Remix Hybrid Modules

Remix Hybrid Modules

Looking to quickly draft a DApp or to test your contract?

Remix can help at different phases of DApp development, even if you are using other tools/frameworks:

- Development
- Compilation
- Static Analysis
- Unit Testing
- Deployment

Smart Contract development

Smart Contract development

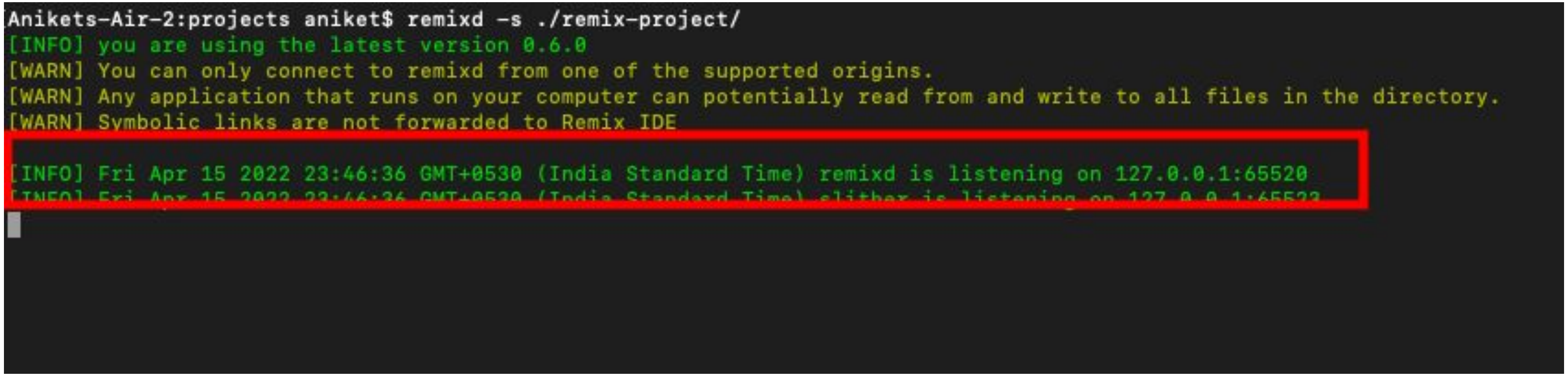
Remixd connects the browser based Remix IDE to…

a folder on your computer’s file system

- **Remixd**: An NPM module that connects a project from a local file system to the browser based Remix IDE
- Create files or folders on the shared directory and they’ll appear in Remix - or the other way around.

Remixd

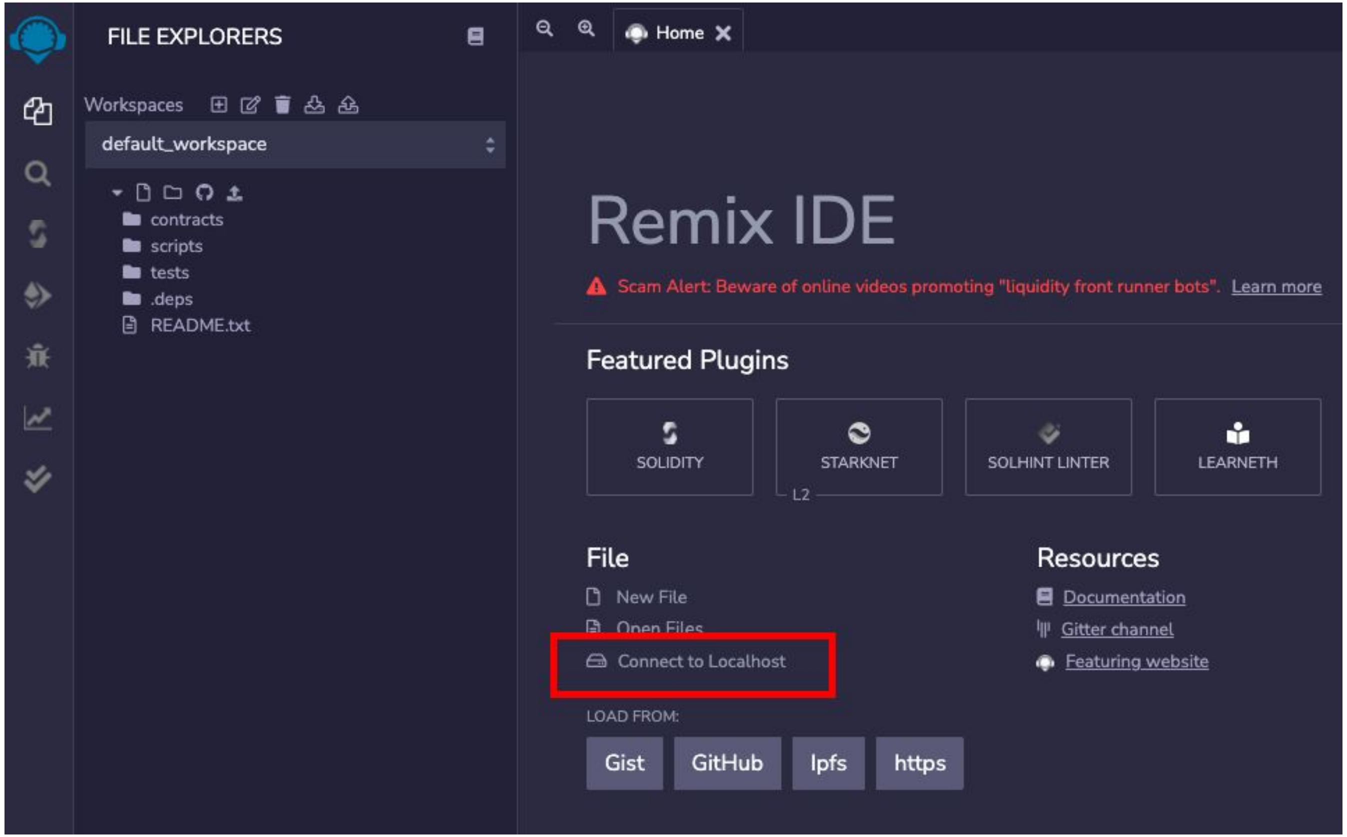
- Install: `npm install @remix-project/remixd -g`
- Run: `remixd -s <shared-folder-path> -u <remix-ide-instance-URL>`
- Start a websocket listener on a specific port for back-and-forth data synchronization



```
Anikets-Air-2:projects aniket$ remixd -s ./remix-project/
[INFO] you are using the latest version 0.6.0
[WARN] You can only connect to remixd from one of the supported origins.
[WARN] Any application that runs on your computer can potentially read from and write to all files in the directory.
[WARN] Symbolic links are not forwarded to Remix IDE

[INFO] Fri Apr 15 2022 23:46:36 GMT+0530 (India Standard Time) remixd is listening on 127.0.0.1:65520
[INFO] Fri Apr 15 2022 23:46:36 GMT+0530 (India Standard Time) slither is listening on 127.0.0.1:65523
```

Remixd

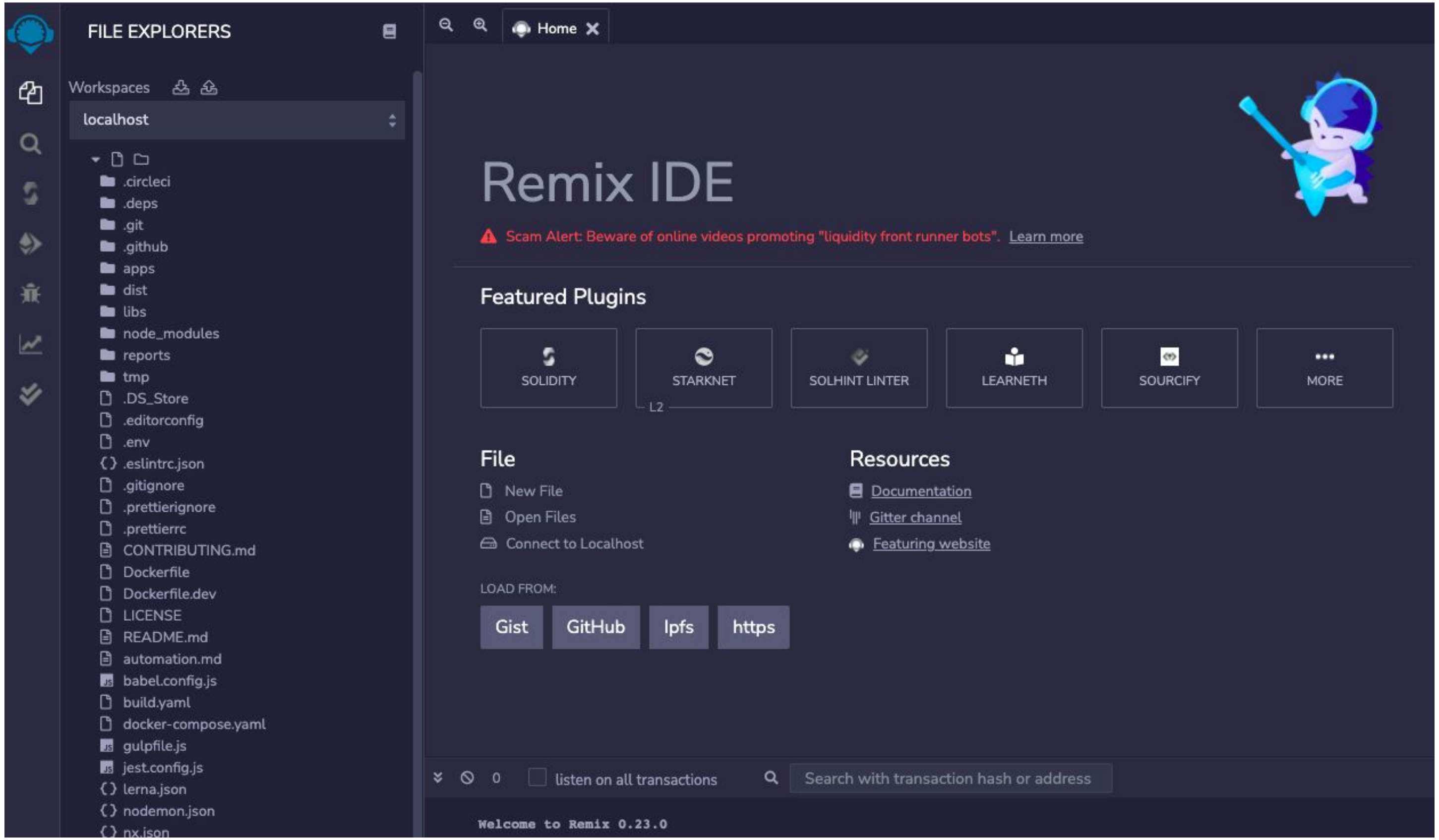
- Once remixd is up, go to Remix IDE, click on ‘Connect to Localhost’ from home tab



Scam Alert: Beware of online videos promoting "liquidity front runner bots". Learn more

Remixd

- Shared project will be loaded under 'localhost' workspace



Scam Alert: Beware of online videos promoting "liquidity front runner bots". Learn more

```
Welcome to Remix 0.23.0
```

Smart Contract Compilation

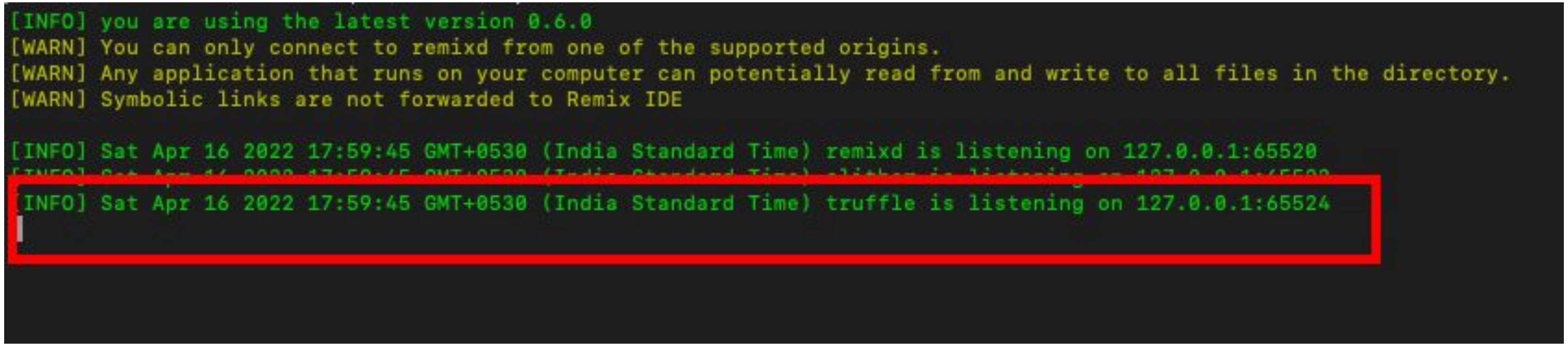
Smart Contract Compilation

Remix connects to Frameworks!

- **Remix's Solidity Compiler**: enables compilation for other frameworks along with Remix
- The supported frameworks are Truffle & Hardhat

Smart Contract Compilation

- When a Truffle project is shared using Remixd
- Starts a Truffle specific action listener a different port
- Truffle should be installed locally on the system along with it its dependencies

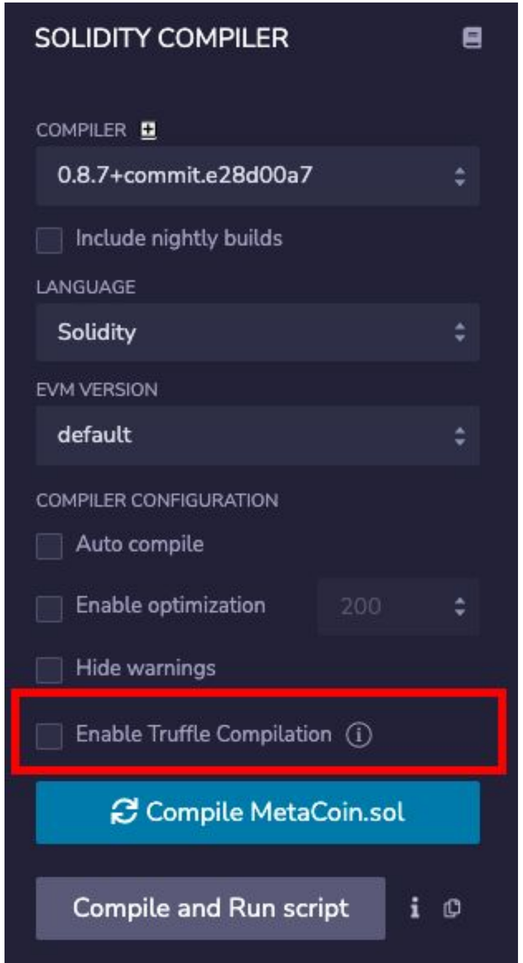


```
[INFO] you are using the latest version 0.6.0
[WARN] You can only connect to remixd from one of the supported origins.
[WARN] Any application that runs on your computer can potentially read from and write to all files in the directory.
[WARN] Symbolic links are not forwarded to Remix IDE

[INFO] Sat Apr 16 2022 17:59:45 GMT+0530 (India Standard Time) remixd is listening on 127.0.0.1:65520
[INFO] Sat Apr 16 2022 17:59:45 GMT+0530 (India Standard Time) slither is listening on 127.0.0.1:65523
[INFO] Sat Apr 16 2022 17:59:45 GMT+0530 (India Standard Time) truffle is listening on 127.0.0.1:65524
```

Smart Contract Compilation

- Solidity Compiler plugin will show an additional checkbox labelled as **‘Enable Truffle Compilation’**

- Check the box and click the Compile button



SOLIDITY COMPILER

COMPILER

0.8.7+commit.e28d00a7

Include nightly builds

LANGUAGE

Solidity

EVM VERSION

default

COMPILER CONFIGURATION

Auto compile

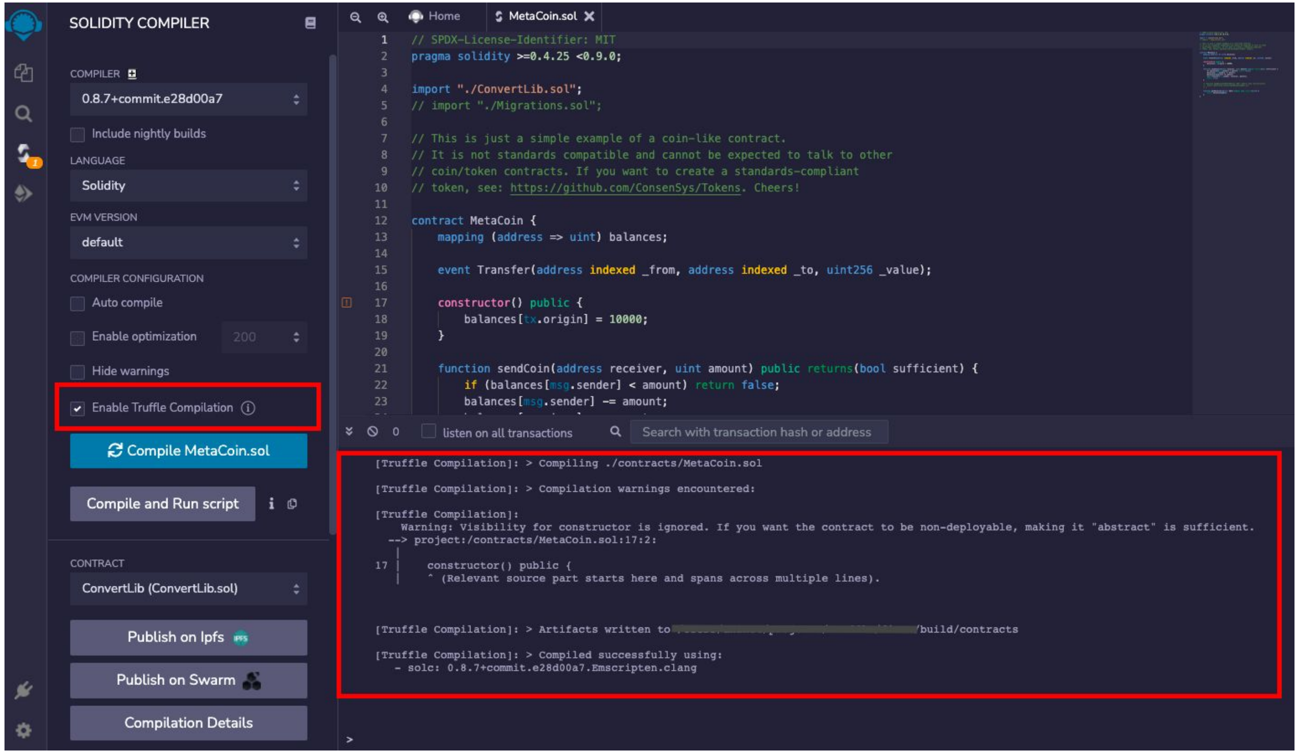
Enable optimization 200

Hide warnings

Enable Truffle Compilation

Smart Contract Compilation

- Performs compilation for both Remix and Truffle
- Shows progress, result, warning or errors on Remix Terminal



SOLIDITY COMPILER

COMPILER: 0.8.7+commit.e28d00a7

Include nightly builds

LANGUAGE: Solidity

EVM VERSION: default

COMPILER CONFIGURATION

Auto compile

Enable optimization 200

Hide warnings

☑ Enable Truffle Compilation

Compile MetaCoin.sol

Compile and Run script

CONTRACT: ConvertLib (ConvertLib.sol)

Publish on Ipfs

Publish on Swarm

Compilation Details

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.4.25 <0.9.0;

import "./ConvertLib.sol";
// import "./Migrations.sol";

// This is just a simple example of a coin-like contract.
// It is not standards compatible and cannot be expected to talk to other
// coin/token contracts. If you want to create a standards-compliant
// token, see: https://github.com/ConsenSys/Tokens. Cheers!

contract MetaCoin {
    mapping (address => uint) balances;

    event Transfer(address indexed _from, address indexed _to, uint256 _value);

    constructor() public {
        balances[tx.origin] = 10000;
    }

    function sendCoin(address receiver, uint amount) public returns(bool sufficient) {
        if (balances[msg.sender] < amount) return false;
        balances[msg.sender] -= amount;
```

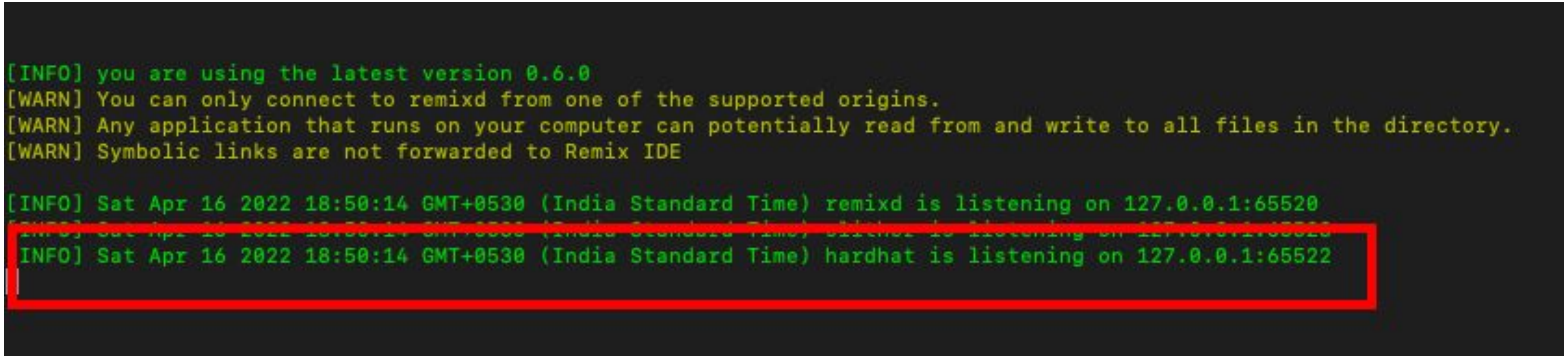
0
listen on all transactions
Search with transaction hash or address

```
[Truffle Compilation]: > Compiling ./contracts/MetaCoin.sol

[Truffle Compilation]: > Compilation warnings encountered:

[Truffle Compilation]:
    Warning: Visibility for constructor is ignored. If you want the contract to be non-deployable, making it "abstract" is sufficient.
  --> project:/contracts/MetaCoin.sol:17:2:
   |
17 |    constructor() public {
   |    ^ (Relevant source part starts here and spans across multiple lines).



[Truffle Compilation]: > Artifacts written to [redacted]/build/contracts

[Truffle Compilation]: > Compiled successfully using:
   - solc: 0.8.7+commit.e28d00a7.Emscripten.clang
```

```
>
```

Smart Contract Compilation

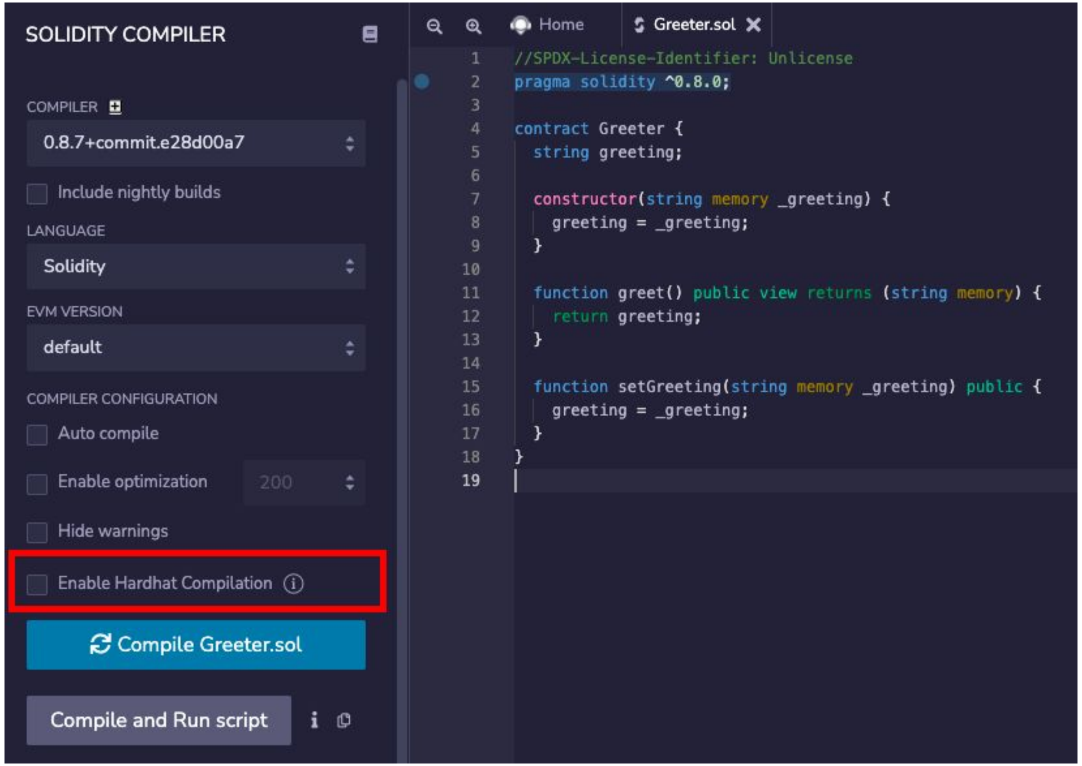
- Similarly, when a Hardhat project is shared using Remixd
- Remixd starts a Hardhat specific action listener
- Hardhat should be installed along with it its dependencies



```
[INFO] you are using the latest version 0.6.0
[WARN] You can only connect to remixd from one of the supported origins.
[WARN] Any application that runs on your computer can potentially read from and write to all files in the directory.
[WARN] Symbolic links are not forwarded to Remix IDE

[INFO] Sat Apr 16 2022 18:50:14 GMT+0530 (India Standard Time) remixd is listening on 127.0.0.1:65520
[INFO] Sat Apr 16 2022 18:50:14 GMT+0530 (India Standard Time) slither is listening on 127.0.0.1:65523
[INFO] Sat Apr 16 2022 18:50:14 GMT+0530 (India Standard Time) hardhat is listening on 127.0.0.1:65522
```

Smart Contract Compilation

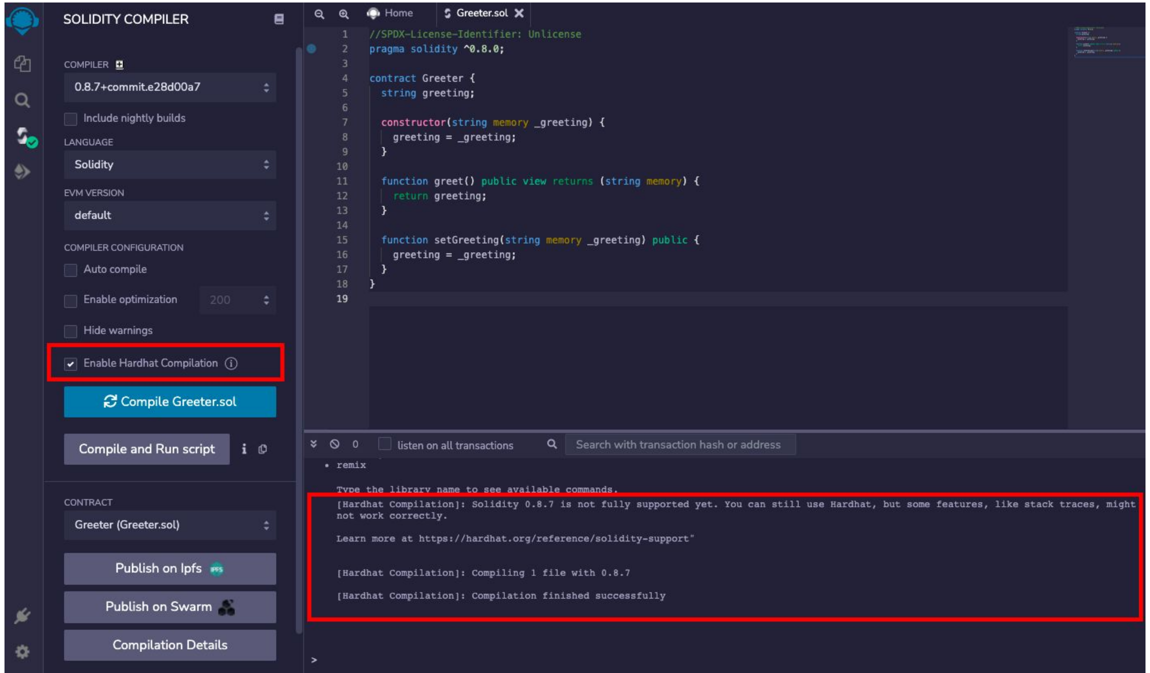
- ‘Solidity Compiler’ plugin will show a checkbox labelled as **‘Enable Hardhat Compilation’**
- Check the box and click on Compile button



```
//SPDX-License-Identifier: Unlicense
pragma solidity ^0.8.0;

contract Greeter {
  string greeting;

  constructor(string memory _greeting) {
    greeting = _greeting;
  }

  function greet() public view returns (string memory) {
    return greeting;
  }

  function setGreeting(string memory _greeting) public {
    greeting = _greeting;
  }
}
```

Smart Contract Compilation

- Compilation will be performed for both Remix and Hardhat
- Compilation for Hardhat will be done **with the configuration set in ‘Solidity Compiler’ Plugin**
- Remix terminal will show the progress and result



SOLIDITY COMPILER

COMPILER: 0.8.7+commit.e28d00a7

[ ] Include nightly builds

LANGUAGE: Solidity

EVM VERSION: default

COMPILER CONFIGURATION

[ ] Auto compile

[ ] Enable optimization 200

[ ] Hide warnings

[✓] Enable Hardhat Compilation

CONTRACT: Greeter (Greeter.sol)

```
//SPDX-License-Identifier: Unlicense
pragma solidity ^0.8.0;

contract Greeter {
  string greeting;

  constructor(string memory _greeting) {
    greeting = _greeting;
  }

  function greet() public view returns (string memory) {
    return greeting;
  }

  function setGreeting(string memory _greeting) public {
    greeting = _greeting;
  }
}
```

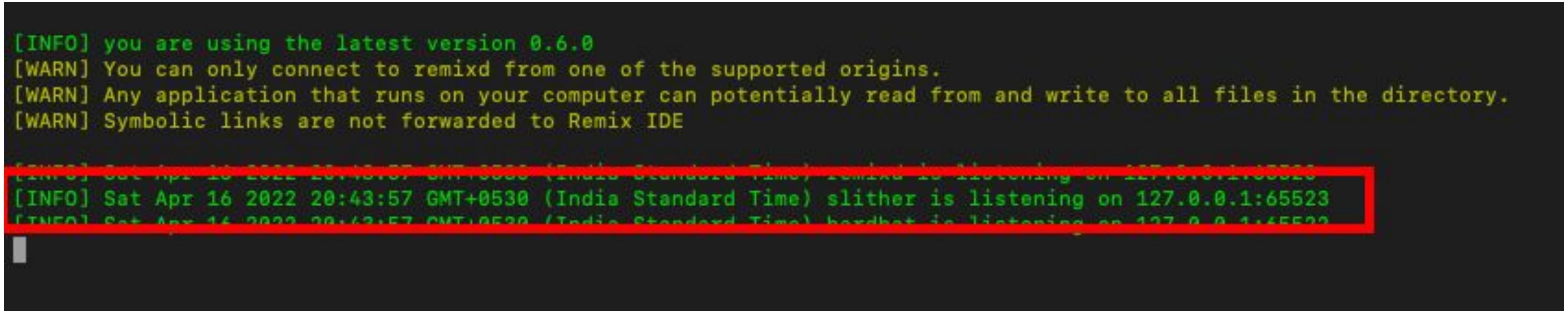
```
• remix

  Type the library name to see available commands.
  [Hardhat Compilation]: Solidity 0.8.7 is not fully supported yet. You can still use Hardhat, but some features, like stack traces, might
  not work correctly.

  Learn more at https://hardhat.org/reference/solidity-support"


  [Hardhat Compilation]: Compiling 1 file with 0.8.7

  [Hardhat Compilation]: Compilation finished successfully
```

Smart Contract Static Analysis

Remix connects to Slither

Smart Contract Static Analysis

- **Solidity Static Analysis**: Remix IDE’s plugin performs smart contract static analysis for Remix and…
- **Slither**
- Remixd starts a Slither specific action listener for each project
- Slither should be installed along with it its dependencies

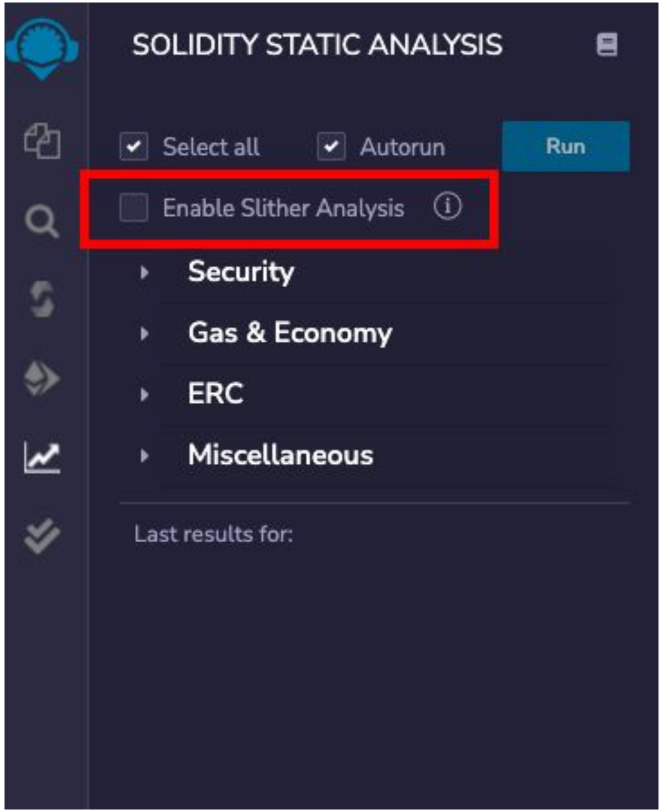


```
[INFO] you are using the latest version 0.6.0
[WARN] You can only connect to remixd from one of the supported origins.
[WARN] Any application that runs on your computer can potentially read from and write to all files in the directory.
[WARN] Symbolic links are not forwarded to Remix IDE

[INFO] Sat Apr 16 2022 20:43:57 GMT+0530 (India Standard Time) remixd is listening on 127.0.0.1:65520
[INFO] Sat Apr 16 2022 20:43:57 GMT+0530 (India Standard Time) slither is listening on 127.0.0.1:65523
[INFO] Sat Apr 16 2022 20:43:57 GMT+0530 (India Standard Time) hardhat is listening on 127.0.0.1:65522
```

Smart Contract Static Analysis

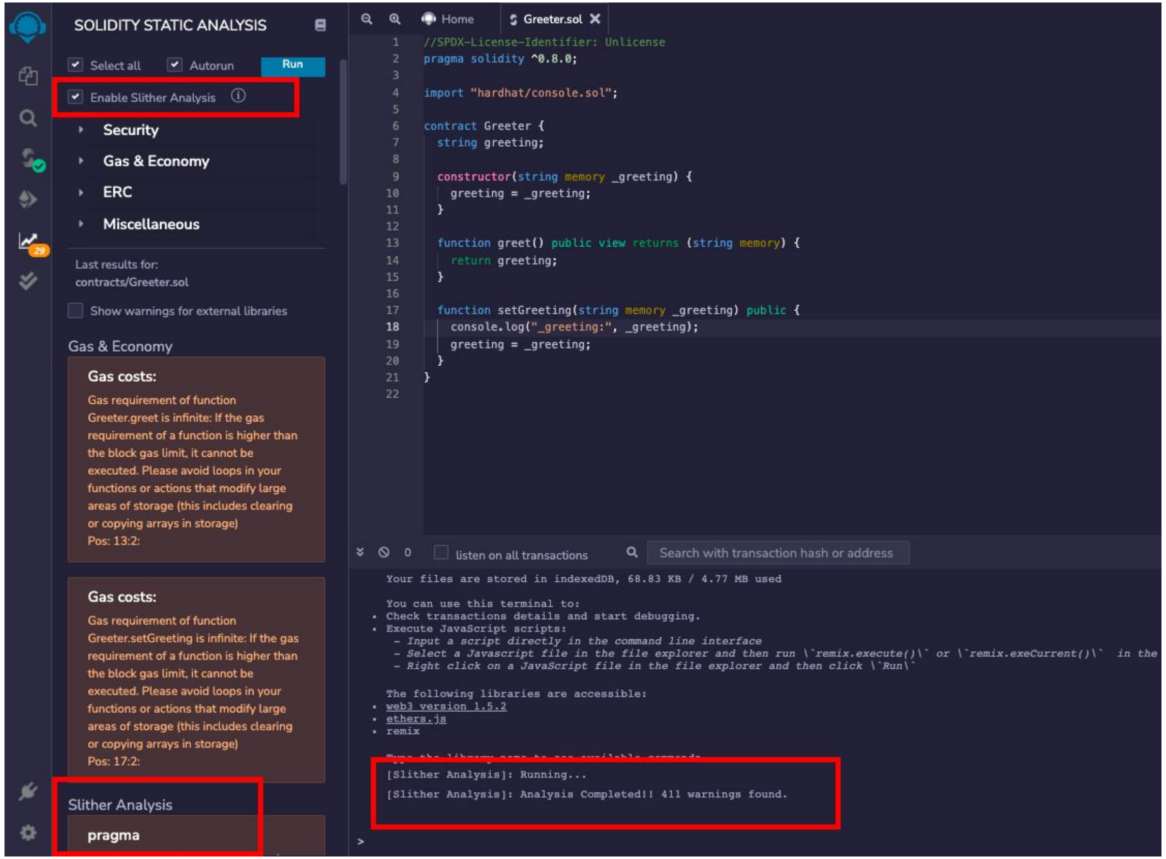
- Solidity Static Analysis plugin will show an additional checkbox labelled as ‘Enable Slither Analysis’

- Check this box to run the analysis for Slither along with Remix’s static analysis.



Smart Contract Static Analysis

- Performs analysis by both Remix and Slither
- Shows the progress in Remix terminal
- Uses compiler configuration from ‘Solidity Compiler’ plugin
- **Saves Slither analysis report**



Last results for:
contracts/Greeter.sol

Show warnings for external libraries

Gas & Economy

Gas costs:

Gas requirement of function Greeter.greet is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 13:2:

Gas costs:

Gas requirement of function Greeter.setGreeting is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 17:2:

Slither Analysis

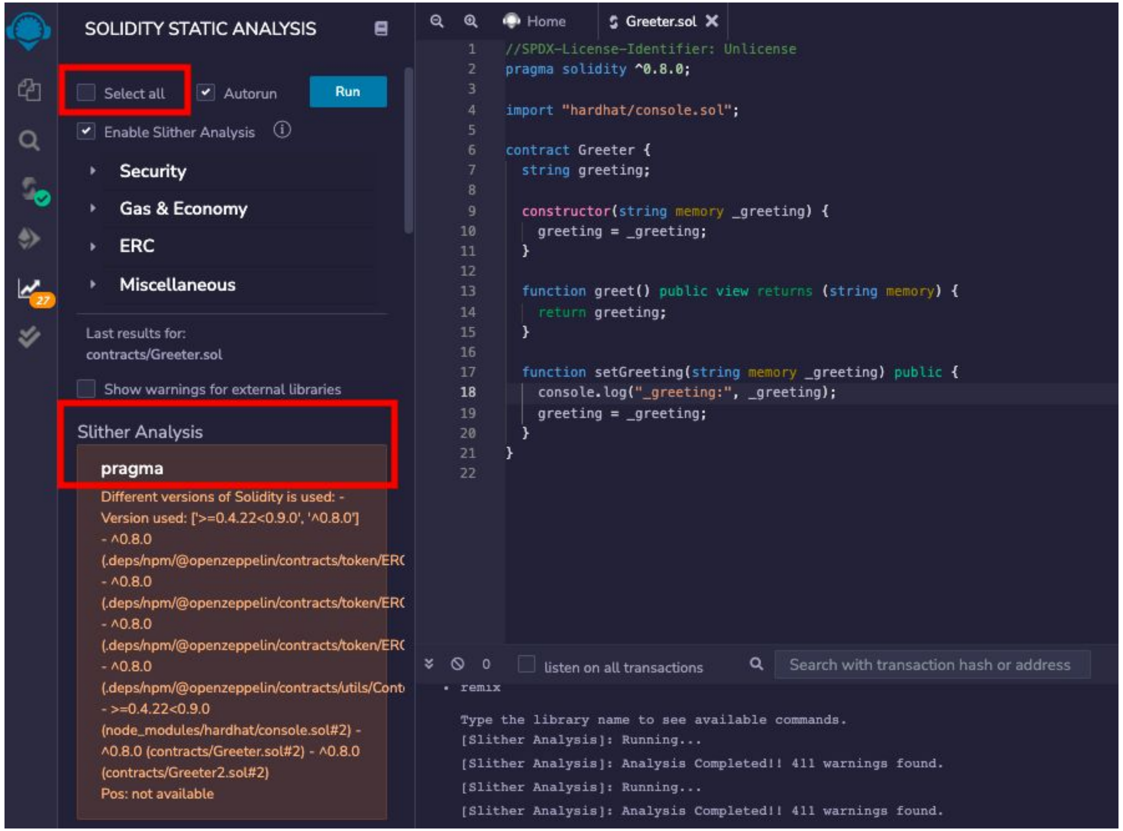
pragma

```
//SPDX-License-Identifier: Unlicense
pragma solidity ^0.8.0;

import "hardhat/console.sol";

contract Greeter {
  string greeting;

  constructor(string memory _greeting) {
    greeting = _greeting;
  }

  function greet() public view returns (string memory) {
    return greeting;
  }

  function setGreeting(string memory _greeting) public {
    console.log("_greeting:", _greeting);
    greeting = _greeting;
  }
}
```

```
  Your files are stored in indexedDB, 68.83 KB / 4.77 MB used

  You can use this terminal to:
• Check transactions details and start debugging.
• Execute JavaScript scripts:
   - Input a script directly in the command line interface
   - Select a Javascript file in the file explorer and then run \`remix.execute()\` or \`remix.exeCurrent()\`  in the
   - Right click on a JavaScript file in the file explorer and then click \`Run\`

  The following libraries are accessible:
• web3 version 1.5.2
• ethers.js
• remix

  [Slither Analysis]: Running...
  [Slither Analysis]: Analysis Completed!! 411 warnings found.
>
```

Smart Contract Static Analysis

- Uncheck `Select all` checkbox to see only the Slither results



[ ] Select all [✓] Autorun

[✓] Enable Slither Analysis

Last results for:
contracts/Greeter.sol

[ ] Show warnings for external libraries

Slither Analysis

pragma

Different versions of Solidity is used: - Version used: ['>=0.4.22<0.9.0', '^0.8.0'] - ^0.8.0 (.deps/npm/@openzeppelin/contracts/token/ERC - ^0.8.0 (.deps/npm/@openzeppelin/contracts/token/ERC - ^0.8.0 (.deps/npm/@openzeppelin/contracts/token/ERC - ^0.8.0 (.deps/npm/@openzeppelin/contracts/utils/Cont - >=0.4.22<0.9.0 (node_modules/hardhat/console.sol#2) - ^0.8.0 (contracts/Greeter.sol#2) - ^0.8.0 (contracts/Greeter2.sol#2)
Pos: not available

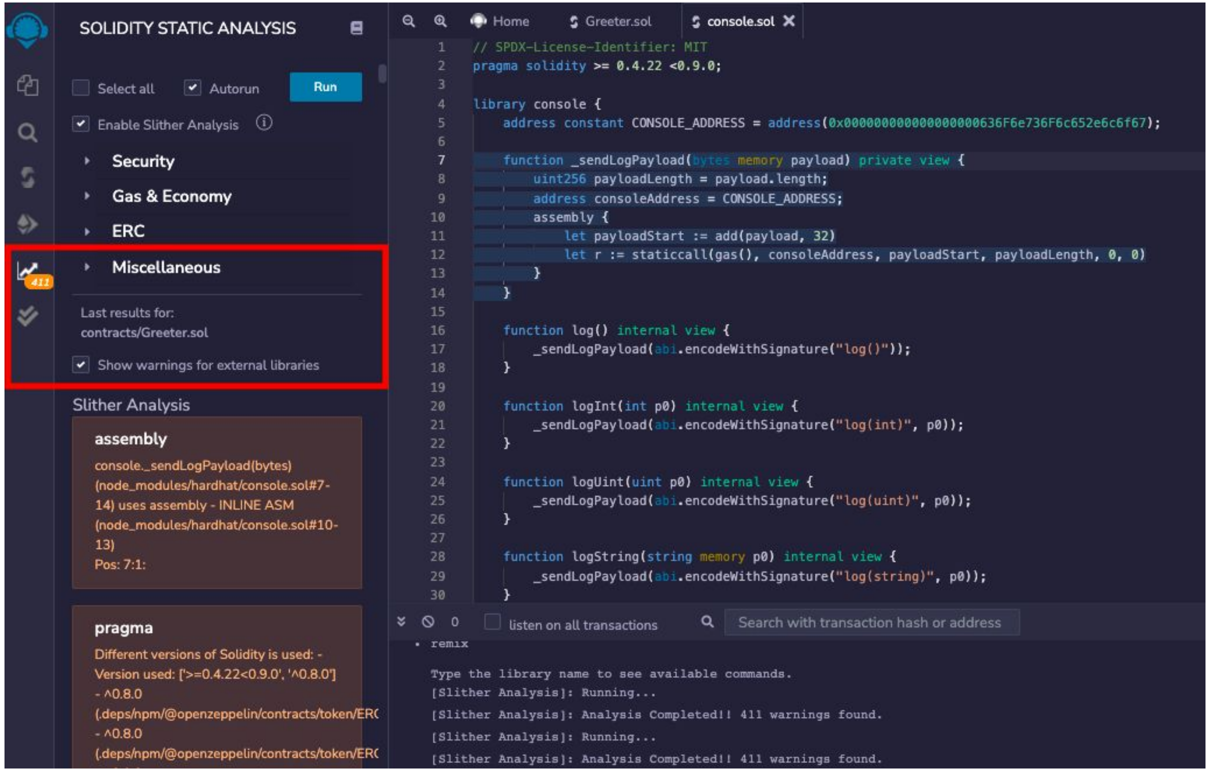
```solidity
//SPDX-License-Identifier: Unlicense
pragma solidity ^0.8.0;

import "hardhat/console.sol";

contract Greeter {
  string greeting;

  constructor(string memory _greeting) {
    greeting = _greeting;
  }

  function greet() public view returns (string memory) {
    return greeting;
  }

  function setGreeting(string memory _greeting) public {
    console.log("_greeting:", _greeting);
    greeting = _greeting;
  }
}
```

0 [ ] listen on all transactions Search with transaction hash or address

```
• remix

Type the library name to see available commands.
[Slither Analysis]: Running...
[Slither Analysis]: Analysis Completed!! 411 warnings found.
[Slither Analysis]: Running...
[Slither Analysis]: Analysis Completed!! 411 warnings found.
```

Smart Contract Static Analysis

- By default, it doesn't show warnings for externally imported libraries
- Select the checkbox labelled as 'Show warnings for external libraries' to see them
- Click on a warning to see its related source code position, if available



[✓ Autorun]

[✓ Enable Slither Analysis]

Last results for:
contracts/Greeter.sol

[✓ Show warnings for external libraries]

Slither Analysis

assembly
console._sendLogPayload(bytes) (node_modules/hardhat/console.sol#7-14) uses assembly - INLINE ASM (node_modules/hardhat/console.sol#10-13)
Pos: 7:1:

pragma
Different versions of Solidity is used: - Version used: ['>=0.4.22<0.9.0', '^0.8.0']
- ^0.8.0
(.deps/npm/@openzeppelin/contracts/token/ERC
- ^0.8.0
(.deps/npm/@openzeppelin/contracts/token/ERC

```
// SPDX-License-Identifier: MIT
pragma solidity >= 0.4.22 <0.9.0;

library console {
    address constant CONSOLE_ADDRESS = address(0x000000000000000000636F6e736F6c652e6c6f67);

    function _sendLogPayload(bytes memory payload) private view {
        uint256 payloadLength = payload.length;
        address consoleAddress = CONSOLE_ADDRESS;
        assembly {
            let payloadStart := add(payload, 32)
            let r := staticcall(gas(), consoleAddress, payloadStart, payloadLength, 0, 0)
        }
    }

    function log() internal view {
        _sendLogPayload(abi.encodeWithSignature("log()"));
    }

    function logInt(int p0) internal view {
        _sendLogPayload(abi.encodeWithSignature("log(int)", p0));
    }

    function logUint(uint p0) internal view {
        _sendLogPayload(abi.encodeWithSignature("log(uint)", p0));
    }

    function logString(string memory p0) internal view {
        _sendLogPayload(abi.encodeWithSignature("log(string)", p0));
    }
```

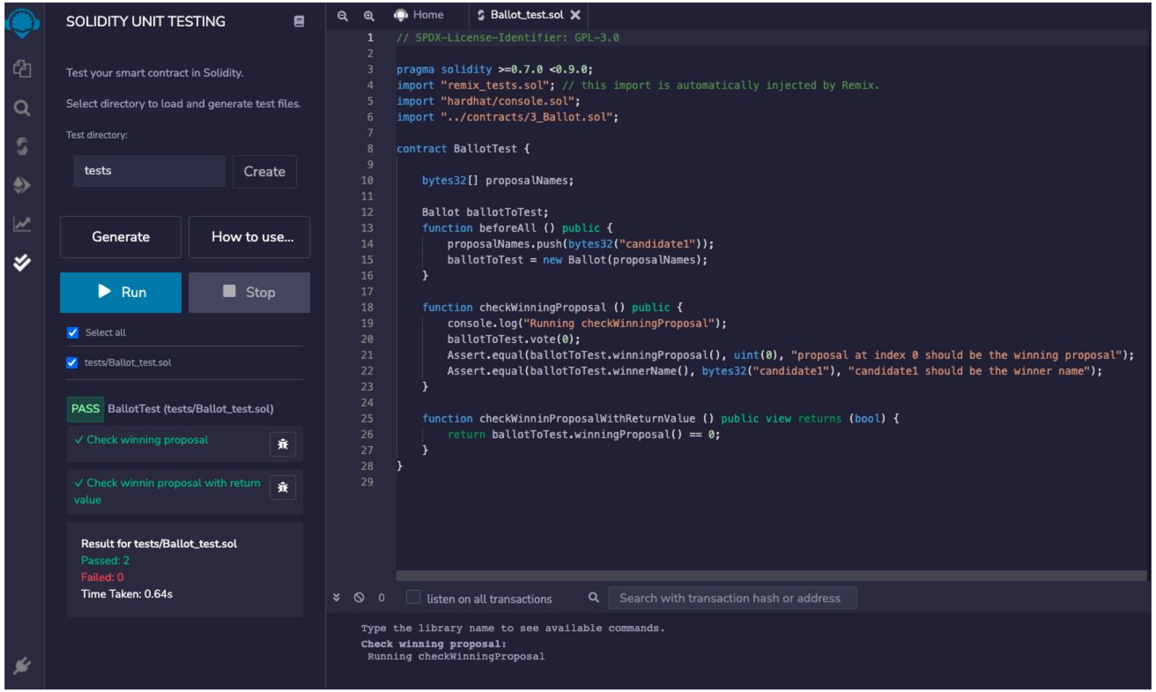
```
• remix

 Type the library name to see available commands.
 [Slither Analysis]: Running...
 [Slither Analysis]: Analysis Completed!! 411 warnings found.
 [Slither Analysis]: Running...
 [Slither Analysis]: Analysis Completed!! 411 warnings found.
```

Smart Contract Unit Testing

Test with Solidity or JS

Smart Contract Unit Testing

- Remix allows execution of unit tests in both Solidity and JS
- **Solidity Unit Testing**: A Remix IDE plugin which allows to run unit tests written in Solidity



Test your smart contract in Solidity.

Select directory to load and generate test files.

Test directory: tests

☑ Select all

☑ tests/Ballot_test.sol

PASS BallotTest (tests/Ballot_test.sol)

✓ Check winning proposal

✓ Check winnin proposal with return value

Result for tests/Ballot_test.sol
Passed: 2
Failed: 0
Time Taken: 0.64s

```solidity
// SPDX-License-Identifier: GPL-3.0

pragma solidity >=0.7.0 <0.9.0;
import "remix_tests.sol"; // this import is automatically injected by Remix.
import "hardhat/console.sol";
import "../contracts/3_Ballot.sol";

contract BallotTest {

    bytes32[] proposalNames;

    Ballot ballotToTest;
    function beforeAll () public {
        proposalNames.push(bytes32("candidate1"));
        ballotToTest = new Ballot(proposalNames);
    }

    function checkWinningProposal () public {
        console.log("Running checkWinningProposal");
        ballotToTest.vote(0);
        Assert.equal(ballotToTest.winningProposal(), uint(0), "proposal at index 0 should be the winning proposal");
        Assert.equal(ballotToTest.winnerName(), bytes32("candidate1"), "candidate1 should be the winner name");
    }

    function checkWinninProposalWithReturnValue () public view returns (bool) {
        return ballotToTest.winningProposal() == 0;
    }
}
```

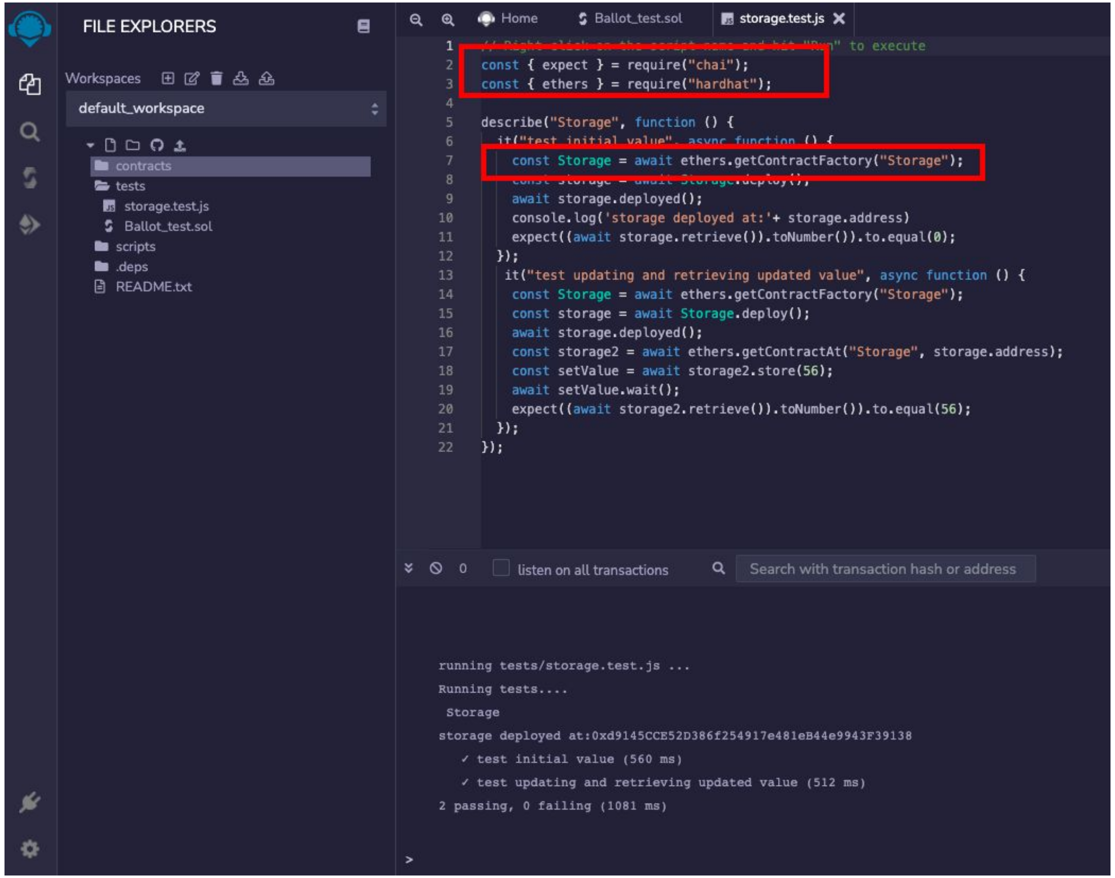
```
Type the library name to see available commands.
Check winning proposal:
 Running checkWinningProposal
```

Smart Contract Unit Testing

- **JS Unit Testing**: Remix supports tests written with **Chai** & executed by **Mocha**

- Put the js tests in a normal js file and run the script

- **Tests from a Hardhat project can be also run** with Remix as it supports hardhat-ethers



```javascript
// Right click on the script name and hit "Run" to execute
const { expect } = require("chai");
const { ethers } = require("hardhat");

describe("Storage", function () {
  it("test initial value", async function () {
    const Storage = await ethers.getContractFactory("Storage");
    const storage = await Storage.deploy();
    await storage.deployed();
    console.log('storage deployed at:'+ storage.address)
    expect((await storage.retrieve()).toNumber()).to.equal(0);
  });
   it("test updating and retrieving updated value", async function () {
    const Storage = await ethers.getContractFactory("Storage");
    const storage = await Storage.deploy();
    await storage.deployed();
    const storage2 = await ethers.getContractAt("Storage", storage.address);
    const setValue = await storage2.store(56);
    await setValue.wait();
    expect((await storage2.retrieve()).toNumber()).to.equal(56);
  });
});
```

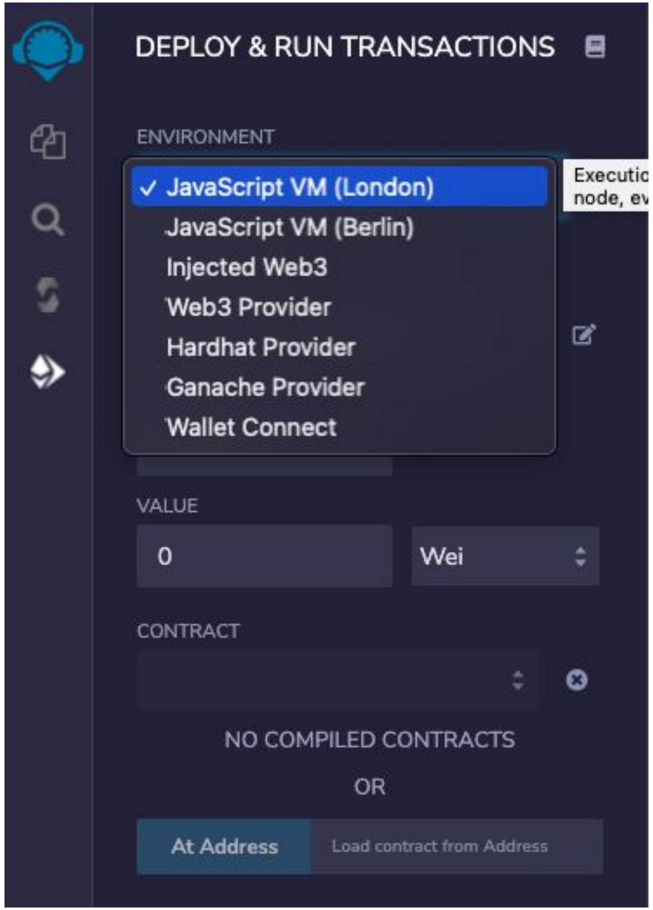
```
running tests/storage.test.js ...
Running tests....
 Storage
storage deployed at:0xd9145CCE52D386f254917e481eB44e9943F39138
   ✓ test initial value (560 ms)
   ✓ test updating and retrieving updated value (512 ms)
2 passing, 0 failing (1081 ms)

>
```

Smart Contract Deployment

Deploy where you like

Smart Contract Deployment

- **Deploy & Run Transactions**: A Remix IDE plugin which allows to deploy a smart contract on different environments


DEPLOY & RUN TRANSACTIONS

ENVIRONMENT

✓ JavaScript VM (London)
JavaScript VM (Berlin)
Injected Web3
Web3 Provider
Hardhat Provider
Ganache Provider
Wallet Connect

Executio
node, ev

VALUE

0 Wei

CONTRACT

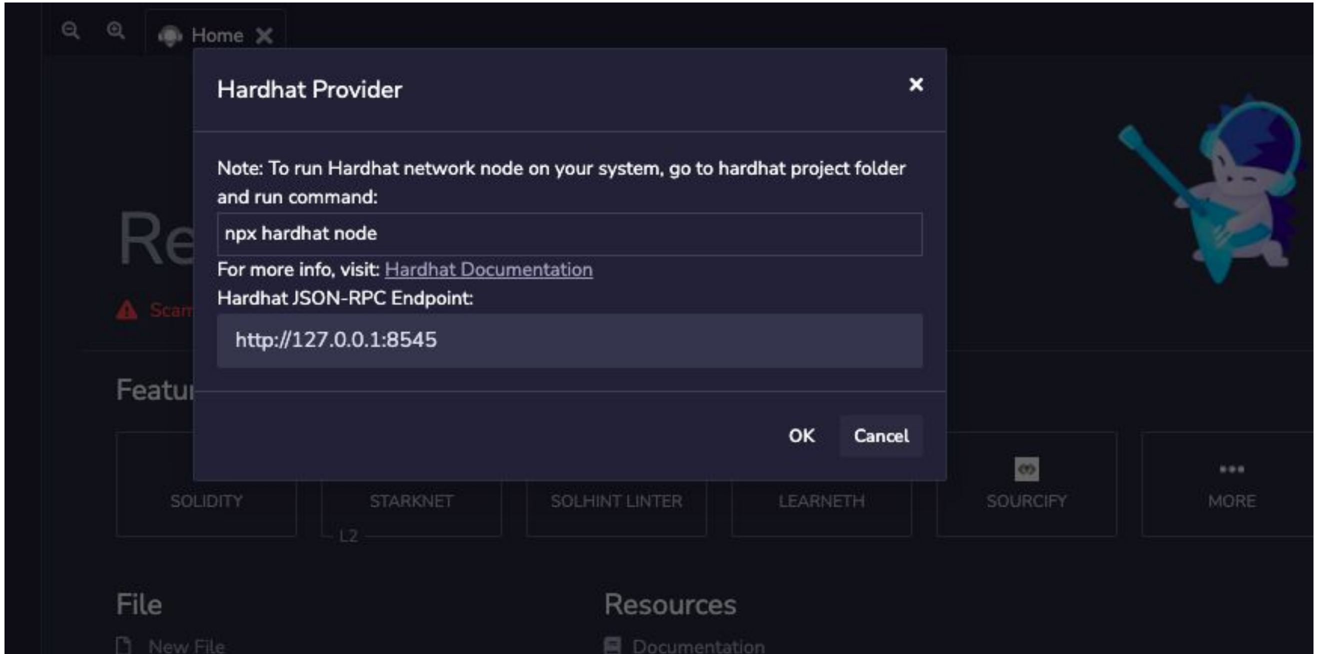
NO COMPILED CONTRACTS

OR

At Address | Load contract from Address

Smart Contract Deployment

- For a Hardhat project, select `Hardhat provider` as environment.
- Connect your local Hardhat node & Deploy contract
- Interact with the deployed contract through Remix Interface


Hardhat Provider

Note: To run Hardhat network node on your system, go to hardhat project folder and run command:

```
npx hardhat node
```

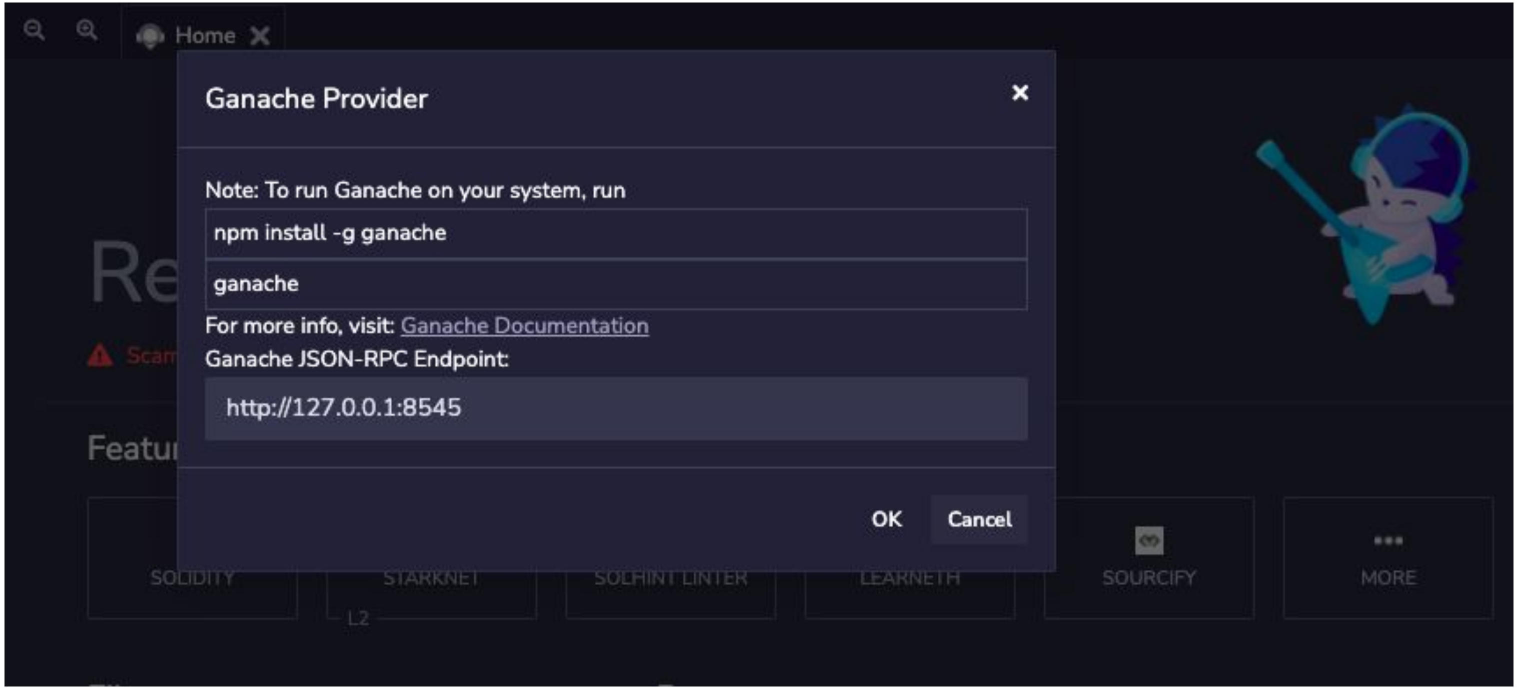
For more info, visit: Hardhat Documentation

Hardhat JSON-RPC Endpoint:

```
http://127.0.0.1:8545
```

Smart Contract Deployment

- Similarly, for a Truffle project, select `Ganache provider` to connect the Ganache
- Deploy & Interact with the contract through Remix Interface



Ganache Provider

Note: To run Ganache on your system, run

```
npm install -g ganache
ganache
```

For more info, visit: Ganache Documentation

Ganache JSON-RPC Endpoint:

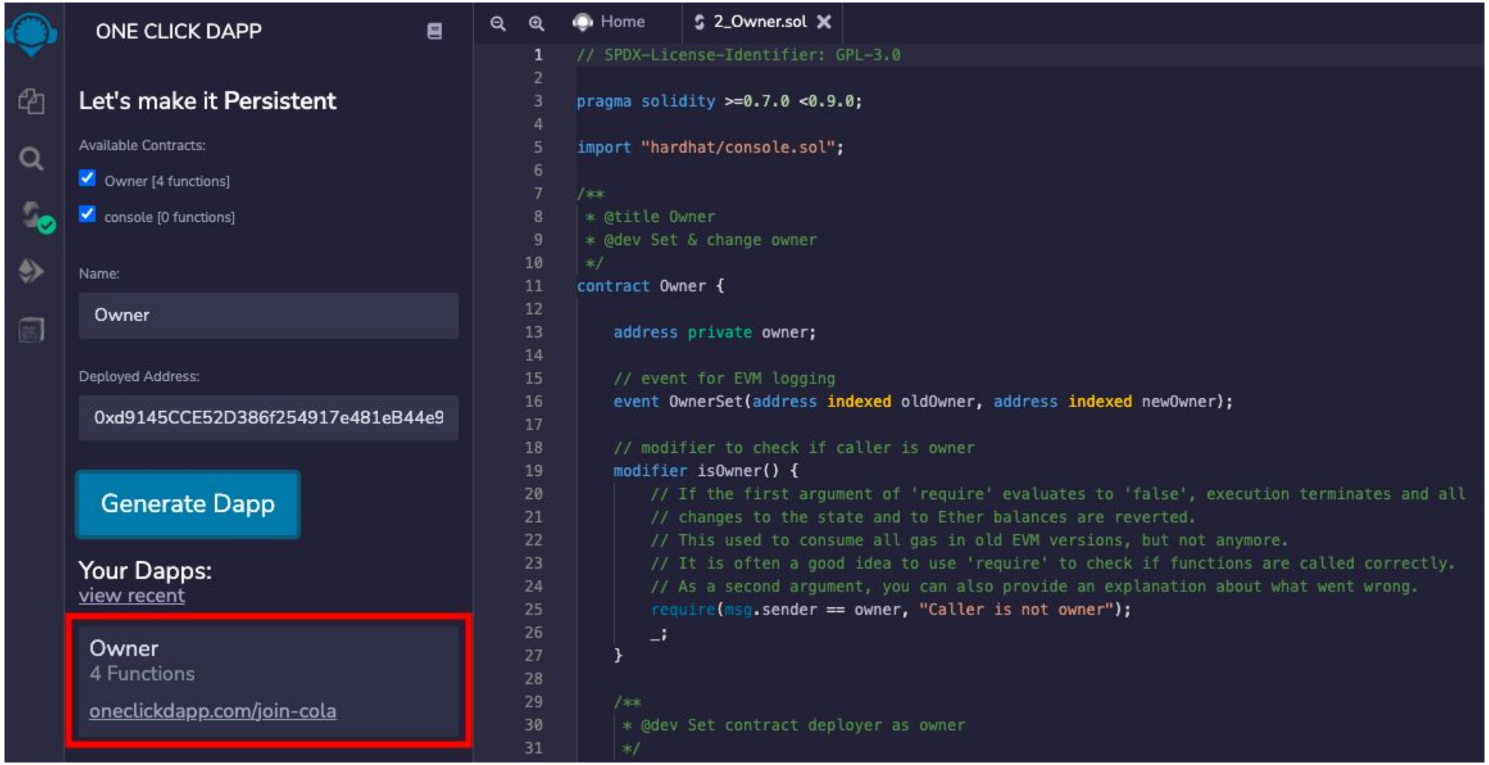
http://127.0.0.1:8545

Make it a DApp…

(quick draft of a DApp)

One Click DApp

- Once a contract is deployed, it can be quickly shared with others as a DApp by using the 'One Click DApp' plugin



Let's make it Persistent

Available Contracts:

- [✓] Owner [4 functions]
- [✓] console [0 functions]

Name: Owner

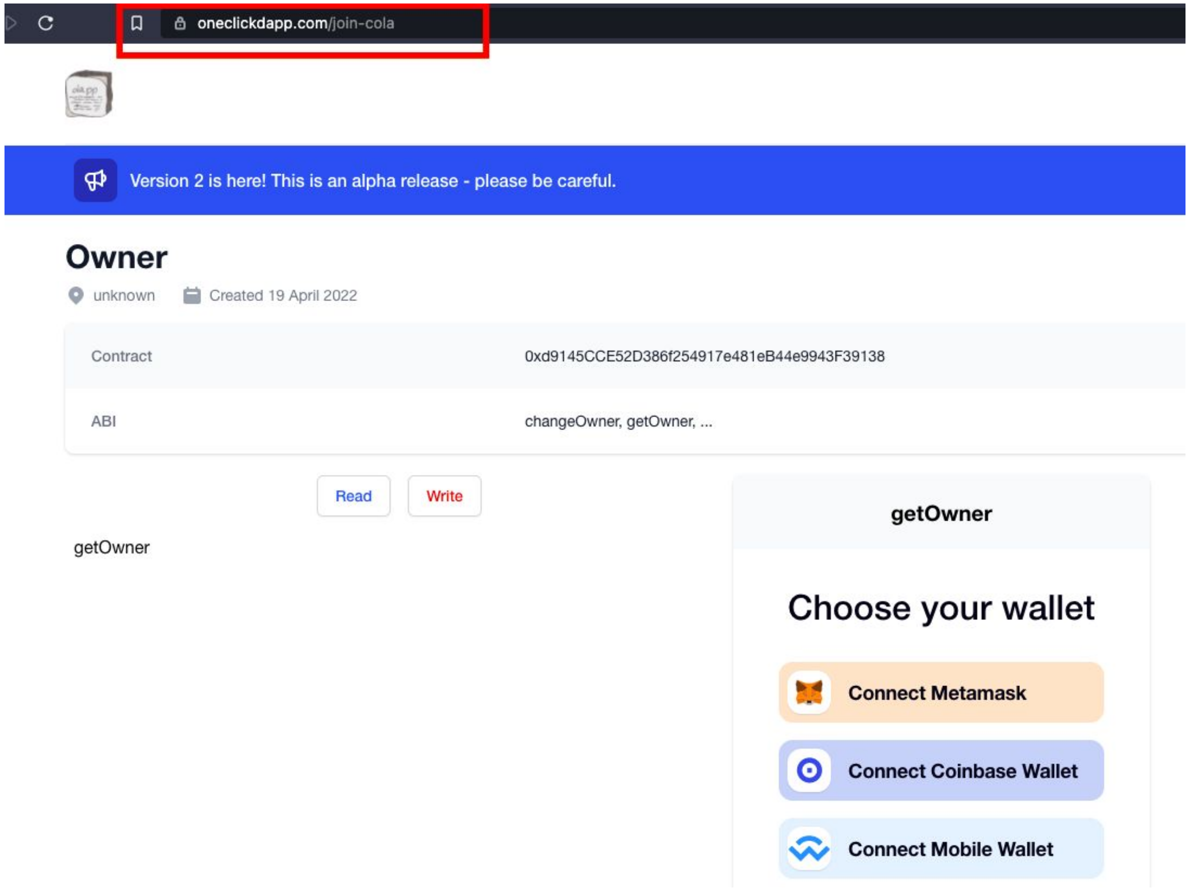
Deployed Address: 0xd9145CCE52D386f254917e481eB44e9

Your Dapps:
view recent

Owner
4 Functions
oneclickdapp.com/join-cola

```
// SPDX-License-Identifier: GPL-3.0

pragma solidity >=0.7.0 <0.9.0;

import "hardhat/console.sol";

/**
 * @title Owner
 * @dev Set & change owner
 */
contract Owner {

    address private owner;

    // event for EVM logging
    event OwnerSet(address indexed oldOwner, address indexed newOwner);

    // modifier to check if caller is owner
    modifier isOwner() {
        // If the first argument of 'require' evaluates to 'false', execution terminates and all
        // changes to the state and to Ether balances are reverted.
        // This used to consume all gas in old EVM versions, but not anymore.
        // It is often a good idea to use 'require' to check if functions are called correctly.
        // As a second argument, you can also provide an explanation about what went wrong.
        require(msg.sender == owner, "Caller is not owner");
        _;
    }

    /**
     * @dev Set contract deployer as owner
     */
```

One Click DApp



oneclickdapp.com/join-cola

Version 2 is here! This is an alpha release - please be careful.

Owner

unknown   Created 19 April 2022

Contract	0xd9145CCE52D386f254917e481eB44e9943F39138
ABI	changeOwner, getOwner, ...

Read   Write

getOwner

getOwner

Choose your wallet

Connect Metamask

Connect Coinbase Wallet

Connect Mobile Wallet

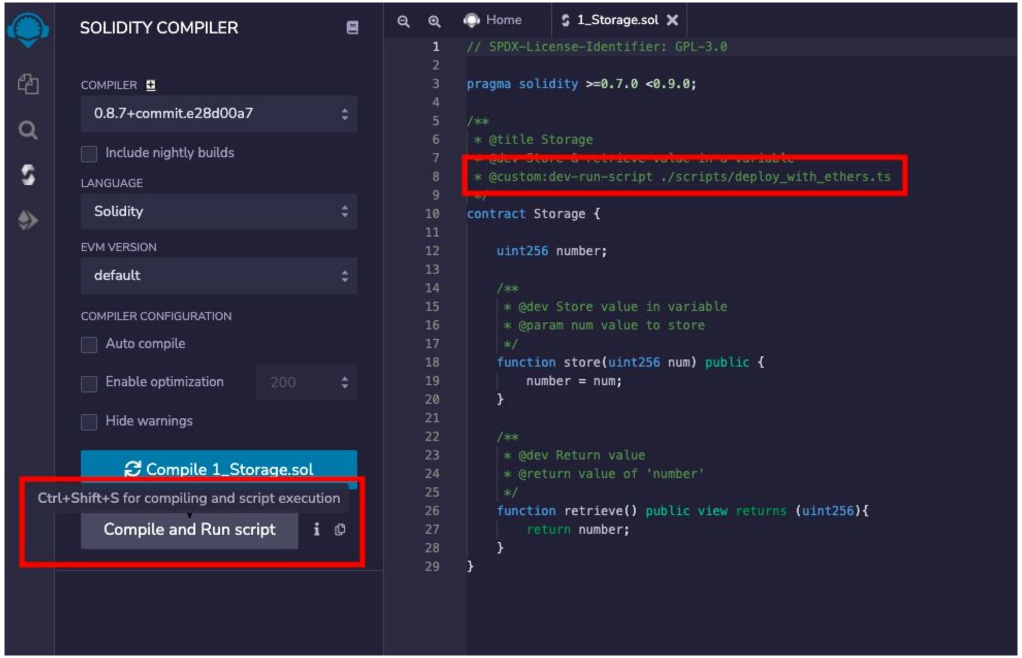
Let’s mix it…

Bind a custom script to contract compilation

In one shot, compile then run a script

- In the contract, the script is specified using the Natspec tag i.e. **@custom:dev-run-script** with the script's location

- Click the 'Compile and Run script' button or just use keyboard shortcut: **CTRL + SHIFT + S**



COMPILER
0.8.7+commit.e28d00a7
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
default
COMPILER CONFIGURATION
Auto compile
Enable optimization 200
Hide warnings

Compile 1_Storage.sol

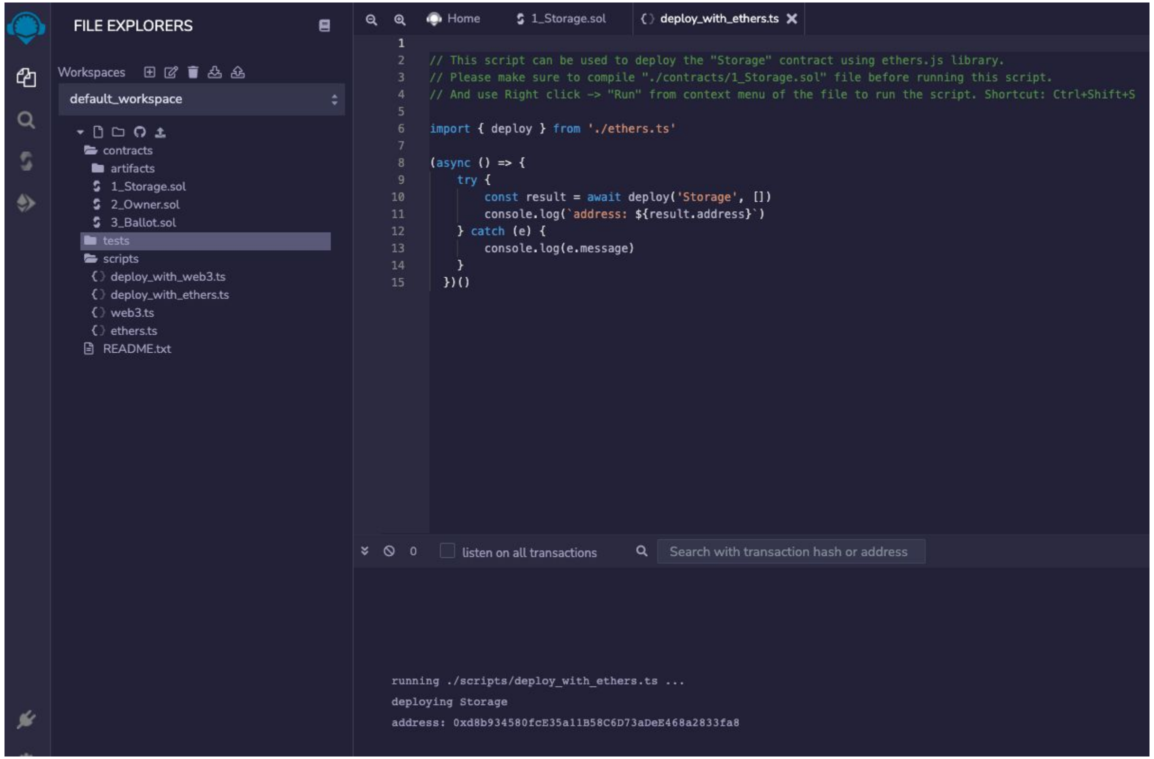
Ctrl+Shift+S for compiling and script execution

Compile and Run script

```
// SPDX-License-Identifier: GPL-3.0

pragma solidity >=0.7.0 <0.9.0;

/**
 * @title Storage
 * @dev Store & retrieve value in a variable
 * @custom:dev-run-script ./scripts/deploy_with_ethers.ts
 */
contract Storage {

    uint256 number;

    /**
     * @dev Store value in variable
     * @param num value to store
     */
    function store(uint256 num) public {
        number = num;
    }

    /**
     * @dev Return value
     * @return value of 'number'
     */
    function retrieve() public view returns (uint256){
        return number;
    }
}
```

Compile & Run Script

- A script contains actions:
    - e.g. deploy a contract, run the js unit tests or anything else



```typescript

// This script can be used to deploy the "Storage" contract using ethers.js library.
// Please make sure to compile "./contracts/1_Storage.sol" file before running this script.
// And use Right click -> "Run" from context menu of the file to run the script. Shortcut: Ctrl+Shift+S

import { deploy } from './ethers.ts'

(async () => {
    try {
        const result = await deploy('Storage', [])
        console.log(`address: ${result.address}`)
    } catch (e) {
        console.log(e.message)
    }
  })()
```

```
running ./scripts/deploy_with_ethers.ts ...
deploying Storage
address: 0xd8b934580fcE35a11B58C6D73aDeE468a2833fa8
```

What’s next

More integrations from the ecosystem:

- Foundry
- Brownie
- More static analysis tools
- ……….

Any feedback or suggestions are always welcome.

Thanks

Aniket @AniketEngg

Remix Project

Github: ethereum/remix-project
Gitter: ethereum/remix
Twitter: @EthereumRemix
Medium: medium.com/remix-ide