
athena-ide-atom

Jul 14, 2020

1	Getting Started	3
1.1	Install	3
1.2	Open	4
1.3	Auto Complete	5
1.4	Lint	6
2	Node	9
2.1	New	9
2.2	Remove	10
3	Account	13
3.1	New	13
3.2	Import	14
3.3	Export	17
3.4	Remove	20
4	Contract	23
4.1	Compile	23
4.2	Deploy	24
4.3	Import & Remove	28
4.4	Execute	30
4.5	Query	35
4.6	Varargs	37
4.7	Redeploy (private mode only)	41

Welcome to the athena ide atom wiki. This is an IDE (Integrated Development Environment) for [aergo smart contract](#). It can compile, deploy and execute smart contract with an account. It also can create, import and export account.

CHAPTER 1

Getting Started

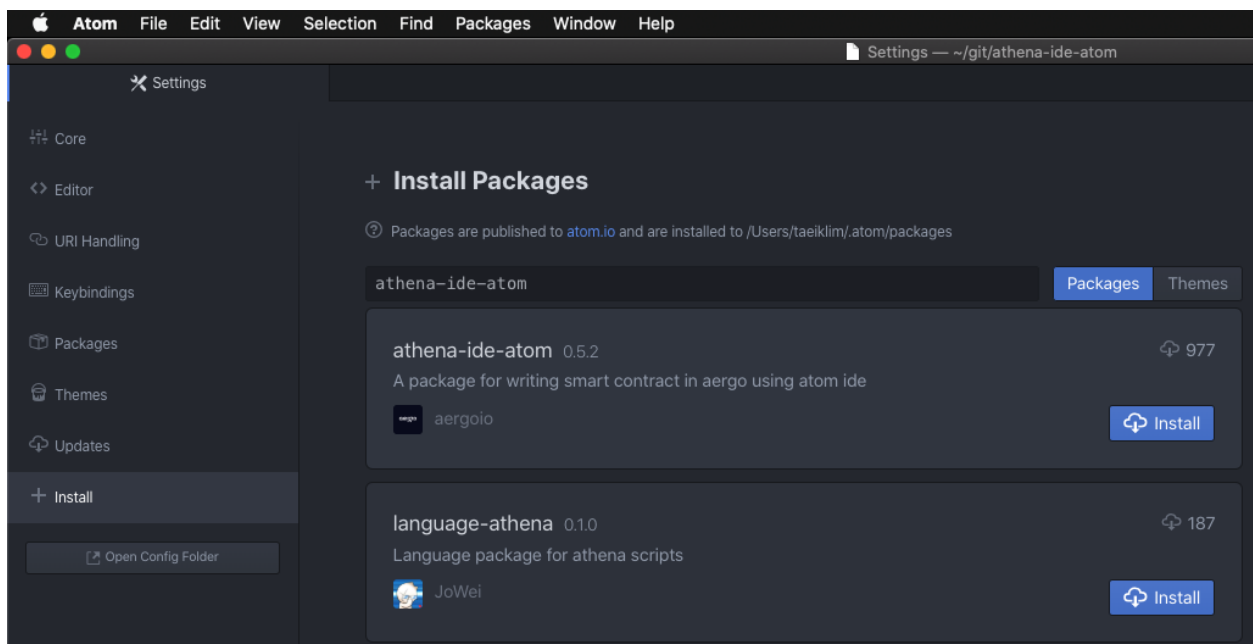
1.1 Install

Athena ide atom is based on [atom](#). You have to install it first.

1.1.1 By apm

Athena ide atom is available in [apm](#) (atom package manager). You can install with it.

Atom -> Preferences -> Install



You can install using cli. For windows user, you can use [git bash](#) for cli environment.

```
> apm install athena-ide-atom
```

1.1.2 By installer

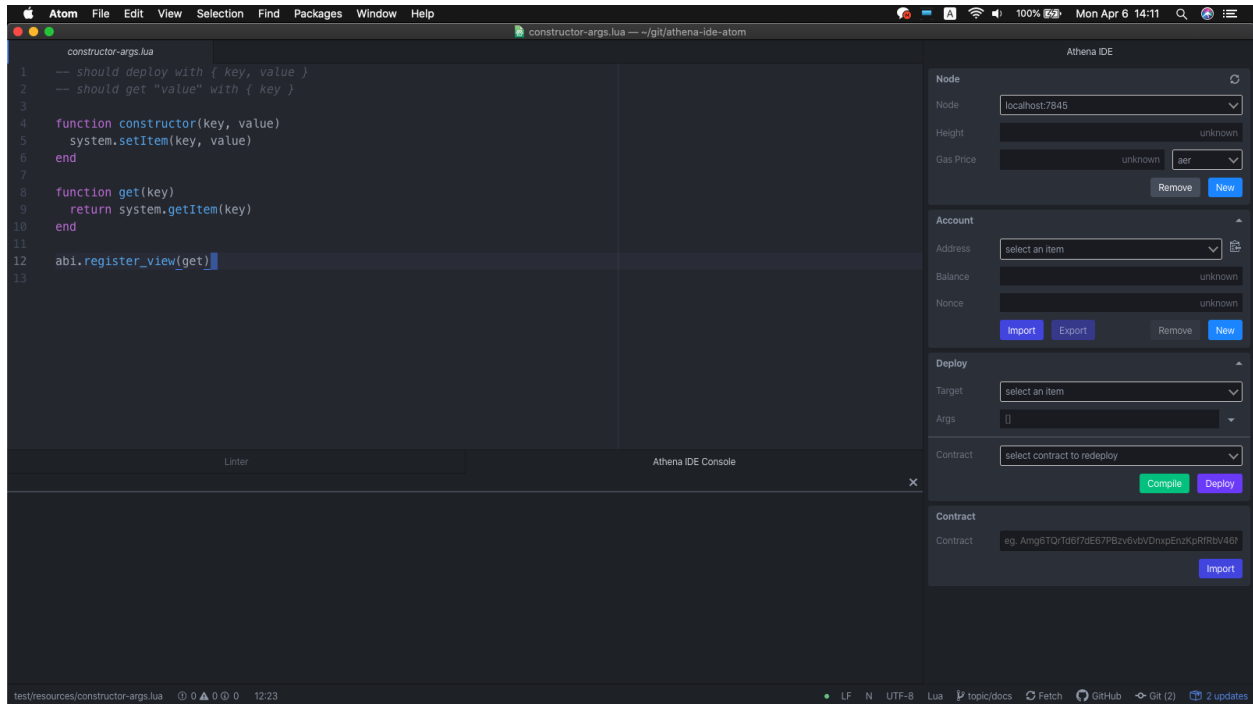
Installing with apm may be slow on slow network status. So we provide custom installer in releases. Download athena-ide-atom-x.x.x-installer.bin from [releases](#) and run in cli. For windows user, you can use [git bash](#) for cli environment.

```
> ./athena-ide-atom-x.x.x-installer.bin
```

1.2 Open

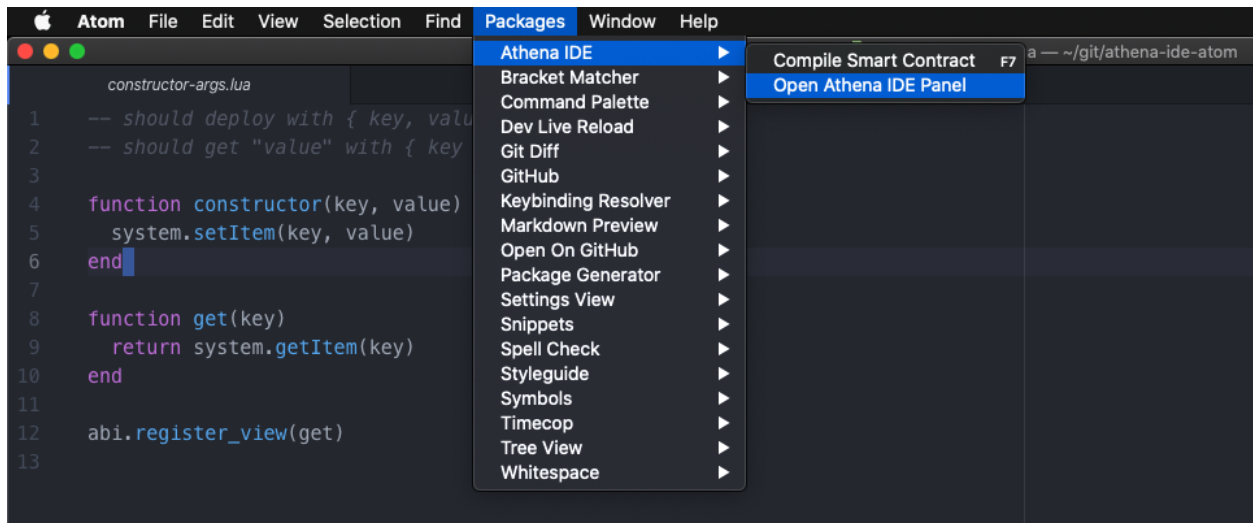
1.2.1 Using shortcut

- Mac : Option + Shift + L
- Windows : Alt + Shift + L



1.2.2 Using menu bar

Packages -> Athena IDE -> Open Athena IDE Panel



1.3 Auto Complete

Athena ide atom provides some basic autocomplete for lua.

```
3
4  function constructor(key, value)
5      system.setItem(key, value)
6  end
7
8  function get(key)
9      system.|
10     re m  getSender()      function
11     end  m  getBlockheight()  function
12
13     abi. m  getTxhash()      function
14
15     m  getTimestamp()      function
16
17     m  getContractID()     function
18
19     m  setItem(key, value)  function
20
21     m  getItem(key)        function
22
23     m  getAmount()         function
24
25     m  getCreator()        function
26
27     m  getOrigin()         function
```

1.4 Lint

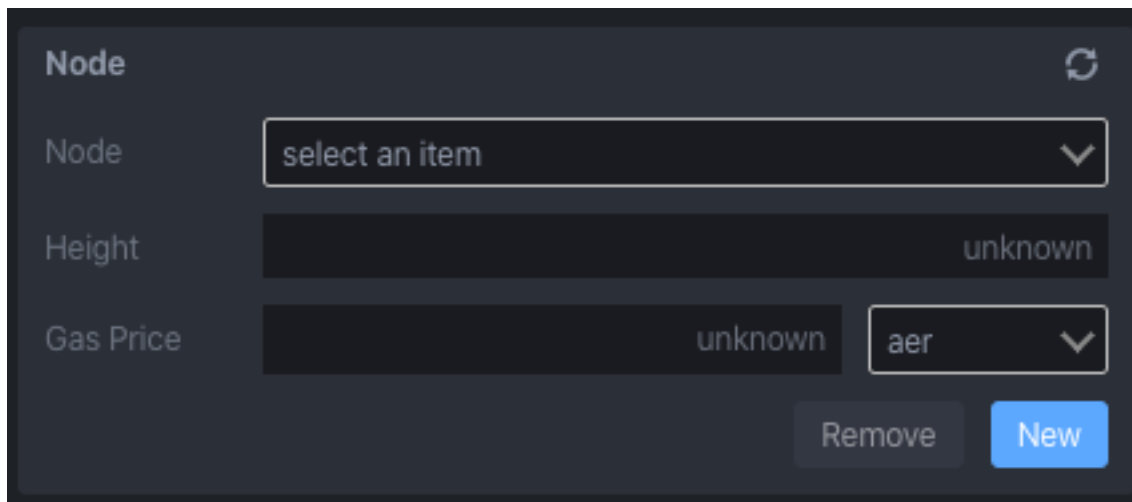
Athena ide atom provides basic lint for lua.

```
3
4  function constructor(key, value)
5      system.setItem(key, value)
6  end
7
8  function get(key)
9      system.
10     • return system.getItem(key)
11     end [10:2] <name> expected near 'return'
12
13     abi.register_view(get)
14
```


You can configure an aergo node whose smart contract interacting with. You can see height and gas price of it. Default configuration is localhost:7845.

2.1 New

Click an new button



The screenshot shows a dark-themed dialog box titled "Node" with a refresh icon in the top right corner. Inside the dialog, there are three rows of configuration fields:

- The first row is labeled "Node" and contains a dropdown menu with the text "select an item" and a downward arrow.
- The second row is labeled "Height" and contains a text field with the value "unknown".
- The third row is labeled "Gas Price" and contains a text field with the value "unknown" and a dropdown menu with the value "aer" and a downward arrow.

At the bottom right of the dialog, there are two buttons: a grey "Remove" button and a blue "New" button.

Enter node endpoint and click ok

New Node

Are you sure you want to add node?

To import, enter node information

Node

You can see height and gas price of it

Node ↻

Node ▼

Height

Gas Price ▼

2.2 Remove

Click a remove button

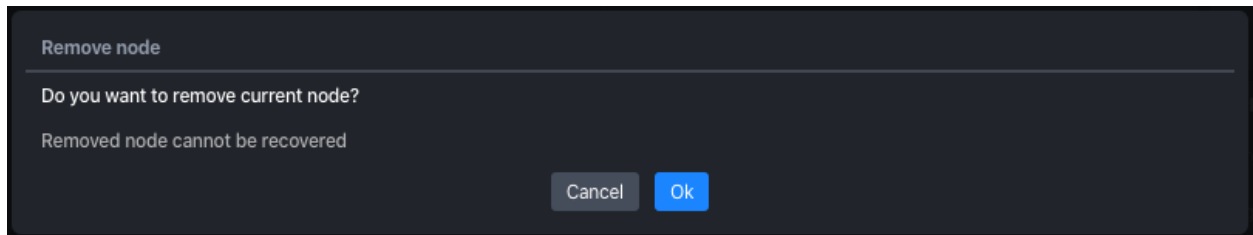
Node ↻

Node ▼

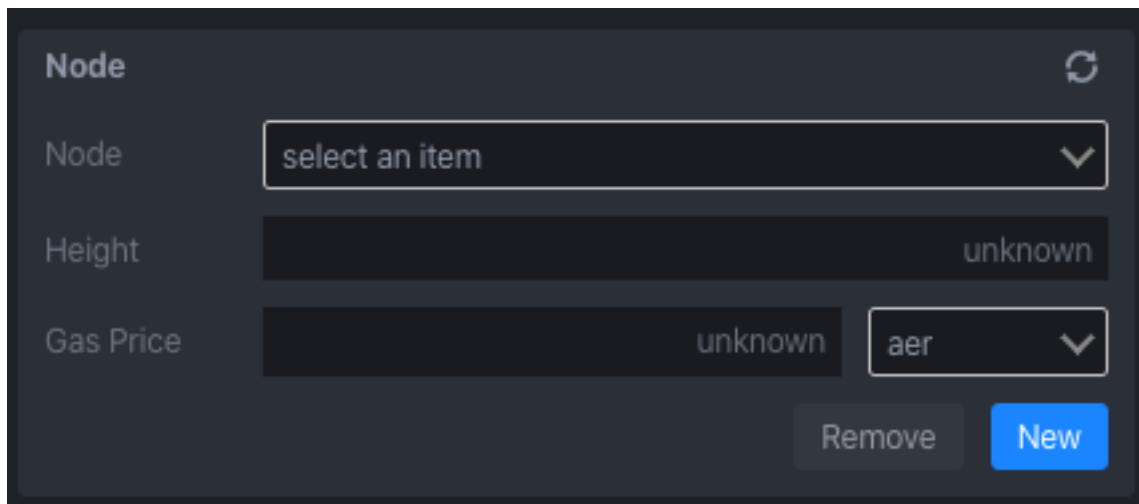
Height

Gas Price ▼

Click ok button



Node removed from list



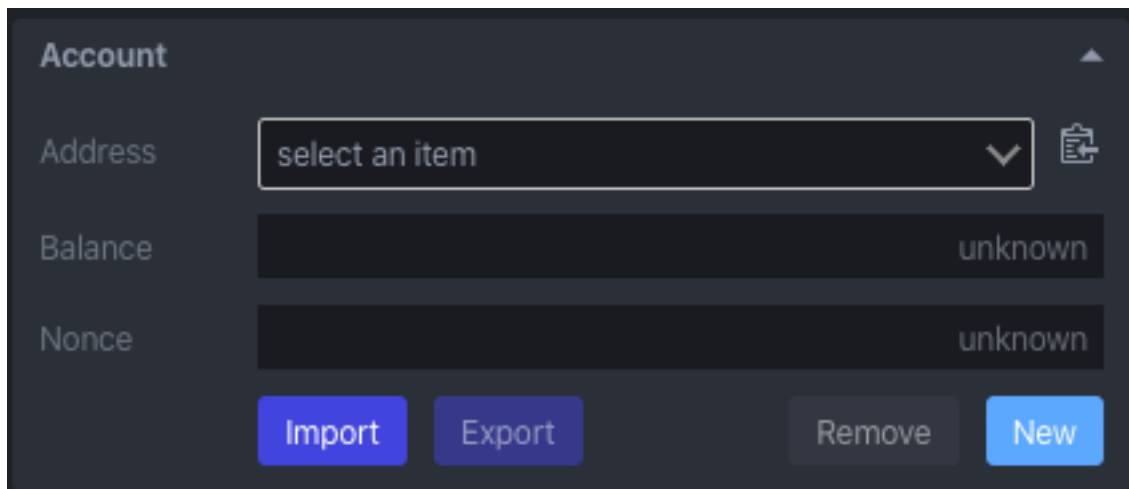
CHAPTER 3

Account


You can configure an account which is deploying, executing and querying smart contract. An account have to have some aergo token to make transaction.

3.1 New

Click an new button



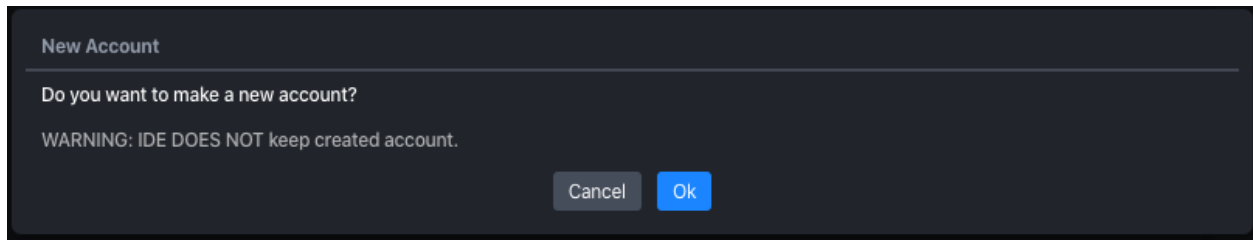
Account

Address 

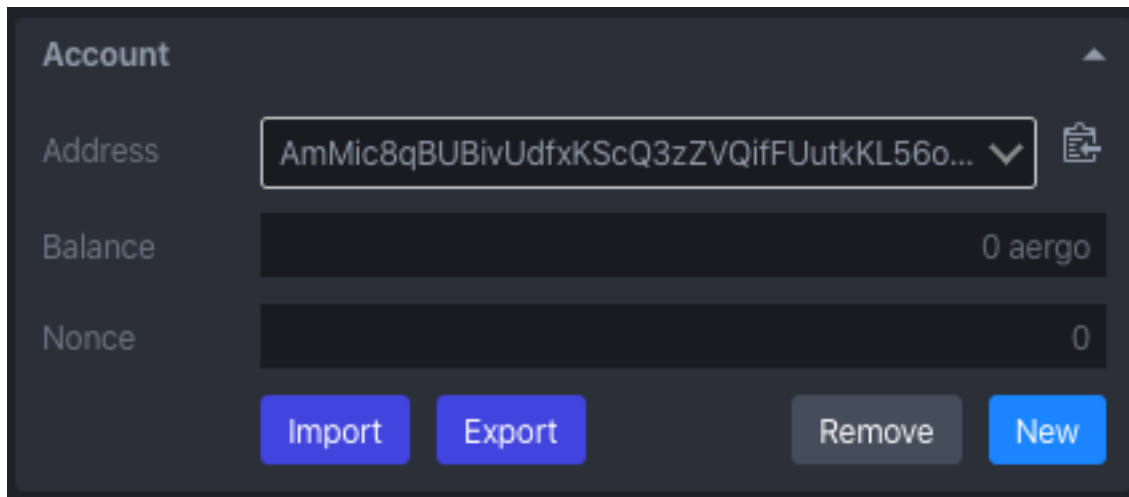
Balance

Nonce

Click a ok button



Account created

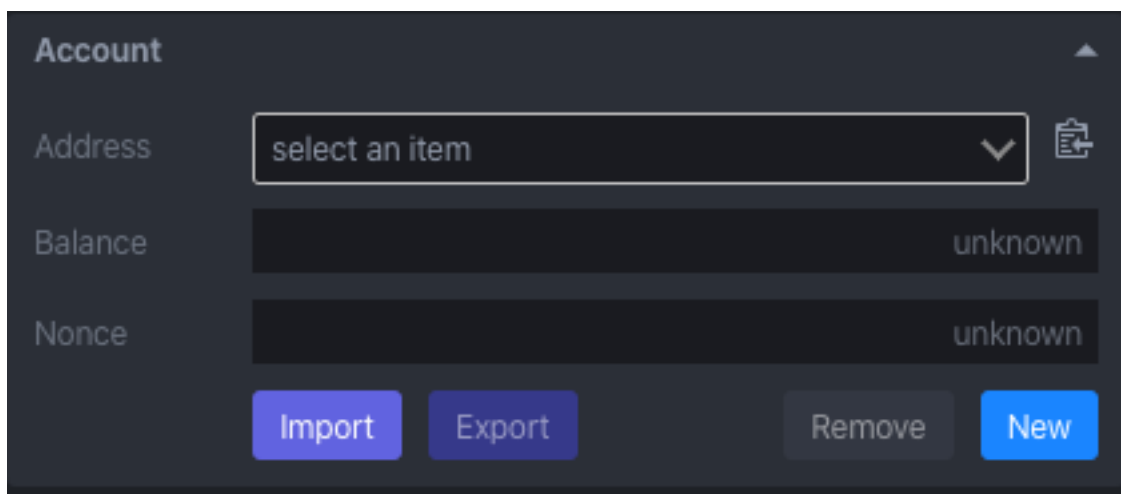


3.2 Import

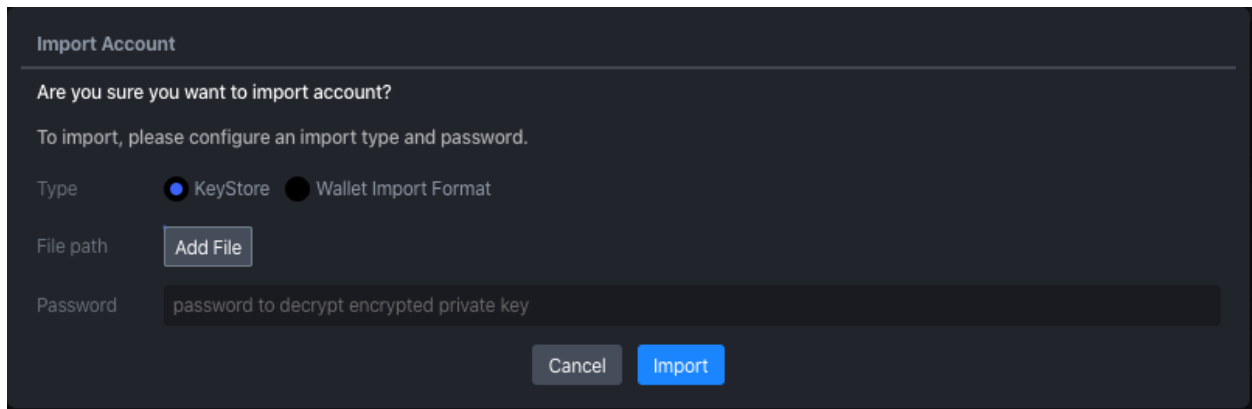
You can import account with an aergo specific keystore format or wallet import format. For details of keystore format, see [aergo keystore proposal](#).

3.2.1 KeyStore

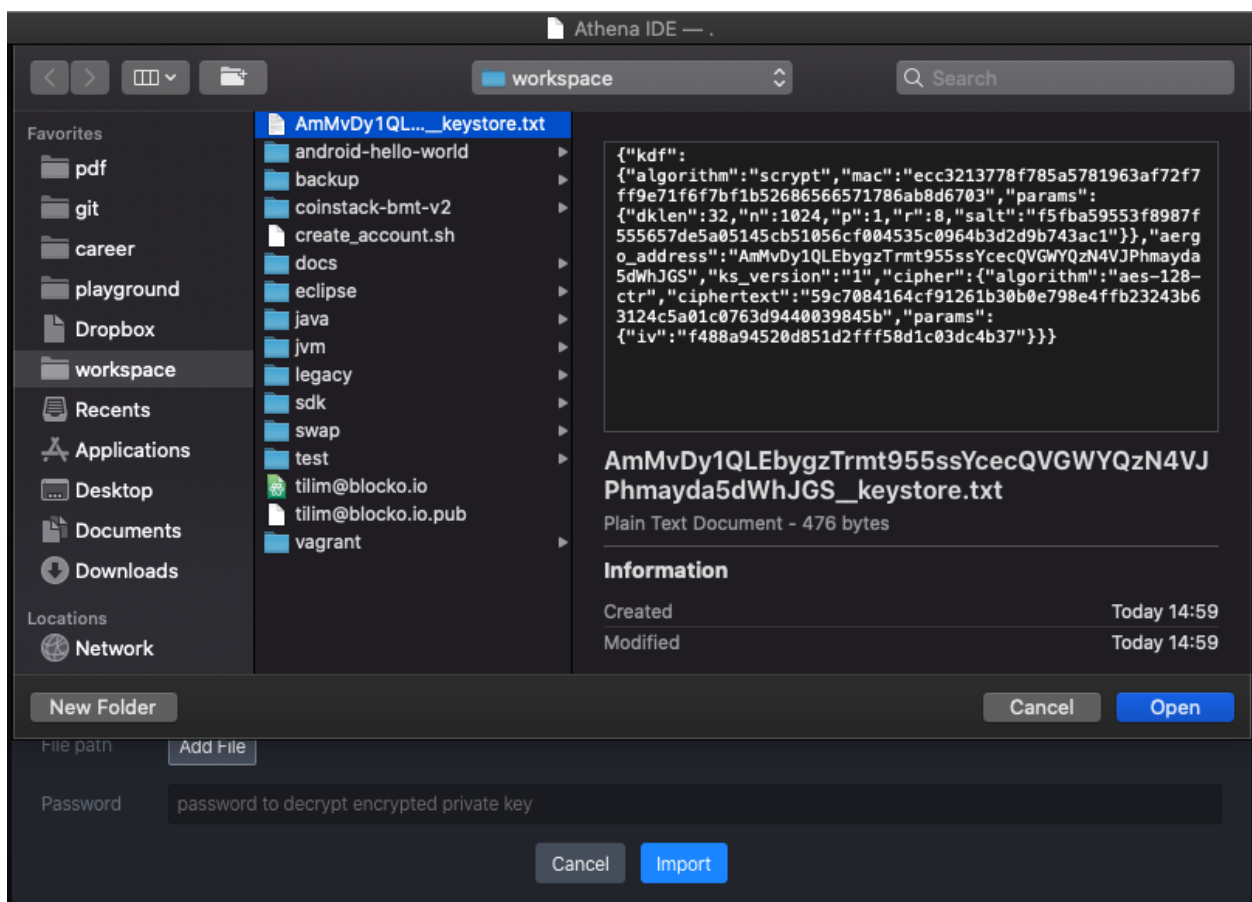
Click an import button



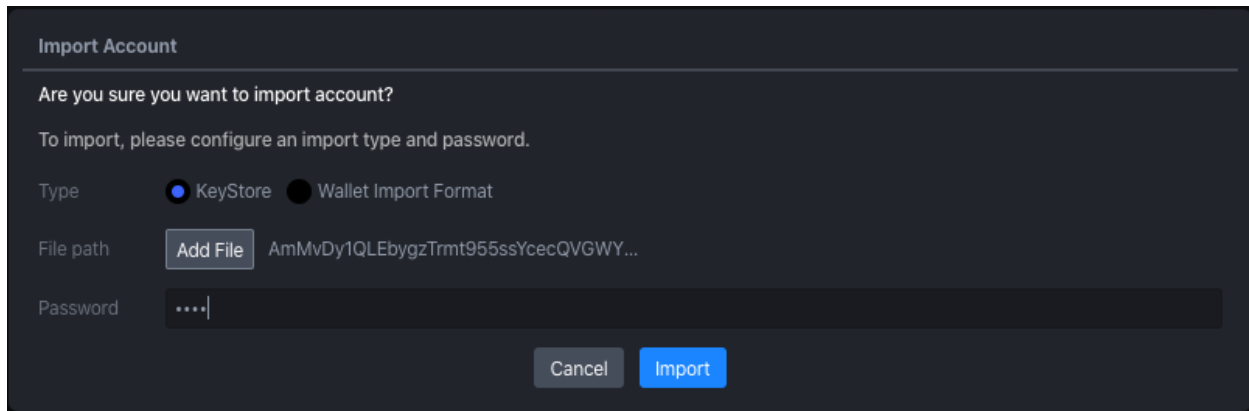
Click an add file button on popup



Select keystore file



Enter password to decrypt keystore



Import Account

Are you sure you want to import account?

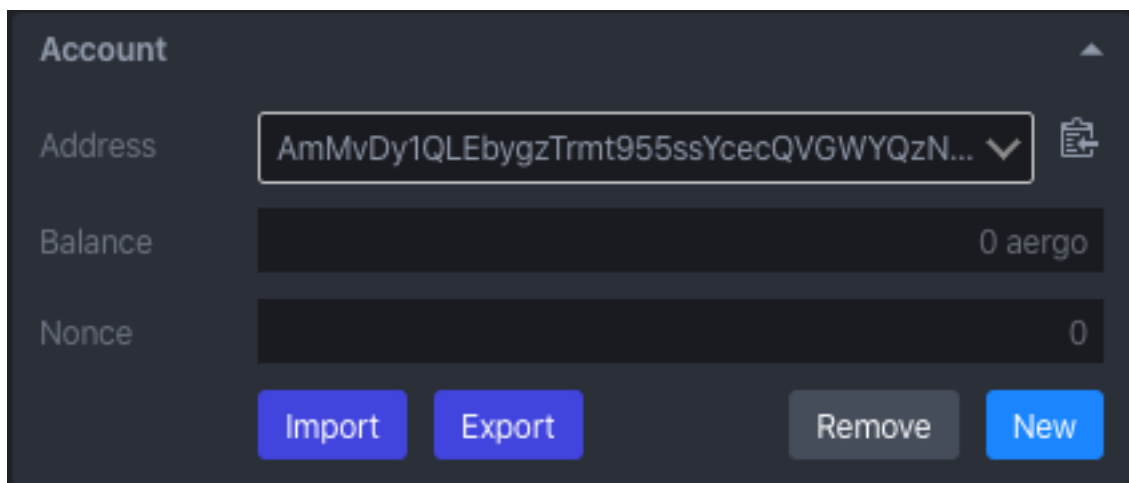
To import, please configure an import type and password.

Type ☒ KeyStore ☐ Wallet Import Format

File path AmMvDy1QLEbygzTrmt955ssYcecQVGWY...

Password

Account imported



Account

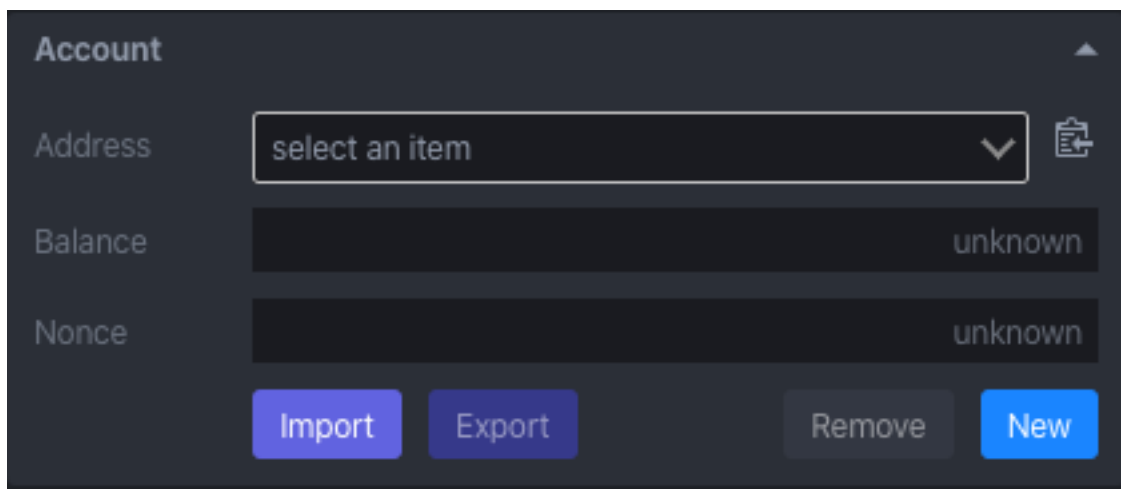
Address

Balance

Nonce

3.2.2 Wallet Import Format

Click an import button



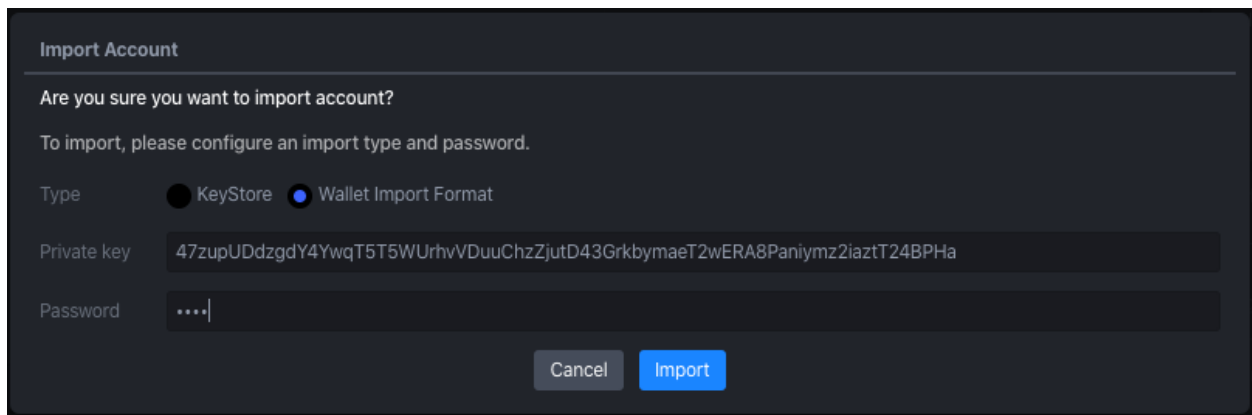
Account

Address

Balance

Nonce

Enter wallet import format & password to decrypt it



Import Account

Are you sure you want to import account?

To import, please configure an import type and password.

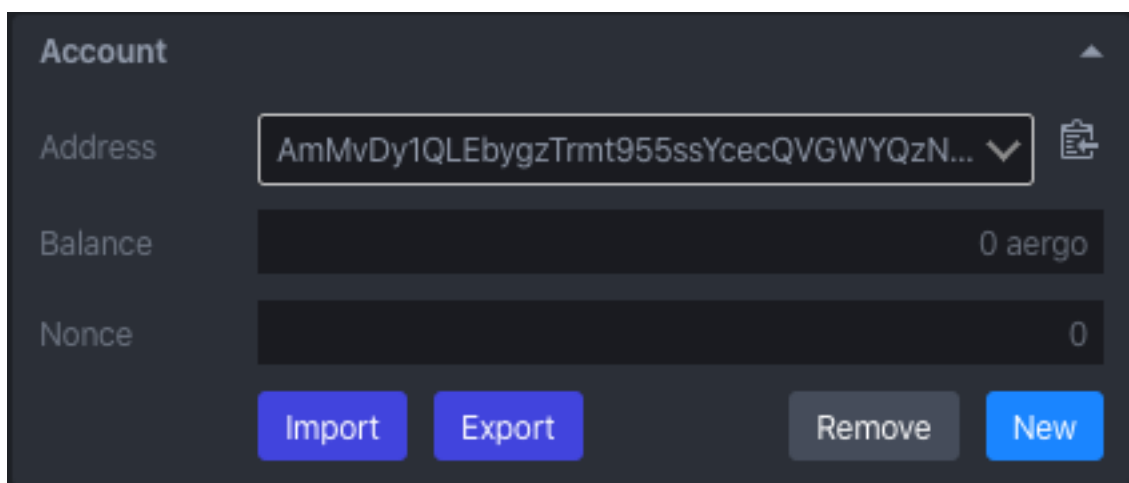
Type ☐ KeyStore ☒ Wallet Import Format

Private key 47zupUDdgdY4YwqT5T5WUrhvVDuuChzZjutD43GrkbymaeT2wERA8Paniymz2iaztT24BPHa


Password|

Cancel Import

Account imported



Account

Address AmMvDy1QLEbygzTrmt955ssYcecQVGWYQzN... 

Balance 0 aergo

Nonce 0

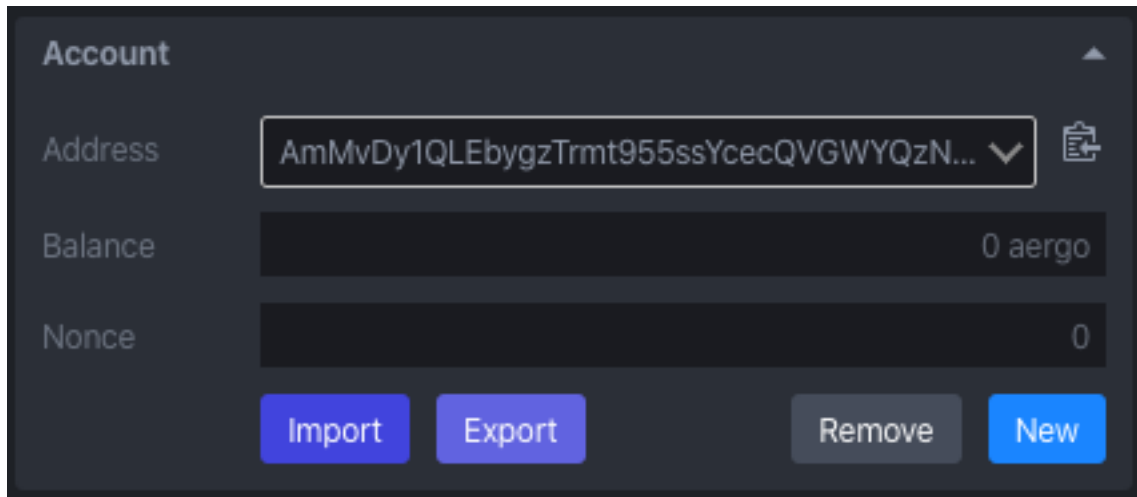
Import Export Remove New

3.3 Export

You can export account with an aergo specific keystore format or wallet import format. For details of keystore format, see [aergo keystore proposal](#).

3.3.1 KeyStore

Click an export button



Account

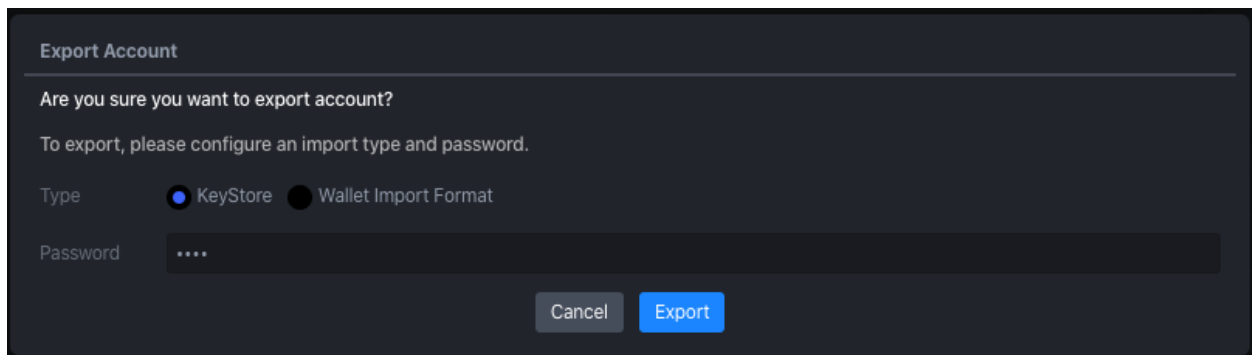
Address: AmMvDy1QLEbygzTrmt955ssYcecQVGWYQzN... ▼

Balance: 0 aergo

Nonce: 0

Buttons: Import, Export, Remove, New

Enter a password to encrypt



Export Account

Are you sure you want to export account?

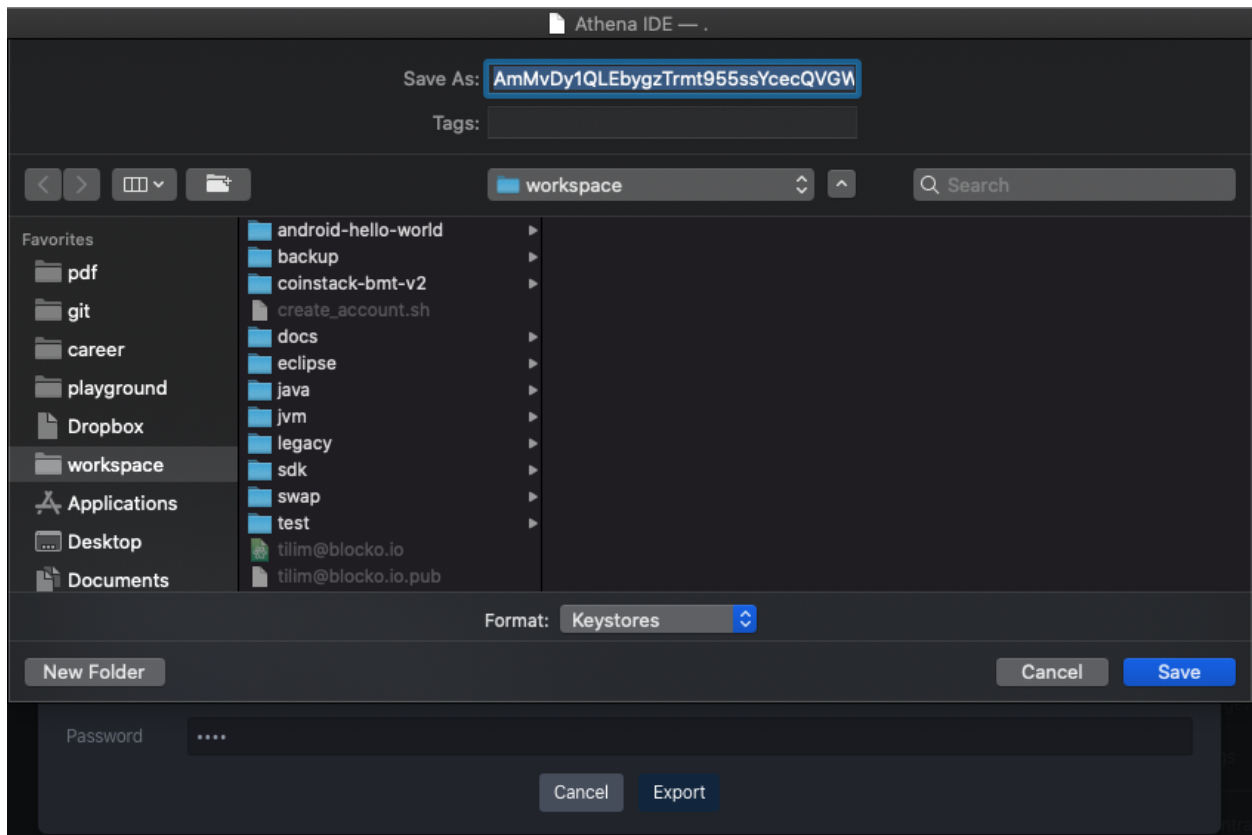
To export, please configure an import type and password.

Type: ☒ KeyStore ☐ Wallet Import Format

Password:

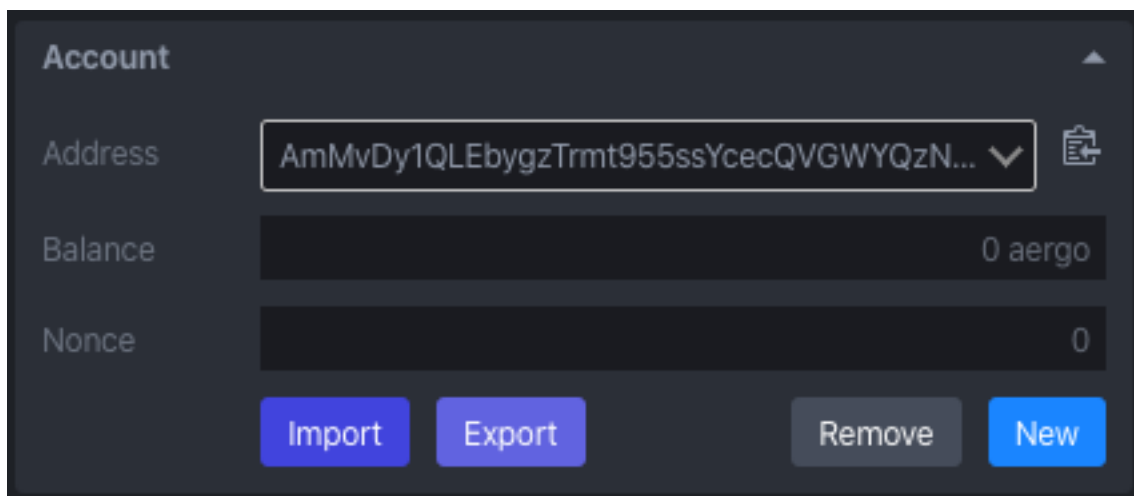
Buttons: Cancel, Export

Choose save location

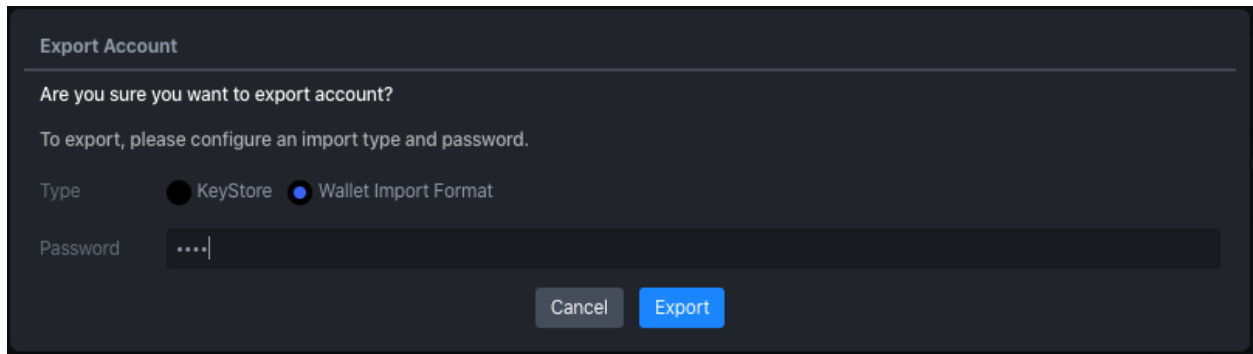


3.3.2 Wallet Import Format

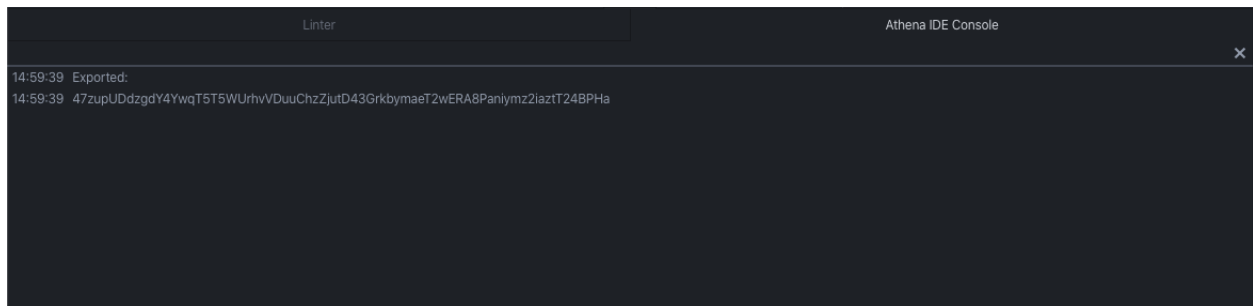
Click an export button



Enter a password to encrypt

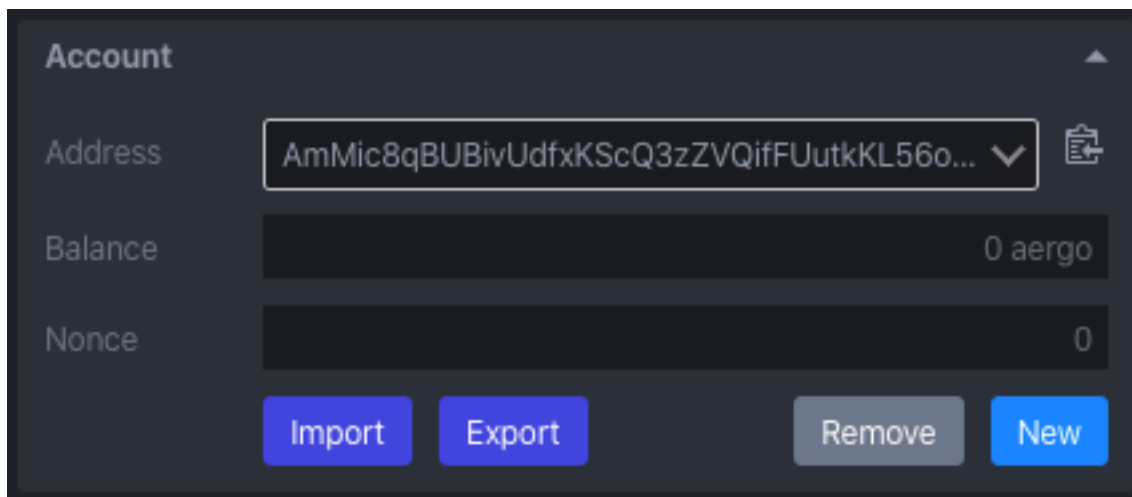


You can see wallet import format in a console

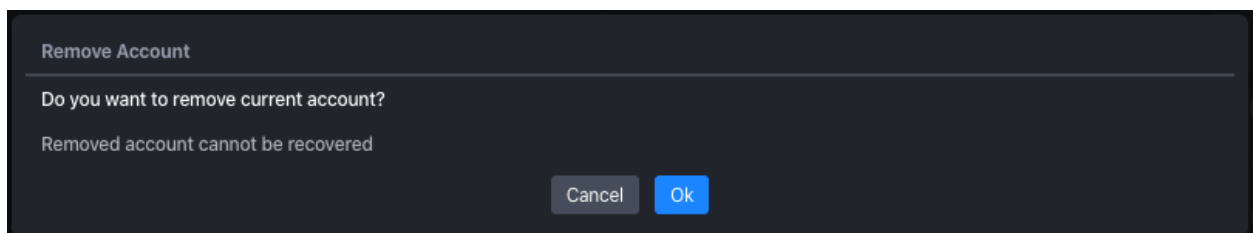


3.4 Remove

Click an remove button



Click a ok button



Account removed from list

Account

Address

select an item

▼

📋

Balance

unknown

Nonce

unknown

Import

Export

Remove

New

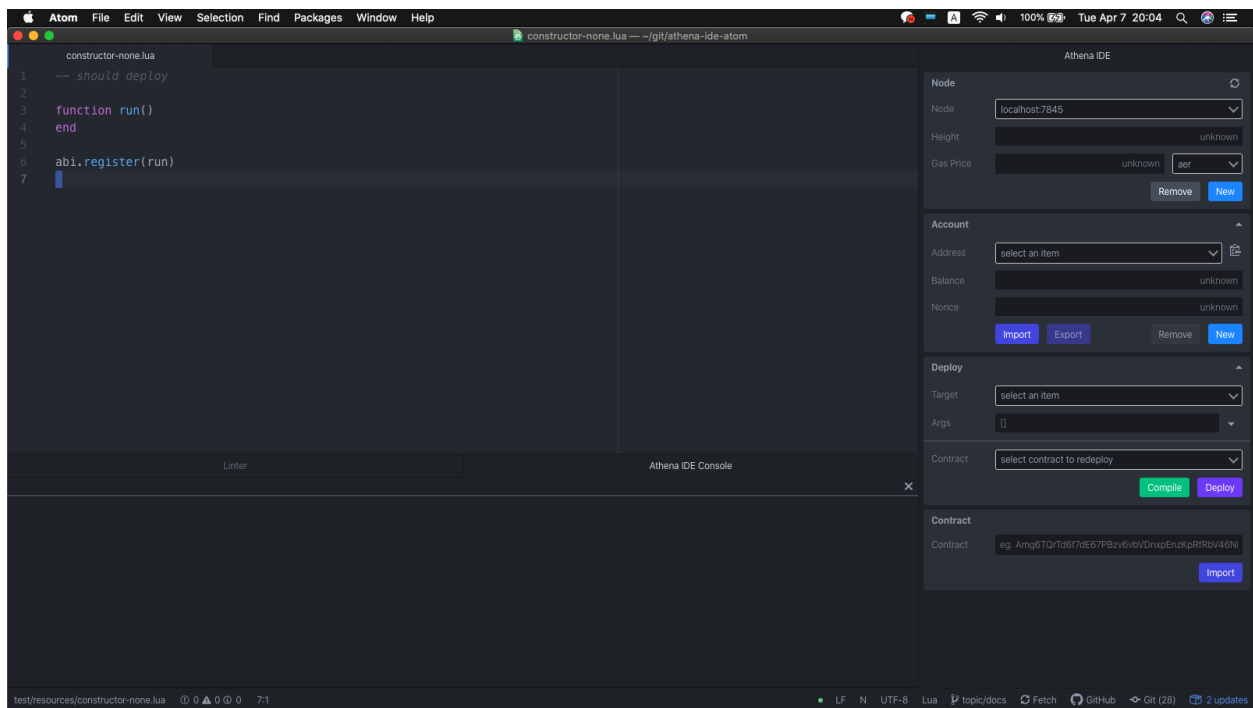
CHAPTER 4

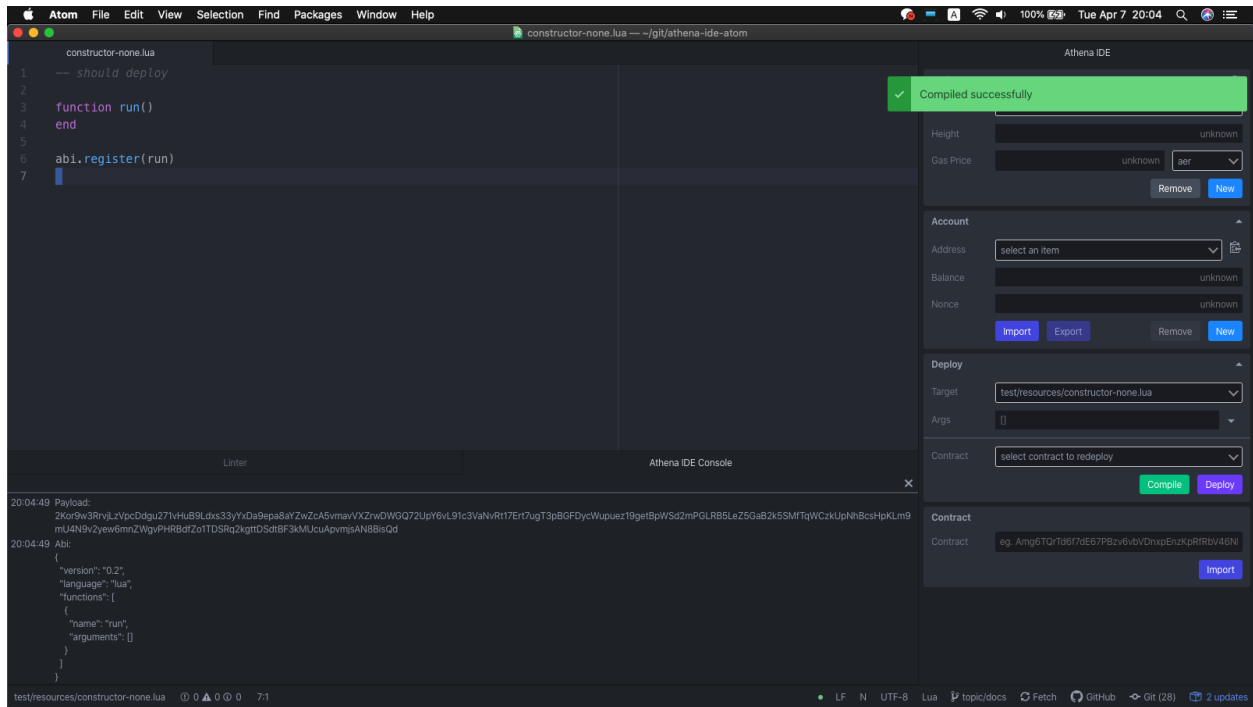
Contract

You can deploy/execute/query smart contract. Deploying and executing is done by making transaction. Make sure that you have enough aergo token to make transaction before deploying and executing contract.

4.1 Compile

Before deploying contract, you have to compile by **clicking compile button** or **pressing F7**.





4.2 Deploy

After compiling contract, you can deploy contract.

4.2.1 Without args

You can deploy contract without constructor arguments.

```

-- no arguments
function constructor()
  system.setItem("k1", "v1")
end

...

```

Deploy

Target

test/resources/constructor-none.lua

Args

[]

Gas Limit

(default: 0)

Contract

select contract to redeploy

Compile

Deploy

4.2.2 With args

You can deploy contract with constructor arguments.

```
-- arguments (key, value)
function constructor(key, value)
  system.setItem(key, value)
end
...
```

Deploy

Targettest/resources/constructor-args.lua

Args

[k1, v1]

keyk1

valuev1

Gas Limit(default: 0)

Contractselect contract to redeploy

CompileDeploy

4.2.3 Gas limit

You can deploy contract with gas limit configuration. 0 limit means infinite (uses as much as possible).

Deploy

Target

test/resources/constructor-args.lua

Args

[k1, v1] (Limit: 200000)

key

k1

value

v1

Gas Limit

200000

Contract

select contract to redeploy

Compile

Deploy

4.2.4 Amount

You can deploy contract with aergo token. Make sure constructor is registered as payable.

```
function constructor()  
  system.setItem("k1", "v1")  
end  
  
...  
  
-- registered as payable  
abi.payable(constructor)
```

Deploy

Target

test/resources/constructor-payable.lua

Args

[] (Amount: 100 aergo)

Gas Limit

(default: 0)

Amount

100

aergo

Contract

select contract to redeploy

Compile

Deploy

4.3 Import & Remove

4.3.1 Import

You can import already deployed contract.

Type contract address & click import button

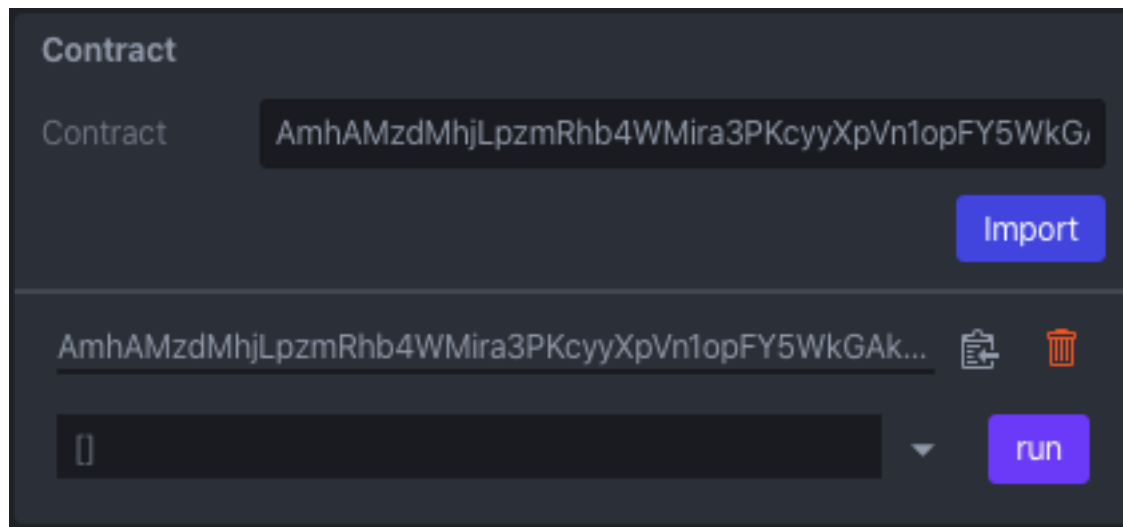
Contract

Contract

AmhAMzdMhjLpzmRhb4WMira3PKcyyXpVn1opFY5WkG,

Import

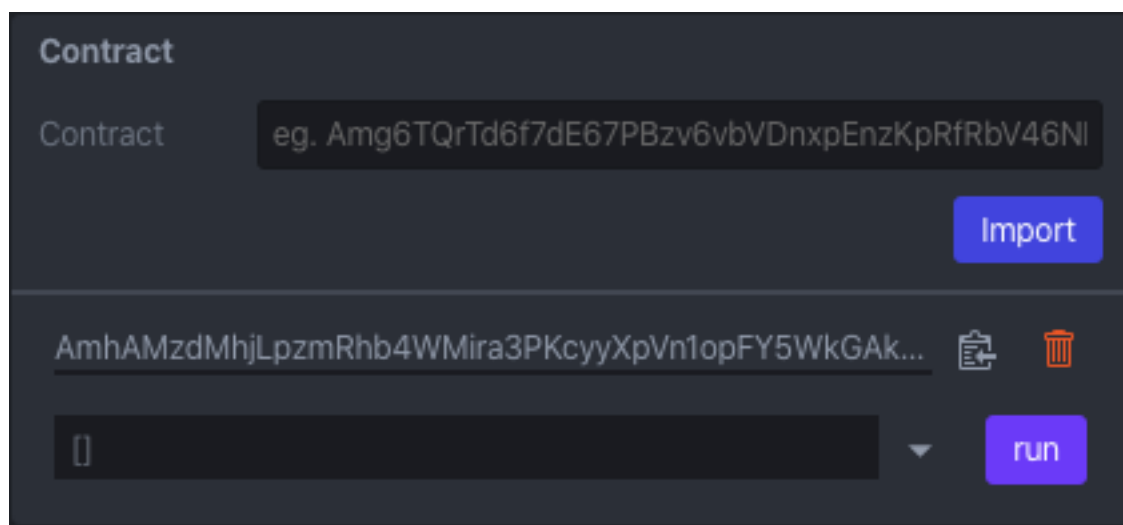
Contract imported



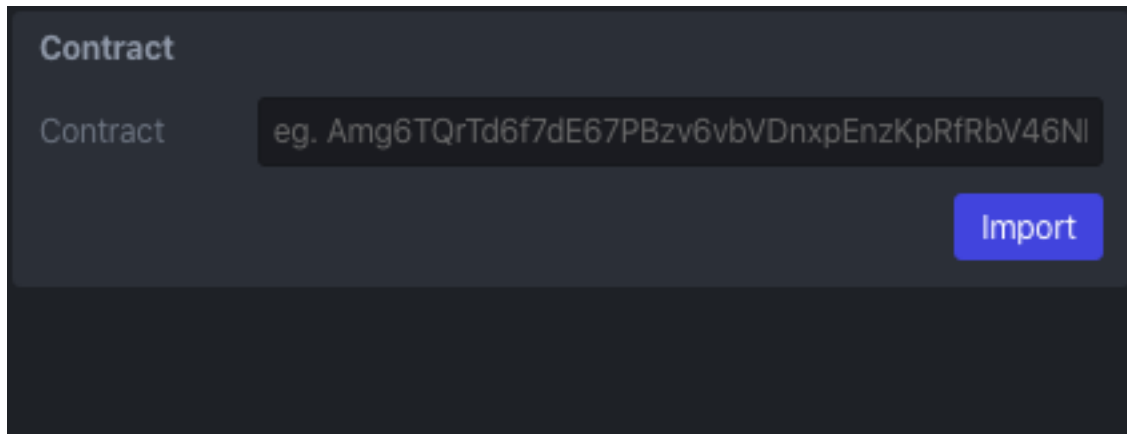
4.3.2 Remove

You can remove contract.

Click trash button



Contract removed



4.4 Execute

Contract execution can change status of contract state db. Any function registered as register can be executed.

4.4.1 Without args

You can execute contract without arguments.

```
...  
  
-- no arguments  
function setDefault()  
  system.setItem("k1", "v1")  
end  
  
...  
  
-- register as execution  
abi.register(setDefault)
```

Contract

Contract

eg. Amg6TQrTd6f7dE67PBzv6vbVDnxpEnzKpRfRbV46NI

Import

AmhQWGAuYzMqPBRK4LPGDHuNLKQizpSSLy8CM3KGHZ...

▲

setDefault

Gas Limit

(default: 0)

▼

get

▼

set

4.4.2 With args

You can execute contract with arguments.

```
...
-- arguments (key, value)
function set(key, value)
  system.setItem(key, value)
end
...
-- register as execution
abi.register(set)
```



Contract

Contract

eg. Amg6TQrTd6f7dE67PBzv6vbVDnxpEnzKpRfRbV46NI

Import

AmhQWGAuYzMqPBRK4LPGDHuNLKQizpSSLy8CM3KGHZ...



[]

▼

setDefault

[key]

▼

get

[key, value]

▲

set

key

key

value

value

Gas Limit

(default: 0)

4.4.3 Gas limit

You can execute contract with configuring gas limit. 0 limit means infinite (uses as much as possible).



Contract

Contract

eg. Amg6TQrTd6f7dE67PBzv6vbVDnxpEnzKpRfRbV46NI

Import

AmfyetWzgzYTRkM8XfGqBbgQUZsPWafQD16xis4vejqbM...

[] (Limit: 200000)

▲

setDefault

Gas Limit

200000

[]

▼

get

[]

▼

set

4.4.4 Amount

You can execute contract with aergo token. Make sure function is registered as payable.

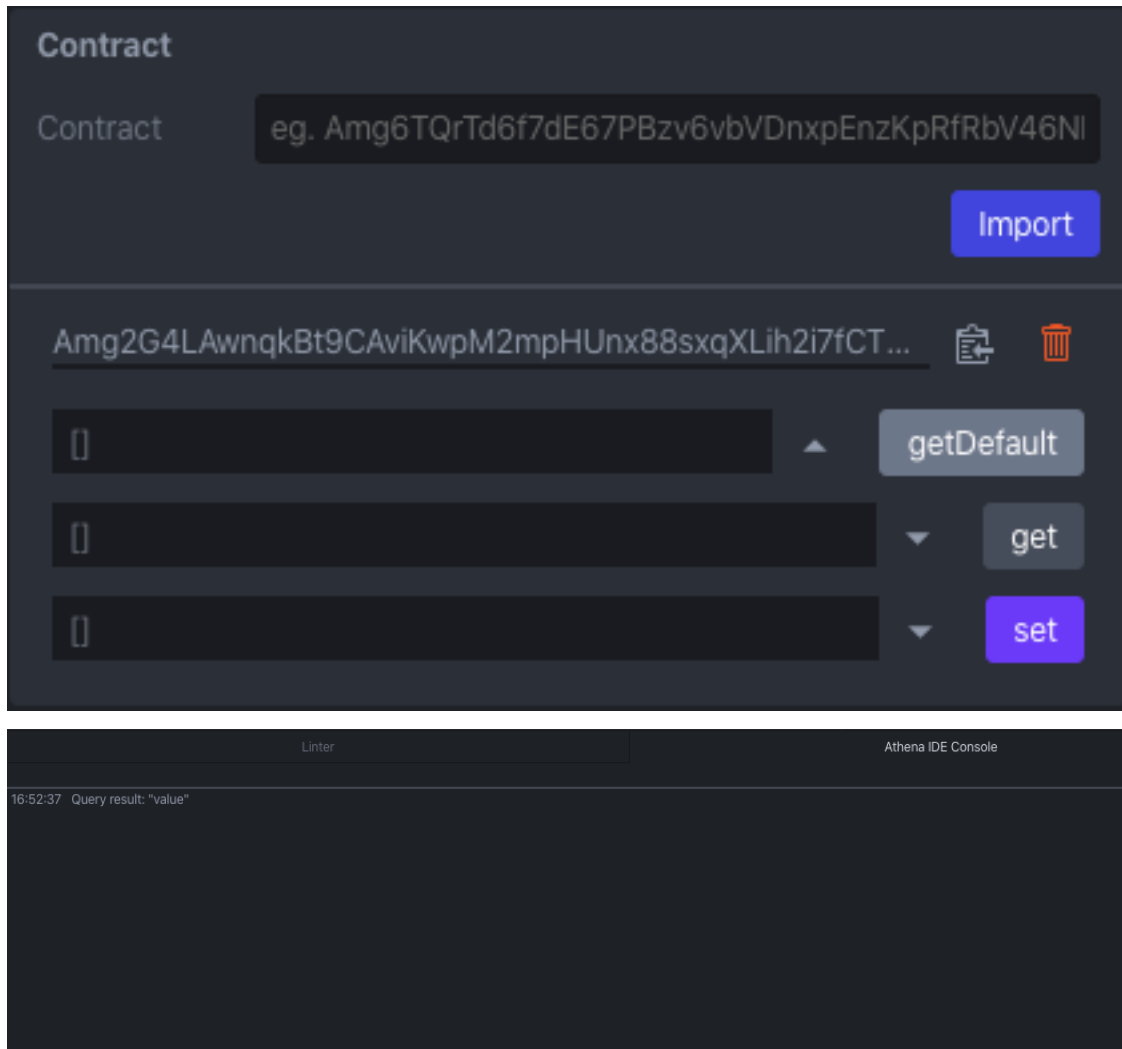
```
...
function run()
end
...
-- registered as payable
abi.payable(run)
```

The screenshot shows the Athena IDE Atom interface for executing a contract. At the top, there is a 'Contract' section with a text input field containing the example address 'eg. Amg6TQrTd6f7dE67PBzv6vbVDnxpEnzKpRfRbV46NI' and an 'Import' button. Below this, a contract address 'AmgUekxBfpwKBKVrt77hphLSdkrRexuK6RH7yKaL3JU3W...' is displayed with copy and delete icons. A dropdown menu shows '[] (Amount: 100 aergo)' with an upward arrow and a 'run' button. Further down, there are input fields for 'Gas Limit' (with '(default: 0)' text) and 'Amount' (with '100' and a unit selector 'aergo' with a dropdown arrow).

4.4.5 Fee delegation

You can execute contract with fee delegation. When contract is executed with fee delegation, the contract pays fee on behalf of contract executor. Make sure function is registered as `fee_delegation` and a contract has enough aergo token.

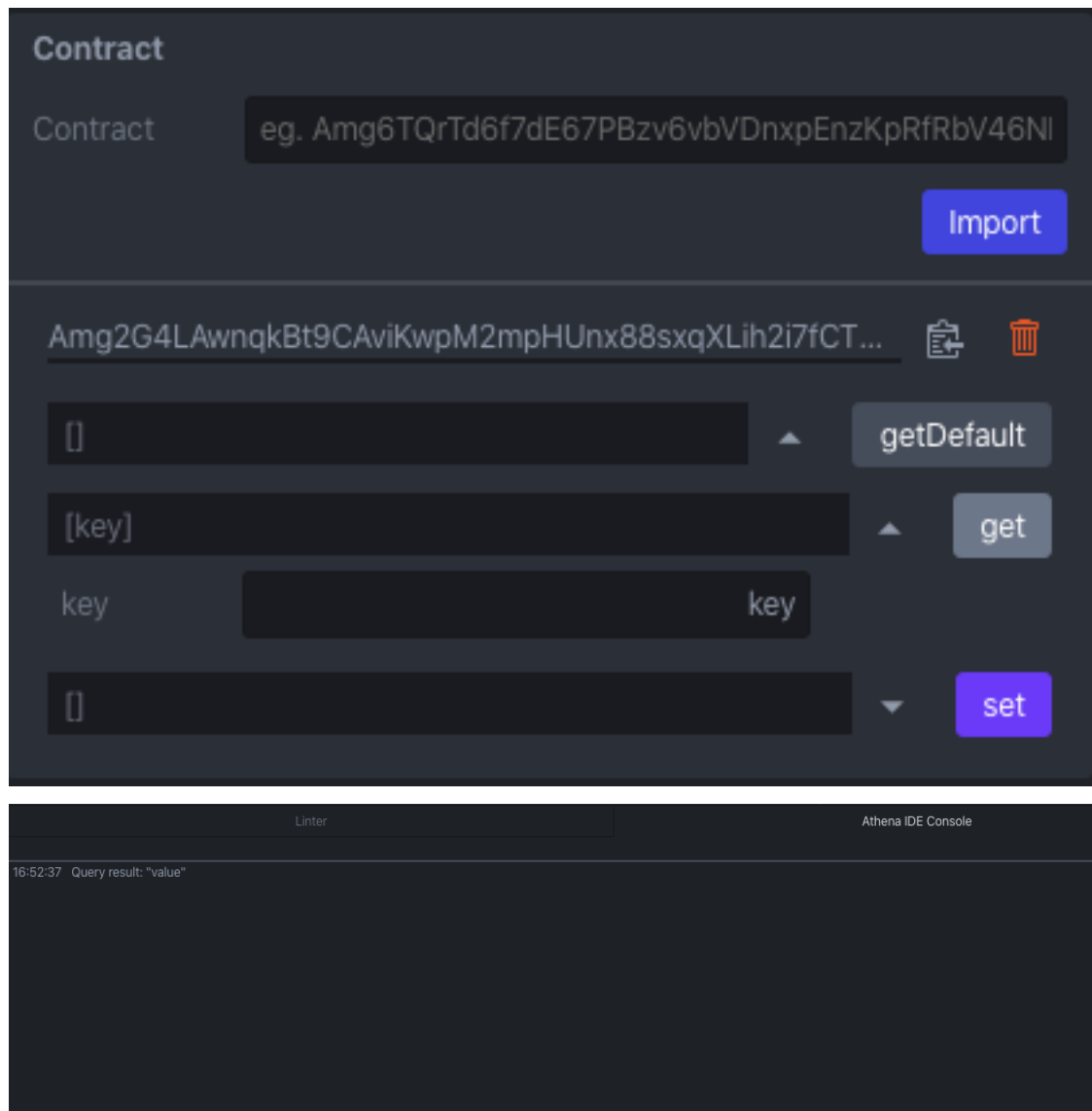
```
...  
  
function run()  
end  
  
-- registered as fee delegation  
abi.fee_delegation(run)  
  
-- register as execution  
abi.register(run)
```

4.5.2 With args

You can query contract status with arguments.

```
...  
  
-- arguments (key)  
function get(key)  
  return system.getItem(key)  
end  
  
-- registered as register_view  
abi.register_view(get)
```

4.6 Varargs

Lua supports varargs. The varargs is denoted by `...` in argument.

```
...
-- ... : varargs
function set(key, ...)
  local s = ""
  for i,v in ipairs{...} do
    s = s .. v
  end
  system.setItem(key, s)
end
...
```

(continues on next page)

(continued from previous page)

```
abi.register(set)
```

4.6.1 Add

Click + button

The screenshot shows the 'Contract' interface in the Athena IDE. At the top, there is a 'Contract' label and a text input field containing 'eg. Amg6TQrTd6f7dE67PBzv6vbVDnxpEnzKpRfRbV46NI'. To the right of this field is a blue 'Import' button. Below this, there is a list of contracts, with the first one being 'AmgQW9YfDRGnvAJWNyUtrFv2cM6TbYowKv1JyHhySY5...'. To the right of this contract name are two icons: a clipboard and a trash can. Below the list, there is a section for adding a new contract. It starts with a dropdown menu showing '[]' and a 'get' button. Below this, there is a dropdown menu showing '[key, v1]' and a 'set' button. To the right of the 'set' button is a '+' button. Below the 'set' button, there are three input fields: 'key', 'arg_1', and 'arg_2'. The 'key' field has a 'key' label. The 'arg_1' field has a 'v1' label. The 'arg_2' field has an 'Optional' label. To the right of the 'arg_2' field is a '+' button. Below these fields is a 'Gas Limit' field with a '(default: 0)' label.

Argument added



Contract

Contract

eg. Amg6TQrTd6f7dE67PBzv6vbVDnxpEnzKpRfRbV46NI

Import

AmgBPce2qVNPae9WcgNY9G1in6ssU3AhATM81fUcTcc9E...



[]

▼

getDefault

[]

▼

get

[v1]

▲

set

key

arg_1

v1

-

arg_2

-

arg_3

Optional

+

Gas Limit

(default: 0)

4.6.2 Remove

Click - button



Contract

Contract

eg. Amg6TQrTd6f7dE67PBzv6vbVDnxpEnzKpRfRbV46NI

Import

Amg62ZaCKSmLW6XBiHohtttFwRp1QCyF83wPYsmhBWQy...



[]

▼

getDefault

[]

▼

get

[v1, v2]

▲

set

key

arg_1

v1

-

arg_2

v2

-

arg_3

Optional

+

Gas Limit

(default: 0)

Argument removed

Contract

Contract

eg. Amg6TQrTd6f7dE67PBzv6vbVDnxpEnzKpRfRbV46NI

Import

Amg62ZaCKSmLW6XBiHohtttFwRp1QCyF83wPYsmhBWQy...

[]

▼

getDefault

[]

▼

get

[v1]

▲

set

key

arg_1

v1

-

arg_2

Optional

+

Gas Limit

(default: 0)

4.7 Redeploy (private mode only)

You can redeploy already deployed contract. This is supported in a private mode only. Make sure redeployer account is deployer of already deployed one.

To redeploy contract, select deployed contract and click deploy button.

Deploy

Target

test/resources/execute-table-vararg.lua

Args

Gas Limit

(default: 0)

Contract

Amg62ZaCKSmLW6XBiHohttFwRp1QCyF83wPYsm...

Compile

Deploy

Contract

Contract

eg. Amg6TQrTd6f7dE67PBzv6vbVDnxpEnzKpRfRbV46NI

Import

Amg62ZaCKSmLW6XBiHohttFwRp1QCyF83wPYsmhBWQy...

