

Human-friendly Contract Interactions with



sourcify Verification

Kaan Uzdoğan
Ethereum Foundation

Devcon VI
11.10.2022 - Bogotá



DEVCON **BOGOTA**
COLOMBIA

Just a normal day in web3...

DATA

HEX DATA: 196 BYTES

[illegible]

New address detected! Click here to add to your address book.

<https://opensea.io>

0x7f2...38E5 : ATOMIC MATCH_ ⓘ


 0.2
 361,23 €

DETAILS	DATA	HEX
---------	------	-----

FUNCTION TYPE: Atomic Match_ (Address[1
 Uint256[18], Uint8[8], Bytes, Bytes, Bytes, B
 Bytes, Bytes, Uint8[2], Bytes32[5])

► **addrs:**

- ▶ uints:

► feeMethodsSidesKindsHowToCalls:

calldataBuy:

[illegible]

New address detected! [Click here](#) to add to your address book.

<https://gov.element.fi>

0x5ae...b966 : CONTRACT INTERACTION ⓘ



0,00 €

DETAILS

DATA

HEX

FUNCTION TYPE: Contract Interaction

proof:

0x1dd85c68a812963b7889151911054ebc98101e43564c2f5b88a923a30eabdfbf1e9ecfa0f5460d2e1779a8c1c0c9482c6328c5358184cc780a2f8d3e2f7e4bf21dae151fbf20b577b215301ac87c89b7fb32541a47564b27f2f18a8c222e70752416867f33c8a6738de9627dbd405e9bfc039d78b9120c8f8f605bea7d0e66b24dbaa858eced9f0d9f4c22725b44e819bba563aaea046024f2342e4355d416c2588483fa3ab875e6facdee244e8cbdbcd2bab2a1b19cea4311fedea4b82944925e0463ccb3b70f9bb4f20c030a6b9398ecad6d8af8356dcfc20a5adee7764552d37e2d8f720bdf6d4c7faa3eb97fa34fa20bb462b488491c408a940ac5a3c5427fde5edd85cfaf754de7e609bd1b1289b3dea92f0088692b740ac9237822ed17f41039943f68695cd77577c760f9246f03595741be6be0bfd8ca9ce40fb15100b17d7cfe986a2c0c7c2a2eb66f86ec5a01c85393elf940d03080334276a6532d9fd4fe70017490a8455cddabaa93c6d48999e328db796a671f43ca4e1cd51c129ada7abc057f56feb39ecf36935b087e35063da5138296ca0

► VS:

▼ rssMetadata:

0x51de7d3b489197787214c5466653447352ebf
36915fe9cd80a85f8f444c3387a
0x6c4e18807799c9b84326502aafc2c166bce1
3c99be74a1a15c72a4a4eb26fc3b
0x51de7d3b489197787214c5466653447352ebf
36915fe9cd80a85f8f444c3387a

DATA

HEX

0x8

000

000

ca7

000

c49

120

000

746

000

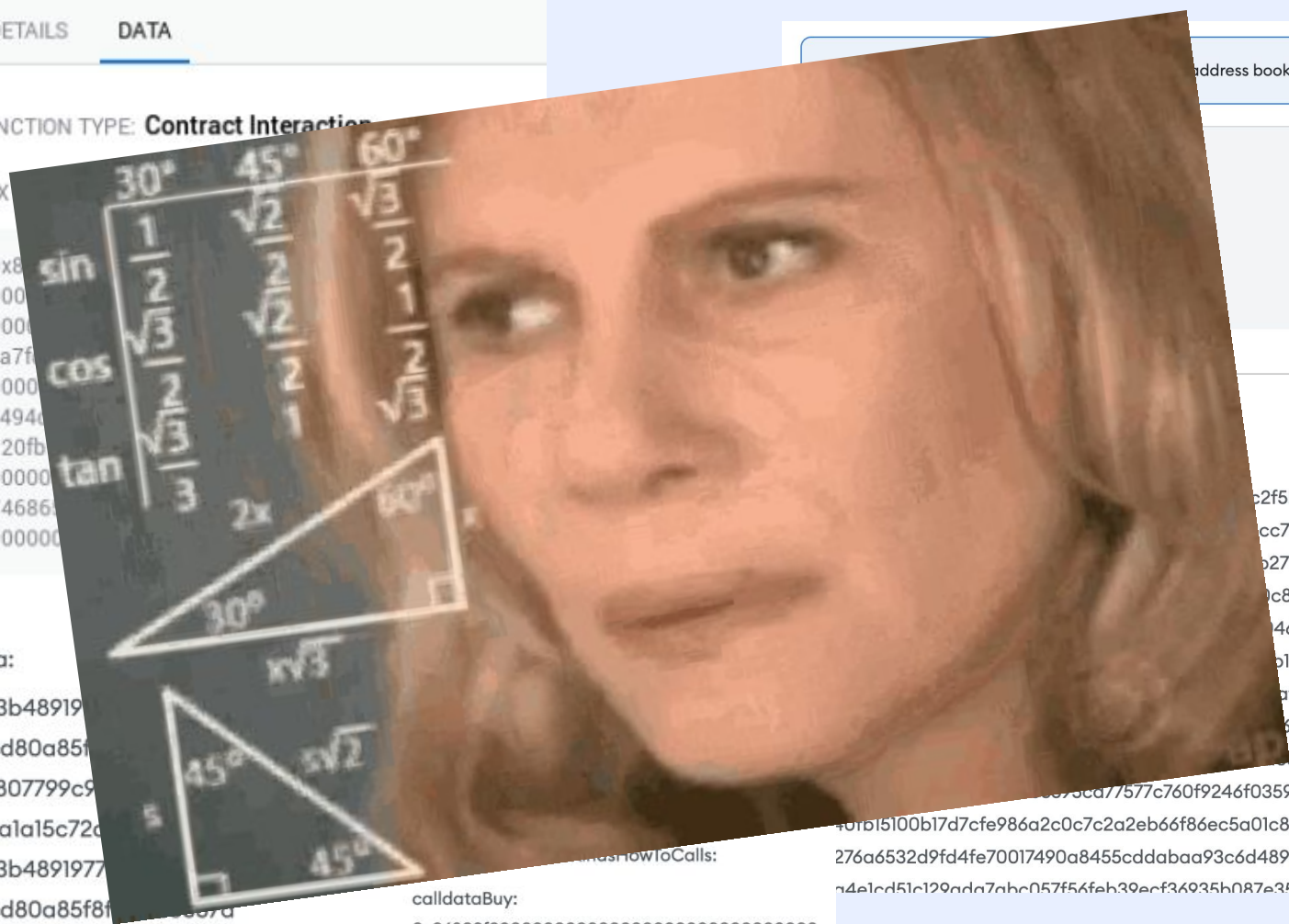
staticExt

▼ rssMetadata:

36915fe9cd80a85f

ON TYPE: Contract Interactions

	30°	45°	60°
sin	$\frac{1}{2}$	$\frac{\sqrt{2}}{2}$	$\frac{\sqrt{3}}{2}$
cos	$\frac{\sqrt{3}}{2}$	$\frac{\sqrt{2}}{2}$	$\frac{1}{2}$
tan	$\frac{\sqrt{3}}{3}$	1	$\sqrt{3}$



address book.

c2f5b88a923a30eabdf

cc780a2f8d3e2f7e4bf

027f2f18a8c222e70752

0c8f8f605bea7d0e66b

46024f2342e4355d41

b19ceq4311fede4b82

af8356dcfc20a5ade

62b488491c408a94

92f0088692b740

cd75cd77577c760f9246f03595741be6be0bfd8ca

40fb15100b17d7cfe986a2c0c7c2a2eb66f86ec5a01c85393e1f940d03080

276a6532d9fd4fe70017490a8455cddabaa93c6d48999e328db796a671f

44e1cd51c129ada7abc057f56feb39ecf36935b087e35063dda5138296ce0

calldataBuy:

0-0480950000000000000000

DATA

FUNCTION

HEX

0x8
000
0000
ca7f
0000
c494
120fb
00000
7468
00000

```
00000000
00000000
00000000
00000000
staticExt
staticExt
```

► VS:

▼ rssMetadata:

0x51de7d3b48
36915fe9cd80
0x6c4e188077
3c99be74a1a1
0x51de7d3b48
36915fe9cd80

88a923a30eabdf
0a2f8d3e2f7e4bf
2f18a8c222e70752
8f605bea7d0e66b
024f2342e4355d41
cea4311fede4b82
3356dcfc20a5ade
b488491c408a94
a92f0088692b740
6741be6be0bfd8ca

707b15100b1/d/cte986a2c0c/c2a2eb66t86ec5a0ic85393ef1f940d03080
276a6532d9fd4fe70017490a8455cddabaa93c6d48999e328db796a671f
g4e1cd51c129ada7nbc057f56feb39ecf34935b087e35063da5138296ce0

calldataBuy:

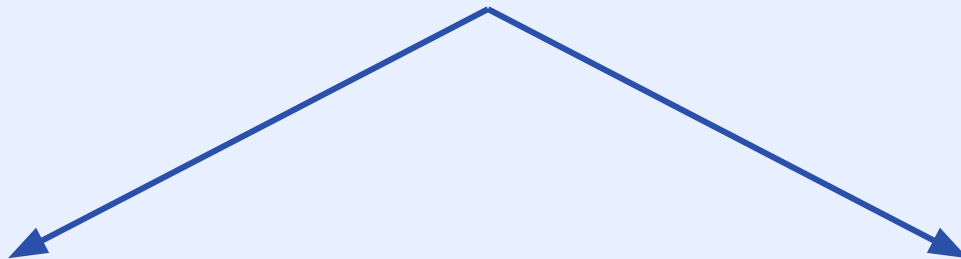
[illegible]



A typical web3 interaction nowadays is **still** a
YOLO-Signing nightmare

What can you do to achieve this...

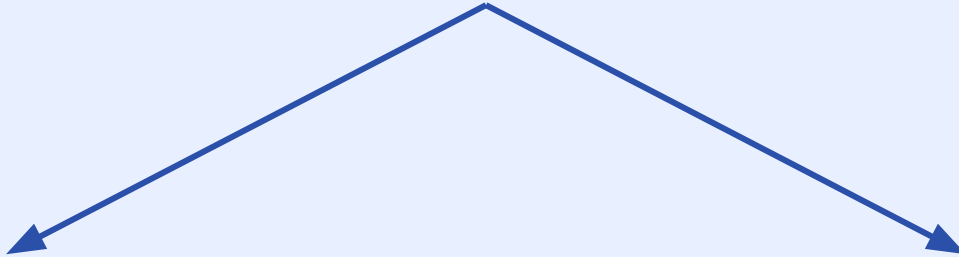
What can you do to achieve this...



...as a smart contract developer?

...as a wallet developer?

What can you do to achieve this...



...as a smart contract developer?

...as a wallet developer?

As a contract developer:

- 1) Use NatSpec documentation
- 2) Source code verification on Sourcify

As a contract developer:

- 1) **Use NatSpec documentation**
- 2) Source code verification on Sourcify

Natspec

<https://docs.soliditylang.org/en/develop/natspec-format.html>

“Ethereum Natural Language Specification Format (NatSpec)”

Natspec

```
contract Wallet {  
...  
    /// @dev Allows to swap/replace an owner from the Safe with another address.  
    ///      This can only be done via a Safe transaction.  
    /// @notice Replaces the owner `oldOwner` in the Safe with `newOwner`.  
    /// @param prevOwner Owner that pointed to the owner to be replaced in the linked list  
    /// @param oldOwner Owner address to be replaced.  
    /// @param newOwner New owner address.  
    function swapOwner(  
        address prevOwner,  
        address oldOwner,  
        address newOwner  
    ) public authorized {...
```

Natspec

```
contract Wallet {
```

```
... *
```

Developer Documentation

```
/// @dev Allows to swap/replace an owner from the Safe with another address.
```

```
///      This can only be done via a Safe transaction.
```

```
/// @notice Replaces the owner `oldOwner` in the Safe with `newOwner`.
```

```
/// @param prevOwner Owner that pointed to the owner to be replaced in the linked list
```

```
/// @param oldOwner Owner address to be replaced.
```

```
/// @param newOwner New owner address.
```

```
function swapOwner(
```

```
    address prevOwner,
```

```
    address oldOwner,
```

```
    address newOwner
```

```
) public authorized {...
```


Natspec

```
contract Wallet {  
    ...  
  
    /// @dev Allows to swap/replace an owner from the Safe with another address.  
    ///      This can only be done via a Safe transaction.  
    /// @notice Replaces the owner `oldOwner` in the Safe with `newOwner`.  
    /// @param prevOwner Owner that pointed to the owner to be replaced in the linked list  
    /// @param oldOwner Owner address to be replaced.  
    /// @param newOwner New owner address.  
  
    function swapOwner(  
        address prevOwner,  
        address oldOwner,  
        address newOwner  
    ) public authorized {...
```

User Documentation

Natspec

```
contract Wallet {  
    ...  
  
    /// @dev Allows to swap/replace an owner from the Safe with another address.  
    ///      This can only be done via a Safe transaction.  
    /// @notice Replaces the owner `oldOwner` in the Safe with `newOwner`.  
    /// @param prevOwner Owner that pointed to the owner to be replaced in the linked list  
    /// @param oldOwner Owner address to be replaced.  
    /// @param newOwner New owner address.
```

Parameters

```
function swapOwner(  
    address prevOwner,  
    address oldOwner,  
    address newOwner  
) public authorized {...
```

Natspec: Dynamic Expressions

```
/// @notice Replaces the owner `oldOwner` in the Safe with `newOwner`.
```

Natspec: Dynamic Expressions

@notice Replaces the owner ``oldOwner`` in the Safe with ``newOwner``

Natspec: Dynamic Expressions

@notice Replaces the owner ``oldOwner`` in the Safe with ``newOwner``

Becomes:

Replaces the owner `0xcC60F45e0507032036033b361d3a6457b9F0283D`

in the Safe with `0x83D0360050703233b361d3a6457b9F2cC60F45e0`



Where to find userdoc (@notice) and devdoc (@dev)?

In Solidity **Contract Metadata:**

<https://docs.soliditylang.org/en/latest/metadata.html>

Solidity Contract Metadata

Introduced in Solidity v0.4.7 (2016-12-15) but was not picked up by the community

Solidity Contract Metadata

Introduced in Solidity v0.4.7 (2016-12-15) but was not picked up by the community

JSON file generated by the Solidity compiler which contains... metadata:

Solidity Contract Metadata

Introduced in Solidity v0.4.7 (2016-12-15) but was not picked up by the community

JSON file generated by the Solidity compiler which contains... metadata:

- ABI
- Userdoc + devdoc
- Compilation info
- Source file info

Solidity Contract Metadata

Introduced in Solidity v0.4.7 (2016-12-15) but was not picked up by the community

JSON file generated by the Solidity compiler which contains... metadata:

- ABI
- Userdoc + devdoc
- Compilation info
- Source file info



How to interact with the contract?

Solidity Contract Metadata

Introduced in Solidity v0.4.7 (2016-12-15) but was not picked up by the community

JSON file generated by the Solidity compiler which contains... metadata:

- ABI
 - Userdoc + devdoc
 - Compilation info
 - Source file info
- } How to interact with the contract?**
- } How to reproduce a contract compilation?**

```
▼ {
  ▼ "compiler": {
    "version": "0.7.6+commit.7338295f"
  },
  "language": "Solidity",
  ▼ "output": {
    ► "abi": [...], // 49 items
    ► "devdoc": {...}, // 5 items
    ► "userdoc": {...} // 3 items
  },
  ▼ "settings": {
    ► "compilationTarget": {...}, // 1 item
    "evmVersion": "istanbul",
    "libraries": {},
    ► "metadata": {...}, // 2 items
    ▼ "optimizer": {
      "enabled": false,
      "runs": 200
    },
    "remappings": []
  },
  ▼ "sources": {
    ► "contracts/GnosisSafe.sol": {...}, // 3 items
    ► "contracts/base/Executor.sol": {...}, // 3 items
    ► "contracts/base/FallbackManager.sol": {...}, // 3 items
    ► "contracts/base/GuardManager.sol": {...}, // 3 items
    ► "contracts/base/ModuleManager.sol": {...}, // 3 items
```

```
▼ {
  ▼ "compiler": {
    "version": "0.7.6+commit.7338295f"
  },
  "language": "Solidity",
  ▼ "output": {
    ▶ "abi": [...], // 49 items
    ▶ "devdoc": {...} // 5 items
    ▶ "userdoc": {...} // 3 items
  },
  ▼ "settings": {
    ▶ "compilationTarget": {...}, // 1 item
    "evmVersion": "istanbul",
    "libraries": {},
    ▶ "metadata": {...}, // 2 items
    ▼ "optimizer": {
      "enabled": false,
      "runs": 200
    },
    "remappings": []
  },
  ▼ "sources": {
    ▶ "contracts/GnosisSafe.sol": {...}, // 3 items
    ▶ "contracts/base/Executor.sol": {...}, // 3 items
    ▶ "contracts/base/FallbackManager.sol": {...}, // 3 items
    ▶ "contracts/base/GuardManager.sol": {...}, // 3 items
    ▶ "contracts/base/ModuleManager.sol": {...}, // 3 items
```



```
▼ "userdoc": {  
  "kind": "user",  
  ▼ "methods": {  
    ▼ "addOwnerWithThreshold(address,uint256)": {  
      "notice": "Adds the owner `owner` to the Safe and updates the threshold to `_threshold`."  
    },  
    ▼ "changeThreshold(uint256)": {  
      "notice": "Changes the threshold of the Safe to `_threshold`."  
    },  
    ▼ "disableModule(address,address)": {  
      "notice": "Disables the module `module` for the Safe."  
    },  
    ▼ "enableModule(address)": {  
      "notice": "Enables the module `module` for the Safe."  
    },  
    ▼ "removeOwner(address,address,uint256)": {  
      "notice": "Removes the owner `owner` from the Safe and updates the threshold to `_threshold`."  
    },  
    ▼ "requiredTxGas(address,uint256,bytes,uint8)": {  
      "notice": "Deprecated in favor of common/StorageAccessible.sol and will be removed in next version."  
    },  
    ▼ "swapOwner(address,address,address)": {  
      "notice": "Replaces the owner `oldOwner` in the Safe with `newOwner`."  
    }  
  },  
},
```

Solidity Contract Metadata

```
$ solc --metadata MyContract.sol
```

Solidity Contract Metadata

```
$ truffle compile
$ cd build/contracts
$ cat MyContract.json
{
  "contractName": "MyToken",
  "abi": [...],
  "metadata": "{\n  \"compiler\": {\n    \"version\": \"0.8.4+commit.c7e474f2\"\n  },\n  \"language\": \"\n    ....\n  \",\n  \"version\": 1\n  }",
  "bytecode": "...",
  ...
}
```

Solidity Contract Metadata

```
$ hardhat compile
$ cd artifacts/build-info
$ cat 901568e56d422b1e1e3f64004cb4dd6e.json
{
  "id": "901568e56d422b1e1e3f64004cb4dd6e",
  "_format": "hh-sol-build-info-1",
  "solcVersion": "0.8.6",
  "solcLongVersion": "0.8.6+commit.11564f7e",
  "input": {
    ...
  },
  "output": {
    "contracts": {
      "MyContract.sol": {
        "MyContract": {
          "abi": [
            ],
          "evm": {
            },
          "metadata": "{\"compiler\":{\"version\":\"0.8.6+commit.11564f7e\"},\"language\":\"Solidity\", \"output\"
        }
      }
    }
  }
}
```

Solidity Contract Metadata

Contract Bytecode

```
919050565b600082610b9a57634e487b7160e01b81526012600452602481fd5b500490565b80825b6001808611610bb157
50610bdc565b818704821115610bc357610bc3610cd5565b80861615610bd057918102915b9490941c938002610ba2565b
94509492505050565b600061046860001960ff851684600082610c0157506001610468565b81610c0e5750600061046856
5b8160018114610c245760028114610c2e57610c5b565b6001915050610468565b60ff841115610c3f57610c3f610cd556
5b6001841b915084821115610c5557610c55610cd5565b50610468565b5060208310610133831016604e8410600b841016
1715610c8e575081810a83811115610c8957610c89610cd5565b610468565b610c9b8484846001610b9f565b8086048211
15610cad57610cad610cd5565b02949350505050565b6000816000190483118215151615610cd057610cd0610cd5565b50
0290565b634e487b7160e01b600052601160045260246000fdfea264697066735822122078b530288f1cffe879bb7d9062
e904deb3aa9b9c8d27ea4ecafa987583c18a6e64736f6c63430008010033
```

Solidity Contract Metadata

- The compiler takes the **IPFS hash** of the metadata file and appends to the bytecode alongside the compiler version.

ipfs://QmWTqspM5B1quNvdhXbS6TbXzyLZ5cUGHnTV8ZWJPqrQqj

Contract Bytecode

919050565b600082610b9a57634e487b7160e01b81526012600452602481fd5b500490565b80825b6001808611610bb157
50610bdc565b818704821115610bc357610bc3610cd5565b80861615610bd057918102915b9490941c938002610ba2565b
94509492505050565b600061046860001960ff851684600082610c0157500001610468565b81610c0e5750600061046856
5b8160018114610c245760028114610c2e57610c5b565b6001915050610468565b60ff841115610c3f57610c3f610cd556
5b6001841b915084821115610c5557610c55610cd5565b50610468565b5060208310610133831016604e8410600b841016
1715610c8e575081810a83811115610c8957610c89610cd5565b610468565b610c9b8484846001610b9f565b8086048211
15610cad57610cad610cd5565b029493505050565b6000816000190483118215151015610cd057610cd0610cd5565b50
0290565b634e487b7160e01b600052601160045260246000fdfeaa264697066735822122078b530288f1cffe879bb7d9062
e904deb3aa9b9c8d27ea4ecafa987583c18a6e64736f6c63430008010033

Contract 0x5ed4a410A612F2fe625a8F3cB4d70f197fF8C8be on Ethereum Mainnet

[View on Sourcify](#)[View on Etherscan](#)

Contract Bytecode

```
919050565b600082610b9a57634e487b7160e01b81526012600452602481fd5b500490565b80825b6001808611610bb157
50610bdc565b818704821115610bc357610bc3610cd5565b80861615610bd057918102915b9490941c938002610ba2565b
94509492505050565b600061046860001960ff851684600082610c0157506001610468565b81610c0e5750600061046856
5b81600181114610c2457600281114610c2e57610c5b565b6001915050610468565b60ff841115610c3f57610c3f610cd556
5b6001841b915084821115610c5557610c55610cd5565b50610468565b5060208310610133831016604e8410600b841016
1715610c8e575081810a83811115610c8957610c89610cd5565b610468565b610c9b8484846001610b9f565b8086048211
15610cad57610cad610cd5565b029493505050565b6000816000190483118215151615610cd057610cd0610cd5565b50
0290565b634e487b7160e01b600052601160045260246000fdfea264697066735822122078b530288f1cffe879bb7d9062
e904deb3aa9b9c8d27ea4ecafa987583c18a6e64736f6c63430008010033
```

CBOR decoding

CBOR length: 51 Bytes [See on CBOR Playground](#)

```
{
  "ipfs": "0x122078b530288f1cffe879bb7d9062e904deb3aa9b9c8d27ea4ecafa987583c18a6e",
  "solc": "0x000801"
}
```

Solidity compiler version (decoded)

0.8.1

Metadata Hash (decoded)

ipfs://QmWTqspM5B1quNvdhXbS6TbXzyLZ5cUGHnTV8ZWJPqrQqj

Try it out in **Playground**: playground.sourcify.dev



Solidity metadata.json playground

● Connected

#1 Ethereum Mainnet



Network list from [ethereum-lists/chains](https://github.com/ethereum-lists/chains)

Enter Contract Address or ENS

0x34a...456d

Decode

Click to decode some example contracts:

SynthetixAMM
on Sourcify

Uniswap
no metadata hash

ENS

XMON

ShibaSwap

PODK
metadata on ipfs

CollateralShort
Optimism Testnet

SushiSwap LP
Polygon

BUSD
BinanceSC

Wonderland DAO
Avalanche

some-old-contract
without appended CBOR

or paste contract bytecode

0x608060405234801561001057600080fd5b5061012f8061002060...

Decode

Try it out in **Playground:** playground.sourcify.dev

As a contract developer:

- 1) Use NatSpec documentation
- 2) **Source code verification on Sourcify**

What is source code verification?



Contract 0x00000000219ab540356cBB839Cbe05303d7705Fa



Buy

Exchange

Earn

Gaming

For more information about the deposit contract and how to stake, please visit the [Eth2 Launchpad](#) or the [BeaconScan Explorer](#).



Contract Overview

Eth2 Deposit Contract

Balance: 7,906,082.000069000000000069 Ether

Value: \$28,229,693,571.91 (@ \$3,570.63/ETH)

Token: >\$736,822.02 >107

More Info



More

My Name Tag: Not Available, [login to update](#)Creator: [0xb20a608c624ca50039...](#) at txn [0xe75fb554e433e03763...](#)

Transactions

Internal Txns

Erc20 Token Txns

Erc721 Token Txns

Contract

Events

Analytics

Code

Read Contract

Write Contract

Contract Source Code Verified (Exact Match)

Contract Name: DepositContract

Compiler Version: v0.6.11+commit.5ef660b1

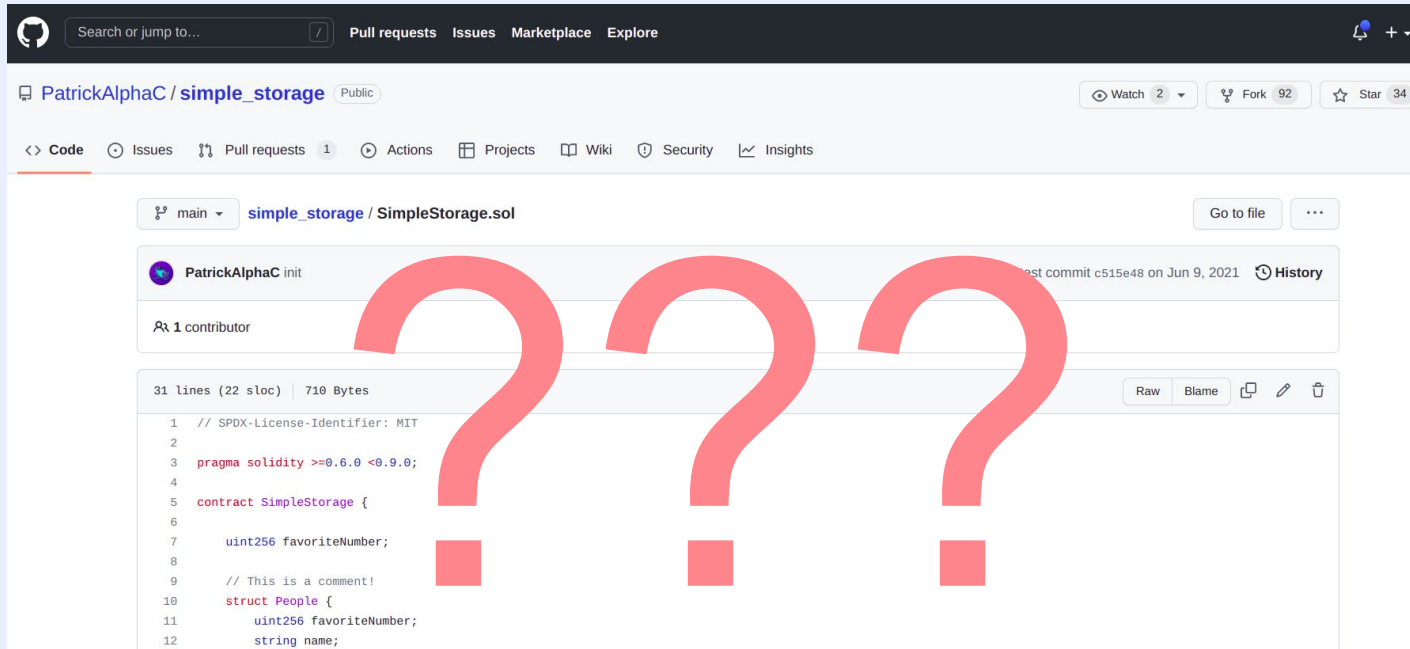
Optimization Enabled: Yes

Other Settings: default



Source Code Verification

- Smart contract code lives in **bytecode** on the blockchain
- Not human-readable, machine readable



The screenshot shows the GitHub interface for the repository 'PatrickAlphaC / simple_storage'. The file 'SimpleStorage.sol' is selected, showing 31 lines of Solidity code. The code includes a license header, a pragma statement for Solidity version, and a contract definition for 'SimpleStorage' with a 'favoriteNumber' variable and a 'People' struct. The page is overlaid with three large red question marks, suggesting a lack of readability or verification.

```
1 // SPDX-License-Identifier: MIT
2
3 pragma solidity >=0.6.0 <0.9.0;
4
5 contract SimpleStorage {
6
7     uint256 favoriteNumber;
8
9     // This is a comment!
10    struct People {
11        uint256 favoriteNumber;
12        string name;
```

Source Code Verification



MyContract.sol



Ownable.sol



ERC20.sol

...

Source Code Verification



MyContract.sol



Ownable.sol



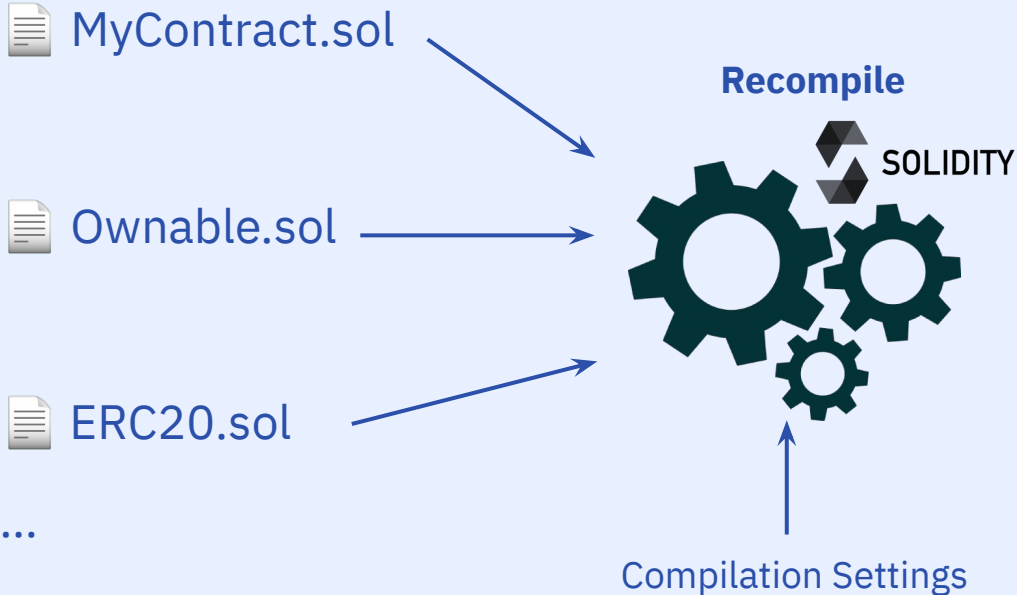
ERC20.sol

...

Compilation Settings

```
· version: "0.8.7+commit.e28d00a7",  
· optimizer: {  
  · enabled: true,  
  · runs: 200  
· },  
· ...
```

Source Code Verification



```
version: "0.8.7+commit.e28d00a7",
optimizer: {
  enabled: true,
  runs: 200
},
...

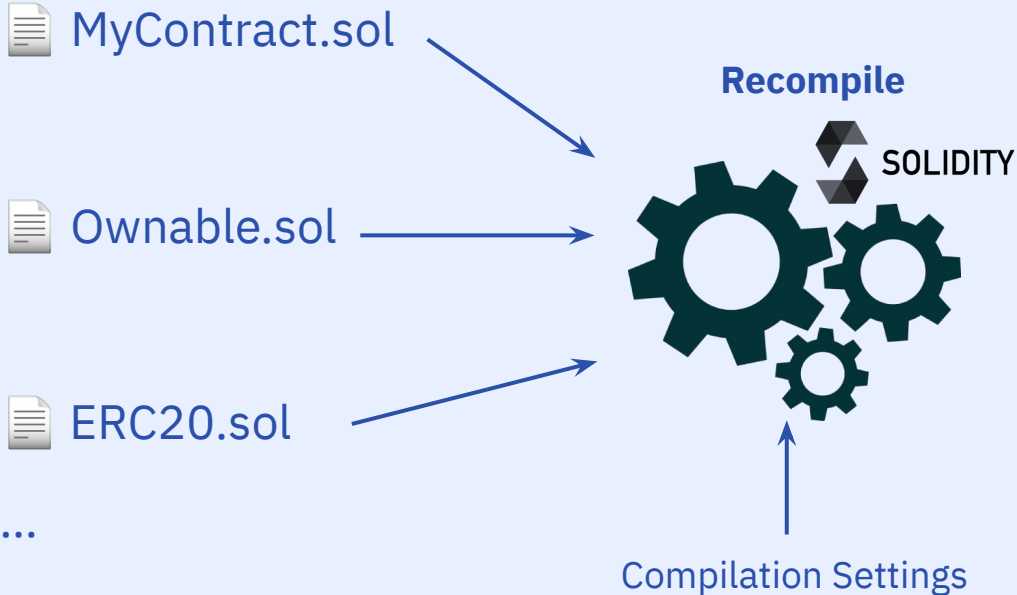
```


Solidity Contract Metadata

JSON file generated by the Solidity compiler which contains... metadata:

- ABI
 - Userdoc + devdoc
 - Compilation info
 - Source file info
- } **How to interact with the contract?**
- } **How to reproduce a contract compilation?**

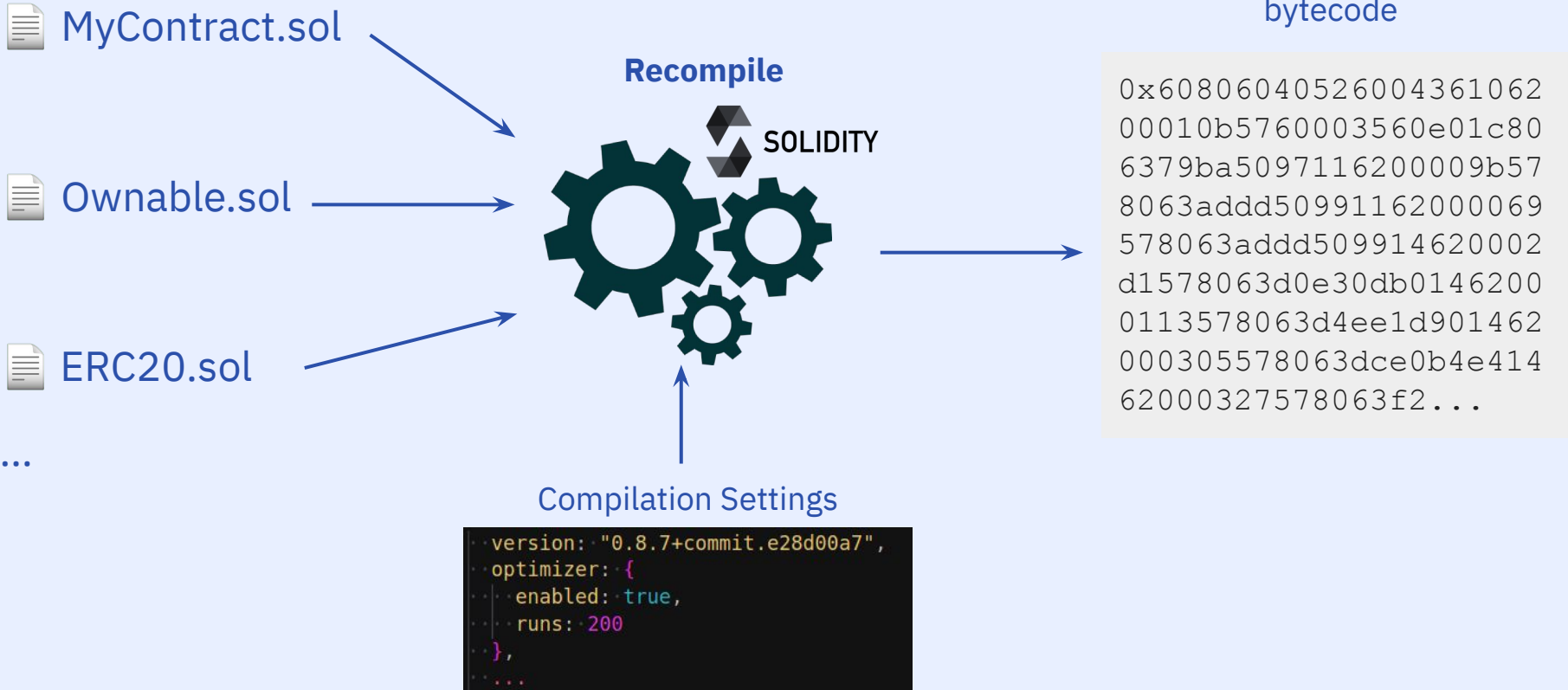
Source Code Verification



```
version: "0.8.7+commit.e28d00a7",
optimizer: {
  enabled: true,
  runs: 200
},
...

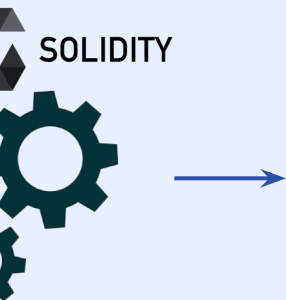
```

Source Code Verification



Source Code Verification

bytecode



```
0x60806040526004361062
00010b5760003560e01c80
6379ba5097116200009b57
8063addd50991162000069
578063addd509914620002
d1578063d0e30db0146200
0113578063d4ee1d901462
000305578063dce0b4e414
62000327578063f2...
```

Settings

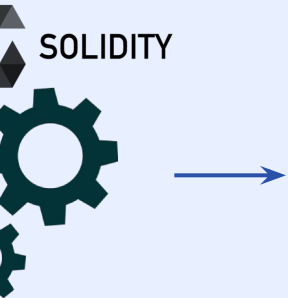
```
...it.e28d00a7",
```

Source Code Verification



bytecode

```
eth_getCode("0x011fDBf...64cc90BB26D0C")
```



```
0x60806040526004361062
00010b5760003560e01c80
6379ba5097116200009b57
8063addd50991162000069
578063addd509914620002
d1578063d0e30db0146200
0113578063d4ee1d901462
000305578063dce0b4e414
62000327578063f2...
```


Settings

```
...it.e28d00a7",
```

Source Code Verification



bytecode



0x60806040526004361062
00010b5760003560e01c80
6379ba5097116200009b57
8063add50991162000069
578063add509914620002
d1578063d0e30db0146200
0113578063d4ee1d901462
000305578063dce0b4e414
62000327578063f2...

eth_getCode("0x011fDBf...64cc90BB26D0C")



0x60806040526004361062
00010b5760003560e01c80
6379ba5097116200009b57
8063add50991162000069
578063add509914620002
d1578063d0e30db0146200
0113578063d4ee1d901462
000305578063dce0b4e414
62000327578063f2...

Settings

...it.e28d00a7",

Source Code Verification



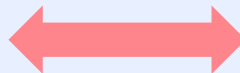
bytecode

`eth_getCode("0x011fDBf...64cc90BB26D0C")`



```
0x60806040526004361062
00010b5760003560e01c80
6379ba5097116200009b57
8063add50991162000069
578063add509914620002
d1578063d0e30db0146200
0113578063d4ee1d901462
000305578063dce0b4e414
62000327578063f2...
```

match?



```
0x60806040526004361062
00010b5760003560e01c80
6379ba5097116200009b57
8063add50991162000069
578063add509914620002
d1578063d0e30db0146200
0113578063d4ee1d901462
000305578063dce0b4e414
62000327578063f2...
```

Settings

`...it.e28d00a7",`

Source Code Verification

😞 **Partial match** = bytecodes match

😊 **Full / Perfect Match** = bytecode + metadata match

Contract Bytecode

```
919050565b600082610b9a57634e487b7160e01b81526012600452602481fd5b500490565b80825b6001808611610bb157
50610bdc565b818704821115610bc357610bc3610cd5565b80861615610bd057918102915b9490941c938002610ba2565b
94509492505050565b600061046860001960ff851684600082610c0157506001610468565b81610c0e5750600061046856
5b8160018114610c245760028114610c2e57610c5b565b6001915050610468565b60ff841115610c3f57610c3f610cd556
5b6001841b915084821115610c5557610c55610cd5565b50610468565b5060208310610133831016604e8410600b841016
1715610c8e575081810a83811115610c8957610c89610cd5565b610468565b610c9b8484846001610b9f565b8086048211
15610cad57610cad610cd5565b02949350505050565b6000816000190483118215151615610cd057610cd0610cd5565b50
0290565b634e487b7160e01b600052601160045260246000fdfea264697066735822122078b530288f1cffe879bb7d9062
e904deb3aa9b9c8d27ea4ecafa987583c18a6e64736f6c63430008010033
```


Source Code Verification

😞 **Partial match = bytecodes match**

😊 **Full / Perfect Match = bytecode + metadata match**

Contract Bytecode

919050565b600082610b9a57634e487b7160e01b81526012600452602481fd5b500490565b80825b6001808611610bb157
50610bdc565b818704821115610bc357610bc3610cd5565b80861615610bdd057918102915b9490941c938002610ba2565b
94509492505050565b600061046860001960ff851684600082610c0157506001610468565b81610c0e5750600061046856
5b8160018114610c245760028114610c2e57610b565b6001915050610468565b600ff841115610c3f57610c3f610cd556
5b6001841b915084821115610c5557610c55610cd5565b50610468565b5060208310610133831016604e8410600b841016
1715610c8e575081810a83811115610c8957610c89610cd5565b610468565b610c9b8484846001610b9f565b8086048211
15610cad57610cad610cd5565b02949350505050565b60008160001904831182151f615610cd057610cd0610cd5565b50
0290565b634e487b7160e01b600052601160045260246000fdfea264697066735822122078b530288f1cffe879bb7d9062
e904deb3aa9b9c8d27ea4ecafa987583c18a6e64736f6c63430008010033

Source Code Verification

😞 **Partial match = bytecodes match**

😊 **Full / Perfect Match = bytecode + metadata match (compilation fingerprint)**

Contract Bytecode

919050565b600082610b9a57634e487b7160e01b81526012600452602481fd5b500490565b80825b6001808611610bb157
50610bdc565b818704821115610bc357610bc3610cd5565b80861615610bdd057918102915b9490941c938002610ba2565b
94509492505050565b600061046860001960ff851684600082610c0157506001610468565b81610c0e5750600061046856
5b8160018114610c245760028114610c2e57610b565b6001915050610468565b60ff841115610c3f57610c3f610cd556
5b6001841b915084821115610c5557610c55610cd5565b50610468565b5060208310610133831016604e8410600b841016
1715610c8e575081810a83811115610c8957610c89610cd5565b610468565b610c9b8484846001610b9f565b8086048211
15610cad57610cad610cd5565b02949350505050565b60008160001904831182151f615610cd057610cd0610cd5565b50
0290565b634e487b7160e01b600052601160045260246000fdfea264697066735822122078b530288f1cffe879bb7d9062
e904deb3aa9b9c8d27ea4ecafa987583c18a6e64736f6c63430008010033

Source Code Verification

😞 **Partial match** = bytecodes match

😊 **Full / Perfect Match** = bytecode + metadata match

Full matches cryptographically guarantee the whole compilation including the Solidity files are exactly the same as when deployed - even comments, variable names etc.

How?

Full/Perfect Verification: How?

 MyContract.sol

 Ownable.sol

 ERC20.sol

...

Full/Perfect Verification: How?

 MyContract.sol $\xrightarrow{\text{Hashed}}$ 0xb6ee9d...

 Ownable.sol $\xrightarrow{\text{Hashed}}$ 0x41e281...

 ERC20.sol $\xrightarrow{\text{Hashed}}$ 0x9fd73f...

...

Full/Perfect Verification: How?

 MyContract.sol → Hashed 0xb6ee9d...

 Ownable.sol → Hashed 0x41e281...

 ERC20.sol → Hashed 0x9fd73f...

...

Metadata

```
{
  ▶ "compiler": { ... }, // 1 item
  "language": "Solidity",
  ▶ "output": { ... }, // 3 items
  ▶ "settings": { ... }, // 6 items
  ▼ "sources": {
    ▼ "contracts/MyContract.sol": {
      "keccak256": "0xb6ee9d528b336942dd70d3b41e2811...",
      "license": "GPL-3.0",
      ▼ "urls": [
        "bzz-raw://fe52c6e3c04ba5d83ede6cc1a43c45fa...",
        "dweb:/ipfs/QmawU3NM1WNWkBauRudYCiFvuFE1tTL..."
      ]
    },
    ▶ "contracts/Ownable.sol": { ... }, // 3 items
    ▶ "contracts/ERC20.sol": { ... } // 3 items
  }
}
```

Full/Perfect Verification: How?

MyContract.sol $\xrightarrow{\text{Hashed}}$ 0xb6ee9d...

Ownable.sol $\xrightarrow{\text{Hashed}}$ 0x41e281...

ERC20.sol $\xrightarrow{\text{Hashed}}$ 0x9fd73f...

...

Metadata

```
{
  ▶ "compiler": { ... }, // 1 item
  "language": "Solidity",
  ▶ "output": { ... }, // 3 items
  ▶ "settings": { ... }, // 6 items
  ▼ "sources": {
    ▼ "contracts/MyContract.sol": {
      "keccak256": "0xb6ee9d528b336942dd70d3b41e2811...",
      "license": "GPL-3.0",
      ▼ "urls": [
        "bzz-raw://fe52c6e3c04ba5d83ede6cc1a43c45fa...",
        "dweb:/ipfs/QmawU3NM1WNWkBauRudYCiFvuFE1tTL...",
      ]
    },
    ▶ "contracts/Ownable.sol": { ... }, // 3 items
    ▶ "contracts/ERC20.sol": { ... } // 3 items
  }
}
```

Full/Perfect Verification: How?

MyContract.sol $\xrightarrow{\text{Hashed}}$ 0xb6ee9d...

Ownable.sol $\xrightarrow{\text{Hashed}}$ 0x41e281...

ERC20.sol $\xrightarrow{\text{Hashed}}$ 0x9fd73f...

...

Metadata

```
{
  "compiler": { ... }, // 1 item
  "language": "Solidity",
  "output": { ... }, // 3 items
  "settings": { ... }, // 6 items
  "sources": {
    "contracts/MyContract.sol": {
      "keccak256": "0xb6ee9d528b336942dd70d3b41e2811...",
      "license": "GPL-3.0",
      "urls": [
        "bzz-raw://fe52c6e3c04ba5d83ede6cc1a43c45fa...",
        "dweb:/ipfs/QmawU3NM1WNWkBauRudYCiFvuFE1tTL..."
      ]
    },
    "contracts/Ownable.sol": { ... }, // 3 items
    "contracts/ERC20.sol": { ... } // 3 items
  }
}
```



```
6942dd70d3b41e2811be10a473776352009fd73f85604f5ed206" ,
```

```
5d83ede6cc1a43c45fa43caa435b207f64707afb17d3af1bcf1" ,  
kBauRudYCiFvuFE1tTLHB98akyBvb9UwWA"
```

```
// 3 items
```

```
3 items
```

Metadata

```
6942dd70d3b41e2811be10a473776352009fd73f85604f5ed206",
```

```
5d83ede6cc1a43c45fa43caa435b207f64707afb17d3af1bcf1",  
kBauRudYCiFvuFE1tTLHB98akyBvb9UWwA"
```

```
// 3 items
```

```
3 items
```

Metadata

IPFS hash

QmWTqspM5B1quNvdhXbS6TbXzyLZ5cUGHnTV8ZWJPqrQqj



QmWTqspM5B1quNvdhXbS6TbXzyLZ5cUGHnTV8ZWJPqrQqj



QmWTqspM5B1quNvdhXbS6TbXzyLZ5cUGHnTV8ZWJPqrQqj

encoded

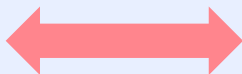


Contract Bytecode

```
919050565b600082610b9a57634e487b7160e01b81526012600452602481fd5b500490565b80825b6001808611610bb157
50610bdc565b818704821115610bc357610bc3610cd5565b80861615610bd057918102915b9490941c938002610ba2565b
94509492505050565b600061046860001960ff851684600082610c0157506001610468565b81610c0e5750600061046856
5b8160018114610c245760028114610c2e57610c5b565b6001915050610468565b60ff841115610c3f57610c3f610cd556
5b6001841b915084821115610c5557610c55610cd5565b50610468565b5060208310610133831016604e8410600b841016
1715610c8e575081810a83811115610c8957610c89610cd5565b610468565b610c9b8484846001610b9f565b8086048211
15610cad57610cad610cd5565b029493505050565b6000816000710483118215151615610cd057610cd0610cd5565b50
0290565b634e487b7160e01b600052601160045260246000fdfea264697066735822122078b530288f1cffe879bb7d9062
e904deb3aa9b9c8d27ea4ecafa987583c18a6e64736f6c63430008010033
```

```
500490565b80825b6001808611610bb157
918102915b9490941c938002610ba2565b
610468565b81610c0e5750600061046856
5b60ff841115610c3f57610c3f610cd556
8310610133831016604e8410600b841016
0c9b8484846001610b9f565b8086048211
15151615610cd057610cd0610cd5565b50
5822122078b530288f1cffe879bb7d9062
```

match?



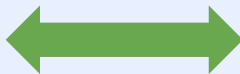
Contract Bytecode

```
919050565b600082610b9a57634e487b7160e01b8157
50610bdc565b818704821115610bc357610bc3610cd5
94509492505050565b600061046860001960ff851684
5b8160018114610c245760028114610c2e57610c5b50
5b6001841b915084821115610c5557610c55610cd550
1715610c8e575081810a83811115610c8957610c8961
15610cad57610cad610cd5565b02949350505050565b
0290565b634e487b7160e01b60005260116004526024
e904deb3aa9b9c8d27ea4ecafa987583c18a6e647361
```

```
500490565b80825b6001808611610bb157
918102915b9490941c938002610ba2565b
610468565b81610c0e5750600061046856
5b60ff841115610c3f57610c3f610cd556
8310610133831016604e8410600b841016
0c9b8484846001610b9f565b8086048211
15151615610cd057610cd0610cd5565b50
5822122078b530288f1cffe879bb7d9062
```



Full match



Contract Bytecode

```
919050565b600082610b9a57634e487b7160e01b8157
50610bdc565b818704821115610bc357610bc3610cd5
94509492505050565b600061046860001960ff851684
5b8160018114610c245760028114610c2e57610c5b50
5b6001841b915084821115610c5557610c55610cd550
1715610c8e575081810a83811115610c8957610c8961
15610cad57610cad610cd5565b02949350505050565b
0290565b634e487b7160e01b60005260116004526024
e904deb3aa9b9c8d27ea4ecafa987583c18a6e64736
```

Full/Perfect Verification: How?

MyContract.sol → Hashed → 0xb6ee9d...

Ownable.sol → Hashed → 0x41e281...

ERC20.sol → Hashed → 0x9fd73f...

...

Metadata

```
{
  "compiler": { ... }, // 1 item
  "language": "Solidity",
  "output": { ... }, // 3 items
  "settings": { ... }, // 6 items
  "sources": {
    "contracts/MyContract.sol": {
      "keccak256": "0xb6ee9d528b336942dd70d3b41...",
      "license": "GPL-3.0",
      "urls": [
        "bzz-raw://fe52c6e3c04ba5d83ede6cc1a43...",
        "dweb:/ipfs/QmawU3NM1WNWkBauRudYCiFvuF..."
      ]
    },
    "contracts/Ownable.sol": { ... }, // 3 items
    "contracts/ERC20.sol": { ... } // 3 items
  }
}
```

Full/Perfect Verification: How?

MyContract-**diff**.sol Hashed → 0xb6ee9d...

Ownable.sol Hashed → 0x41e281...

ERC20.sol Hashed → 0x9fd73f...

...

Metadata

```
{
  "compiler": { ... }, // 1 item
  "language": "Solidity",
  "output": { ... }, // 3 items
  "settings": { ... }, // 6 items
  "sources": {
    "contracts/MyContract.sol": {
      "keccak256": "0xb6ee9d528b336942dd70d3b41...",
      "license": "GPL-3.0",
      "urls": [
        "bzz-raw://fe52c6e3c04ba5d83ede6cc1a43...",
        "dweb:/ipfs/QmawU3NM1WNWkBauRudYCiFvuF..."
      ]
    },
    "contracts/Ownable.sol": { ... }, // 3 items
    "contracts/ERC20.sol": { ... } // 3 items
  ]
}
```


Full/Perfect Verification: How?

MyContract-**diff**.sol Hashed → 0xa2fc16...

Ownable.sol Hashed → 0x41e281...

ERC20.sol Hashed → 0x9fd73f...

...

Metadata

```
{
  "compiler": { ... }, // 1 item
  "language": "Solidity",
  "output": { ... }, // 3 items
  "settings": { ... }, // 6 items
  "sources": {
    "contracts/MyContract.sol": {
      "keccak256": "0xb6ee9d528b336942dd70d3b41...",
      "license": "GPL-3.0",
      "urls": [
        "bzz-raw://fe52c6e3c04ba5d83ede6cc1a43...",
        "dweb:/ipfs/QmawU3NM1WNWkBauRudYCiFvuF..."
      ]
    },
    "contracts/Ownable.sol": { ... }, // 3 items
    "contracts/ERC20.sol": { ... } // 3 items
  ]
}
```

Full/Perfect Verification: How?

MyContract-**diff**.sol Hashed → 0xa2fc16...

Ownable.sol Hashed → 0x41e281...

ERC20.sol Hashed → 0x9fd73f...

...

Metadata

```
{
  "compiler": { ... }, // 1 item
  "language": "Solidity",
  "output": { ... }, // 3 items
  "settings": { ... }, // 6 items
  "sources": {
    "contracts/MyContract.sol": {
      "keccak256": "0xa2fc161e281128b336942dd7...",
      "license": "GPL-3.0",
      "urls": [
        "bzz-raw://d83e43c45fe52c6e3c04ba5e6...",
        "dweb:/ipfs/QmawU3NM1WNWkCiFvu8akyFE1..."
      ]
    },
    "contracts/Ownable.sol": { ... }, // 3 items
    "contracts/ERC20.sol": { ... } // 3 items
  }
}
```

```
b336942dd70e10a47304f5ed2776352009d3b4bfd73f85606",
```

```
e3c04ba5e6cc1ab207f64707afba43caa435d17d3af1bcf1",  
iFvu8akyFE1tTLBauRudYHB9Bvb9UWwA"
```

```
3 items  
items
```

Metadata

IPFS hash

QmWTqspM5B1quNvdhXbS6TbXzyLZ5cUGHnTV8ZWJPqrQqj

```
b336942dd70e10a47304f5ed2776352009d3b4bfd73f85606",
```


```
e3c04ba5e6cc1ab207f64707afba43caa435d17d3af1bcf1",  
iFvu8akyFE1tTLBauRudYHB9Bvb9UWwA"
```

```
3 items  
items
```

Metadata

IPFS hash

QmawU3NM1WNWkCiFvu8akyFE1tTLBauRudYHB9Bvb9UWwA




QmawU3NM1WNWkCiFvu8akyFE1tTLBauRudYHB9Bvb9UWwA

Contract Bytecode

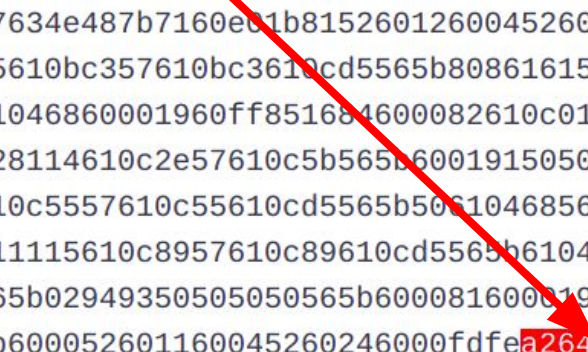


```
919050565b600082610b9a57634e487b7160e01b81526012600452602481fd5b500490565b80825b6001808611610bb157
50610bdc565b818704821115610bc357610bc3610cd5565b80861615610bd057918102915b9490941c938002610ba2565b
94509492505050565b600061046860001960ff851684600082610c0157506001610468565b81610c0e5750600061046856
5b8160018114610c245760028114610c2e57610c5b565b6001915050610468565b60ff841115610c3f57610c3f610cd556
5b6001841b915084821115610c5557610c55610cd5565b50610468565b5060208310610133831016604e8410600b841016
1715610c8e575081810a83811115610c8957610c89610cd5565b610468565b610c9b8484846001610b9f565b8086048211
15610cad57610cad610cd5565b029493505050565b6000816000710483118215151615610cd057610cd0610cd5565b50
0290565b634e487b7160e01b600052601160045260246000fdfea264697066735822122078b530288f1cffe879bb7d9062
e904deb3aa9b9c8d27ea4ecafa987583c18a6e64736f6c63430008010033
```



QmawU3NM1WNWkCiFvu8akyFE1tTLBauRudYHB9Bvb9UWwA

Contract Bytecode

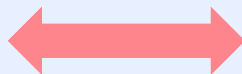


```
919050565b600082610b9a57634e487b7160e01b81526012600452602481fd5b500490565b80825b6001808611610bb157
50610bdc565b818704821115610bc357610bc3610cd5565b80861615610bd057918102915b9490941c938002610ba2565b
94509492505050565b6000061046860001960ff851684600082610c0157506001610468565b81610c0e57506000061046856
5b8160018114610c245760028114610c2e57610c5b565b6001915050610468565b60ff841115610c3f57610c3f610cd556
5b6001841b915084821115610c5557610c55610cd5565b50610468565b5060208310610133831016604e8410600b841016
1715610c8e575081810a83811115610c8957610c89610cd5565b610468565b610c9b8484846001610b9f565b8086048211
15610cad57610cad610cd5565b02949350505050565b60000816000190483118215151615610cd057610cd0610cd5565b50
0290565b634e487b7160e01b600052601160045260246000fdfea2646970667358e879bb7d9062e3aa9b9c22122078b530
288f1c904debff8d27ea4eca18a6e6fa987583c4736f6c63430008010033
```



```
00490565b80825b6001808611610bb157
18102915b9490941c938002610ba2565b
10468565b81610c0e5750600061046856
b60ff841115610c3f57610c3f610cd556
310610133831016604e8410600b841016
c9b8484846001610b9f565b8086048211
5151615610cd057610cd0610cd5565b50
8e879bb7d9062e3aa9b9c22122078b530
```

match?



Contract Bytecode

```
919050565b600082610b9a57634e487b7160e01b8157
50610bdc565b818704821115610bc357610bc3610cd5
94509492505050565b600061046860001960ff851684
5b8160018114610c245760028114610c2e57610c5b50
5b6001841b915084821115610c5557610c55610cd550
1715610c8e575081810a83811115610c8957610c8963
15610cad57610cad610cd5565b02949350505050565f
0290565b634e487b7160e01b60005260116004526024
e904deb3aa9b9c8d27ea4ecafa987583c18a6e64736
```

```
00490565b80825b6001808611610bb157
18102915b9490941c938002610ba2565b
10468565b81610c0e5750600061046856
b60ff841115610c3f57610c3f610cd556
310610133831016604e8410600b841016
c9b8484846001610b9f565b8086048211
5151615610cd057610cd0610cd5565b50
8e879bb7d9062e3aa9b9c22122078b530
```



Partial match



Contract Bytecode

```
919050565b600082610b9a57634e487b7160e01b815
50610bdc565b818704821115610bc357610bc3610cd
94509492505050565b600061046860001960ff85168
5b8160018114610c245760028114610c2e57610c5b5
5b6001841b915084821115610c5557610c55610cd55
1715610c8e575081810a83811115610c8957610c896
15610cad57610cad610cd5565b02949350505050565
0290565b634e487b7160e01b6000526011600452602
e904deb3aa9b9c8d27ea4ecafa987583c18a6e64736
```


How to verify?

sourcify.dev UI

Verifier

Verify smart contracts by recompiling with the Solidity source code and metadata.

Old verifier UI available at legacy.sourcify.dev

File Add Zone

Add the Solidity source files and metadata of all contracts you want to verify.

+ Import from remote

 Import from Etherscan

 Import from GitHub

CLEAR FILES

Added Files (2)

- 0x00878Ac0D6B8d981ae72BA7cDC967eA0Fae69df4/metadata.json
- 0x00878Ac0D6B8d981ae72BA7cDC967eA0Fae69df4/sources/contracts/1_Storage.sol

Contracts

COLLAPSE ALL

Storage

Chain & Address Missing



API

Sends provided files for verification.

URL : `/verify` or `/`

Method : `POST`

Content-Type : `multipart/form-data` or `application/json`

```
{
  "address": ...,
  "chain": ...,
  "files": {
    "file-name1.sol": ...,
    "file-name2.sol": ...
  }
}
```

API

- Check by addresses (full match) : `GET /check-by-addresses?addresses={addresses}&chainIds={chainIds}`
- Check by addresses (full or partial match) : `GET /check-by-all-addresses?addresses={addresses}&chainIds={chainIds}`
- Get file tree (full match) : `GET /files/tree/:chain/:address`
- Get file tree (full or partial match) : `GET /files/tree/any/:chain/:address`
- Get source files (full match) : `GET /files/:chain/:address`
- Get source files (full or partial match) : `GET /files/any/:chain/:address`
- Get contract addresses (full or partial match) : `GET /files/contracts/:chain`

docs.sourcify.dev

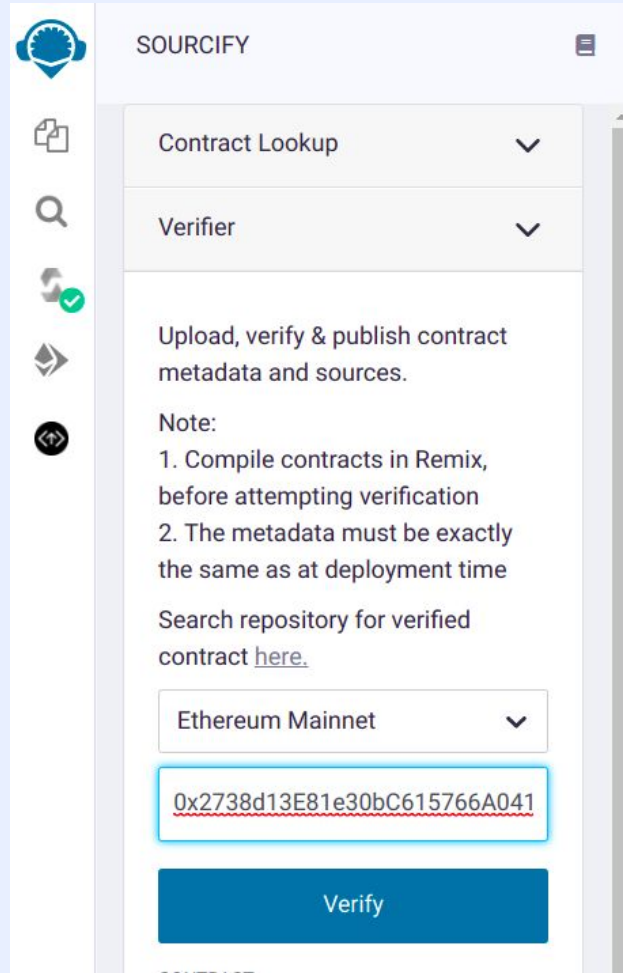
Tooling

[@wighawag/hardhat-deploy](#)

```
$ hardhat --network mainnet sourcify
```

Tooling

Remix Plugin



The screenshot shows the SOURCIFY plugin interface. On the left is a sidebar with icons for file explorer, search, and other tools. The main panel has a header with the SOURCIFY logo and a menu icon. Below the header are two expandable sections: 'Contract Lookup' and 'Verifier'. The 'Verifier' section is currently expanded, showing instructions on how to upload, verify, and publish contract metadata and sources. It includes a note with two steps: 1. Compile contracts in Remix, before attempting verification; 2. The metadata must be exactly the same as at deployment time. Below the note is a link to 'Search repository for verified contract [here](#).' and a dropdown menu set to 'Ethereum Mainnet'. A text input field contains the contract address '0x2738d13E81e30bC615766A041', which is highlighted with a red dashed border. At the bottom is a blue 'Verify' button.

SOURCIFY

Contract Lookup

Verifier

Upload, verify & publish contract metadata and sources.

Note:

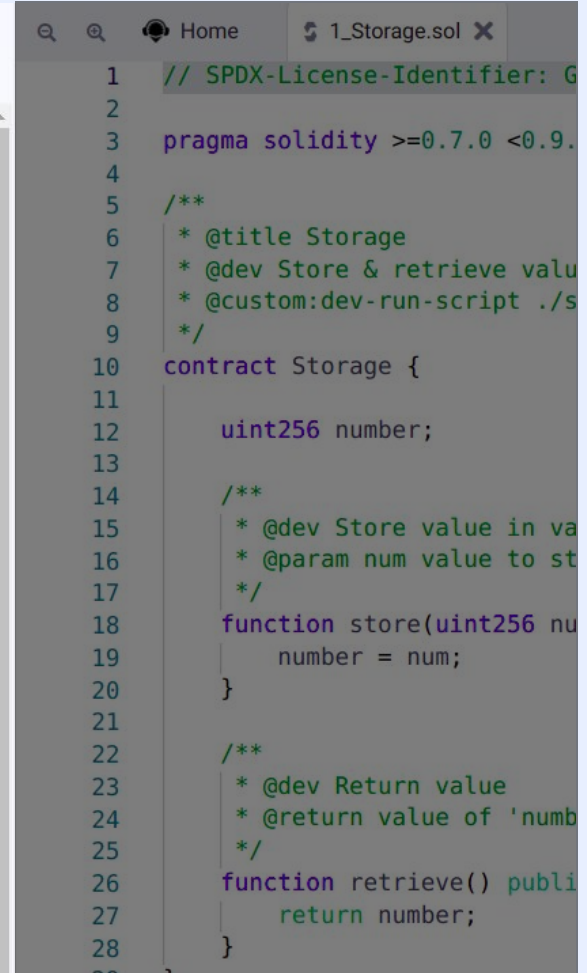
1. Compile contracts in Remix, before attempting verification
2. The metadata must be exactly the same as at deployment time

Search repository for verified contract [here](#).

Ethereum Mainnet

0x2738d13E81e30bC615766A041

Verify



The screenshot shows a Solidity code editor with a file named '1_Storage.sol'. The code defines a 'Storage' contract with a 'uint256' variable 'number'. It includes a 'store' function that takes a 'uint256' parameter and assigns it to 'number', and a 'retrieve' function that returns the value of 'number'. The code is annotated with SPDX license identifiers and developer comments.

```
// SPDX-License-Identifier: G
pragma solidity >=0.7.0 <0.9.
/**
 * @title Storage
 * @dev Store & retrieve valu
 * @custom:dev-run-script ./s
 */
contract Storage {
    uint256 number;

    /**
     * @dev Store value in va
     * @param num value to st
     */
    function store(uint256 nu
        number = num;
    }

    /**
     * @dev Return value
     * @return value of 'numb
     */
    function retrieve() publi
        return number;
    }
```

Tooling

Foundry



```
$ forge create Contract --verify --verification-provider sourcify
```

```
$ forge verify-contract <address> --verifier sourcify
```

```
$ forge verify-check <address> --verifier sourcify
```

Automatic Verification

- i.e. **Monitor**
- Catches contract creations
- Tries to fetch
 - 1) Metadata
 - 2) Source files from IPFS
 - Metadata file contains source file IPFS hashes too
- Automatically compiles and verifies

As a contract developer:

- 1) Use NatSpec documentation
- 2) **Source code verification on Sourcify**

As a contract developer:

- 1) Use NatSpec documentation
- ~~2) Source code verification on Sourcify~~

As a contract developer:

- 1) Use NatSpec documentation
- 2) **Publish (pin) metadata and source files on IPFS**

As a contract developer:

- 1) Use NatSpec documentation
- 2) **Publish (pin) metadata and source files on IPFS**

If you publish on IPFS we verify for you! 🎉

Contract repository

- Served over HTTP (repo.sourcify.dev) and IPFS
- We pin the verified contract source files and metadata so that they are accessible by decoding the bytecode.

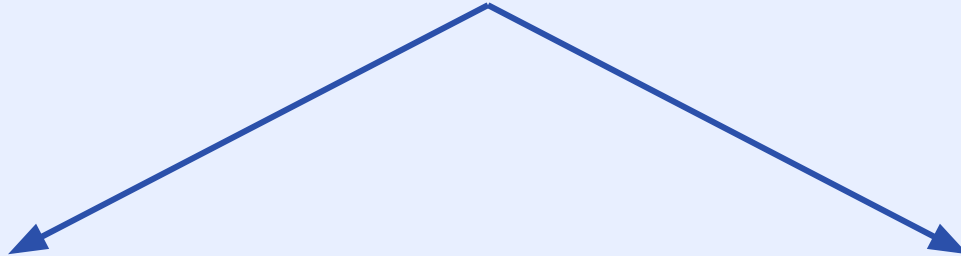
```
15610cad57610cad610cd5565b02949350505050565b60000816000190483118215151615610cd057610cd0610cd5565b50  
0290565b634e487b7160e01b6000052601160045260246000fdfea264697066735822122078b530288f1cffe879bb7d9062  
e904deb3aa9b9c8d27ea4ecafa987583c18a6e64736f6c63430008010033
```

Metadata Hash (decoded)

ipfs://QmWTqspM5B1quNvdhXbS6TbXzyLZ5cUGHnTV8ZWJPqrQqj

- IPNS updated regularly
- [/ipns/repo.sourcify.dev](https://ipns/repo.sourcify.dev)

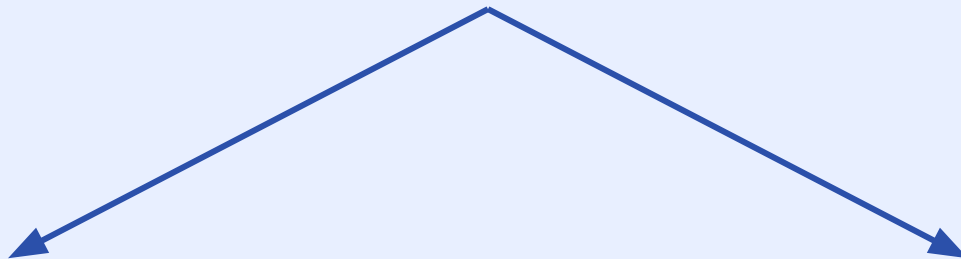
What can you do to achieve this...



...as a smart contract developer?

...as a wallet developer?

What can you do to achieve this...



...as a smart contract developer?

...as a wallet developer?

What is the goal again?

[illegible]

decode with
ABI JSON

Human-readable
description

Function

```
depositETH(address,address,uint)
```

Arguments

pool1:0xd9Db270c1B5E3Bd161E8c8503c55cEABeE709552

onBehalfOf:0x68b3465833fb72A70ecDF485E0e4C7bD8665Fc45

referralCode:0

deposits WETH into the pool
0xd9Db270c1B5E3Bd161E8c8503c55cEA
BeE709552, on behalf of the account
0x68b3465833fb72A70ecDF485E0e4C7b
D8665Fc45 using native ETH. A
corresponding amount of the overlying
asset is minted

As a wallet developer

1) Fetch metadata

https://repo.sourcify.dev/contracts/full_match/{chainId}/{address}/metadata.json

As a wallet developer

1) Fetch metadata

https://repo.sourcify.dev/contracts/full_match/{chainId}/{address}/metadata.json

Remember you can get it through the contract bytecode!

As a wallet developer

1) Get contract bytecode `eth_getCode("0x1fa47...b2c53f")`

```
15610cad57610cad610cd5565b02949350505050565b6000816000190483118215151615610cd057610cd0610cd5565b50
0290565b634e487b7160e01b600052601160045260246000fdfea264697066735822122078b530288f1cffe879bb7d9062
e904deb3aa9b9c8d27ea4ecafa987583c18a6e64736f6c63430008010033
```

As a wallet developer

1) Get contract bytecode `eth_getCode("0x1fa47...b2c53f")`

```
15610cad57610cad610cd5565b02949350505050565b6000816000190483118215151615610cd057610cd0610cd5565b50
0290565b634e487b7160e01b600052601160045260246000fdfea264697066735822122078b530288f1cffe879bb7d9062
e904deb3aa9b9c8d27ea4ecafa987583c18a6e64736f6c63430008010033
```

2) Fetch metadata via the IPFS hash

We pinned it for you!

Metadata Hash (decoded)

`ipfs://QmWTqspM5B1quNvdhXbS6TbXzyLZ5cUGHnTV8ZWJPqrQqj`

Solidity Contract Metadata

JSON file generated by the Solidity compiler which contains... metadata:

- ABI
 - Userdoc + devdoc
 - Compilation info
 - Source file info
- }** **How to interact with the contract?**
- }** **How to reproduce a contract compilation?**

As a wallet developer

1) Get contract bytecode `eth_getCode("0x1fa47...b2c53f")`

```
15610cad57610cad610cd5565b02949350505050565b6000816000190483118215151615610cd057610cd0610cd5565b50
0290565b634e487b7160e01b600052601160045260246000fdfea264697066735822122078b530288f1cffe879bb7d9062
e904deb3aa9b9c8d27ea4ecafa987583c18a6e64736f6c63430008010033
```

2) Fetch metadata via the IPFS hash

We pinned it for you!

Metadata Hash (decoded)

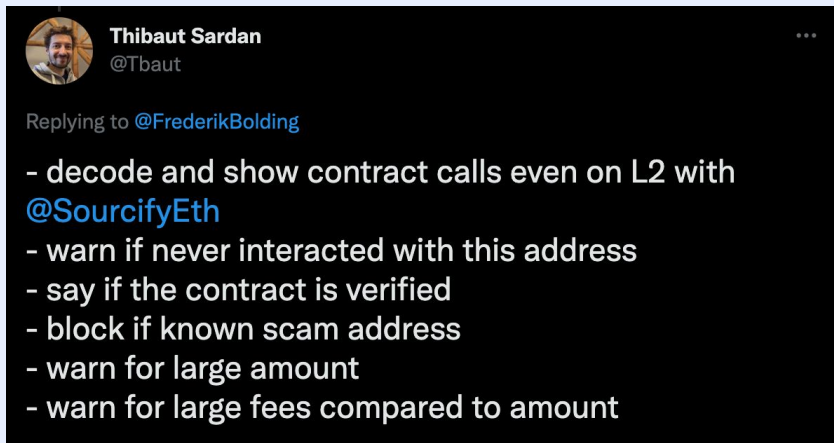
`ipfs://QmWTqspM5B1quNvdhXbS6TbXzyLZ5cUGHnTV8ZWJPqrQqj`

3) Decode **ABI** + populate **NatSpec** comments for the method

Ways to “human-friendliness”

- **Sourcify**
 - Idea: show human readable descriptions via the NatSpec comments found in the metadata
- **EIP-4430 & EIP-3224**
 - Richard Moore (@ricmoo) and Nick Johnson (@arachnid)
 - Idea: Include a `eipXXXdescribe` function *inside the contract* that will return a human readable description
 - Can decode things like an ENS `commit(bytes32 hash)` but extra data (gas) in contract.
- **EIP: Rich Site-Proposed Contract Metadata**
 - Dan Finlay (@danfinlay)
 - Idea: The first point of contact with the contract address proposes metadata to the wallet
 - visit `app.uniswap.org` → receive ABI + method describers from the website → save to the wallet
 - Backwards compatible and flexible but not bound to the contract itself

Ways to better UX:



We can do better! 💪

- How many times was this contract interacted with?
- When was this contract deployed?
- Is this contract audited and by whom?

Recap: What is Sourcify?



Technically:

- An open-sourced **automatic** smart contract verification service (**Monitor**)
- A user interface (**UI**) , API (**Server**), and tooling (**Plugins**) to verify contracts manually
- A public, decentralized content-addressed storage (IPFS) of verified contracts (**Repo**)

More generally:

- A **base layer** and **public good** for other tools to build on top
- An initiative to foster the use of Solidity contract metadata, NatSpec, and full verification
- An ongoing effort to improve smart contract UX, safety, and transparency

Thank you



sourcify.dev



ethereum/sourcify



@sourcifyeth



Matrix chat: [#ethereum_source-verify:gitter.im](https://matrix.to/#/#ethereum_source-verify:gitter.im)



sourcify.dev