

EOSFACTORY

SMART-CONTRACT DEVELOPMENT & TESTING IDE

Presented by

TOKENIKA



WHAT IS EOSFACTORY?

A Python-based EOS smart-contract development & testing framework

WHY IS EOSFACTORY NEEDED?

Automation is essential for efficient unit testing

WHAT ARE THE MAIN FEATURES?

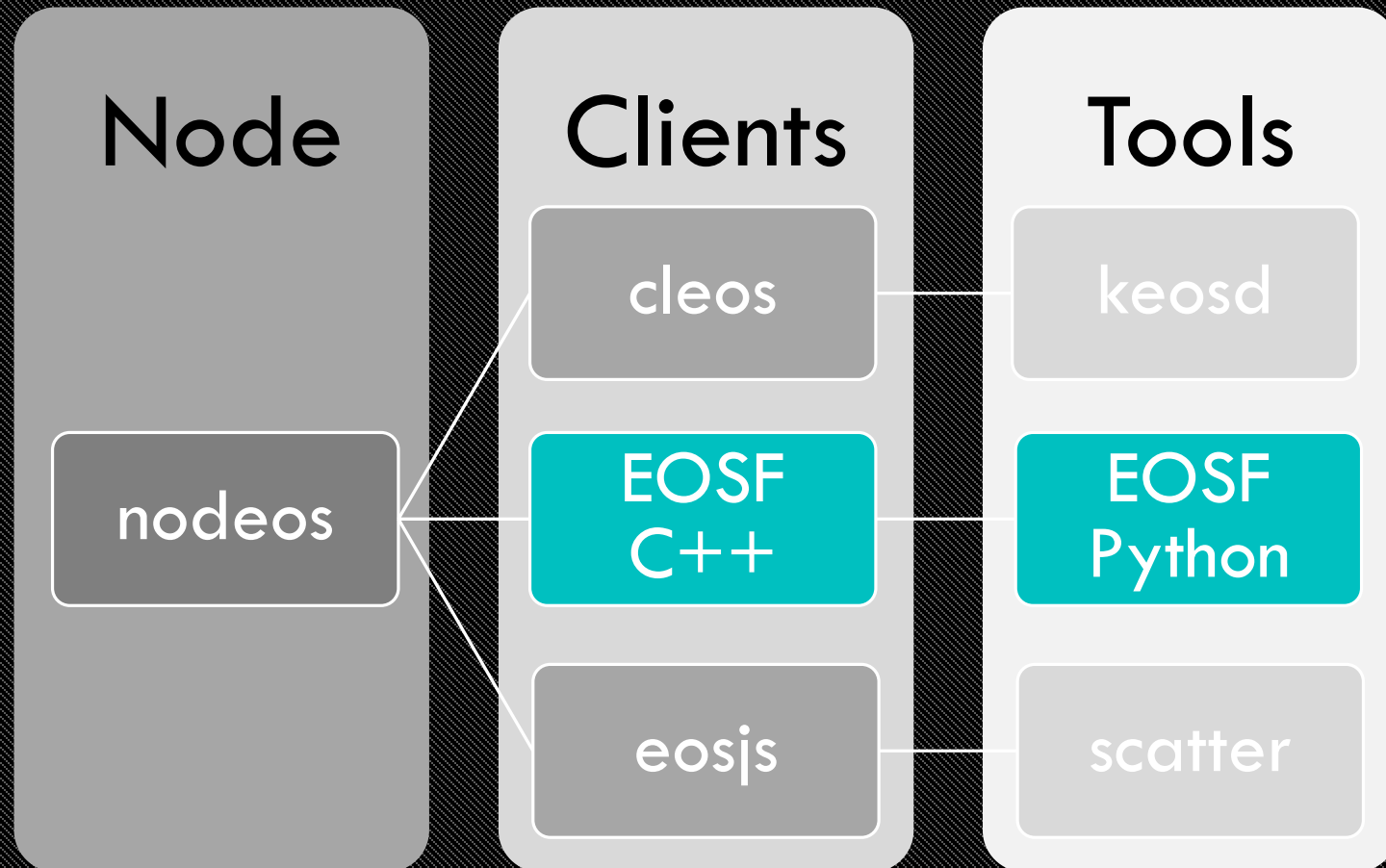
Object-oriented, Simple syntax, Real testnet, Truly cross-platform



“ An **integrated development environment** (IDE) is a software application that provides comprehensive facilities to computer programmers for software development. ”

An IDE normally consists of **a source code editor, build automation tools, and a debugger.**

EOSIO INFRASTRUCTURE



EOSFACTORY COMPONENTS

Demos, Templates,
Unit Tests

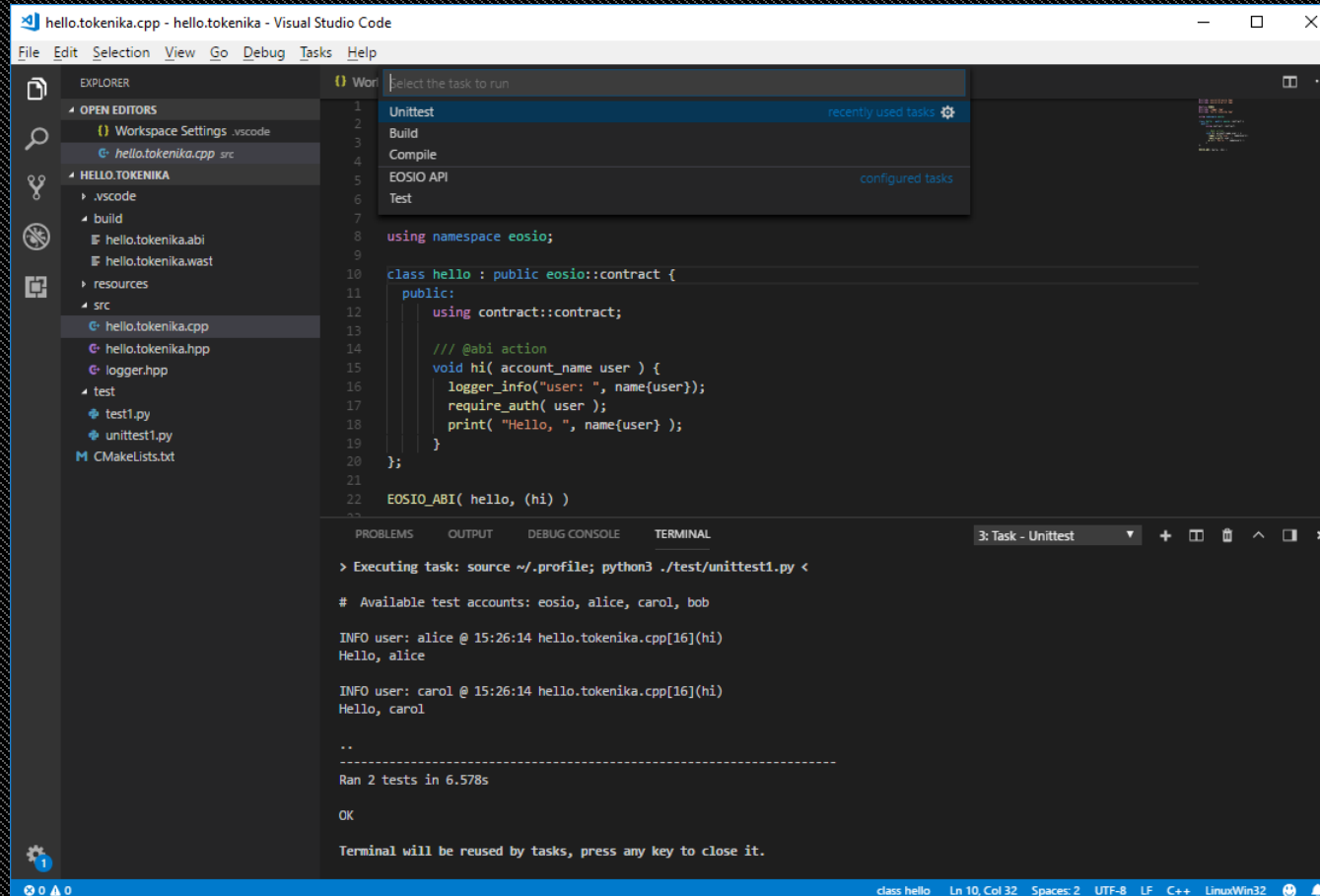
Python CLI

C++

MAIN FEATURES IN V1.0

1. Standard unit-testing framework
2. Support for user-defined workspaces
3. Support for debugging
4. Preliminary code verification
5. Visual Studio Code integration

EOSFACTORY UNIT TESTS



The screenshot shows the Visual Studio Code interface with a C++ file named `hello.tokenika.cpp` open. The file contains an EOSIO contract class `hello` with a `hi` action. The terminal window shows the execution of the unit tests, displaying the available test accounts and the results of the tests.

```
hello.tokenika.cpp - hello.tokenika - Visual Studio Code
File Edit Selection View Go Debug Tasks Help

EXPLORER
  WORKSPACE SETTINGS .vscode
  hello.tokenika.cpp src
  HELLO.TOKENIKA
    .vscode
    build
    hello.tokenika.abi
    hello.tokenika.wasm
    resources
    src
      hello.tokenika.cpp
      hello.tokenika.hpp
      logger.hpp
    test
      test1.py
      unittest1.py
    CMakeLists.txt

1 Select the task to run
2 Unittest recently used task
3 Build
4 Compile
5 EOSIO API configured task
6 Test

7
8 using namespace eosio;
9
10 class hello : public eosio::contract {
11 public:
12     using contract::contract;
13
14     /// @abi action
15     void hi( account_name user ) {
16         logger_info("user: ", name{user});
17         require_auth( user );
18         print( "Hello, ", name{user} );
19     }
20 };
21
22 EOSIO_ABI( hello, (hi) )

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
3: Task - Unittest

> Executing task: source ~/.profile; python3 ./test/unittest1.py <

# Available test accounts: eosio, alice, carol, bob

INFO user: alice @ 15:26:14 hello.tokenika.cpp[16](hi)
Hello, alice

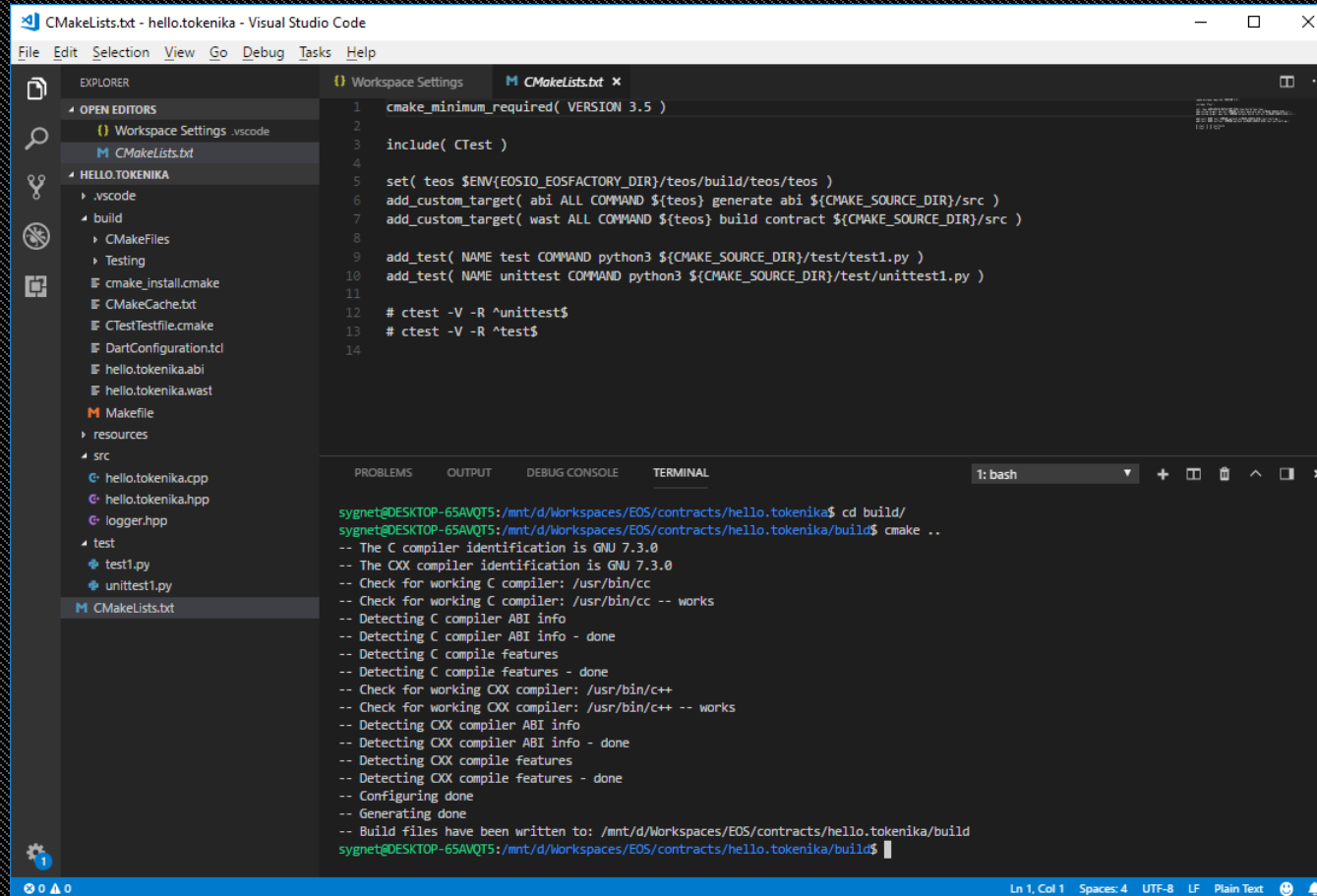
INFO user: carol @ 15:26:14 hello.tokenika.cpp[16](hi)
Hello, carol

..
-----
Ran 2 tests in 6.578s

OK

Terminal will be reused by tasks, press any key to close it.
```

EOSFACTORY CMAKE SUPPORT



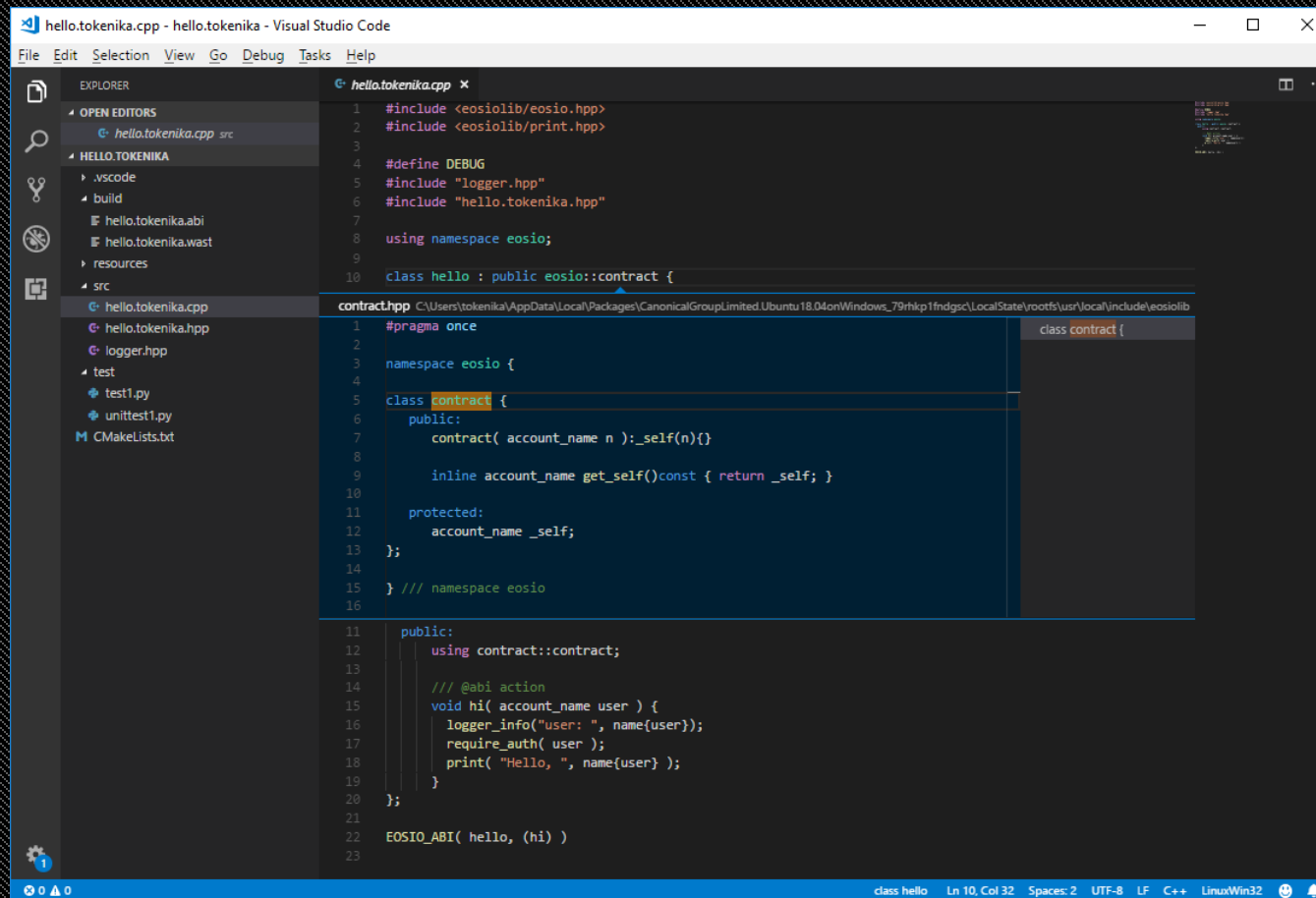
The screenshot displays the Visual Studio Code interface for a project named 'hello.tokenika'. The Explorer sidebar on the left shows the project structure, including files like 'CMakeLists.txt', 'cmake_install.cmake', 'CMakeCache.txt', 'CTestTestfile.cmake', 'DartConfiguration.tcl', 'hello.tokenika.abi', 'hello.tokenika.wasm', 'Makefile', 'resources', 'src', 'hello.tokenika.cpp', 'hello.tokenika.hpp', 'logger.hpp', 'test', 'test1.py', 'unittest1.py', and 'CMakeLists.txt'. The main editor window shows the 'CMakeLists.txt' file with the following content:

```
1 cmake_minimum_required( VERSION 3.5 )
2
3 include( CTest )
4
5 set( teos $ENV{EOSIO_EOSFACTORY_DIR}/teos/build/teos/teos )
6 add_custom_target( abi ALL COMMAND ${teos} generate abi ${CMAKE_SOURCE_DIR}/src )
7 add_custom_target( wast ALL COMMAND ${teos} build contract ${CMAKE_SOURCE_DIR}/src )
8
9 add_test( NAME test COMMAND python3 ${CMAKE_SOURCE_DIR}/test/test1.py )
10 add_test( NAME unittest COMMAND python3 ${CMAKE_SOURCE_DIR}/test/unittest1.py )
11
12 # ctest -V -R ^unittest$
13 # ctest -V -R ^test$
14
```

The terminal window at the bottom shows the output of the 'cmake' command executed in the 'build' directory:

```
sygnet@DESKTOP-65AVQTS:/mnt/d/Workspaces/EOS/contracts/hello.tokenika$ cd build/
sygnet@DESKTOP-65AVQTS:/mnt/d/Workspaces/EOS/contracts/hello.tokenika/build$ cmake ..
-- The C compiler identification is GNU 7.3.0
-- The CXX compiler identification is GNU 7.3.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: /mnt/d/Workspaces/EOS/contracts/hello.tokenika/build
sygnet@DESKTOP-65AVQTS:/mnt/d/Workspaces/EOS/contracts/hello.tokenika/build$
```


EOSFACTORY INTELLISENSE

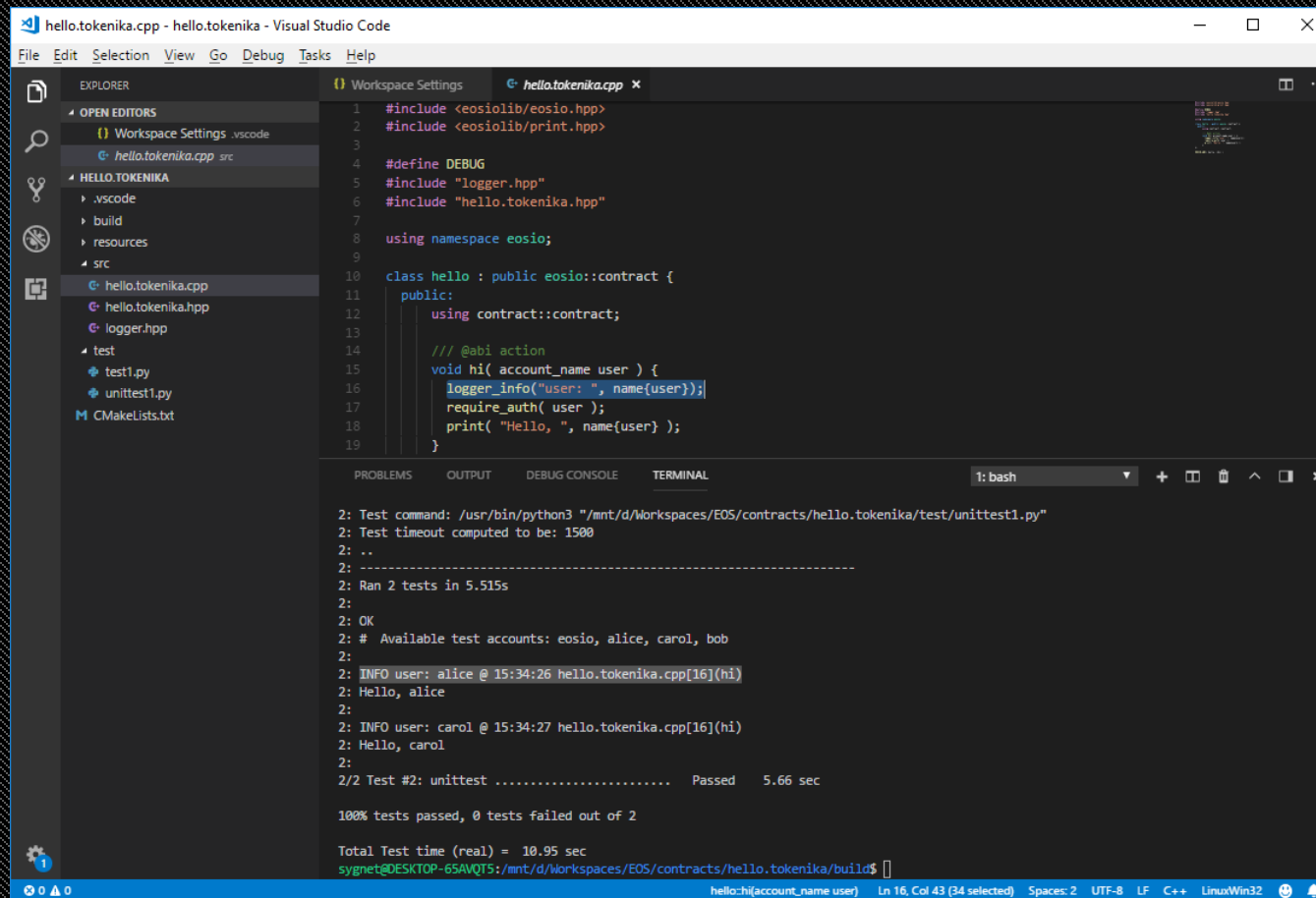


```
hello.tokenika.cpp - hello.tokenika - Visual Studio Code
File Edit Selection View Go Debug Tasks Help

EXPLORER
  OPEN EDITORS
    hello.tokenika.cpp src
  HELLO.TOKENIKA
    .vscode
    build
    hello.tokenika.abi
    hello.tokenika.wasm
    resources
    src
      hello.tokenika.cpp
      hello.tokenika.hpp
      logger.hpp
      test
      test1.py
      unittest1.py
      CMakeLists.txt

hello.tokenika.cpp
1 #include <eosiolib/eosio.hpp>
2 #include <eosiolib/print.hpp>
3
4 #define DEBUG
5 #include "logger.hpp"
6 #include "hello.tokenika.hpp"
7
8 using namespace eosio;
9
10 class hello : public eosio::contract {
11
12 public:
13     contract( account_name n ): _self(n){}
14
15     inline account_name get_self()const { return _self; }
16
17 protected:
18     account_name _self;
19 };
20
21 } // namespace eosio
22
23 public:
24     using contract::contract;
25
26     /// @abi action
27     void hi( account_name user ) {
28         logger_info("user: ", name(user));
29         require_auth( user );
30         print( "Hello, ", name(user) );
31     }
32 };
33
34 EOSIO_ABI( hello, (hi) )
```

EOSFACTORY DEBUGGING



```
hello.tokenika.cpp - hello.tokenika - Visual Studio Code
File Edit Selection View Go Debug Tasks Help

EXPLORER
  OPEN EDITORS
    Workspace Settings .vscode
    hello.tokenika.cpp src
  HELLO.TOKENIKA
    .vscode
    build
    resources
    src
    hello.tokenika.cpp
    hello.tokenika.hpp
    logger.hpp
    test
    test1.py
    unittest1.py
    CMakeLists.txt

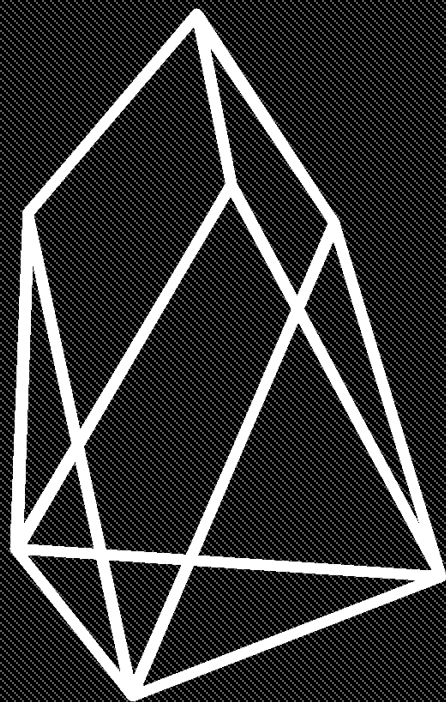
Workspace Settings hello.tokenika.cpp x
1 #include <eosiolib/eosio.hpp>
2 #include <eosiolib/print.hpp>
3
4 #define DEBUG
5 #include "logger.hpp"
6 #include "hello.tokenika.hpp"
7
8 using namespace eosio;
9
10 class hello : public eosio::contract {
11 public:
12     using contract::contract;
13
14     /// @abi action
15     void hi( account_name user ) {
16         logger_info("user: ", name{user});
17         require_auth( user );
18         print( "Hello, ", name{user} );
19     }
20 }

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: bash

2: Test command: /usr/bin/python3 "/mnt/d/Workspaces/EOS/contracts/hello.tokenika/test/unittest1.py"
2: Test timeout computed to be: 1500
2: ..
2: -----
2: Ran 2 tests in 5.515s
2:
2: OK
2: # Available test accounts: eosio, alice, carol, bob
2:
2: INFO user: alice @ 15:34:26 hello.tokenika.cpp[16](hi)
2: Hello, alice
2:
2: INFO user: carol @ 15:34:27 hello.tokenika.cpp[16](hi)
2: Hello, carol
2:
2/2 Test #2: unittest ..... Passed 5.66 sec

100% tests passed, 0 tests failed out of 2

Total Test time (real) = 10.95 sec
sygnet@DESKTOP-65AVQT5:/mnt/d/Workspaces/EOS/contracts/hello.tokenika/build$
```



THANK YOU

ANY QUESTIONS?

EOSFACTORY

eosfactory.io

github.com/tokenika/eosfactory

Presented by

TOKENIKA

tokenika.io

contact@tokenika.io