Kraken	4,199.9 ETH	0.02532067
GENESIS_001d14804b...	-	0	3395 days ago	GENESIS	IN 0x001D1480...0804feC0b	4,200 ETH	0

A red box highlights the "To" column header in the table, and another red box highlights the "To" field for the Genesis transaction, which is labeled "Genesis block". A red arrow points from the "To" field to the "Genesis block" label.

Search the Ether Token

Bancor: Old ETH Token | Addr x +

etherscan.io/address/0xd76b5c2a23ef78368d8e34288b5b65d616b746ae

ETH Price: \$2,595.81 (-0.80%) Gas: 18.864 Gwei

Search by Address / Txn Hash / Block / Token / Domain

Etherscan Home Blockchain Tokens NFTs Resources Developers More Sign In

Contract 0xD76b5c2A23ef78368d8E34288B5b65D616B746aE Buy Exchange Play Gaming

Sponsored: Metawin: Win 3 ETH worth \$7k+! Grab Your FREE ENTRY Today. [Enter Here.](#)

Bancor: Old ETH Token Source Code # Bancor # Old Contract

Overview

ETH BALANCE
107.701676227902966279 ETH

ETH VALUE
\$279,573.04 (@ \$2,595.81/ETH)

TOKEN HOLDINGS
\$0.59 (26 Tokens)

More Info

PRIVATE NAME TAGS
+ Add

CONTRACT CREATOR
Bancor: Deployer 2 at txn 0x0704cf41ef1...

TOKEN TRACKER
Ether Token (ETH)

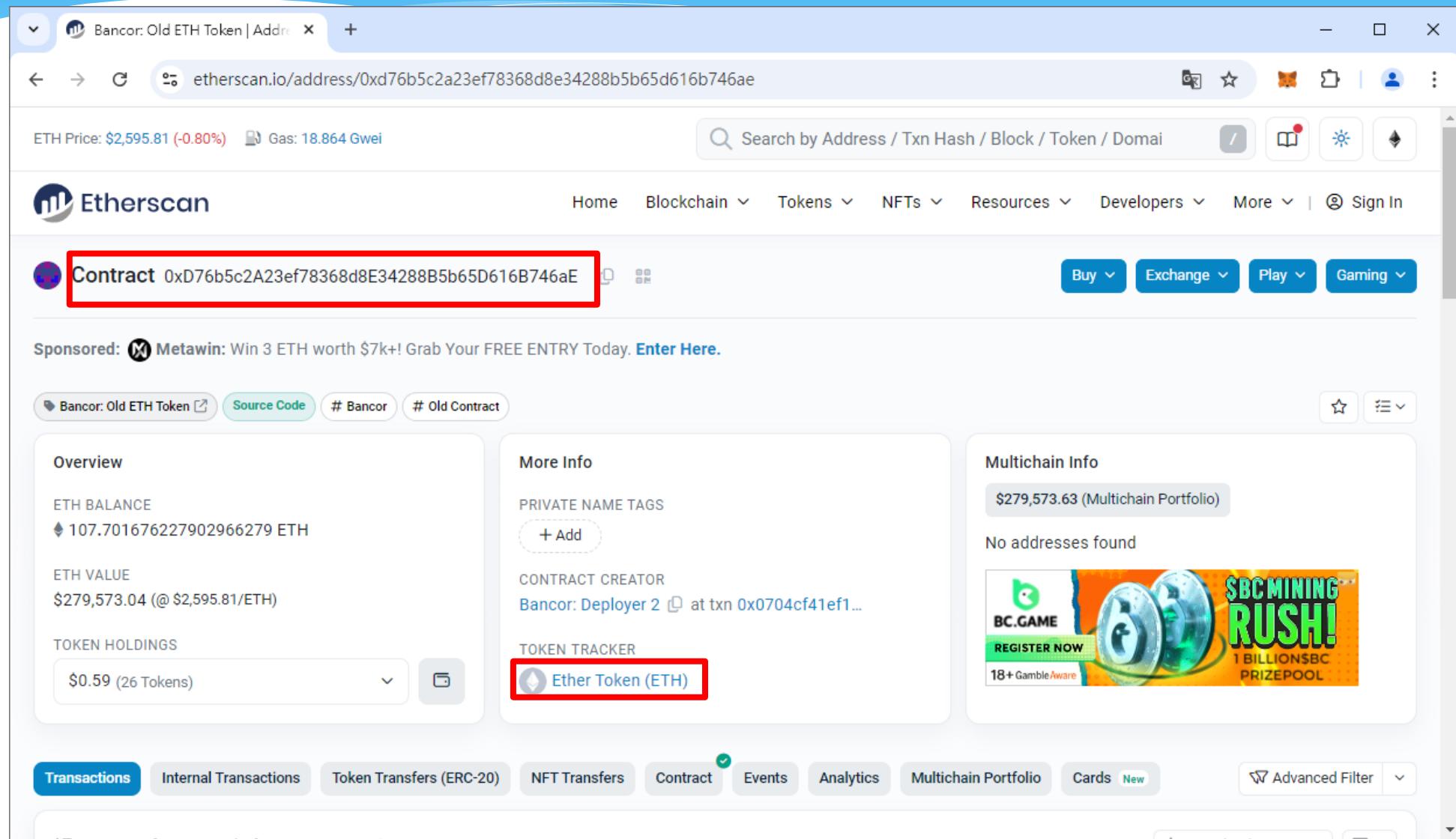
Multichain Info

\$279,573.63 (Multichain Portfolio)

No addresses found

BC.GAME REGISTER NOW SBC MINING RUSH! 1 BILLION\$BC PRIZEPOOL 18+ GambleAware

Transactions Internal Transactions Token Transfers (ERC-20) NFT Transfers Contract Events Analytics Multichain Portfolio Cards New Advanced Filter



Search the Ether Token

Ether Token (ETH) Token Track + ↻

etherscan.io/token/0xd76b5c2a23ef78368d8e34288b5b65d616b746ae

ETH Price: \$2,596.05 (-0.79%) Gas: 22.33 Gwei

Search by Address / Txn Hash / Block / Token / Domain Name

Etherscan Home Blockchain Tokens NFTs Resources Developers More Sign In

Token Ether Token (ETH) ⓘ

Buy Exchange Play Gaming

Sponsored: Stake Stake: 200% Bonus, 75k Raffle, Best VIP Program, Instant Withdrawals - Provably Fair. [Claim Bonus](#)

ERC-20 # Old Contract

bancor.network

MAX TOTAL SUPPLY
107.70167622790296... ETH

OVERVIEW

HOLDERS
298 (0.00%)

TOTAL TRANSFERS
22,188 ⓘ

Market

ONCHAIN MARKET CAP ⓘ
\$0.00

CIRCULATING SUPPLY MARKET CAP
-

OTHER INFO

TOKEN CONTRACT (WITH 18 DECIMALS)
[0xd76b5c2a23ef78368d8e34288b5b65d616b746ae](#) 🔗

BC.GAME REGISTER NOW 18+ GambleAware AND BETS IN CRYPTO TOP CRYPTO GAMES

Transfers Holders Info Contract Analytics Cards New

A total of 22,188 transactions found
(Showing the last 10k records)

Download Page Data Advanced Filter First < Page 1 of 400 > Last

249

Search the Ether Token

The screenshot shows a web browser window for Etherscan.io, displaying the details of the EtherToken contract. The URL in the address bar is etherscan.io/address/0xd76b5c2a23ef78368d8e34288b5b65d616b746ae#code. The page header indicates an ETH Price of \$2,595.81 (-0.80%) and Gas cost of 18.864 Gwei.

The navigation bar includes tabs for Transactions, Internal Transactions, Token Transfers (ERC-20), NFT Transfers, Contract (highlighted with a red box), Events, Analytics, Multichain Portfolio, Cards (New), and Advanced Filter.

Below the tabs, there are buttons for Code (selected), Read Contract, and Write Contract, along with a search bar for Source Code.

Contract details shown:

- Contract Name: EtherToken
- Optimization Enabled: Yes with 200 runs
- Compiler Version: v0.4.11+commit.68ef5810
- Other Settings: default evmVersion

A note indicates that the Contract Source Code is Verified (Exact Match).

The Contract Source Code (Solidity) is displayed in a code editor:

```
1 // SPDX-License-Identifier: MIT
2 // Submitted for verification at Etherscan.io on 2017-06-22
3 /*
4  * pragma solidity ^0.4.11;
5  *
6  */
7 /*
8  * Overflow protected math functions
9  */
10 contract SafeMath {
11     /**
12      constructor
13  }
```

At the bottom left, the URL is <https://remix.ethereum.org/address/0xd76b5c2a23ef78368d8e34288b5b65d616b746ae>.

On the right side, a dropdown menu for IDE options shows "Remix IDE" highlighted with a red box. Other options include Blockscan IDE, Code Reader (Beta), and More Options.

Search the Ether Token

需展開mainnet才可看到EtherToken.sol

FILE EXPLORER

WORKSPACES ▾

Sign in

code-sample

.deps

npm

@openzeppelin

contracts

utils

cryptography

ECDSA.sol

mainnet

0xd76b5c2a23ef78368d8e34288b5b65d616b746ae

EtherToken.sol

sepolia

0xd76b5c2a23ef78368d8e34288b5b65d616b746ae

@openzeppelin

contracts

MetaMultiSigWallet.sol

MetaMultiSigWallet.sol

```
// SPDX-License-Identifier: MIT
//
// Off-chain signature gathering multisig that streams funds - @austingriffith
//
// started from scaffold-eth - meta-multi-sig-wallet example https://github.com/
// (off-chain signature based multi-sig)
// added a very simple streaming mechanism where `onlySelf` can open a withdraw-b...
//
pragma solidity >=0.8.0 <0.9.0;
// Not needed to be explicitly imported in Solidity 0.8.x
// pragma experimental ABIEncoderV2;

import "@openzeppelin/contracts/utils/cryptography/ECDSA.sol";

contract MetaMultiSigWallet {
    using ECDSA for bytes32;

    event Deposit(address indexed sender, uint amount, uint balance);
    event ExecuteTransaction(address indexed owner, address payable to, uint256 value);
    event Owner(address indexed owner, bool added);
    mapping(address => bool) public isOwner;
    uint public signaturesRequired;
    uint public nonce;
}
```

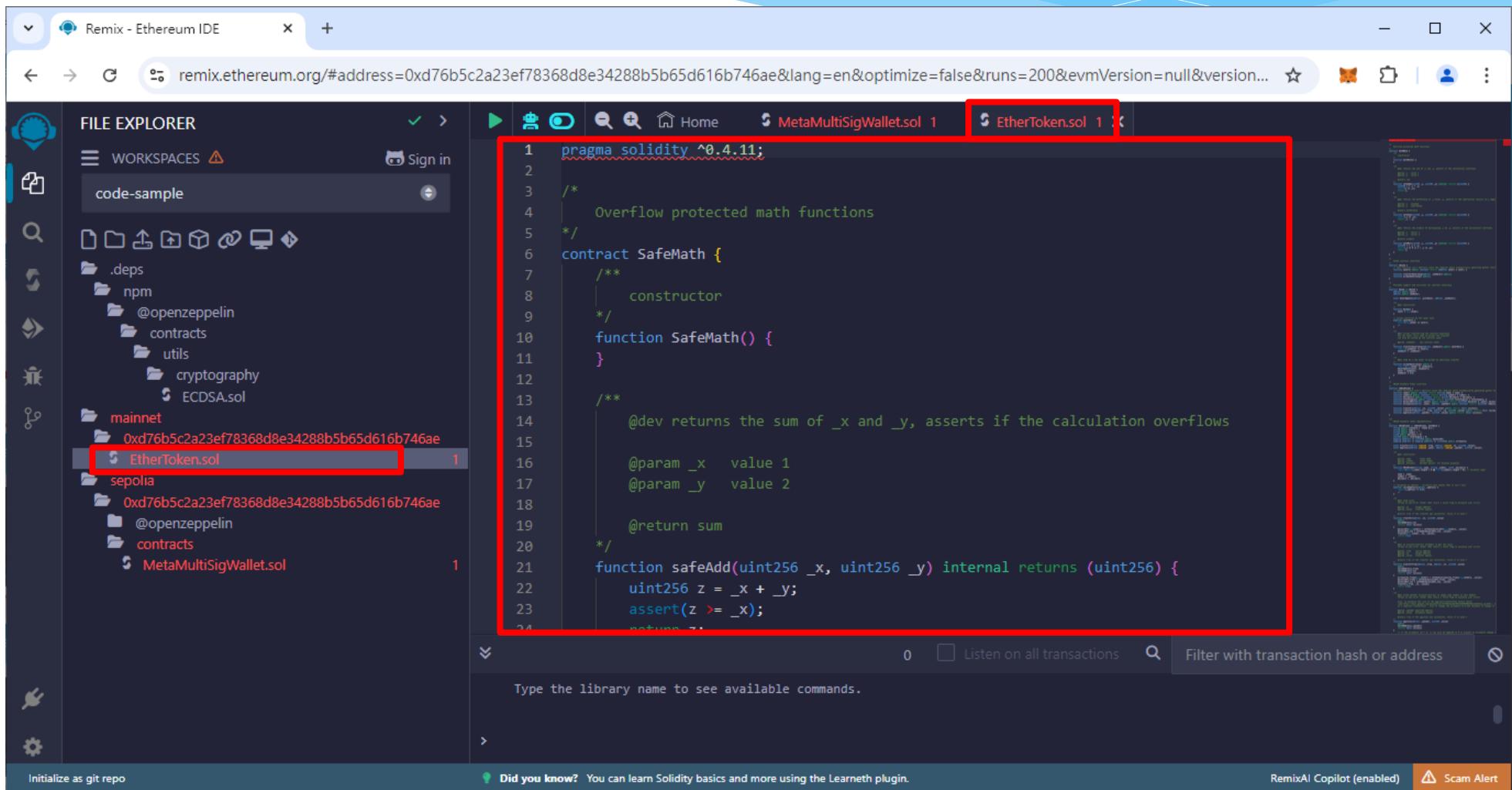
0 Listen on all transactions Filter with transaction hash or address

Looking for contract(s) verified on different networks of Etherscan for contract address
0xd76b5c2a23ef78368d8e34288b5b65d616b746ae

Initialize as git repo Did you know? You can learn Solidity basics and more using the LearnETH plugin.

RemixAI Copilot (enabled) Scam Alert

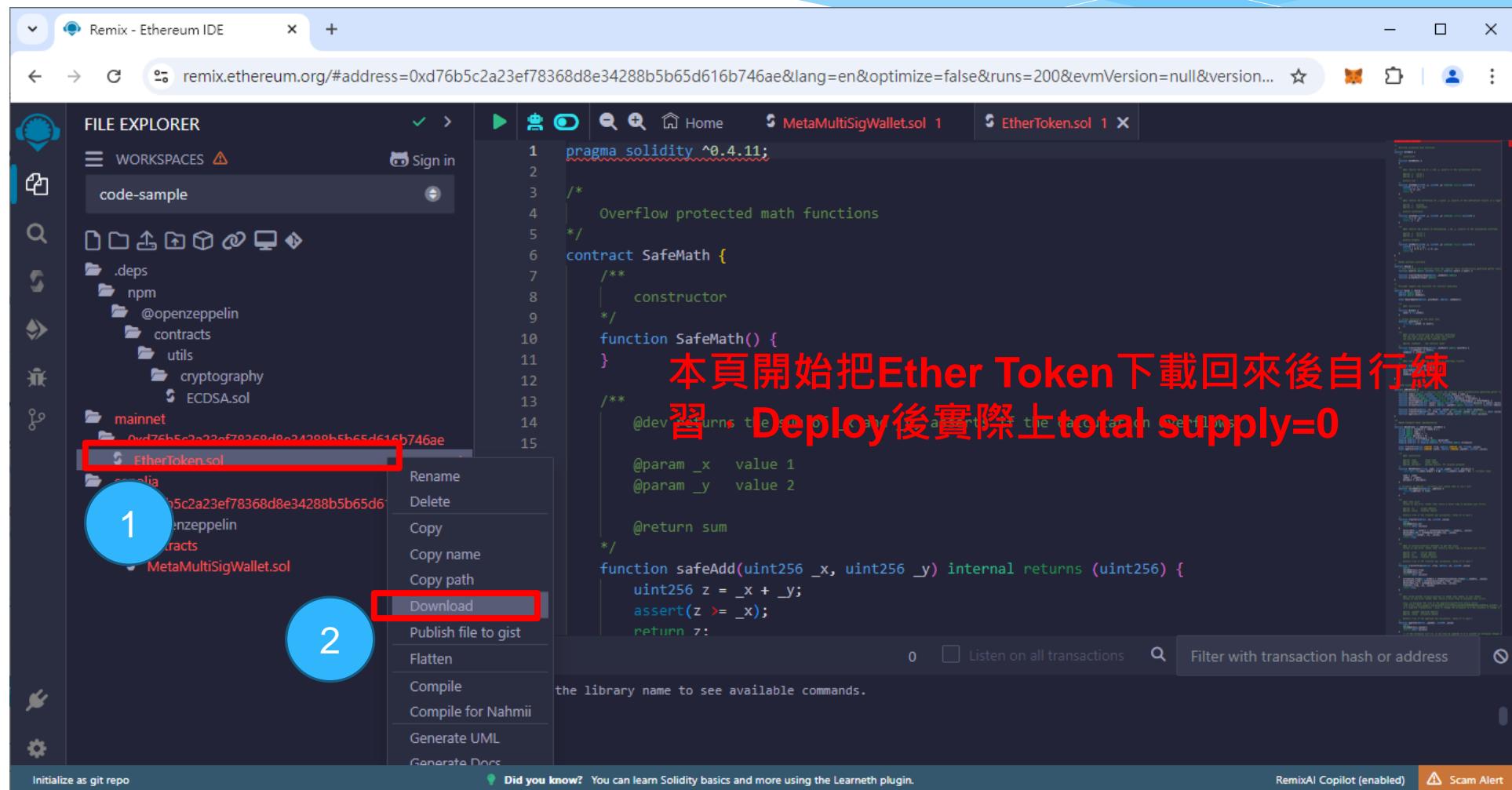
EtherToken.sol



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

- File Explorer:** On the left, it shows a workspace named "code-sample". Inside, there are folders ".deps", ".npm", "mainnet", and "sepolia". Under "mainnet", the file "EtherToken.sol" is selected and highlighted with a red box. Other files like "0xd76b5c2a23ef78368d8e34288b5b65d616b746ae" and "MetaMultiSigWallet.sol" are also listed.
- Code Editor:** The central area displays the Solidity code for "EtherToken.sol". The code includes imports from OpenZeppelin's SafeMath library and defines a SafeMath contract with a safeAdd function that performs addition and asserts that the result is greater than or equal to the first operand.
- Right Panel:** On the right, there is a sidebar with various tabs and sections, some of which are visible through a red box.
- Bottom Bar:** At the bottom, there are buttons for "Initialize as git repo", "Did you know?", "RemixAI Copilot (enabled)", and "Scam Alert".

Download the EtherToken.sol



Flatten the EtherToken.sol

The screenshot shows the Remix Ethereum IDE interface. The left sidebar contains a 'FILE EXPLORER' with a 'WORKSPACES' section. A folder named 'code-sample' is expanded, showing subfolders '.deps', 'mainnet', and 'sepolia'. Inside 'mainnet', there is a file named 'EtherToken.sol' which is highlighted with a red box and circled with a blue number '1'. In the main workspace, two tabs are open: 'MetaMultiSigWallet.sol 1' and 'EtherToken.sol 1'. The code editor displays the Solidity code for the SafeMath library and the EtherToken contract. A context menu is open over the 'EtherToken.sol' code, with the 'Flatten' option highlighted and also circled with a blue number '2'. Other options in the menu include Rename, Delete, Copy, Copy name, Copy path, Download, Publish file to gist, Compile, Compile for Nahmii, and Generate UML.

```
pragma solidity ^0.4.11;

/*
    Overflow protected math functions
*/
contract SafeMath {
    /**
     | constructor
     */
    function SafeMath() {}

    /**
     | @dev returns the sum of _x and _y, asserts if the calculation overflows
     |
     | @param _x  value 1
     | @param _y  value 2
     |
     | @return sum
     */
    function safeAdd(uint256 _x, uint256 _y) internal returns (uint256) {
        uint256 z = _x + _y;
        assert(z >= _x);
        return z;
    }
}
```

1 EtherToken.sol
2 Flatten

Add SPDX to EtherToken.sol

The screenshot shows the Remix Ethereum IDE interface. On the left, the FILE EXPLORER panel displays a workspace named 'code-sample' containing several files and folders, with 'EtherToken.sol' highlighted by a red box and a blue numbered circle '1'. The main code editor window shows the Solidity code for the EtherToken contract. A specific line of code, `// SPDX-License-Identifier: MIT`, is highlighted with a red box and a blue numbered circle '2'. A blue arrow points from the text 'Add this line' to this highlighted line. The code editor also shows other parts of the contract, including the SafeMath library and the transferFrom function.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.4.11;

/*
    Overflow protected math functions
*/
contract SafeMath {
    /**
     * constructor
     */
    function SafeMath() {}

    /**
     * @dev returns the sum of _x and _y, asserts if the calculation overflows
     *
     * @param _x    value 1
     * @param _y    value 2
     *
     * @return sum
     */
    function add(uint _x, uint _y) internal pure returns (uint) {
        require(_x <= uint(2**256 - 1));
        require(_y <= uint(2**256 - 1));
        uint256 sum = _x + _y;
        require((sum - _x) == _y);
        return sum;
    }

    /**
     * @dev returns true if the transfer was successful, false if it wasn't
     *
     * @param _from address to transfer from
     * @param _to   target address
     * @param _value transfer amount
     *
     * @return true if the transfer was successful, false if it wasn't
     */
    function transferFrom(address _from, address _to, uint256 _value)
        public
        validAddress(_from) // Use only one instance of this modifier
    {
        require(_value <= balanceOf(_from));
        require(_value >= balanceOf(_to));
        require(_value <= allowance[_from][msg.sender]);
        require(_value >= allowance[_from][msg.sender] - _value);
        allowance[_from][msg.sender] -= _value;
        transfer(_from, _to, _value);
    }
}
```

FILE EXPLORER

WORKSPACES

EtherToken.sol

1

2

Add this line

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

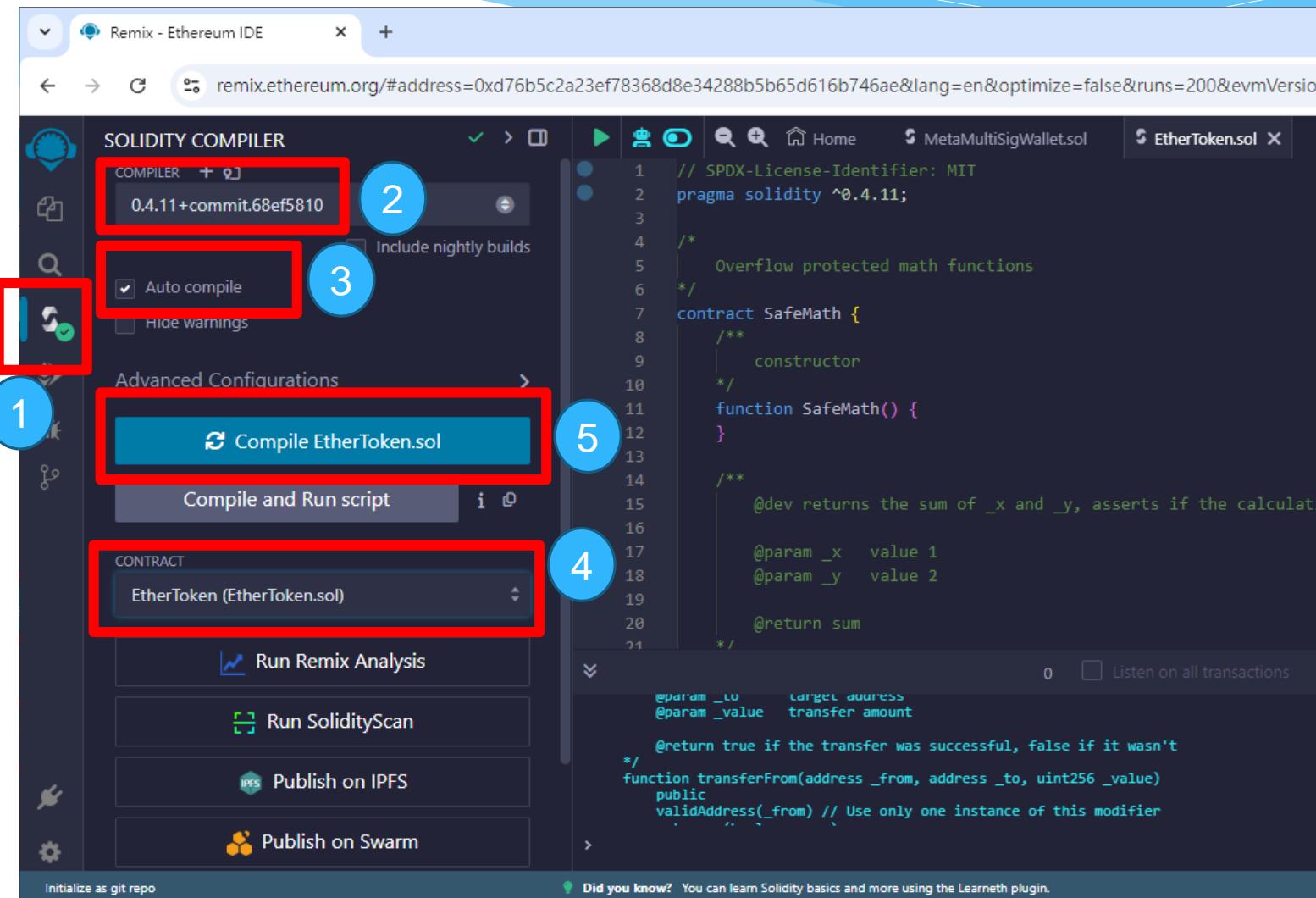
Initialize as git repo

RemixAI Copilot (enabled)

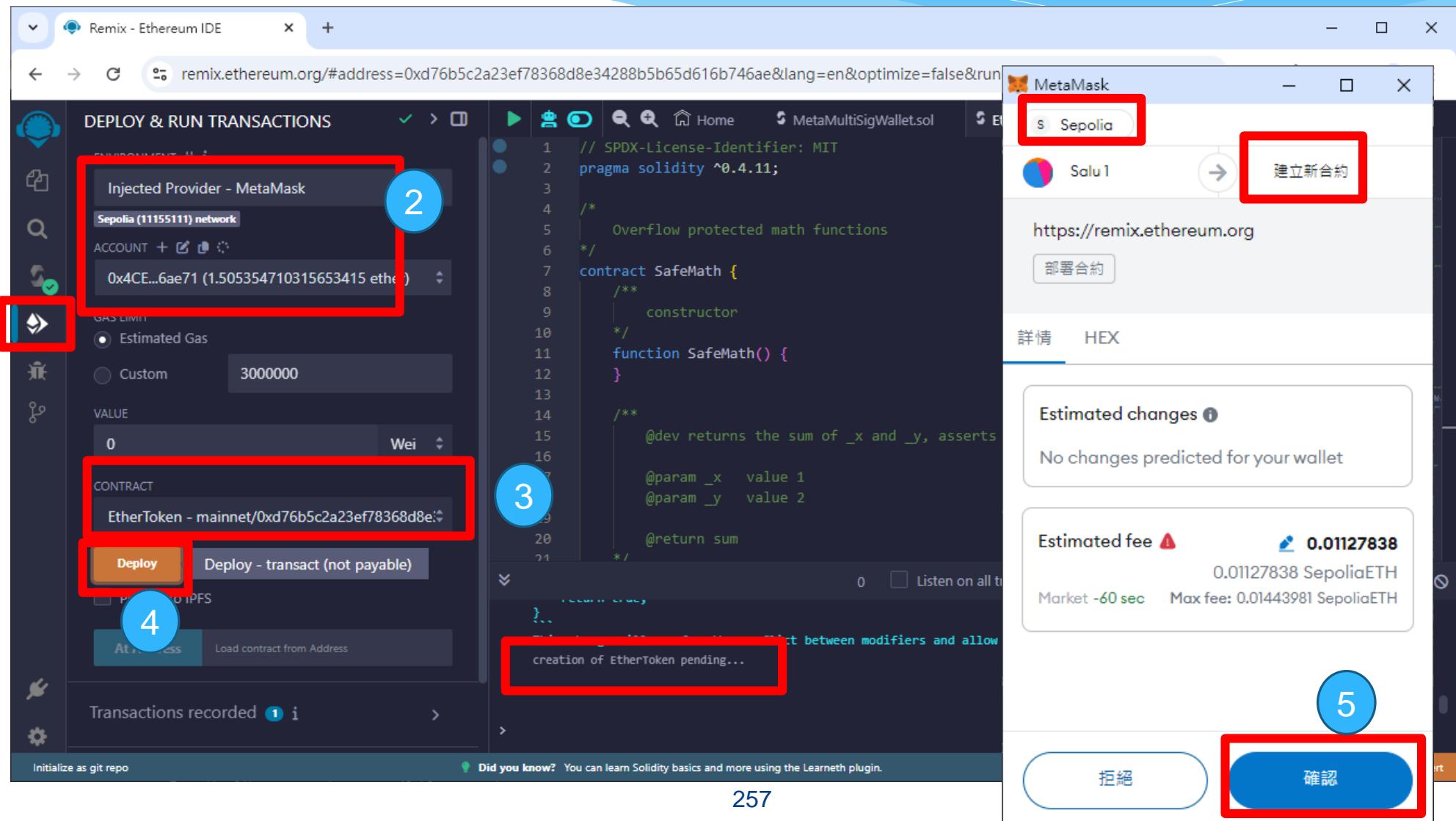
Scam Alert

255

Compile Smart Contract



Deploy an ERC-20 Token



Deploy and Run Transaction

The image shows three main components illustrating the process of deploying and running a transaction:

- Remix - Ethereum IDE (Left):** A screenshot of the Remix IDE interface. It displays the "DEPLOY & RUN TRANSACTIONS" sidebar on the left, which includes fields for "ENVIRONMENT" (set to "Injected Provider - MetaMask"), "ACCOUNT" (showing address 0x4CE13...6ae71 with 1.4939 SepoliaETH), "FACILITATOR" (set to "Estimated Gas"), and "VALUE" (set to 0 Wei). The "CONTRACT" section shows the deployment of "EtherToken - mainnet/0xd76b5c2a23ef78368d8e..." and includes a "Deploy" button. A red box highlights the "Deploy" button.
- Solidity Contract (Center):** A screenshot of the Solidity code for the SafeMath contract. The code defines a constructor and a function named `SafeMath()`. A red box highlights the "view on etherscan" link at the bottom of the code area.
- MetaMask Extension (Right):** A screenshot of the MetaMask extension interface. It shows the user's account balance as 1.4939 SepoliaETH. Below the balance are buttons for "Buy & Sell", "發送" (Send), "Swap", "Bridge", and "Portfolio". A red box highlights the "已確認" (Confirmed) status of the deployment transaction from Oct 18, 2024.

Deploy and Run Transaction

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

- Left Sidebar:** DEPLOY & RUN TRANSACTIONS panel with settings for ENVIRONMENT (Injected Provider - MetaMask, Sepolia network), ACCOUNT (0x4CE...6ae71), GAS LIMIT (Estimated Gas or Custom 3000000), VALUE (0 Wei), CONTRACT (EtherToken - mainnet/0xd76b5c2a23ef78368d8e34288b5b65d616b746ae), Deploy button, and Publish to IPFS option.
- Middle Panel:** Solidity code editor for `SafeMath.sol`. The code defines a SafeMath contract with a constructor and a add function that returns the sum of two parameters.
- Bottom Panel:** Transaction details for a deployed contract. It includes a link to view on etherscan, transaction hash [block:6896075 txIndex:21] from: 0x4ce...6ae71 to: EtherToken, and a note about logs and hash.
- Right Panel:** Deployment status window titled "部署合約" (Deploy Contract) showing transaction details:

Status	View on block explorer
已確認	複製交易 ID
來源帳戶	目的帳戶
0x4CE13...	建立新合約
交易	7
Nonce	-0 SepoliaETH
數量	1497135
Gas 上限 (單位)	1484188
Gas 用量 (單位)	6.236256086
Base fee (GWEI)	1.5
Priority fee (GWEI)	0.011482 SepoliaETH
Total gas fee	0.00000001 SepoliaETH
Max fee per gas	0.01148206 SepoliaETH
總量	

Annotations with red boxes highlight the "view on etherscan" button, the "View on block explorer" button in the deployment status window, and the "Did you know?" footer message.

View Transaction on Sepolia

The screenshot shows a web browser window displaying the Etherscan Sepolia Testnet transaction details for the hash `0x0a56409f3ba94e4e1c198f566217a27f2af711350c05d77668be038bb5496c68`. The transaction was successful, having 37 block confirmations at block 6896075. It was timestamped 9 mins ago on Oct-18-2024 at 03:49:00 AM UTC. The transaction action was a call to `0x60606040` via method `0x4CE135aB...64E06ae71`. The transaction originated from address `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71` and was sent to address `0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888`, which was created as part of the transaction.

Sepolia Transaction Hash (Txh) sepolia.etherscan.io/tx/0x0a56409f3ba94e4e1c198f566217a27f2af711350c05d77668be038bb5496c68

Etherscan

Transaction Details

Overview State

[This is a Sepolia Testnet transaction only]

Transaction Hash: `0x0a56409f3ba94e4e1c198f566217a27f2af711350c05d77668be038bb5496c68`

Status: Success

Block: 6896075 37 Block Confirmations

Timestamp: 9 mins ago (Oct-18-2024 03:49:00 AM UTC)

Transaction Action: Call `0x60606040` Method by `0x4CE135aB...64E06ae71`

From: `0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71`

To: `0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888` Created

Smart Contract Information

The screenshot shows a browser window displaying the Etherscan interface for a smart contract on the Sepolia Testnet. The URL in the address bar is <https://sepolia.etherscan.io/address/0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888>.

The page features a navigation bar with tabs for Sepolia Testnet, Home, Blockchain, Tokens, NFTs, and More. A search bar at the top right allows users to search by Address / Txn Hash / Block / Token.

The main content area includes:

- Contract Address:** 0x99c4Ae2324Ba2D9dA8FB32EA1458502d8B9C5888 (highlighted with a red box).
- More Info:** CONTRACT CREATOR [0x4CE135aB...64E06ae71](#) at txn [0x0a56409f3ba...](#) (highlighted with a red box and a red arrow pointing to the transaction link).
- Multichain Info:** N/A

Below these sections are tabs for Transactions, Token Transfers (ERC-20), Contract, and Events. The Transactions tab is selected, showing a single transaction entry:

Transaction Hash	Method	Block	Age	From	To	Amount	Txn Fee
0x0a56409f3ba...	0x60606040	6896075	13 mins ago	0x4CE135aB...64E06ae71	Contract Creation	0 ETH	0.01148205

At the bottom right of the table, there is a link to [Download: CSV Export].

Smart Contract Information

The screenshot shows the Etherscan interface for the Ether Token (ETH) on the Sepolia Testnet. The URL in the browser is sepolia.etherscan.io/token/0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888.

Token Overview: Ether Token (ETH)

MAX TOTAL SUPPLY: 0 ETH

HOLDERS: 0

TOTAL TRANSFERS: 0

Market:

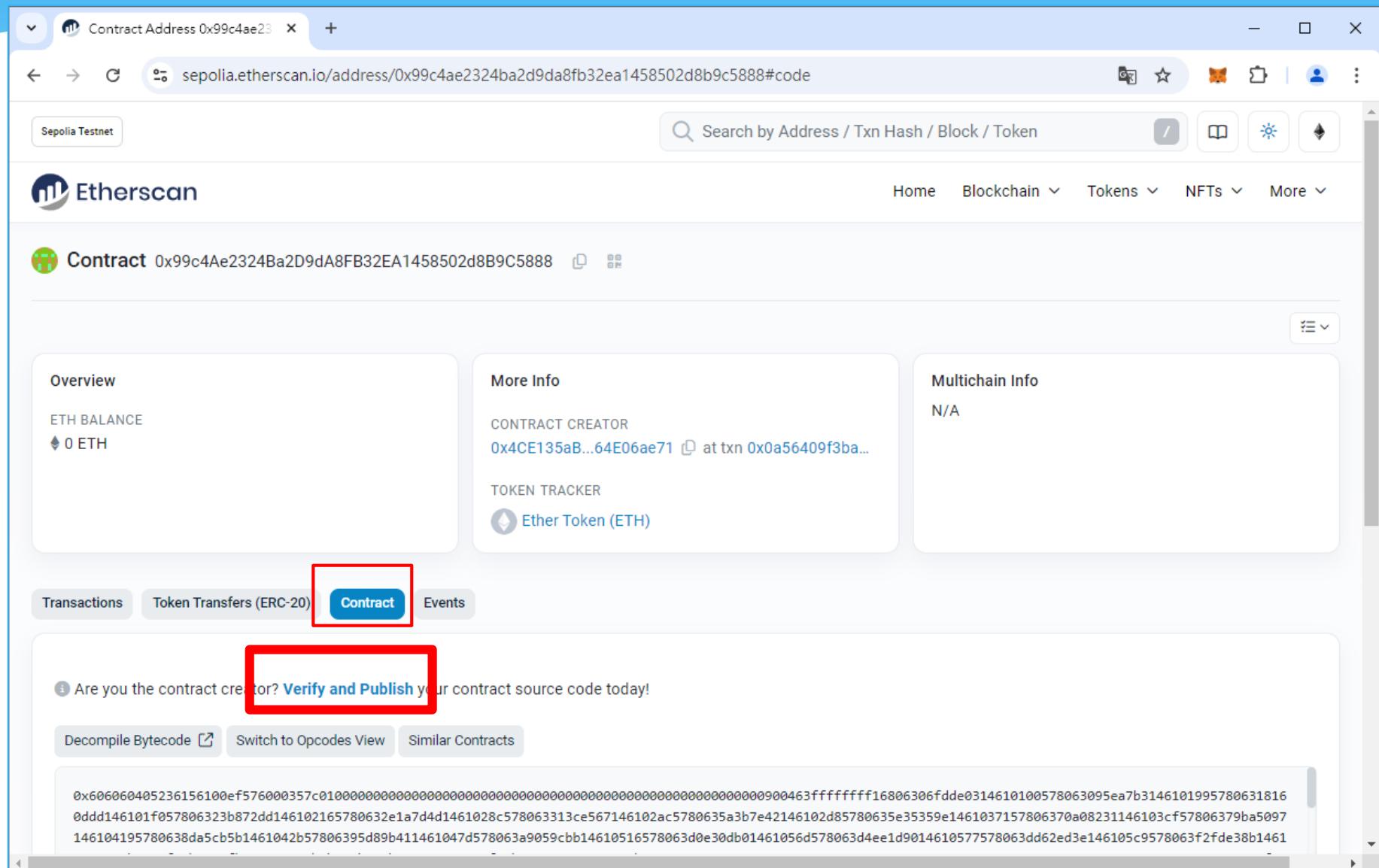
- ONCHAIN MARKET CAP: \$0.00
- CIRCULATING SUPPLY MARKET CAP: -

Other Info:

- TOKEN CONTRACT (WITH 18 DECIMALS): [0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888](#)

Transfers: Transaction Hash, Method, Block, Age, From, To, Amount

View & Publish



View & Publish Contract Source Code

The screenshot shows a web browser window titled "Verify & Publish Contract Source Code" on the Sepolia Testnet. The URL in the address bar is <https://sepolia.etherscan.io/verifyContract?a=0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888>. The page displays a form for entering contract details. The fields are as follows:

- Contract Address:** 0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888
- Compiler Type:** Solidity (Single file)
- Compiler Version:** v0.4.11+commit.68ef5810
- Open Source License Type:** 3) MIT License (MIT)
- Agreement:** I agree to the terms of service (checkbox checked)

At the bottom of the form are two buttons: "Continue" (highlighted with a red box) and "Reset". A tooltip message "Select Compiler Type before selecting Compiler Version" appears above the Compiler Version dropdown.

View & Publish Contract Source Code

The screenshot shows a web browser window for the Sepolia Solidity Contract Source page on etherscan.io. The URL in the address bar is `sepolia.etherscan.io/verifyContract-solc?a=0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888&c=v0.4.11%2bcommit.68ef5810...`. The page title is "Verify & Publish Contract Source Code". A sub-header explains that source code verification provides transparency and matches uploaded code with the blockchain. It also notes a simple interface for single-file contracts. Below this, two steps are outlined: "Enter Contract Details" (step 1) and "Verify & Publish" (step 2). The "Upload Contract Source Code" section contains instructions for compilation and verification. In the "Contract Address" field, the value `0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888` is highlighted with a red box. The "Compiler Type" is listed as "SINGLE FILE / CONCATENATED METHOD" and the "Compiler Version" is "v0.4.11+commit.68ef5810".

Sepolia Solidity Contract Sour x +

sepolia.etherscan.io/verifyContract-solc?a=0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888&c=v0.4.11%2bcommit.68ef5810...

Sepolia 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

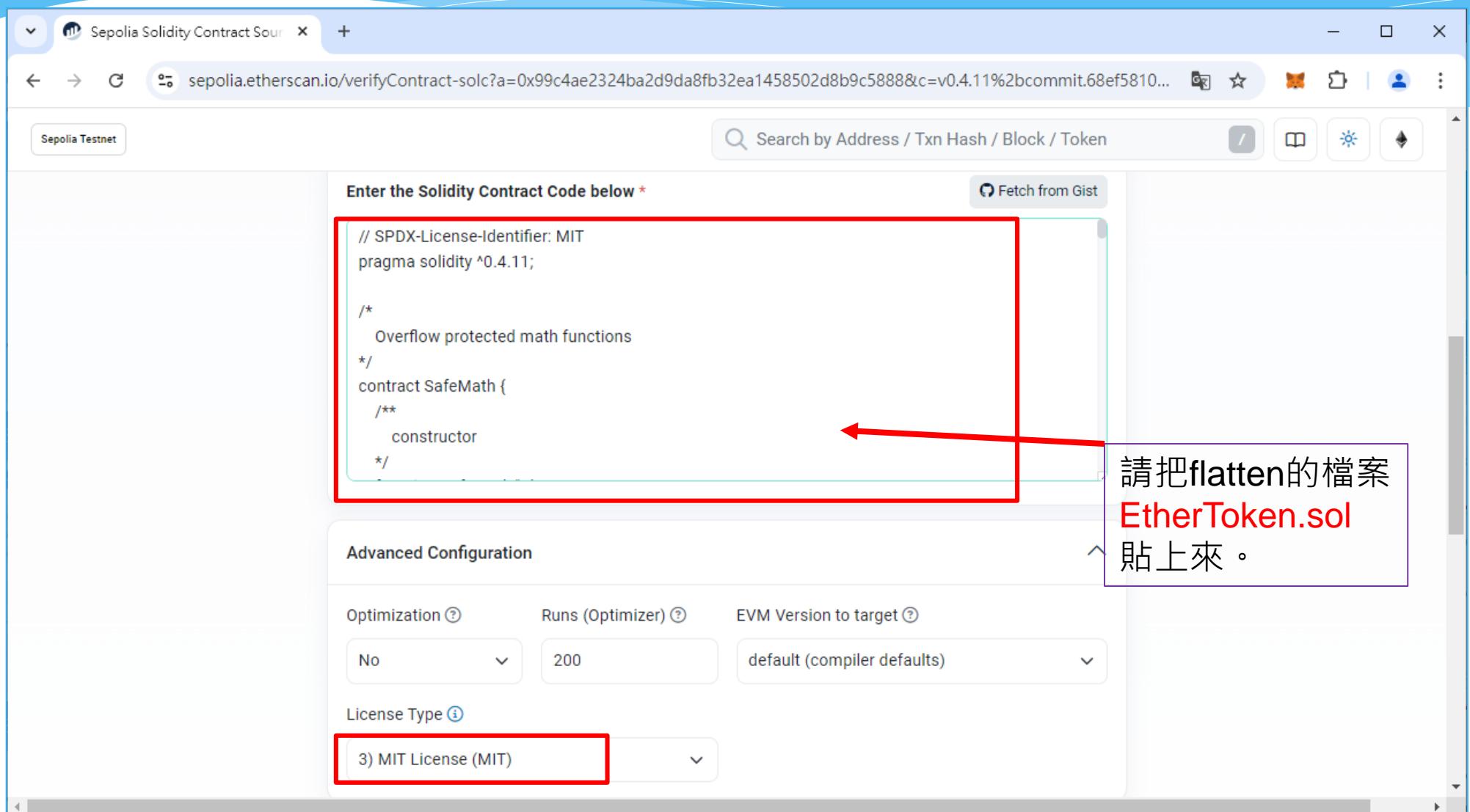
1. If the contract compiles correctly at [REMXI](#), it should also compile correctly here.
2. We have limited support for verifying contracts created by another contract and there is a timeout of up to 45 seconds for each contract compiled.
3. For programmatic contract verification, check out the [Contract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

Contract Address: `0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888`

Compiler Type: SINGLE FILE / CONCATENATED METHOD

Compiler Version: v0.4.11+commit.68ef5810

View & Publish Contract Source Code



Sepolia Solidity Contract Sourc... +

sepolia.etherscan.io/verifyContract-solc?a=0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888&c=v0.4.11%2bcommit.68ef5810...

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

Enter the Solidity Contract Code below *

// SPDX-License-Identifier: MIT
pragma solidity ^0.4.11;

/*
 * Overflow protected math functions
 */
contract SafeMath {
 /**
 * constructor
 */

Fetch from Gist

Advanced Configuration

Optimization ⓘ Runs (Optimizer) ⓘ EVM Version to target ⓘ

No 200 default (compiler defaults)

License Type ⓘ

3) MIT License (MIT)

請把flatten的檔案
EtherToken.sol
貼上來。

View & Publish Contract Source Code

The screenshot shows a web browser window with the title "Sepolia Solidity Contract Sour" and the URL "sepolia.etherscan.io/verifyContract-solc?a=0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888&c=v0.4.11%2bcommit.68ef5810...". The page displays a form for publishing a contract source code. At the top left, there is a "Sepolia Testnet" button. A search bar at the top right contains the placeholder "Search by Address / Txn Hash / Block / Token". Below the search bar, there are several icons: a magnifying glass, a clipboard, a lightbulb, and a gear.

Constructor Arguments ABI-encoded
For contracts that were created with constructor parameters

For additional information on Constructor Arguments, see our KB entry

Contract Library Address (for contracts that use libraries, supports up to 10 libraries)

A success message box is displayed, showing a green checkmark icon and the text "成功!". To the right of the message is the Cloudflare logo. Below the message box are two buttons: "Verify and Publish" (highlighted with a red border) and "Reset".

View & Publish Contract Source Code

The screenshot shows a web browser window for the Sepolia Testnet on etherscan.io. The URL in the address bar is `sepolia.etherscan.io/verifyContract-solc?a=0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888&c=v0.4.11%2bcommit.68ef5810...`. The page title is "Verify & Publish Contract Source Code".
The main content area includes:

- A heading: "Verify & Publish Contract Source Code".
- A sub-heading: "Source code verification provides transparency for users interacting with smart contracts."
- Text: "By uploading the source code, Etherscan will match the compiled code with that on the blockchain. [Read more.](#)"
- A note: "A simple and structured interface for verifying smart contracts that fit in a single file."
- Two numbered steps:
 - 1 Enter Contract Details
 - 2 Verify & Publish
- A success message box with a red border containing:
 - A green checkmark icon.
 - Text: "Successfully generated Bytecode and ABI for Contract Address [0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888]"
- A note: "Learn how to verify your contract on multiple blockchains with a single API key [here](#)".
- A section titled "Code Reader" with a "Prompt" field.

A red arrow points from the text "click" to the "Verify & Publish" button.

View Contract Source Code

The screenshot shows the Etherscan interface for the EtherToken contract. The URL in the address bar is highlighted with a red box: sepolia.etherscan.io/address/0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888#code. The page displays the following information:

- Contract Source Code Verified (Exact Match)** (highlighted with a red box)
- Contract Name:** EtherToken
- Compiler Version:** v0.4.11+commit.68ef5810
- Optimization Enabled:** No with 200 runs
- Other Settings:** default evmVersion, MIT license

The **Contract Source Code (Solidity)** section shows the following Solidity code (also highlighted with a red box):

```
1 /**
2  *Submitted for verification at Etherscan.io on 2024-11-01
3 */
4
5 // SPDX-License-Identifier: MIT
6 pragma solidity ^0.4.11;
7
8 /*
9  *      Overflow protected math functions
10 */
11 contract SafeMath {
12     /**
13      constructor
14     */
15     function SafeMath() {
16     }
17
18     /**
19      @dev returns the sum of _x and _y, asserts if the calculation overflows
20  }
```

View & Publish Contract Source Code

The screenshot shows the Etherscan interface for a contract at address 0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888. The browser title bar reads "EtherToken | Address 0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888". The URL in the address bar is "sepolia.etherscan.io/address/0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888#code". The page header includes a "Sepolia Testnet" button, a search bar, and various navigation icons.

Contract Details: The contract address is 0x99c4Ae2324Ba2D9dA8FB32EA1458502d8B9C5888. The "Source Code" tab is selected. The "Overview" section shows an ETH BALANCE of 0 ETH. The "More Info" section displays the CONTRACT CREATOR as 0x4CE135aB...64E06ae71, created at txn 0xa56409f3ba..., and the TOKEN TRACKER as Ether Token (ETH). The "Multichain Info" section indicates N/A.

Contract Interaction Buttons: Below the tabs, there are buttons for "Transactions", "Token Transfers (ERC-20)", "Contract" (which is highlighted with a red box), and "Events". Further down, there are buttons for "Code" (highlighted with a red box), "Read Contract", and "Write Contract".

Verification Status: A message at the bottom left states "Contract Source Code Verified (Exact Match)".

Contract Name and Optimization: The contract name is EtherToken, and optimization is enabled with 200 runs.

Page Footer: The footer contains the URL "https://sepolia.etherscan.io/address/0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888#code" and the page number "270".

Connect to Web3

The screenshot shows the Etherscan.io interface for the address `0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888`. A modal window titled "sepolia.etherscan.io 顯示" (Display) contains a note about the beta feature and two buttons: "確定" (Confirm) and "取消" (Cancel). A red arrow points from the "Connect to Web3" button in the bottom-left corner of the main content area to the "確定" button in the modal. The "Read Contract" button is also highlighted with a red box.

EtherToken | Address 0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888#readContract

sepolia.etherscan.io/address/0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888#readContract

Sepolia Testnet

Source Code

Overview

ETH BALANCE
0 ETH

Transactions Token Transfers (ERC-20) Contract Events

Code Read Contract Write Contract

● Connect to Web3

TOKEN TRACKER
Ether Token (ETH)

sepolia.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.

確定 取消

Block / Token

Multichain Info
I/A

[Expand all] [Reset]

Connect a Wallet-MetaMask

The screenshot shows a web browser window displaying the Etherscan interface for the Sepolia Testnet. The URL in the address bar is sepolia.etherscan.io/address/0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888#readContract. 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

Connected Web3 using account

The screenshot shows a web browser window with the title "EtherToken | Address 0x99c4a" and the URL "sepolia.etherscan.io/address/0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888#readContract". The browser interface includes a search bar, a tab bar with "Transactions", "Token Transfers (ERC-20)", "Contract", and "Events", and a sidebar with "Sepolia Testnet". The main content area displays the "Contract" tab with sections for "Code", "Read Contract", and "Write Contract". A red box highlights the "Connected - Web3 [0x4ce1...ae71]" status message. Below it is a list of function names: 1. name, 2. totalSupply, 3. decimals, 4. standard, 5. balanceOf, 6. owner, 7. symbol, 8. newOwner, and 9. allowance. To the right of the browser is a MetaMask extension window titled "Salu 1". It shows a balance of "1.5581 SepoliaETH" and five buttons: "Buy & Sell", "发送" (Send), "Swap", "Bridge", and "Portfolio". The "Tokens" tab is selected, showing "SepoliaETH" with a balance of "1.5581 SepoliaETH". Other tabs include "NFTs" and "交易紀錄" (Transactions). Buttons for "Import Tokens", "Refresh list", and "MetaMask 支援" (Support) are also present.

Read Contract - balanceOf

The screenshot shows the Etherscan interface for the Sepolia Testnet. The URL in the browser is <https://sepolia.etherscan.io/address/0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888#readContract>. The tab title is "EtherToken | Address 0x99c4...". The main navigation tabs are "Code", "Read Contract" (which is selected), and "Write Contract". A message at the top says "Connected - Web3 [0x4ce1...ae71]". Below this, under "Read Contract Information", there is a list of fields: 1. name, 2. totalSupply, 3. decimals, 4. standard, and 5. balanceOf. The "balanceOf" field has a sub-section labeled "<input> (address)" containing the value "0x4CE135aB2eB8e482D16B8011ba9415D64E06ae71". This input field is highlighted with a red rectangle. Below it is a "Query" button, also highlighted with a red rectangle. Under the query result, there is a section labeled "[balanceOf(address) method Response]" which shows "» uint256 : 0". This response section is also highlighted with a red rectangle.

Read Contract-decimals

EtherToken | Address 0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888#readContract

Sepolia Testnet

Search by Address / Txn Hash / Block / Token

Overview

ETH BALANCE
0 ETH

More Info

CONTRACT CREATOR
0x4CE135aB...64E06ae71 at txn 0xa56409f3ba...

TOKEN TRACKER
Ether Token (ETH)

Multichain Info
N/A

Transactions Token Transfers (ERC-20) **Contract** Events

Code Read Contract Write Contract

Connect to Web3

Read Contract Information

[Expand all] [Reset]

1. name

2. totalSupply

3. decimals

18 uint8

Read Contract-name

The screenshot shows the EtherToken contract details page on Etherscan. The URL in the browser is <https://sepolia.etherscan.io/address/0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888#readContract>. The page is set to the Sepolia Testnet. The main navigation tabs are Transactions, Token Transfers (ERC-20), Contract (selected), and Events. Under the Contract tab, there are three sub-tabs: Code, Read Contract (selected), and Write Contract. A red box highlights the 'Read Contract' tab. Below it, a 'Connect to Web3' button is shown. The 'Read Contract Information' section contains the following fields:

- 1. name: Ether Token string (highlighted with a red box)
- 2. totalSupply
- 3. decimals: 18 uint8

At the bottom right of the page, there are 'Expand all' and 'Reset' buttons.

Read Contract-symbol

EtherToken | Address 0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888#readContract

Sepolia Testnet

Search by Address / Txn Hash / Block / Token

1. name
Ether Token string

2. totalSupply

3. decimals
18 uint8

4. standard

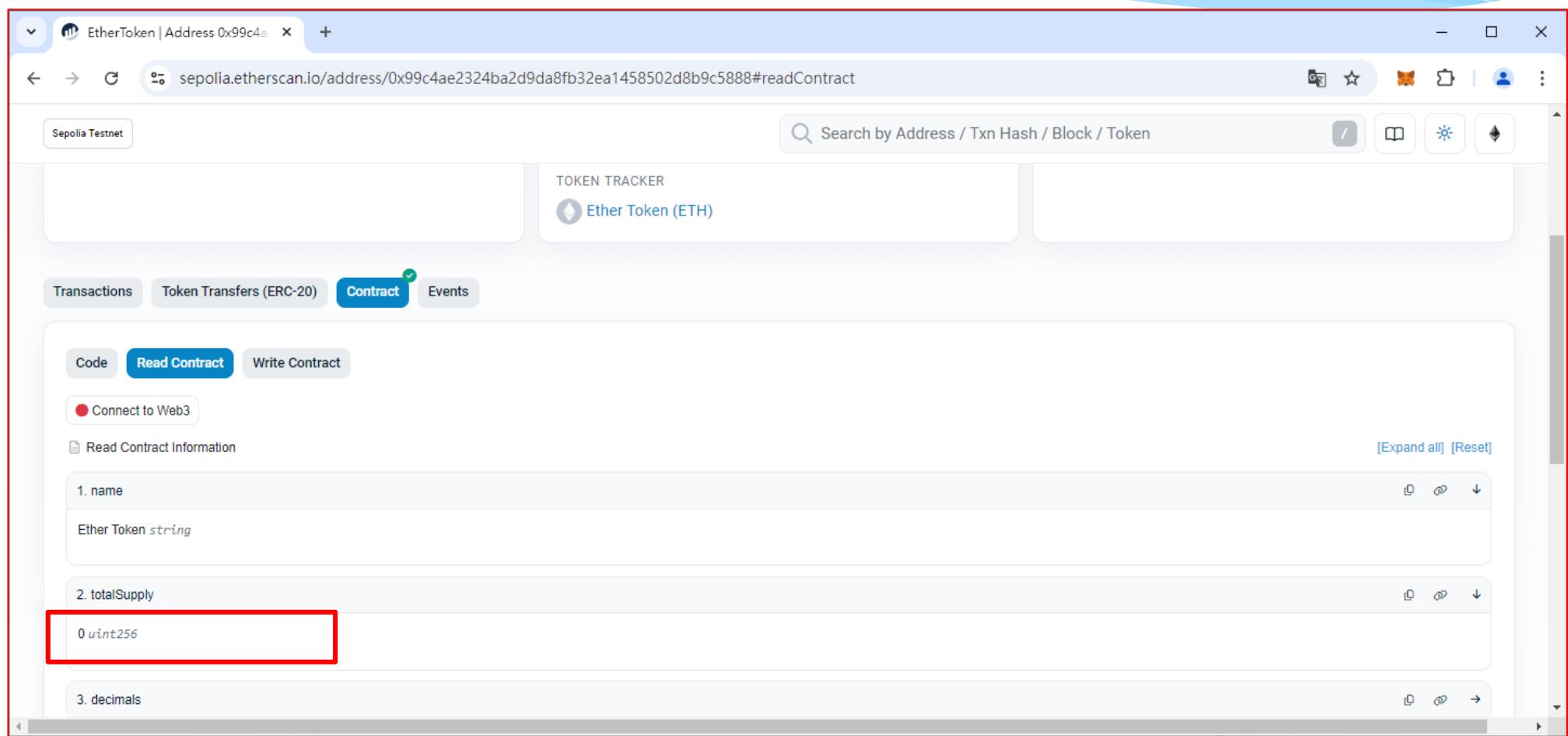
5. balanceOf

6. owner

7. symbol
ETH string

8. newOwner

Read Contract-totalSupply

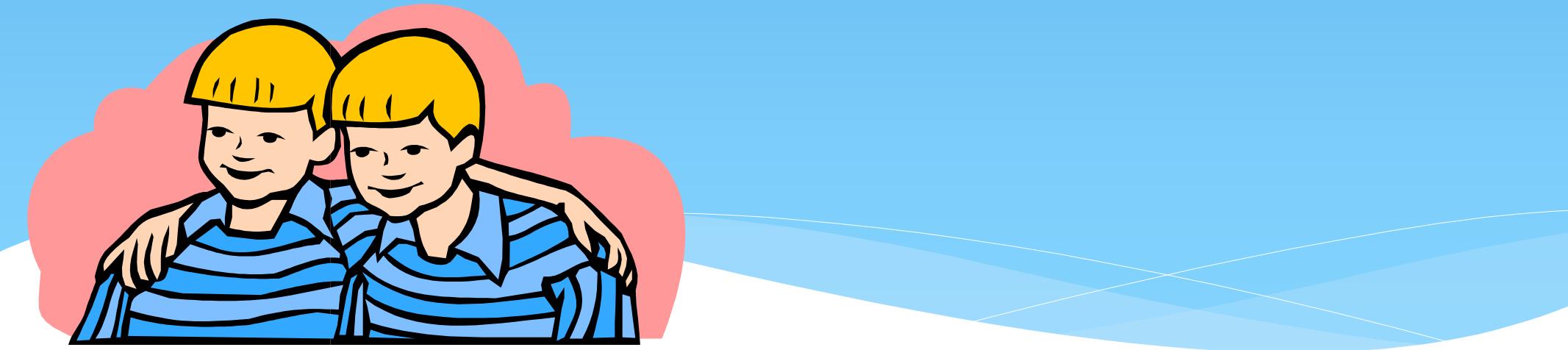


The screenshot shows the Etherscan interface for the Sepolia Testnet. The address being analyzed is EtherToken | Address 0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888. The "Contract" tab is selected. Under the "Read Contract" section, the "totalSupply" field is highlighted with a red box. The value shown is 0 uint256.

sepolia.etherscan.io/address/0x99c4ae2324ba2d9da8fb32ea1458502d8b9c5888#readContract



Thank you !



References

References

■ MetaMask

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

■ Remix

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

■ Openzeppelin

<https://docs.openzeppelin.com/>

<https://github.com/OpenZeppelin/openzeppelin-contracts>

References

■ Ethereum

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

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

<https://ethereum.org/en>

<https://geth.ethereum.org/docs/fundamentals/private-network>

■ Solidity Documentation

<https://solidity.readthedocs.io>

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

■ Smart Contract

<https://www.quicknode.com/guides/tags/ethereum>