BOTTOS
New AI Ecosystem

BOTTOS Developer Documentation

BOTTOS 开发手册

Table of Contents

Bottos Introduction

Bottos is an infrastructure built for the artificial intelligence, and its underlying public blockchain hasn't only been designed based on data, but it is also meant as a data circulation platform serving the whole artificial intelligence and its derivative industries. By means of data mining and using smart contracts, Bottos realizes a consensus-based, scalable, easy-to-develop, and collaborative one-stop application platform based on data, models, calculation, and storage of multi-tiered shared services.

Bottos can be deployed in fields such as big data, artificial intelligence, smart hardware, robotics, IOT, VR/AR, etc,. Simply put, it is not just an artificial intelligence public chain, it also offers data and models exchange services, which realizes the exchange value of artificial intelligence and its industry based on data feeding, finally forming a new ecosystem of distributed AI.

As a public chain designed to support the artificial intelligence industry, Bottos' performance advantages are focused in the data exchange, as well as calculation power, shared storage and so on, with strong industrial attributes. Its original technical features include: Intelligent Currency Design, Lottery DPOS algorithm and Distributed Storage with AI algorithm, which has a leading competitive advantage in data privacy protection, oversize data storage and so on.

The target of Bottos public chain is to create a decentralized data exchange platform and to solve the difficult pain points of artificial intelligence training data acquisition, then attract artificial intelligence and its derivative industries to actively enter the platform. Through data and model certainty, users are encouraged to share and speed up the volume of transactions. Through the smart contract lifecycle management, Bottos helps AI and its derivative industries eliminate trust costs and increase efficiency.

The application development layer of Bottos is chiefly based on DApps which will be divided into three categories: Artificial Intelligence, Artificial Intelligence Derivative and Block Chain Technical Service categories. The blockchain technology is always through continuing iterations, such as Lightning Network, ORALCE, side chain, cross-chain, etc.,; In order to improve the Bottos ecosystem, the artificial intelligence DApp is represented by simple software, including the features of data feeding model iterative upgrading; Artificial intelligence related Dapps which include intelligent hardware (in the form of combination of software and hardware), IOT, robots, etc., collects new data through hardwares that are the source channels of AI data. These DApps form a closed loop of data between each other, which can eventually form a global scale of data pools, while there is also a demand between DApps smart modules to form a demand map through blockchain technology, which will eventually result in a large pool of modules.

Bottos Architecture

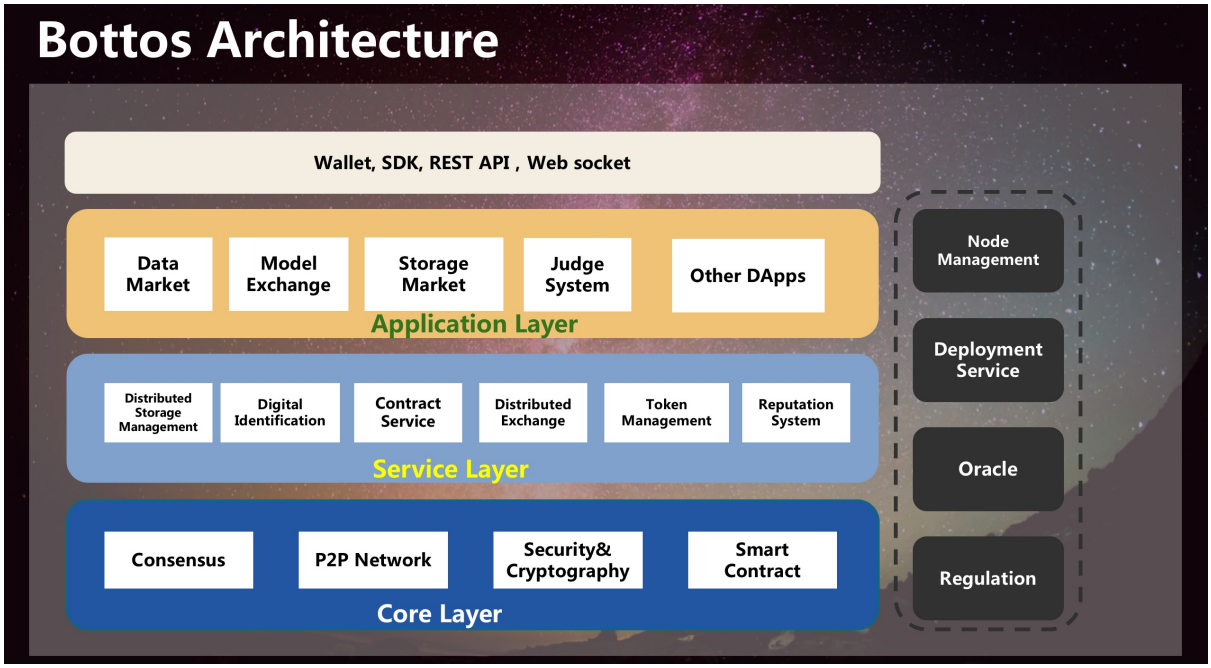
Bottos is a basic infrastructure which focuses on the field of artificial intelligence. It is not only a public underlying blockchain based on data, but it is also a data exchange platform serving the whole artificial intelligence and its derivative industries. The data exchange upon Bottos' blockchain is the world's largest data collection pool based on blockchain technology, addressing the difficult pain points of high-quality data acquisition in the artificial intelligence (AI) industry, creating smart data equity contracts, using data mining to achieve personal data wealth sharing, and establishing a value link between personal data and AI models.

Therefore, the hierarchical architecture is adopted in the design of Bottos system, which facilitates the dynamic expansion and flexible deployment of the system. The overall system is divided into three layers, as shown below.

Core layer: The Chainbase layer, providing chain service based on blockchain, constructing trust public chain infrastructure and realizing the basic function of block chain.

Service Layer: Provides basic services based on Bottos chain development based on Chainbase, such as providing distributed identity services, distributed storage, contracts and token management services, in order to achieve rapid DApp development;

Application Layer: A collection of applications based on Bottos development. Currently, this article chiefly introduces how to use the services of core layer, (that is to say, the block chain’s basic services), and we provide the corresponding development interface and use cases, the basic interface and mode of wallet. The overall technical architecture of the Bottos system is shown in the following figure.



Node Introduction

Bottos nodes are divided into: basic service nodes, application-oriented nodes, light nodes and ultra-light nodes. Each node checks and balances and complements each other, forming a dynamic balance of the self-consistent node system.

(1) Basic service nodes

The Basic Service node provides the fundamental service of the whole Bottos blockchain system, in addition to supporting the core functions of the blockchain core, such as transaction consensus, contract deployment, block production, etc., it also supports the main functions that provided by the Bottos service layer, such as unified identity system, storage management services, credit services, Token management and contract management and other key functions, to provide the most important production services for the overall Bottos network.

(2) Application nodes

The kind of application node primarily serves different Dapps' deployments in the ecosystem, and application node can be deployed on itself, or be deployed on the fundamental nodes. (Logical applications on each fundamental node can charge a certain deployment threshold fee in future.)

(3) Lightweight nodes

The lightweight node chiefly works as the data verification node of Bottos, and it does not provide the corresponding function of the service layer. It only carries on the block data's synchronization and so on, and can also support the Bottos core Layer basic service.

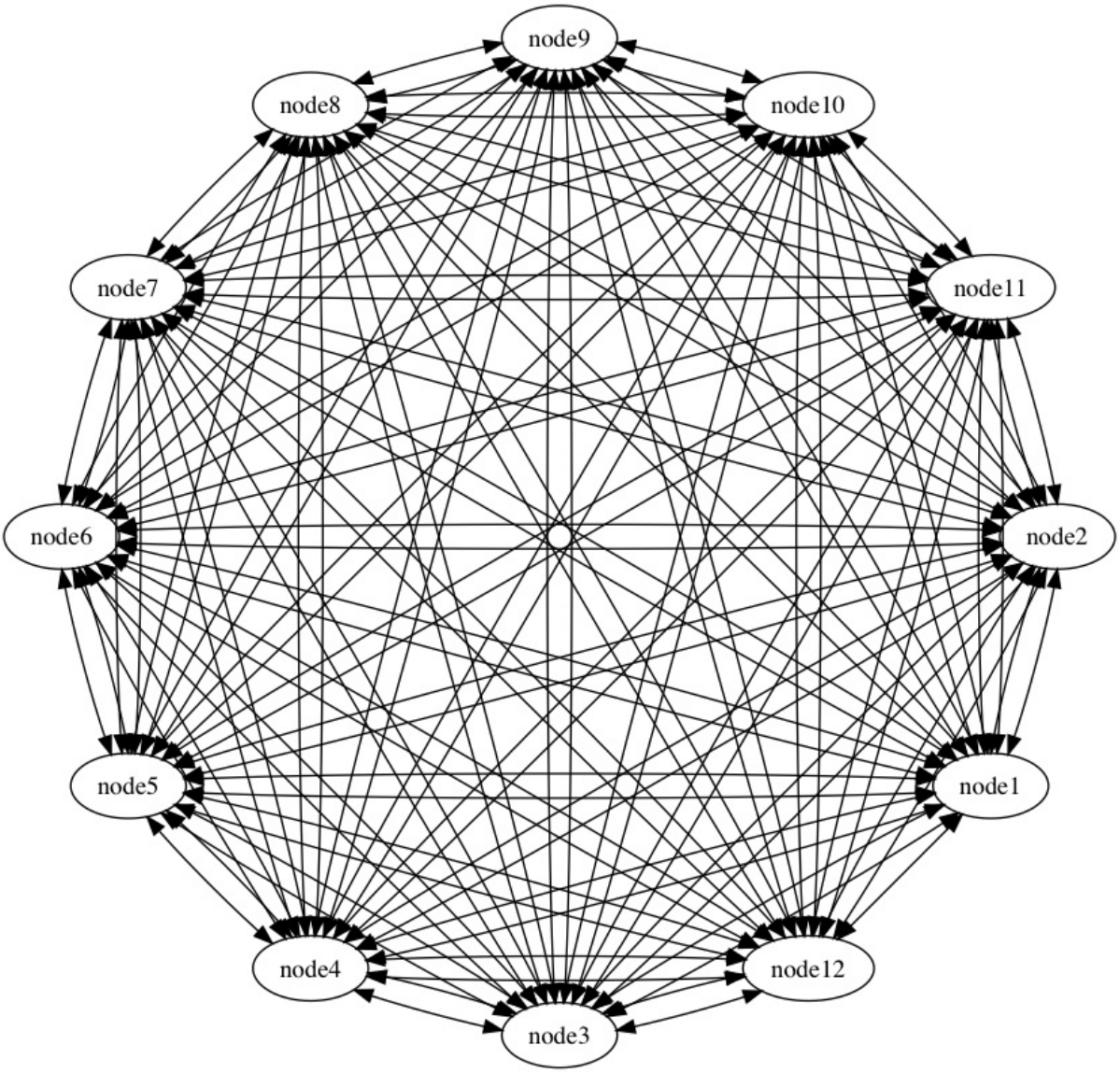
（4）Ultra lightweight nodes

The ultra lightweight node chiefly handles the block correlation head verification, and synchronizes the key hash data, etc.

Network Topology

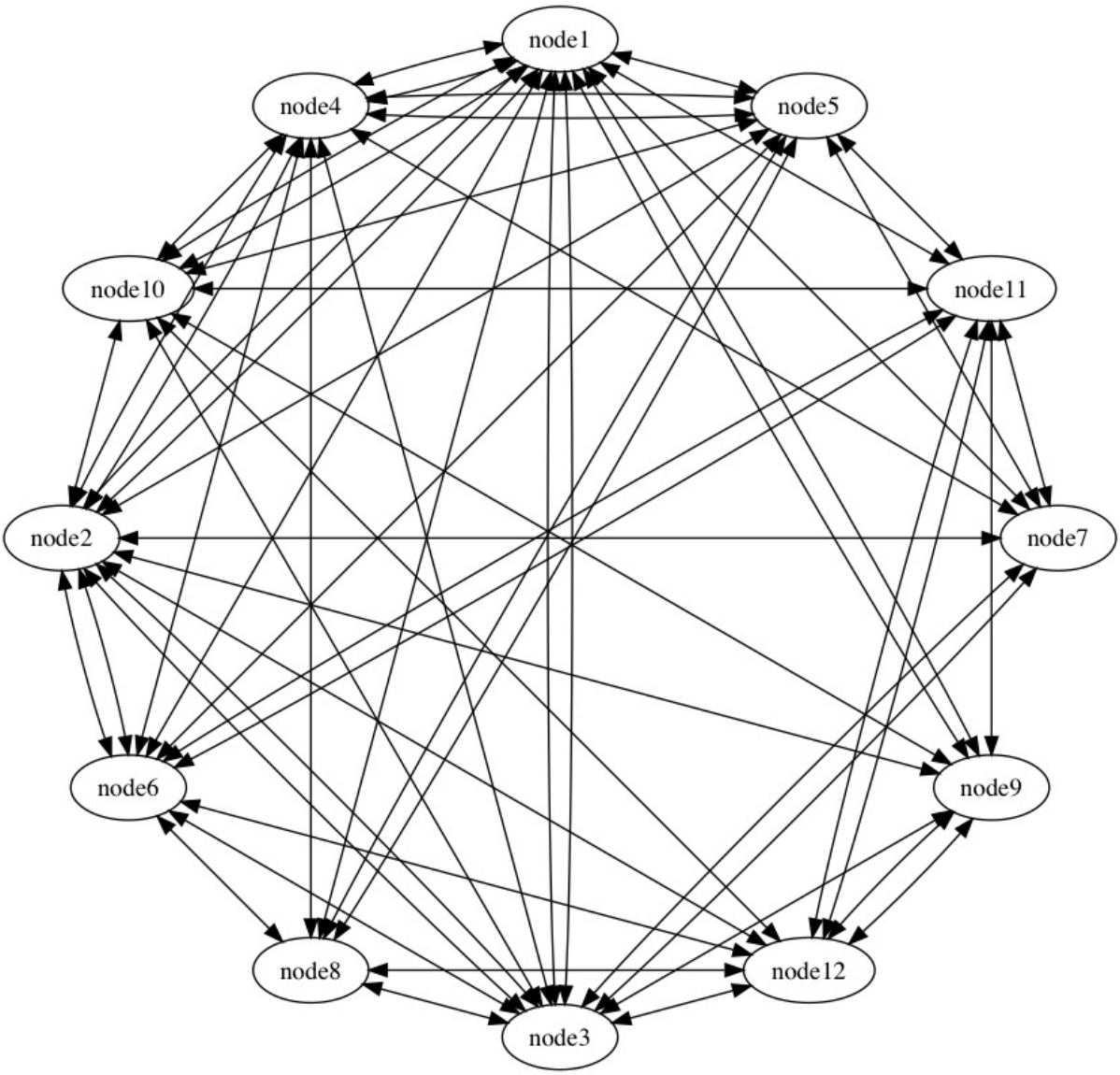
Super node network

The super node undertakes the implementation and the writing of the records of all transactions in the whole network, it is the fundation of the overall stability of the global network. Therefore, Bottos uses a fully connected mode of networking by its super node's network-building design, shown as the following figure (for instance, the full connection of 10 super nodes. The actual number of super nodes is 49, amnong which 29 members will be elected in each round, the other 20 nodes are spare nodes)



Connection diagram among genernal and super nodes

The network among the nodes other than super nodes are loosely organized, just similar to the mesh-structure-connected way in the following figure.


node1
node4
node5
node10
node11
node2
node7
node6
node9
node8
node12
node3

Bottos BCLI Usage Specification

1 BCLI Tool Briefly Description

Bottos BCLI realizes the human-computer interaction command line, and chiefly help to realize the accounts creation, information querying from the chain, balances transfer, wallet functions, etc., by interacting with the block chain based on RESTFUL API.

2 BCLI Tool Installation and Deployment

Do 'go build' command under the BCLI directory of BOTTOS code's directory, then the bcli executable file will be generated after a successful compilation.

3 BCLI Command Line Description

The Global Option Parameter Description

Gloabl Options	Parameters description
--servaddr	The global restful address' configuration

The global help information

```
./bcli --help

NAME:
Bottos bcli tool - a tool that makes user communicate with bottos blockchain

USAGE:
bcli [global options] command [command options] [arguments...]

VERSION:
0.0.1

COMMANDS:
    getinfo      get chian info
    getblock     get block info
    gettable     get table info
    account      create or get account
    transfer     for user transfering bto
    transaction  get or push transactions
    contract     get or deploy contract
    p2p          for p2p connection
    delegate     for delegate operations
    wallet       For wallet operations
    genesis      for genesis node operations
    help, h      Shows a list of commands or help for one command
```

```
GLOBAL OPTIONS:
--servaddr value  (default: "127.0.0.1:8689")
--help, -h        show help
--version, -v     print the version
```

command line function description

Command	Parameter list	Parameter description
./bcli	getinfo	Get block header information
./bcli	getblock	Get specific block information
./bcli	gettable	Get table information of a contract
./bcli	account	Create / get user's information, stake/unstake/claim balances, etc,.
./bcli	transfer	BTO transferring function
./bcli	transaction	Get / call a transaction
./bcli	contract	Query/deploy contract and ABI
./bcli	p2p	P2P commands are not supported temporary
./bcli	delegate	Register/Unregieter a producer，vote for producer/cancel vote for a producer, etc.
./bcli	wallet	Create /lock/unlock/query a wallet
./bcli	genesis	Operations relates to the genesis node, setting the original producers，transferring the authority of producing blocks，cancel the operation authority of node, etc,.

1. BCLI user account function commands

The BCLI user account function commands are chiefly be responsible for creating new accounts, gettting specific user information, staking or unstaking BTO, and claiming staking BTO, etc,.

Chief command help information

```
./bcli account --help

NAME:
Bottos bcli tool account - create or get account

USAGE:
Bottos bcli tool account command [command options] [arguments...]

COMMANDS:
    create   create account
    get      get account info
    stake    stake of account
```

```
    unstake  unstake of account
    claim    claim of stake

OPTIONS:
--help, -h  show help
```

Function Description of command line

chief command	parameter list	parameter description
./bcli account	create	create new account
./bcli account	get	get account's information
./bcli account	stake	stake BTO
./bcli account	unstake	unstake BTO
./bcli account	claim	claim parts of user's BTO

Create a User Command Description

help information

```
./bcli account create --help

NAME:
Bottos bcli tool account create - create account
USAGE:
Bottos bcli tool account create [command options] [arguments...]

OPTIONS:
```

--username value acocunt name --pubkey value account public key (default: "0454f1c2223d553aa6ee53ea1ccea8b7bf78b8ca99f3ff622a3bb3e62dedc712089033d6091d77296547bc071022ca2838c9e86dec29667cf740e5c9e654b6127f") --referrer value account referrer

Parameter Description

chief command	parameter list	Parameter Description	mandatory
bcli account create	--username	The created user name	Yes
	--pubkey	The new public key of the new creared user	Yes
	--referrer	The referrer user，default will use inside default account，otherwise use the referrer to do the signature	No

Return Information

This will output the succeeded Transaction information sent by BCLI, chiefly include user name, public key, sender and signature (--referrer information, default is the inside bottos account), etc,.

Sample

```
./bcli account create --username user1 --pubkey 0454f1c2223d553aa6ee53ea1ccea8b7bf7
8b8ca99f3ff622a3bb3e62dedc712089033d6091d77296547bc071022ca2838c9e86dec29667cf740e5
c9e654b6127f
```

Outputs

```
Create account: user1 Succeed
Trx:
{
    "version": 1,
    "cursor_num": 1093,
    "cursor_label": 3607725829,
    "lifetime": 1536230581,
    "sender": "bottos",
    "contract": "bottos",
    "method": "newaccount",
    "param": {
        "name": "user1",
        "pubkey": "0454f1c2223d553aa6ee53ea1ccea8b7bf78b8ca99f3ff622a3bb3e62dedc712
089033d6091d77296547bc071022ca2838c9e86dec29667cf740e5c9e654b6127f"
    },
    "param_bin": "dc0002da00057573657231da00823034353466316332323233643535336161366
565353365613163636561386237626637386238636139396633666636323261336262336536326465646
633731323038393033336436303931643737323936353437626330373130323263613238333863396538
366465633239363637636637343065356339653635346231323766",
    "sig_alg": 1,
    "signature": "90936cc7c453a737ec16a70d4c63d1032b1f3d3419a3b20f428d6c98490d59345
53bcf2bb5ee6c21914b1421fcbb41146fe8beafd0f0bc78d8708811769dd6db"
}
TrxHash: fc83e64f1e05fd6a78256250c6e58a782567ead312db711a6549a6675eafb696
```

Sample of getting user information

Help information

```
./bcli account get --help
NAME:
Bottos bcli tool account get - get account info

USAGE:
Bottos bcli tool account get [command options] [arguments...]

OPTIONS:
    --username value  account name
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
bcli account get	--username	user name	Yes

Return Information

This will return the BTO information of the account.

Sample

```
./bcli account get --username bottos
```

Outputs

```
Account: bottos
Balance: 999710000.00000000 BTO
```

Sample of user staking BTO

Help information

```
./bcli account stake --help

NAME:
Bottos bcli tool account stake - stake of account

USAGE:
Bottos bcli tool account stake [command options] [arguments...]

OPTIONS:
    --account value  account name
    --amount value   amount of bto
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
bcli account stake	--account	user name	Yes
	--amount	amount of staking BTO	Yes

Return Information

This will return the transaction information after the command successfully sent.

Sample

```
./bcli account stake --account lyp --amount 2
```

Outputs

```
Transfer Succeed:
Trx:
{
    "version": 1,
    "cursor_num": 3909,
    "cursor_label": 4191535575,
    "lifetime": 1536560467,
    "sender": "bottos",
    "contract": "bottos",
    "method": "stake",
    "param": "dc0001c500200000000000000000000000000000000000000000000000000000000000000000
0000002",
    "param_bin": "dc0001c500200000000000000000000000000000000000000000000000000000000000000000
000000000002",
    "sig_alg": 1,
    "signature": "fcb341de3ada22108107285b511fe35d8777d31150545678eb278ad4457303c80
59e1b1ff2fa39fb545a0ea743b45e830c6d1bbf86611087a4807116c5351acd"
}
TrxHash: cbea44984ca17dc640cec7bc5349b6c839229ff86462a24fa0e1196d2b73cedd
```

Sample of user unstaking BTO

Help information

```
./bcli account unstake --help

NAME:
Bottos bcli tool account unstake - unstake of account

USAGE:
Bottos bcli tool account unstake [command options] [arguments...]

OPTIONS:
    --account value  account name
    --amount value   amount of bto
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
bcli account unstake	--account	user name	Yes
	--amount	amount of unstaking BTO	Yes

Return Information

This will return the transaction information after the command successfully sent.

Sample

```
./bcli account unstake --account lyp --amount 2
```

Outputs

```
Transfer Succeed:
Trx:
{
    "version": 1,
    "cursor_num": 4067,
    "cursor_label": 3077673547,
    "lifetime": 1536560941,
    "sender": "bottos",
    "contract": "bottos",
    "method": "unstake",
    "param": "dc0001c500200000000000000000000000000000000000000000000000000000000000000000000000000002",
    "param_bin": "dc0001c500200000000000000000000000000000000000000000000000000000000000000000000000000000000002",
    "sig_alg": 1,
    "signature": "9f14f894a5dd68a7ec010e95ae9c0de52bf368964e8a539d4f2a2c68240e1e4164d4e7a0b34943399c821c81c7b5526ff96370f2cd7d175c4dab062a98f354e5"
}
TrxHash: 336b46af3a8eefbc8879cd46e91c42eb122152111c3f61c312422eb41b77e383
```

Sample of user claiming BTO

Help information

```
./bcli account claim --help

NAME:
Bottos bcli tool account claim - claim of stake

USAGE:
Bottos bcli tool account claim [command options] [arguments...]

OPTIONS:
    --account value  acocunt name
    --amount value   amount of bto
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
bcli account claim	--account	user name	Yes
	--amount	Amount of claiming staked BTO	Yes

Return Information

This will return the transaction information after the command successfully sent.

Sample

```
./bcli account claim --account lyp --amount 2
```

Outputs

```
Transfer Succeed:
Trx:
{
    "version": 1,
    "cursor_num": 4148,
    "cursor_label": 2794796656,
    "lifetime": 1536561184,
    "sender": "bottos",
    "contract": "bottos",
    "method": "claim",
    "param": "dc0001c500200000000000000000000000000000000000000000000000000000000000000000000000000002",
    "param_bin": "dc0001c500200000000000000000000000000000000000000000000000000000000000000000000000000002",
    "sig_alg": 1,
    "signature": "e05082ecf9028bc9e1e0aa003f6f13cffabba4fc28118ad3d3e9de678525d5b049035818d14c8067b739db34408d8aee1aa50deb99a0fcd9973ad2dcd7fdc627"
}
TrxHash: fe3952e5743db0b35bf669b68e5a3d243c19836e1f5c17b970f524a9d59b8eab
```

2. BCLI contract function command

BCLI contract function command chiefly releaizes the deployment of the contract file and its ABI file onto block chain, also the function of querying the contract code (or file) and ABI information(or file) from block chain.

Chief Help information

```
./bcli contract --help

NAME:
Bottos bcli tool contract - get or deploy contract

USAGE:
Bottos bcli tool contract command [command options] [arguments...]

COMMANDS:
    deploy      contract deploy
    deploycode  contract  deploycode
```

```
    deployabi   Contract  deployabi
    get         get contract

OPTIONS:
    --help, -h  show help
```

Commandline Function Description

chief command	parameter list	Parameter Description
./bcli contract	deploy	Deploy the ABI file and WASM contract
./bcli contract	deploycode	Deploy contract
./bcli contract	deployabi	Deploy ABI file
./bcli contract	get	Get specific contract and abi info , then save to files

BCLI Contract Deployment Command

Help information

```
./bcli contract deploycode --help
NAME:
Bottos bcli tool contract deploycode - contract  deploycode

USAGE:
    Bottos bcli tool contract deploycode [command options] [arguments...]

OPTIONS:
    --name value  the contract name
    --code value  the contract's wasm file path ( includes wasm file name )
    --user value  the user account
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
bcli contract deploycode	--name	contract name	Yes
	--code	Contract file（.WASM）'s path	Yes
	--user	user who signs the contract	Yes

Return Information

This will return the transaction information after the command successfully be sent.

Sample

```
./bcli contract deploycode --name nodeclustermng --code nodeclustermng.wasm --user
bottos
```

Outputs

```
Deploy contract: nodeclustermng Succeed
Trx:
{
    "version": 1,
    "cursor_num": 4948,
    "cursor_label": 3178968225,
    "lifetime": 1536563584,
    "sender": "nodeclustermng",
    "contract": "bottos",
    "method": "deploycode",
    "param": {
        "name": "nodeclustermng",
        "vm_type": 1,
        "vm_version": 1,
        "contract_code": "0061736d01000000012907600 27f7f017f60017f0060027f7f0060067
f7f7f7f7f7f017f60037f7f7f017f60000060017f017f02560603656e760867657450617 2616d000003
656e76066d656d637079000403656e76066d656d736574000403656e7606..."
    },
    "param_bin": "dc0004da000e6e6f6465636c75737465726d6e67cc01cc01c5162f0061736d010
0000001290760027f7f017f60017f0060027f7f0060067f7f7f7f7f7f017f60037f7f7f017f60000060
017f017f02560603656e760867657450617 2616d000003656e76...",
    "sig_alg": 1,
    "signature": "aa80b0d7ab8779f112e38cd9ae6eff759dc3997c2d852c33130baac8cf4c641d5
49b43d69759c3e2b1c617a7552134be5d68ff11dbb0b2fd70d51a2606b853be"
}
TrxHash: ad45ac9a840642e569f07d0a9ba7fb9331e67bc96576e157b68ee09840d430f7
```

BCLI ABI deploy commands

Help information

```
./bcli contract deployabi --help
NAME:
Bottos bcli tool contract deployabi - Contract  deployabi

USAGE:
    Bottos bcli tool contract deployabi [command options] [arguments...]

OPTIONS:
    --name value  the contract name
    --abi value   the contract's abi file path ( includes abi file name )
    --user value  the user account
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
bcli contract deployabi	--name	contract name	Yes

	--abi	abi description file（.abi）'s path	Yes
	--user	user who signs the contract	Yes

Return Information

This will return the transaction information after the command successfully sent.

Sample

```
./bcli contract deployabi --name nodeclustermng --abi nodeclustermng.abi --user bot
tos
```

Outputs

```
{
    "errcode": 0,
    "msg": "trx receive succ",
    "result": {
        "trx": {
            "version": 1,
            "cursor_num": 5264,
            "cursor_label": 890555306,
            "lifetime": 1536564532,
            "sender": "nodeclustermng",
            "contract": "bottos",
            "method": "deployabi",
            "param": "dc0002da000e6e6f6465636c75737465726d6e67c504297b0a09227479706
573223a205b5d2c0a092273747275637473223a205b0a2020202020202020202020202020207b0a202020
2020202020202020202009226e616d65223a20224e6f6465436c7573746572526567222c0a2020202
0202020202020202009226262617365223a2022222c0a2020202020202020202020202020202009226669
656c6473223a207b0a2020202020202020202020202020200909226e6f64654950223a2022737472696e6
7222c0a2020202020202020202020202020200909226c75737465724950223a2022737472696e67222c
0a2020202020202020202020202020200909227575696422 3a2022737472696e67222c0a2020202020202
0202020202020200909226361706163697479223a2022737472696e67220a2020202020202020202020
2020202020207d0a2020202020202020202020202020200909227575696422 3a2022737472696e67222c
0a2020202020202020202020202020200909226361706163697479223a2022737472696e67220a2020202
0202020202020202020202020202020207d0a2020202020202020202020202020200909227575696422
3a2022737472696e67222c0a2020202020202020202020202020200909226361706163697479223a2022
737472696e67220a2020202020202020202020202020202020207d0a2020202020202020202020202020
20207d0a2020202020202020202020202020200909227575696422 3a2022737472696e67222c0a202020
2020202020202020202020202020207d0a2020202020202020202020202020200909227461626c6573223a
205b0a2020202020202020202020202020202020207d0a2020202020202020202020202020202020
20207d0a2020202020202020202020202020200909227461626c655f6e616d65223a20226e6f6465636c
7573746572696e666f222c0a2020202020202020202020202020200922696e6465785f74797065223a202
2737472696e67222c0a2020202020202020202020202020202009226b65795f6e616d6573223a20205b0a20
2020202020202020202020202020200909226e6f64654950220a2020202020202020202020202020202009205d2
```

```
c0a2020202020202020202020202009226b65795f7479706573223a20205b0a202020202020202020
2020202020090922737472696e67220a2020202020202020202020202009205d2c0a2020202020202
0202020202020200922747970653a20224e6f6465436c7573746572496e666f220a20202020202020
2020202020207d0a202020202020205d0a7d0a",
            "sig_alg": 1,
            "signature": "1807799fb6c007dff7a49641f1a84b41c083d346af300bbafa0dd1894
a01ab6e51d1885b9ded14a66850b85e5ef4c38cc515bd9934f7ceaf26f2db33cfc6973f"
        },
        "trx_hash": "62472648f88eece3a959d4cc1ff6e5dca11f05b1c6fb214e49ac4a9b3458ed
1e"
    }
}
```

Note： BOTTOS ABI file describes the Apis and structs needed by contract, for instances, in this case, the nodeclustermng.abi file's information is shown as following.

```
"structs" represents for the definition of structs which be used in this contract

"actions" represents for all the contract methods nane and the corresponding struct
 names which will be used in this contract

"structs" represents for the definition of structs which be used in this contract

"tables" represents that the contract information will be written to the block chai
n after the contract is called(take the effect).  It represents for the inputting p
arameters from querying the block chain, which includes the keywords and the corres
ponding structs' information.


Sample

{
    "types": [],
    "structs": [
            {
                "name": "NodeClusterReg",
                "base": "",
                "fields": {
                    "nodeIP": "string",
                    "clusterIP": "string",
                    "uuid": "string",
                    "capacity": "string"
                    }
                },
            {
                "name": "NodeClusterInfo",
                "base": "",
                "fields": {
                    "clusterIP": "string",
                    "uuid": "string",
```

```
                    "capacity": "string"
                    }
                }
    ],
    "actions": [
            {
                "action_name": "reg",
                "type": "NodeClusterReg"
            }
    ],
    "tables": [
            {
                "table_name": "nodeclusterinfo",
                "index_type": "string",
                "key_names":  [
                    "nodeIP"
                ],
                "key_types":  [
                    "string"
                ],
                "type": "NodeClusterInfo"
            }
    ]
}
```

Both Deploy BCLI CODE and ABI's command

Help information

```
./bcli contract deploy --help
NAME:
Bottos bcli tool contract deploy - contract deploy

USAGE:
    Bottos bcli tool contract deploy [command options] [arguments...]

OPTIONS:
    --name value  the contract name
    --code value  the contract's wasm file path ( includes wasm file name )
    --abi value   the contract's abi file path ( includes abi file name )
    --user value  the user account
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
bcli contract deploy	--name	contract name	Yes
	--code	contract name	Yes
	--abi	abi description file（.abi）'s path	Yes

	--user	user who signs the contract	Yes

Return Information

This will return the transaction information after the command successfully sent.

Sample

```
./bcli contract deploy --name nodeclustermng --code nodeclustermng.wasm --abi nodec
lustermng.wasm --user nodeclustermng
```

Outputs

```
Deploy contract: nodeclustermng Succeed
Trx:
{
    "version": 1,
    "cursor_num": 5730,
    "cursor_label": 2562596760,
    "lifetime": 1536565930,
    "sender": "nodeclustermng",
    "contract": "bottos",
    "method": "deploycode",
    "param": {
        "name": "nodeclustermng",
        "vm_type": 1,
        "vm_version": 1,
        "contract_code": "0061736d01000000012907600027f7f017f60017f0060027f7f0060067
f7f7f7f7f7f017f60037f7f7f017f60000060017f017f02560603656e760867657450617261 6d000003
656e76066d656d637079000403656e76066d656d736574000403656e7606..."
    },
    "param_bin": "dc0004da000e6e6f6465636c75737465726d6e67cc01cc01c5162f0061736d010
0000001290760027f7f017f60017f0060027f7f0060067f7f7f7f7f7f017f60037f7f7f017f60000060
017f017f02560603656e760867657450617261 6d000003656e76...",
    "sig_alg": 1,
    "signature": "5611a1ccaa1286616e01e93517c04d4353143b5fe3d831ce9f67c48d7a19f39c5
05e6bd8e190f65ea74f8ade93a5db528c5299b8314cfa37243a8a66df0d900a"
}
TrxHash: 5be2a59d92486a5b9cdc9ab8197f9640d0276949d08c7f8887374a9cd5b36bdc
{
    "errcode": 0,
    "msg": "trx receive succ",
    "result": {
        "trx": {
            "version": 1,
            "cursor_num": 5730,
            "cursor_label": 2562596760,
            "lifetime": 1536565930,
            "sender": "nodeclustermng",
            "contract": "bottos",
```

```
            "method": "deployabi",
            "param": "dc0002da000e6e6f6465636c75737465726d6e67c5162f0061736d0100000
001290760027f7f017f60017f0060027f7f0060067f7f7f7f7f7f017f60037f7f7f017f600000060017f
017f02560603656e76086765745061726 16d000003656e76066d656d637079000403656e76066d656d7
36574000403656e76067072696e7469000103656e76067072696e7473000203656e760b736574537472
56616c756500030305040505060004040170000005030100010 73d04066d656d6f7279020004696e697
4000705737461727 40008215f474c4f42414c5f5f7375625f495f6e6f6465636c75737465726d6e672e
63707000060ada2804f2020041004200370 28c4041004200370294404100420037029c4041004200370
2a440410042003702ac40410041003602b440410041003602b840410041003602bc40410041003602c0
40410041003602c440410041003602c840410041003602cc40410041003602d040410041003602d4404
10041003602d840410041003602dc40410041003602e040410041003602e440410041003602e8404100
41003602ec40410041003602f040410041003602f440410041003602f840410041003602fc404100410
036028041410041003602844141004100360288414100410036028c4141004100360290414100410036
029441410041003602984141004100360 29c41410041003602a041410041003602a441410041003602a
841410041003602ac41410041003602b041410041003602b441410041003602b841410041003602bc41
410041003602c041410041003602c441410041003602c841410041003602cc41410041003602d041410
041003602d4410b02000bec1b01087f4100410028020441c0196b220836020402402402402400240240
02402402400 41f200470d00200841c0096a410041801010021a20084180056a410041ba0410021a200
841c0096a418010100022051003024020054 50d00200841c0096a210020052107034020002d00001003
200041016a21002007417f6a22070d000b0b200841003a00f004200820053602f804200841003602fc0
42008200841c0096a3602f404200841f0046a200841306a1009450d0420082d00304106470d01200828
02384104470d04200841f0046a200841306a1009450d0420082d00304105470d0220082802382206410
16a410a490d03200841013a00f0040c040b200841c0096a41b0c300412310011a200841c0096a410172
210541002100034020052000 6a2107200041016a2206210020072d00000d000b200841c0096a2006100
40c040b200841063a00f0040c020b200841063a00f0040c010b0240024020082802f4042207450d0020
0841fc046a280200220020066a200841f8046a2802004d0d010b200841043a00f0040c010b024020064
50d00200720006a210020084180056a2107200621050340200720002d00003a0000200741016a210720
0041016a21002005417f6a22050d000b200841fc046a28020021000b20084180056a20066a41003a000
0200841fc046a200020066a360200200841f0046a200841306a1009450d000240024020082d00304105
470d0020082802382206 41016a41c303490d01200841013a00f0040c020b200841063a00f0040c010b0
240024020082802f4042207450d00200841fc046a280200220020066a200841f8046a2802004d0d010b
200841043a00f0040c010b20084189056a210102402006450d00200720006a2100200121072006210 50
340200720002d00003a0000200741016a2107200041016a21002005417f6a22050d000b200841fc046a
28020021000b200841fc046a200020066a360200200841 84180056a20066a41096a41003a0000200841f00
46a200841306a1009450d000240024020082d00304105470d0020082802382206 41016a41e500490d01
200841013a00f0040c020b200841063a00f0040c010b0240024020082802f4042207450d00200841fc0
46a280200220020066a200841f8046a2802004d0d010b200841043a00f0040c010b200841cb086a2102
02402006450d00200720006a2100200221072006210503402007200 02d00003a0000200741016a21072
00041016a21002005417f6a22050d000b200841fc046a28020021000b200841fc046a200020066a3602
0020084180056a20066a41cb036a41003a0000200841f0046a200841306a1009450d000240024020082
d00304105470d0020082802382203 41016a410c490d01200841013a00f0040c020b200841063a00f004
0c010b0240024020082802f4042205450d00200841fc046a280200220720036a200841f8046a2802004
d0d010b200841043a00f0040c010b200841af096a210002402003450d00200520076a21072000210520
0321060340200520072d00003a0000200541016a2105200741016a21072006417f6a22060d000b20084
1fc046a28020021070b200841fc046a200720036a360200410021072008418005 6a20036a41af046a41
003a0000200841366a41002d0086423a0000200841346a41002f0084423b0100200841002800804236 0
230200841306a4101722106034020062007 6a2105200741016a2203210720052d00000d000b20084130
6a200310044100210741002103024020082d008005450d0020084180056a4101722104410021050340 2
00420056a2106200541016a2203210520062d00000d000b0b20084180056a200310042008413 86a4100
2f0098423b0100200841002900904237023020084 1306a4101722106034020062007 6a2105200741016
a2203210720052d00000d000b200841306a200310044100210741002103024020084189056a2d000045
```

```
0d002008418a056a2104410021050340200420056a2106200541016a2203210520062d00000d000b0b2
0012003100420084134 6a41002d00a4423a0000200841002800a042360230200841306a410172210603
40200620076a2105200741016a2203210720052d00000d000b200841306a2003100441002107410021 0
30240200841cb086a2d0000450d00200841cc086a2104410021050340200420056a2106200541016a22
03210520062d00000d000b0b2002200310042008413 86a41002d00b8423a0000200841002900b042370
230200841306a4101722106034020062007 6a2105200741016a2203210720052d00000d000b20084130
6a2003100441002106024020084 1af096a2d0000450d00200841b0096a2103410021070340200320076
a2105200741016a2206210720052d00000d000b0b2000200610042008413 06a410041b10410021a2008
41306a2107034020072001 2d000022053a0000200741016a2107200141016a210120050d000b200841f
2036a2107034020072002 2d000022053a0000200741016a2107200241016a210220050d000b200841d6
046a2107034020072000 2d000022053a0000200741016a2107200041016a210020050d000b200841f00
46a41086a4180103602002008 1fc046a2201410336020041002105200841003a00f004200841dc013a
00c00920084180063b00c1092008200841c0096a3602f404024020082d00302203450d00200841306a4
1017221064100210003402006200 6a2107200041016a2205210020072d00000d000b0b410621022001
4106360200200841da013a00c309200820053a00c5092008200541087441808 0fc07714110763a00c40
9024002400240024020054 1ffff03712206450d0041052100200641066a4180104b0d01200820033a00
c6090240200641014 0d00410120066b2105200841c0096a4107722100200841306a410172210703402
00020072d00003a0000200041016a2100200741016a2107200541016a22050d000b0b200841fc046a22
0020002802002006 6a22023602000b02400240200841f2036a2d00002201450d00200841f3036a21064
1002100034020062000 6a2107200041016a2205210020072d00000d000c020b0b410021050b41032100
20082802f4042207450d00200241016a200841f0046a41086a2802004b0d00200720026a41da013a000
0200841fc046a2200200028020022074101 6a2206360200024020082802f4042202450d004108210020
0741036a200841f0046a41086a2802004b0d01200220066a2200200541187420054108744180 0fc077
1722005410876418 0fe037120054118767272220741187 3a0001200020074110763a0000200841fc04
6a2200200028020022074102 6a2202360200024020054 1fff03712206450d00410521002008280 2f40
42203450d02200220066a200841f8046a2802004b0d02200320026a20013a0000024020064101460d00
410120066b2105200320076a41036a2100200841f3036a2107034020002007 d00003a0000200041016
a2100200741016a2107200541016a22050d000b0b200841fc046a2200200028020020066a2202360200
0b02400240200841d6046a2d00002201450d00200841d7046a2106410021000340200620006a2107200
041016a2205210020072d00000d000c020b0b410021050b41032100200820 2f4042207450d01200241
016a200841f0046a41086a2802004b0d01200720026a41da013a0000200841fc046a2200200028020020
20741016a2206360200200820 2f4042202450d00410821002007410 6a200841f0046a41086a280200
4b0d01200220066a2200200541187420054108744180 0fc0771722005410876418 0fe0371200541187
6727222074118763a0001200020074110763a0000200841fc046a2200200028020022074102 6a220236
02000240200541ffff03712206450d00410521002008280 2f4042203450d02200220066a200841f8046
a2802004b0d02200320026a20013a0000024020064101460d00410120066b2105200320076a41036a21
00200841d7046a2107034020002007 d00003a0000200041016a2100200741016a2107200541016a220
50d000b0b200841fc046a2200200028020020066a22023602000b4100210020084100 2903e842370328
200841002903e0423703202008412 6a4101722105034020052000 6a2107200041016a2206210020072
d00000d000b20082d008005450d0220084180056a4101722101410021000340200120006a2107200041
016a2205210020072d00000d000c040b0b410821000b200820003a00f0044100210020084 106a41002
d00d0423a0000200841002903c842370308200841002903c042370300200841017221050340200520006
6a2107200041016a2206210020072d00000d000b2008200610040c030b410021050b0240200841206a2
00620084180056a200520082802f40420021005450d0041002100200841146a41002d00a4433a000020
0841106a41002802a0433602002008410029039843370308200841002903904337030020084101722 0
503402005200 6a2107200041016a2206210020072d00000d000b2008200610044100210 00c030b4100
2100200841146a41002d0084433a0000200841106a4100280280433602002008410029 3f8423703082
00841002903f042370300200841017221050340200520006a2107200041016a2206210020072d00000d
000b2008200610040c010b4100210020084 1c4006a41002f01f4413b0100200841c0006a41002802f04
1360200200841002903e84137033820084100290 3e041370330200841306a4101722105034020052000
6a2107200041016a2206210020072d00000d000b200841306a200610040b410121000b4100200841c01
```

```
96a36020420000bf20902047f017e0240024020002802042202450d00200028020c220341016a220420
002802084d0d010b200041023a000041000f0b200220036a2d000021022000410c6a200436020002400
240024002400240024002400240024002400240024002400240200241bb7e6a220241174b0d00024020
020e180001010101010102030405010101010101010101060107000b200141003a000002402000410
46a2802002202450d002000410c6a220328020022044102 6a2205200041086a2802004d0d080b200041
073a000041000f0b200041063a000041000f0b200141013a0000024020004104 6a2802002202450d002
000410c6a220328020022044101 6a200041086a2802004d0d070b200041043a000041000f0b20014102
3a0000024020004104 6a2802002202450d002000410c6a220328020022044102 6a200041086a2802004
d0d070b200041043a000041000f0b200141033a0000024020004104 6a2802002202450d002000410c6a
220328020022044104 6a200041086a2802004d0d070b200041043a000041000f0b200141043a0000024
0200041046a2802002202450d002000410c6a220328020022044108 6a200041086a2802004d0d070b20
0041043a000041000f0b200141053a0000024020004104 6a2802002202450d002000410c6a220328020
0220441026a2205200041086a2802004d0d070b200041043a000041000f0b200141063a000002402000
41046a2802002202450d002000410c6a220328020022044102 6a2205200041086a2802004d0d070b200
041043a000041000f0b200220046a22002d0000210220002d0001210020032005360200200141086a20
022000410874722200411874200041087441808 0fc07717241107636020041010f0b200141086a20022
0046a2d00003a0000200320032802004 1016a36020041010f0b200141086a2200200220046a22022d00
003a0000200141096a20022d00013a0000200320032802004 1026a3602002000200 2f0100220141187
420014108744180 80fc0771724110763b010041010f0b200141086a2205200220046a22002d00003a00
00200141096a20002d00013a00002001410a6a200041026a2d00003a00002001410b6a200041036a2d0
0003a0000200320032802004 1046a3602002005200528020022004118742000410874418080fc077172
20004108764180fe037120004118767272360200 41010f0b200141086a2205200220046a22002d00003
a0000200141096a20002d00013a00002001410a6a200041026a2d00003a00002001410b6a200041036a
2d00003a00002001410c6a200041046a2d00003a00002001410d6a200041056a2d00003a00002001410
e6a200041066a2d00003a00002001410f6a200041076a2d00003a0000200320032802004 1086a360200
200520052903002206423886200642288642808080808080c0ff008384200642188642808080808080e03
f832006420886428080808 0f01f838484200642088842808080f80f832006421888428080fc07838420
064228884280fe038320064238888484843703004 1010f0b200220046a22002d0000210220002d00012
100200320053602002001410 6a200220004108747222004118742000410874418080fc077172411076
36020041010f0b200220046a22002d0000210220002d0001210020032005360200200141086a2002200
041087472220041187420004108744180 80fc07717241107636020041010b0bf3010b0041040b04e061
00000041e0c1000b166e6f646520636c7573746572207265672066661696c00004180c2000b076e6f646
5495000004190c2000b0a636c757374657249500000041a0c2000b0575756964000041b0c2000b096361
706163697479000041c0c2000b117061636b20696e666f206661696c6564000041e0c2000b106e6f646
5636c7573746572696e666f000041f0c2000b157365742073747220766 16c7565206661696c65640000
4190c3000b15726567206e6f646520636c7573746572207375630000041b0c3000b23696e76616c69642
06d6574686f642066726f6d206e6f6465636c75737465726d6e6700",
            "sig_alg": 1,
            "signature": "fecde87d6c5cd6c53b3a4f7c79e6335c5d229bad22092db9188501301
63cc1bd688bfbf79b188352129ec6c1240d56f8c7c90ad16f953a8cd41a71e63aa426f2"
        },
        "trx_hash": "f55af648b151466499d19d8de3a62ab304a3ff49616e10fa12f78443ec12c4
e4"
    }
}
```


BCLI Getting CODE file and ABI File

Help information

```
./bcli contract get --help
NAME:
Bottos bcli tool contract get - get contract

USAGE:
Bottos bcli tool contract get [command options] [arguments...]

OPTIONS:
    --name value  the contract name
    --code value  the contract's wasm file path ( includes wasm file name )
    --abi value   the contract's abi file path ( includes abi file name )
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
bcli contract deploy	--name	contract name	Yes
	--code	The path that save the contract	Yes
	--abi	The path that save the abi(.abi) file	Yes

Return Information

The output gives the contract raw data and abi file information, and save the contract and abi file to corresponding path.

Sample

```
./bcli contract get --name nodeclustermng --code ~/nodeclustermng.wasm --abi ~/node
clustermng.abi
```

Outputs

==========CONTRACTCODE==========

"0061736d01000000012907600027f7f017f60017f0060027f7f0060067f7f7f7f7f017f60037f7f7f017f60
000060017f017f02560603656e760867657450617261 6d000003656e76066d656d637079000403656
e76066d656d736574000403656e76067072696e7469000103656e76067072696e7473000203656e7
60b73657453747256616c7565000303050405050600040401700000050301000 1073d04066d656d6f
7279020004696e6974000705737461727400082 15f474c4f42414c5f5f7375625f495f6e6f6465636c75
737465726d6e672e63707000060ada2804f202004100420037028c4041004200370294404100420037
7029c40410042003702a440410042003702ac40410041003602b440410041003602b840410041003
602bc40410041003602c040410041003602c440410041003602c840410041003602cc404100410036
02d040410041003602d440410041003602d840410041003602dc40410041003602e0404100410036
02e440410041003602e840410041003602ec40410041003602f040410041003602f44041004100360
2f840410041003602fc4041004100360280414100410036028441410041003602884141004100360
28c4141004100360290414100410036029441410041003602984141004100360 29c41410041003602
a041410041003602a441410041003602a841410041003602ac41410041003602b041410041003602

b441410041003602b841410041003602bc41410041003602c041410041003602c441410041003602
c841410041003602cc41410041003602d041410041003602d4410b02000bec1b01087f41004100280
20441c0196b22083602040240024002400240024002400240200041f200470d00200841c0096a4100
41801010021a20084180056a410041ba0410021a200841c0096a418010100022051003024020054 5
0d00200841c0096a2100200521070340200 02d0000100320004 1016a21002007417f6a22070d000b0
b200841003a00f004200820053602f80420084 1003602fc042008200841c0096a3602f404200841f004
6a200841306a1009450d0420082d00304106470d012008280238410447 0d04200841f0046a2008413
06a1009450d0420082d00304105470d022008280238220641016a410a490d03200841013a00f0040c
040b200841c0096a41b0c300412310011a200841c0096a41017221054100210003402005200 06a21
07200041016a2206210020072d00000d000b200841c0096a200610040c040b200841063a00f0040c0
20b200841063a00f0040c010b0240024020082802f4042207450d00200841fc046a280200220020066
a200841f8046a2802004d0d010b200841043a00f0040c010b02402006450d00200720006a21002008
4180056a21072006210503402007200 02d00003a0000200741016a2107200041016a21002005417f6
a22050d000b200841fc046a28020021000b20084180056a20066a41003a0000200841fc046a200020
066a360200200841f0046a200841306a1009450d000240024020082d00304105470d0020082802382
20641016a41c303490d01200841013a00f0040c020b200841063a00f0040c010b0240024020082802f
4042207450d00200841fc046a280200220020066a200841f8046a2802004d0d010b200841043a00f00
40c010b20084189056a210102402006450d00200720006a2100200121072006210503402007200 02
d00003a0000200741016a2107200041016a21002005417f6a22050d000b200841fc046a2802002100
0b200841fc046a200020066a36020020084180056a20066a41096a41003a0000200841f0046a20084
1306a1009450d000240024020082d00304105470d0020082802382206 41016a41e500490d0120084
1013a00f0040c020b200841063a00f0040c010b0240024020082802f4042207450d00200841fc046a2
80200220020066a200841f8046a2802004d0d010b200841043a00f0040c010b200841cb086a210202
402006450d00200720006a2100200221072006210503402007200 02d00003a0000200741016a2107
200041016a21002005417f6a22050d000b200841fc046a28020021000b200841fc046a200020066a36
020020084180056a20066a41cb036a41003a0000200841f0046a200841306a1009450d00024002402
0082d00304105470d0020082802382203 41016a410c490d01200841013a00f0040c020b200841063a
00f0040c010b0240024020082802f4042205450d00200841fc046a280200220720036a200841f8046a
2802004d0d010b200841043a00f0040c010b200841af096a210002402003450d00200520076a21072
0002105200321060340200520072d00003a0000200541016a2105200741016a21072006417f6a2206
0d000b200841fc046a28020021070b200841fc046a200720036a360200410021072008418 0056a200
36a41af046a41003a0000200841366a41002d0086423a0000200841346a41002f0084423b01002008
4100280080423602302008413 06a41017221060340200620076a2105200741016a2203210720052d
00000d000b200841306a200310044100210741002103024020082d008005450d0020084180056a41
0172210441002105034020042005 6a2106200541016a2203210520062d00000d000b0b2008418005
6a200310042008413 86a41002f0098423b0100200841002900904237023020084 1306a41017221060
340200620076a2105200741016a2203210720052d00000d000b200841306a200310044100210741 0
021030240200841890 56a2d0000450d002008418a056a210441002105034020042005 6a210620054
1016a2203210520062d00000d000b0b200120031004200841346a41002d00a4423a0000200841002
800a042360230200841306a41017221060340200620076a2105200741016a2203210720052d00000
d000b200841306a200310044100210741002103024020084 1cb086a2d0000450d00200841cc086a2
104410021050340200420056a2106200541016a2203210520062d00000d000b0b200220031004200
841386a41002d00b8423a000020084100290 0b042370230200841306a41017221060340200620076
a2105200741016a2203210720052d00000d000b200841306a200310044100210602402008 41af096a

2d0000450d00200841b0096a21034100210703402003200 76a2105200741016a2206210720052d00
000d000b0b200020061004200841306a410041b10410021a200841306a21070340200720012d0000
22053a0000200741016a2107200141016a210120050d000b200841f2036a21070340200720022d000
022053a0000200741016a2107200241016a210220050d000b200841d6046a21070340200720002d0
00022053a0000200741016a2107200041016a210020050d000b200841f0046a41086a418010360200
200841fc046a220141033602004100210520084100 3a00f004200841dc013a00c00920084180063b0
0c1092008200841c0096a3602f404024020082d00302203450d00200841306a410172210641002100
0340200620006a2107200041016a2205210020072d00000d000b0b4106210220014106360200200 8
41da013a00c309200820053a00c5092008200541087441808 0fc07714110763a00c40902400240024
00240200541ffff03712206450d00410521002006410 66a4180104b0d01200820033a00c6090240200
64101460d00410120066b2105200841c0096a4107722100200841306a4101722107034020002007 2
d00003a0000200041016a2100200741016a2107200541016a22050d000b0b200841fc046a22002000
28020020066a22023602000b02400240200841f2036a2d00002201450d00200841f3036a210641002
100034020062000 6a2107200041016a2205210020072d00000d000c020b0b410021050b410321002
0082802f4042207450d00200241016a200841f0046a41086a2802004b0d00200720026a41da013a00
00200841fc046a220020002802002007 41016a220636020002402008 2802f4042202450d004108210
0200741036a200841f0046a41086a2802004b0d01200220066a220020054118742005410874418080
fc07717220054108764180fe037120054118767272220741187 63a000120002007 4110763a0000200
841fc046a220020002802002207 41026a220236020002402005 41ffff03712206450d00410521002008
2802f4042203450d02200220066a200841f8046a2802004b0d02200320026a20013a0000024020064
101460d00410120066b2105200320076a41036a2100200841f3036a21070340200020072d00003a00
00200041016a2100200741016a2107200541016a22050d000b0b200841fc046a220020002802002 00
66a22023602000b02400240200841d6046a2d00002201450d00200841d7046a21064100210003402
00620006a2107200041016a2205210020072d00000d000c020b0b410021050b41032100208 2802f4
042207450d01200241016a200841f0046a41086a2802004b0d01200720026a41da013a0000200841f
c046a220020002802002207 41016a220636020020082802f4042202450d00410821002007 41036a20
0841f0046a41086a2802004b0d01200220066a220020054118742005410874418080fc07717220054
108764180fe037120054118767272220741187 63a000120002007 4110763a0000200841fc046a2200
200028020022074102 6a220236020002402005 41ffff03712206450d00410521002008 2802f40422034
50d02200220066a200841f8046a2802004b0d02200320026a20013a0000024020064101460d004101
20066b2105200320076a41036a2100200841d7046a21070340200020072d00003a0000200041016a
2100200741016a2107200541016a22050d000b0b200841fc046a220020002802002 0066a220236020
00b4100210020084100290 3e84237032820084100290 3e04237032020084120 6a410172210503402
00520006a2107200041016a2206210020072d00000d000b20082d008005450d0220084180056a410
172210141002100034020012000 6a2107200041016a2205210020072d00000d000c040b0b4108210
00b200820003a00f00441002100200841106a41002d00d0423a0000200841002903c8423703082008
41002903c04237030020084101722105034020052000 6a2107200041016a2206210020072d00000d
000b2008200610040c030b410021050b0240200841206a200620084180056a200520082802f404200
21005450d00410021002008411 46a41002d00a4433a0000200841106a41002802a04336020020084
100290398433703082008410029039043370300200841017221050340200520006a2107200041016
a2206210020072d00000d000b200820061004410021000c030b410021002008411 46a41002d00844
33a0000200841106a41002802804336020020084100290 3f84237030820084100290 3f04237030020
084101722105034020052000 6a2107200041016a2206210020072d00000d000b2008200610040c01
0b41002100200841c4006a41002f01f4413b0100200841c0006a41002802f041360200200841002903

e84137033820084100290 3e041370330200841306a41017221050340200520006a2107200041016a2206210020072d00000d000b200841306a200610040b410121000b4100200841c0196a3602042000 0bf20902047f017e024002402000280204220245 0d00200028020c220341016a22042000280208 4d0d010b200041023a000041000f0b200220036a2d00002102200041 0c6a200436020002400240024002 4002400240024002400240024002400240024002400240200241bb7e6a220241174b0d00024020020e18 000101010101010203040501010101010101010101060107000b200141003a00000240200041046a 2802002202450d002000410c6a220328020022044102 6a2205200041086a2802004d0d080b200041 073a000041000f0b200041063a000041000f0b200141013a0000024020004104 6a2802002202450d0 02000410c6a220328020022044101 6a200041086a2802004d0d070b200041043a000041000f0b2001 41023a00000240200041046a2802002202450d002000410c6a220328020022044102 6a200041086a 2802004d0d070b200041043a000041000f0b200141033a00000240200041046a2802002202450d002 000410c6a220328020022044102 6a200041086a2802004d0d070b200041043a000041000f0b200141 043a00000240200041046a2802002202450d002000410c6a220328020022044108 6a200041086a28 02004d0d070b200041043a000041000f0b200141053a00000240200041046a2802002202450d00200 0410c6a220328020022044102 6a2205200041086a2802004d0d070b200041043a000041000f0b2001 41063a00000240200041046a2802002202450d002000410c6a220328020022044102 6a2205200041 086a2802004d0d070b200041043a000041000f0b200220046a22002d0000210220002d00012100200 320053602002001410 86a200220004108747222004118742000410874418080fc0771724110763602 0041010f0b200141086a200220046a2d00003a0000200320032802004101 6a36020041010f0b20014 1086a2200200220046a22022d00003a0000200141096a20022d00013a0000200320032802004102 6a360200200020002f0100220141187420014108744180 80fc0771724110763b010041010f0b2001410 86a2205200220046a22002d00003a0000200141096a20002d00013a00002001410a6a200041026a2 d00003a00002001410b6a200041036a2d00003a0000200320032802004104 6a360200200520052800 2002200411874200041087441808 0fc07717220004108764180fe037120004118767272360200410 1 0f0b200141086a2205200220046a22002d00003a0000200141096a20002d00013a00002001410a6a2 00041026a2d00003a00002001410b6a200041036a2d00003a00002001410c6a200041046a2d00003 a00002001410d6a200041056a2d00003a00002001410e6a200041066a2d00003a00002001410f6a20 0041076a2d00003a0000200320032802004108 6a360200200520052903002206423886200642288 6 42808080808080c0ff00838420064218864280808080e03f832006420886428080808 0f01f8384842 00642088842808080f80f832006421888428080fc07838420064228884280fe038320064238888484 8 437030041010f0b200220046a22002d0000210220002d00012100200320053602002001410 86a2002 2000410874722200411874200041087441808 0fc07717241107636020041010f0b200220046a22002 d0000210220002d00012100200320053602002001410 86a200220004108747222004118742000410 874418080fc07717241107636020041010b0bf3010b0041040b04e06100000041e0c1000b166e6f646 520636c7573746572207265672066616 96c00004180c2000b076e6f6465495000004190c2000b0a63 6c7573746572495000004 1a0c2000b0575756964000041b0c2000b0963617061636974790000041c0c 2000b117061636b20696e666f206661696c6564000041e0c2000b106e6f6465636c7573746572696e 666f000041f0c2000b1573657420737472207661 6c7565206661696c656400004190c3000b1572656 7206e6f646520636c7573746572207375630000 41b0c3000b23696e76616c6964206d6574686f6420 66726f6d206e6f6465636c75737465726d6e6700"

==========ABI==========

{ "types": [], "structs": [{ "name": "NodeClusterReg", "base": "", "fields": { "nodeIP": "string", "clusterIP": "string", "uuid": "string", "capacity": "string" } }, { "name": "NodeClusterInfo", "base": "", "fields": { "clusterIP": "string", "uuid": "string", "capacity": "string" } }], "actions": [{ "action_name": "reg", "type": "NodeClusterReg" }], "tables": [{ "table_name": "nodeclusterinfo", "index_type": "string", "key_names": ["nodeIP"], "key_types": ["string"], "type": "NodeClusterInfo" }] }

BCLI get Contract TABLE Information

Help information

```
./bcli gettable --help
NAME:
bcli gettable - get table info

USAGE:
    bcli gettable [command options] [arguments...]

OPTIONS:
    --contract value  contract name (default: "usermng")
    --table value     table name
    --key value       Key value
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
bcli gettable	--contract	contract name	Yes
	--table	Table name that is required to query block chain（refer to the description of 'table of abi file')	Yes
	--key	Keyword name that is required to query block chain（refer to the description of 'table of abi file')	Yes

Return Information

This will return the Transaction information sent by BCLI.

Sample

```
(No yet, the contract is under debugging)
```

3. BCLI Candidate Node Voting Command

Bcli candidate node voting chiefly realize the functions of: register the node as the producer, unregister the producer node, let part/all the producer nodes' information, vote(vote one node as the producer), cancel vote, etc,.

Help information

```
./bcli delegate --help
NAME:
Bottos bcli tool delegate - for delegate operations

USAGE:
    Bottos bcli tool delegate command [command options] [arguments...]

COMMANDS:
    reg         reg delegate
    unreg       unreg delegate
    list        list delegates
    vote        Vote for producers
    cancelvote  cancel vote for producers

OPTIONS:
    --help, -h  show help
```

Commandline Function Description

chief command	parameter list	Parameter Description
./bcli delegate	reg	register as the producer
./bcli delegate	unreg	unregister the producer
./bcli delegate	list	watch the producer list
./bcli delegate	vote	vote for the producer
./bcli delegate	cancelvote	cancel vote

BCLI Registing as Producer Command

Help information

```
./bcli delegate reg --help
NAME:
Bottos bcli tool delegate reg - reg delegate

USAGE:
    Bottos bcli tool delegate reg [command options] [arguments...]

OPTIONS:
    --account value      account name
    --signkey value      public sign key (default: "0454f1c2223d553aa6ee53ea1ccea8b
7bf78b8ca99f3ff622a3bb3e62dedc712089033d6091d77296547bc071022ca2838c9e86dec29667cf7
40e5c9e654b6127f")
    --location value     location
    --description value  description
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
bcli delegate reg	--account	user name	Yes
	--signkey	Public key defined by user（Default is the inside default value）	Yes
	--location	The city name of voter	No
	--description	The description defined by user	No

Return Information

This will return the Transaction information sent by BCLI.

Sample

```
./bcli delegate reg --account lyp --location "SHANGHAI" --description "Reg lyp as a
 producer"
```

Outputs

```
Transfer Succeed:
Trx:
{
    "version": 1,
    "cursor_num": 6115,
    "cursor_label": 789697330,
    "lifetime": 1536567085,
    "sender": "bottos",
    "contract": "bottos",
    "method": "regdelegate",
    "param": "dc0004da00036c7970da00823034353466316332323233643535336161366565353365613163636561386237626637386238636139396633666636323261336262336536326465646337313230383930333364363039316437373239363534376263303731303232636132383338633965383664656332393636376366373430653563396536353462363132376 6da00085348414e47484149da0015526567206c797020617320612070726f6475636572",
    "param_bin": "dc0004da00036c7970da00823034353466316332323233643535336161366565353365613163636561386237626637386238636139396633666636323261336262336536326465646337313230383930333364363039316437373239363534376263303731303232636132383338633965383664656332393636376366373430653563396536353462363132376 6da00085348414e47484149da0015526567206c797020617320612070726f6475636572",
    "sig_alg": 1,
    "signature": "eb0f741d4466c8d650c867dd3da988d2b81fc7c035018a461c01b9adab0767177239a041c9e6835e0952d2a95ab48900255a97ef8145a61115b286f1bbbfd3bb"
}
TrxHash: 816dc08c5cb3c4e059d04ac4842ceb583b34c58c865311c889d2d9d9acf9658b
```

Note： The precondition before this command be used is: the genesis node must transfer the generating block authority in secceed.

Please refer to the command './bcli genesis blkprodtrans --sender lyp --actblknum 10'

BCLI Unregisting Producer Command

Help information

```
./bcli  delegate unreg  --help

NAME:
Bottos bcli tool delegate unreg - unreg delegate

USAGE:
    Bottos bcli tool delegate unreg [command options] [arguments...]

OPTIONS:
    --account value  account
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
bcli delegate unreg	--account	user name	Yes

Return Information

This will return the Transaction information sent by BCLI.

Sample

```
./bcli delegate unreg --account lyp
```

Outputs Transfer Succeed: Trx: { "version": 1, "cursor_num": 6237, "cursor_label": 2162498930, "lifetime": 1536567466, "sender": "bottos", "contract": "bottos", "method": "unregdelegate", "param": "dc0001da00036c7970", "param_bin": "dc0001da00036c7970", "sig_alg": 1, "signature": "48e3f949527851c13e9a82e5eb5bd605548ca4491ea05f3afed8405cab6985c037b6d59175d7880658da9bad33ab61e14da77b846305913d1b9326afd0b326dd" } TrxHash: 8eacc1d64d722565b0033ca7afbd090ab73ae56f6ab2bf10fe934a3d305ffd6f

BCLI Listing Current Producer Nodes Command

Help information

```
./bcli delegate list --help
NAME:
Bottos bcli tool delegate list - list delegates

USAGE:
```

```
    Bottos bcli tool delegate list [command options] [arguments...]

OPTIONS:
    --limit value  (default: 100)
    --start value  (default: 0)
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
bcli delegate list	--start	Output the head N records	No
	--limit	Output from head N record and stop within the limits	No

Return Information

This will return the Transaction information sent by BCLI.

Sample

```
./bcli delegate list
```

Outputs

```
[
    {
        "account_name": "adhil",
        "active": false,
        "desc": "",
        "last_confirmed_block_num": 6358,
        "last_slot": 1147647,
        "location": "",
        "report_key": "0454f1c2223d553aa6ee53ea1ccea8b7bf78b8ca99f3ff622a3bb3e62ded
c712089033d6091d77296547bc071022ca2838c9e86dec29667cf740e5c9e654b6127f",
        "total_missed": 0
    },
    {
        "account_name": "albireo",
        "active": false,
        "desc": "",
        "last_confirmed_block_num": 6377,
        "last_slot": 1147667,
        "location": "",
        "report_key": "0454f1c2223d553aa6ee53ea1ccea8b7bf78b8ca99f3ff622a3bb3e62ded
c712089033d6091d77296547bc071022ca2838c9e86dec29667cf740e5c9e654b6127f",
        "total_missed": 0
    },
    .....
```

```
    ....
    .... (略)
```

BCLI Producer Voting Command

Help information

```
./bcli delegate vote --help
NAME:
Bottos bcli tool delegate vote - Vote for producers

USAGE:
    Bottos bcli tool delegate vote [command options] [arguments...]

OPTIONS:
    --voter value     voter account
    --delegate value  delegate account
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
bcli delegate vote	--voter	the voter	Yes
	--delegate	vote to which producer node（must vote to the nodes which has been already registerd as the producer node）	Yes

Return Information

This will return the Transaction information sent by BCLI.

Sample

```
./bcli delegate vote --voter bottos --delegate lyp
```

Outputs

```
Transfer Succeed:
Trx:
{
    "version": 1,
    "cursor_num": 6616,
    "cursor_label": 1568844234,
    "lifetime": 1536568645,
    "sender": "bottos",
    "contract": "bottos",
    "method": "votedelegate",
```

```
    "param": "dc0003cc01da0006626f74746f73da00036c7970",
    "param_bin": "dc0003cc01da0006626f74746f73da00036c7970",
    "sig_alg": 1,
    "signature": "86968717552104084b995c8ac46aca0f20fde5535daaf6f634af3cef9ecef8045
08577ee0c4c5c41f33e10feec45c22fccb5569e712fa62149f38d816d76ce35"
}
```

BCLI Producer Cancel Voting Command

Help information

```
./bcli delegate cancelvote --help
NAME:
Bottos bcli tool delegate cancelvote - cancel vote for producers

USAGE:
    Bottos bcli tool delegate cancelvote [command options] [arguments...]

OPTIONS:
    --voter value     voter account
    --delegate value  delegate account
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
bcli delegate cancelvote	--voter	The voter	Yes
	--delegate	cancel vote to which producer node（must vote to the nodes which has been already registerd as the producer node）	Yes

Return Information

This will return the Transaction information sent by BCLI.

Sample

```
./bcli delegate cancelvote --voter bottos --delegate lyp
```

Outputs

```
Transfer Succeed:
Trx:
{
    "version": 1,
    "cursor_num": 6690,
    "cursor_label": 482796464,
```

```
    "lifetime": 1536568876,
    "sender": "bottos",
    "contract": "bottos",
    "method": "votedelegate",
    "param": "dc0003cc00da0006626f74746f73da00036c7970",
    "param_bin": "dc0003cc00da0006626f74746f73da00036c7970",
    "sig_alg": 1,
    "signature": "946291c599105d2906d004b73fbec002e4ae2d8e113a4d71bbc17fa7129affb32
4ee6b6e1a101f9da217eee14ff1bac697f55bb29e634cdd4eda20b33801d116"
}
TrxHash: f90e5cb5c6e5990c6bceb6e0fd7ec5a2f629760dd809b7fbd9c2832f916319ee
```

4. BCLI Block Information Getting Command

The BCLI block information chiefly includes: Getting current block information, block head information.

BCLI Getting Block Header Information command

Help information

```
./bcli getinfo --help
NAME:
    bcli getinfo - get chian info

USAGE:
    bcli getinfo [arguments...]
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
./bcli getinfo	(Null)	(Null)	(Null)
	(Null)	(Null)	(Null)

Return Information

The output gives the latest block header of realtime.

Sample

```
./bcli getinfo
```

Outputs

```
==Chain Info==

{
    "head_block_num": 7047,
```

```
    "head_block_hash": "34460313889fc988137e42394009df8880e5e82118fcae4579fc0e33d91
3309e",
    "head_block_time": 1536569883,
    "head_block_delegate": "sulafat",
    "cursor_label": 3641913502,
    "last_consensus_block_num": 7026,
    "chain_id": "5c7c2ea85df042747b38b183c84c8313b499177ed4abbf29d0947f4908934941"
}
```

BCLI Getting Latest block Information Command

Help information

```
./bcli getblock --help
NAME:
    bcli getblock - get block info

USAGE:
    bcli getblock [command options] [arguments...]

OPTIONS:
    --num value   get block by number (default: 100)
    --hash value  get block by hash
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
./bcli getblock	(Null)	(Null)	(Null)
	(Null)	(Null)	(Null)

Return Information

```
The output gives the latest block information of realtime.
```

Sample

```
./bcli getblock

Outputs

==Block Info==

{
    "prev_block_hash": "6de5f9fb40511d307a55f5838e936a4d8c211177a0750287010bc7e6793
5954f",
    "block_num": 100,
    "block_hash": "e97df19ae9b73ba46b5f5681b159f87a56a4aa8426dc0407f6ea4ed12fb1857d
",
```

```
    "cursor_block_label": 800163197,
    "block_time": 1536548940,
    "trx_merkle_root": "0000000000000000000000000000000000000000000000000000000000000000
00000",
    "delegate": "vega",
    "delegate_sign": "fa3504d657168fb9b16ae9ee3d281ead80adb43fbb3859213198f856732b1
2c33459b2ff820af6fdf15c9dce7dde9a29fe0446a09568b1a34a3909cd64a73815",
    "trxs": null
}
```

5. BCLI Genesis Node Functions Command

The BCLI genesis node chiefly includes: Add original producers，transfer generating block autority, cancel the operation authority of genesis node.

Help information

```
./bcli genesis --help

NAME:
Bottos bcli tool genesis - for genesis node operations

USAGE:
    Bottos bcli tool genesis command [command options] [arguments...]

COMMANDS:
    setdelegate      set delegate
    blkprodtrans     for genesis node transfering the permission of producing block
s
    cancelprevilige  cancel genesis node permission

OPTIONS:
    --help, -h  show help
```

Commandline Function Description

chief command	parameter list	Parameter Description
./bcli genesis	setdelegate	The original procuders be designated in genesis node
./bcli genesis	blkprodtrans	transfer the authority of generating blocks
./bcli genesis	cancelprevilige	cancel the operation authority of genesis node

BCLI Genesis Node Adding Original Producer Functions Command

Help information

```
./bcli genesis setdelegate --help
NAME:
Bottos bcli tool genesis setdelegate - set delegate
```

```
USAGE:
    Bottos bcli tool genesis setdelegate [command options] [arguments...]

OPTIONS:
    --sender value       sender account
    --account value      account name
    --signkey value      public sign key (default: "0454f1c2223d553aa6ee53ea1ccea8b
7bf78b8ca99f3ff622a3bb3e62dedc712089033d6091d77296547bc071022ca2838c9e86dec29667cf7
40e5c9e654b6127f")
    --location value     location
    --description value  description
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
./bcli genesis setdelegate	--sender	The signer（user name must be regietered at first，default value will be inside bottos account）	No
	--account	The producer that will be added	Yes
	--signkey	user's public key	No
	--location	The city designated by user	No
	--description	The description defined by user	No

Return Information

This will return the Transaction information sent by BCLI.

Sample

```
./bcli genesis setdelegate --sender bottos --account lyp
```

Outputs

```
Transfer Succeed:
Trx:
{
    "version": 1,
    "cursor_num": 7356,
    "cursor_label": 2458267119,
    "lifetime": 1536570943,
    "sender": "bottos",
    "contract": "bottos",
    "method": "setdelegate",
    "param": "",
    "param_bin": "",
```

```
    "sig_alg": 1,
    "signature": "fe901957f6b55a41aef893ec6c285bd8f3c6624a1a0950d96cea4dab5896fe3d4
476f602b910cdd7d51e9f74ee69af9c17b126ca37a85c040d6aee0827e33e3d"
}
TrxHash: ec4654e3edb61c661ed63cf6b02d44cd255ca7a604c463074fb1307f70025181
```

BCLI Genesis Node Transfer the Authority Function Command

Help information

```
./bcli genesis blkprodtrans --help
NAME:
Bottos bcli tool genesis blkprodtrans - for genesis node transfering the permission
 of producing blocks

USAGE:
    Bottos bcli tool genesis blkprodtrans [command options] [arguments...]

OPTIONS:
    --sender value     sender account
    --actblknum value  the threshold number of active block for which genesis node
transfering the permission of producing blocks after exceeding it (default: 0)
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
./bcli genesis blkprodtrans	--sender	The signer（user name must be regietered at first，default value will be inside bottos account）	No

Return Information

This will return the Transaction information sent by BCLI.

Sample

```
 ./bcli genesis blkprodtrans --sender lyp
```

Outputs

```
Transfer Succeed:
Trx:
{
    "version": 1,
    "cursor_num": 7431,
    "cursor_label": 974139076,
    "lifetime": 1536571174,
    "sender": "lyp",
```

```
    "contract": "bottos",
    "method": "blkprodtrans",
    "param": "",
    "param_bin": "",
    "sig_alg": 1,
    "signature": "393abd00bc0c7bb2da1bec6fb03d755e553e17cddfd7b4e40daecdf372d188854
517d099a5d166057e6e85e7b4edae98d5e75d48ec09f891d1f6eca3c94b979f"
}
TrxHash: 2e25dd5f8bc7c598a3c6b81f245b6413143973ab9c9bc46f695a9ef6b33b4507
```

BCLI Cancel the Genesis Node Operation Authority Command

Help information

```
./bcli genesis cancelprevilige --help
NAME:
Bottos bcli tool genesis cancelprevilige - cancel genesis node permission

USAGE:
    Bottos bcli tool genesis cancelprevilige [command options] [arguments...]

OPTIONS:
    --sender value  sender account
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
./bcli genesis cancelprevilige	--sender	The signer（the user name must be registered at first,the default value is the bottos account）	No

Return Information

This will return the Transaction information sent by BCLI.

Sample

```
 ./bcli genesis cancelprevilige --sender lyp
```

Outputs

```
Transfer Succeed:
Trx:
{
    "version": 1,
    "cursor_num": 7479,
    "cursor_label": 4050849039,
    "lifetime": 1536571324,
```

```
    "sender": "lyp",
    "contract": "bottos",
    "method": "cancelprevilige",
    "param": "",
    "param_bin": "",
    "sig_alg": 1,
    "signature": "bb88393a4d1b526c079f2c3d0d38bbbc90f46084fb0c3e21cde9a5eaed5e0ce14
0a7a5dd30192ae63df489b46fa54d0ad4ff077c5f7286e559221e64c5d1ee9f"
}
TrxHash: 794fb7f03c8d9ae0545660d66394f45e5cc261966fc66342dab72a70a6bbd2ea
```

6. BCLI Transferring Function Command

BCLI transfer functions are responsible for transferring BTO from user designated by 'from' to user designated by 'to' account.

BCLI transferring Command

Help information

```
./bcli transfer --help
NAME:
bcli transfer - for user transfering bto

USAGE:
    bcli transfer [command options] [arguments...]

OPTIONS:
    --from value    the [from] user account (default: "bottos")
    --to value      the [to] user account (default: "bottos")
    --amount value  the amount of bto (default: "0")
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
./bcli transfer	--from	transferring user	No
	--to	target user	No
	--amount	BTO amount to be transferred	No

Return Information

This will return the Transaction information sent by BCLI.

Sample

```
./bcli transfer --from bottos --to lyp --amount 2
```

Outputs

```
Transfer Succeed
    From: bottos
    To: lyp
    Amount: 200000000
Trx:
{
    "version": 1,
    "cursor_num": 7762,
    "cursor_label": 117036287,
    "lifetime": 1536572206,
    "sender": "bottos",
    "contract": "bottos",
    "method": "transfer",
    "param": {
        "from": "bottos",
        "to": "lyp",
        "value": 2
    },
    "param_bin": "dc0003da0006626f74746f73da00036c7970c500200000000000000000000000000000000000000000000000000000000000000000bebc200",
    "sig_alg": 1,
    "signature": "3aa45652b368827fd5480721de1588ed08c2289de2cfa97ce352d65c8f1acc6e4f04b2c8492caacedb61097dea2728019d4c40c932d763ed7e26d051ca27a188"
}
TrxHash: 86cbc8bd54d85fc817f3675103e62c5693fe7ef1520f2218f57c6406be05b46b
```

7. BCLI Transaction Committing and Querying Commands

BCLI Transaction committing and querying commands take the responsibilities of committing a raw transaction defined by user himself to the blockchain, or querying an existing contract by the TrxHash.

Help information

```
./bcli transaction --help

NAME:
Bottos bcli tool transaction - get or push transactions

USAGE:
Bottos bcli tool transaction command [command options] [arguments...]

COMMANDS:
    get   get tx details
    push  push transaction

OPTIONS:
--help, -h  show help
```

Commandline Function Description

chief command	parameter list	Parameter Description
./bcli transaction	get	Get the specific transaction information via TrxHash
./bcli transaction	push	Push a Transaction

BCLI Transaction Querying Command

Help information

```
./bcli transaction get --help

NAME:
    Bottos bcli tool transaction get - get tx details

USAGE:
    Bottos bcli tool transaction get [command options] [arguments...]

OPTIONS:
    --trxhash value
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
./bcli transaction get	--trxhash	Transaction hash index	Yes

Return Information

This will return the Transaction information of the designated Txhash sent by BCLI.

Sample

```
./bcli transaction get --trxhash 86cbc8bd54d85fc817f3675103e62c5693fe7ef1520f2218f5
7c6406be05b46b
```

Outputs

```
{
    "contract": "bottos",
    "cursor_label": 117036287,
    "cursor_num": 7762,
    "lifetime": 1536572206,
    "method": "transfer",
    "param": null,
    "sender": "bottos",
    "sig_alg": 1,
    "signature": "3aa45652b368827fd5480721de1588ed08c2289de2cfa97ce352d65c8f1acc6e4
f04b2c8492caacedb61097dea2728019d4c40c932d763ed7e26d051ca27a188",
```

```
    "version": 1
}
```

BCLI Transaction Pushing Command

Help information

```
./bcli transaction push --help
NAME:
    Bottos bcli tool transaction push - push transaction

USAGE:
    Bottos bcli tool transaction push [command options] [arguments...]

OPTIONS:
    --sender value    acocunt name
    --contract value  contract name
    --method value    method name
    --param value     param value
    --sign value      sign value
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
./bcli transaction push	--sender	The signer（default value is the bottos account）	No
./bcli transaction push	--contract	contract name	Yes
./bcli transaction push	--method	contract's method name	Yes
./bcli transaction push	--param	parameter key value pair	Yes
./bcli transaction push	--sign	public key defined by user（default value is defined by inside）	No

Return Information

This will return the Transaction information of the designated Txhash sent by BCLI.

Sample

```
（Omitted.）
```

8. BCLI Wallet Command

Wallet functions take the chief responsibilities for user to push the Transactions of actions: create wallet, bind account, unlok the wallet, lock the wallet, query wallet, etc,.

Help information

```
./bcli wallet --help

NAME:
Bottos bcli tool wallet - For wallet operations

USAGE:
Bottos bcli tool wallet command [command options] [arguments...]

COMMANDS:
    generatekey  generate key pairs
    create       create wallet
    lock         lock wallet
    unlock       unlock wallet
    list         list wallet
    listkey      listkey of wallet

OPTIONS:
--help, -h  show help
```

Commandline Function Description

chief command	parameter list	Parameter Description
./bcli wallet	generatekey	Create a new public and private key pair
./bcli wallet	create	Create wallet
./bcli wallet	lock	Lock wallet
./bcli wallet	unlock	Unlock wallet
./bcli wallet	list	List all wallet
./bcli wallet	listkey	List the key pair of wallet（Wallet must be unlocked firstly）

BCLI Create Wallet's Public and Private Key Command

Help information ./bcli wallet generatekey --help

```
NAME:
    Bottos bcli tool wallet generatekey - generate key pairs

USAGE:
    Bottos bcli tool wallet generatekey [arguments...]
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
./bcli wallet generatekey	(Null)	(Null)	(Null)

Return Information

This will output a new generated public and private key pair.

Sample

```
./bcli wallet generatekey
```

Outputs { "private_key": "b8b890ebc315a8e1c3a6f7b78977d68ca1e9274c986314ccbe967b964cf68b66", "public_key": "0485fccecf8c8e6260d8558e1a61adca3a888127e34ba0d052dfeb1c31d419bf0494482e7e8a447d63394cff713fc00aa8e64c24b73a8173661a91884b71407bce" }

BCLI Creating Wallet Command

Help information

```
./bcli wallet create --help

NAME:
Bottos bcli tool wallet create - create wallet

USAGE:
Bottos bcli tool wallet create [command options] [arguments...]

OPTIONS:
--account value  account name
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
./bcli wallet create	--account	The user name binded by wallet	Yes

Return Information

This will return the user's keystore information file, which will be saved under path of /home/bottos/bot, format as .keystore of the file.

Sample

```
./bcli wallet create --account lyp
```

Outputs

Please input your password for your wallet:

Please input your private key for your wallet:

{ "wallet_name": "lyp.keystore" }

Note： In this case, user should input the wallet password and his used private key by manual.

BCLI Lock Wallet Command

Help information

```
./bcli wallet lock --help

NAME:
Bottos bcli tool wallet lock - lock wallet

USAGE:
Bottos bcli tool wallet lock [command options] [arguments...]

OPTIONS:
--account value  account name
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
./bcli wallet lock	--account	The user name binded by wallet	Yes

Return Information

This will output the wallet locked state flag.

Sample

./bcli wallet lock --account lyp

Outputs

./bcli wallet lock --account lyp { "lock": true }

BCLI 解锁钱包命令行

Help information

```
./bcli wallet unlock --help

NAME:
Bottos bcli tool wallet unlock - unlock wallet

USAGE:
Bottos bcli tool wallet unlock [command options] [arguments...]
```

```
OPTIONS:
--account value  account name
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
./bcli wallet unlock	--account	The user name binded by wallet	Yes

Return Information

This will output the wallet unlocked state flag.

Sample

```
 ./bcli wallet unlock --account lyp
```

Outputs

```
Please input your password for your wallet:

{
    "unlock": true
}
```

Note： In this case, user should input the wallet password and his used private key from generating wallet by manual.

BCLI Watching All Wallet List Command

Help information

```
./bcli wallet list --help

NAME:
Bottos bcli tool wallet list - list wallet

USAGE:
Bottos bcli tool wallet list [arguments...]
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
./bcli wallet list	(Null)	(Null)	(Null)

Return Information

This will output the wallet List.

Sample

```
 ./bcli wallet list
```

Outputs

```
[
    {
        "account_name": "lyp",
        "public_key": "0454f1c2223d553aa6ee53ea1ccea8b7bf78b8ca99f3ff622a3bb3e62ded
c712089033d6091d77296547bc071022ca2838c9e86dec29667cf740e5c9e654b6127f",
        "wallet_path": "/home/bottos/bot/lyp.keystore"
    }
]
```

BCLI Watching All the Public and Private Key Command

Note： This command must be used after the wallet be under the unlocked state.

Help information

```
./bcli wallet listkey --help

NAME:
Bottos bcli tool wallet listkey - listkey of wallet

USAGE:
    Bottos bcli tool wallet listkey [command options] [arguments...]

OPTIONS:
    --account value  account name
```

Parameter Description

chief command	parameter list	Parameter Description	mandatory
./bcli wallet listkey	--account	The user name binded by wallet	(Null)

Return Information

This will output the public and private key pair of the wallet's binded account.

Sample

```
./bcli wallet listkey --account lyp
```

Outputs

{ "account_name": "lyp", "private_key":
"b799ef616830cd7b8599ae7958fbee56d4c8168ffd5421a16025a398b8a4be45", "public_key":
"0454f1c2223d553aa6ee53ea1ccea8b7bf78b8ca99f3ff622a3bb3e62dedc712089033d6091d772965
47bc071022ca2838c9e86dec29667cf740e5c9e654b6127f" }

Write smart contract in C++

1. Compilation of contract（C/C++ version）

C/C++'s contract compilation tool chain is under following library:

https://github.com/bottos-project/contract-tool-cpp.git

This tool is used for compiling C/C++ contracts, generating wasm/wast files, and scanning and generating the abi file. We will describe it in following 2 samples from the library.

1. Sample of testHelloWorld

Put the contract code file put into a directory, and put that directory into the directory of ""“contract-tool-cpp", like sample of testHelloWorld.

As you need to compile the contract, go through into the directory of contract, like come into testHelloWorld's directory, then use the follwing command to compile the contract, while after then the 'testHelloWorld.wast' and 'testHelloWorld.wasm' will be generated under the same directory:

```
python ../gentool.py wasm testHelloWorld.cpp
```

Since this sample does not related to abi, we will describe what does the abi file be generated like in following cases.

2. Sample of testRegUser

Like above case, compile the contract as coming into the directory of 'testRegUser' and compiling the contract via using following command:

```
python ../gentool.py wasm testRegUser.cpp
```

Now we mainly focus on how does the contract abi file be generated. By now, let us introduce what is abi file composed of:

- structs： The struct descriptions from scanning, which will be used later.
- actions： The description of contract methods, in which the action_name stands for the name of the function, type stands for the parameters that called by contract；
- tables： The contract persistence data drovider description, where table_name is the table's name, index_type is the type of index, key_names and Key_types are the name and type of the key value, and type is the structure definition of the content.

The ABI file is generated by scanning the hpp file, in which it tells the scanner specific definition through comments.

- "//@abi action reguser"：

  A method reguser is defined, and the corresponding entry parameter is defined as UserInfo;

- "//@abi table userinfo:[index_type:string， "key_names:userName， key_types:string] "：

  A table is defined, and the structure of the table content is defined as Userbaseinfo.

Under the Testreguser folder, you can scan the ABI file for the hpp file by using the following command:

```
python ../gentool testRegUser.hpp
```

2. contract's editing（C/C++ version）

2.1. A simplest contract

The entry function of the contract is the start function, and we create a simple contract, that is, when the contract is tuned, print "Hello World in Start":

```
#include "contractcomm.hpp"

int start(char* method)
{
    myprints("hello world in start");

    return 0;
}
```

The "Contractcomm.hpp" is a common basic interface declaration file. Call this contract, and the "Hello World in Start" will be printed in the log of the node:

```
2018-08-06 16:05:20 [INF] vm/wasm/exec/env_func.go:390 prints(): VM: func prints: h
ello world in start
```

2.2. Gets the method that invokes the contract

As calling a contract, we need to specify a specific method for calling the contract, that is, the following "method" parameter "Test_method":

```
{"version":1, "cursor_num":28,"cursor_label":3745260307,"lifetime":15270819998,"sen
der":"example", "contract":"example", "method":"test_method", "param":"", "sig_alg"
:1, "signature":""}
```

This method is passed into the contract through the entry of the start function, that is, the following method parameter, then we add the statement to print the parameter in the contract:

```
#include "contractcomm.hpp"

int start(char* method)
{
    myprints("hello world in start");
    myprints(method);

    return 0;
}
```

We construct a transaction that sets method to a specific string, the following "Test_method": (The exact value of the signature is omitted here, the same below)

```
{"version":1, "cursor_num":28,"cursor_label":3745260307,"lifetime":15270819998,"sen
der":"example", "contract":"example", "method":"test_method", "param":"", "sig_alg"
:1, "signature":""}
```

From calling this contract, there will be the following log output: Print the set "method" parameter, that is, "Test_method":

```
2018-08-07 14:40:22 [INF] vm/wasm/exec/env_func.go:412 prints(): VM: func prints: h
ello word in start

2018-08-07 14:40:22 [INF] vm/wasm/exec/env_func.go:412 prints(): VM: func prints: t
est_method
```

We can provide different contract implementations in the contract according to method, that is, by comparing the method parameters and then going to different branch branches, such as the "add" and "Del" methods provided in the following contracts:

```
#include "contractcomm.hpp"
#include "string.hpp"

int start(char* method)
{
    myprints("hello word in start");
    myprints(method );

    if (0 == strcmp("add", method))
    {
        myprints("it is method add");
    }
    else if (0 == strcmp("del", method))
    {
```

```
        myprints("it is method del");
    }

    return 0;
}
```

2.3. Gets the call contract parameter

When calling a contract, in addition to specifying the method of calling the contract, you also need to bring the corresponding parameters, that is, the following "param" parameter:

```
{"version":1, "cursor_num":28,"cursor_label":3745260307,"lifetime":15270819998,"sen
der":"example", "contract":"example", "method":"test_method", "param":"", "sig_alg"
:1, "signature":""}
```

This parameter can be obtained through the "Getparam" interface in the contract, for example:

```
#include "contractcomm.hpp"
#define ERROR_PARAM (-1)

int start(char* method)
{
    char param[PARAM_MAX_LEN];
    uint32_t paramLen = 0;

    paramLen = getParam(param, PARAM_MAX_LEN);

    if (0 == paramLen)
    {
        myprints("paramLen is 0, error");
        return ERROR_PARAM;
    }

    myprints("paramLen is:");
    printi(paramLen);
    myprints("param detail:");
    for(int i = 0;i<paramLen;i++)
    {
        printi(param[i]);
    }

    return 0;
}
```

We construct a transaction: The parameter is a value that is serialized by Messagepack, for example, we need to pass a serialized parameter to [97,98,99], convert its serialized value into a 16-binary string: "616263", and then fill in the Param:

```
{"version":1, "cursor_num":28,"cursor_label":3745260307,"lifetime":15270819998,"sender":"example", "contract":"example", "method":"test_method", "param":"616263", "sig_alg":1, "signature":""}
```

By calling this contract, there will be the following log output, that is, you can get the value after serialization to [97,98,99] in the contract:

```
2018-08-08 17:25:26 [INF] vm/wasm/exec/env_func.go:431 prints(): VM: func prints: paramLen is:

2018-08-08 17:25:26 [INF] vm/wasm/exec/env_func.go:405 printi(): VM: from contract: example, method:test_method, func printi: 3

2018-08-08 17:25:26 [INF] vm/wasm/exec/env_func.go:431 prints(): VM: func prints: param detail:

2018-08-08 17:25:26 [INF] vm/wasm/exec/env_func.go:405 printi(): VM: from contract: example, method:test_method, func printi: 97

2018-08-08 17:25:26 [INF] vm/wasm/exec/env_func.go:405 printi(): VM: from contract: example, method:test_method, func printi: 98

2018-08-08 17:25:26 [INF] vm/wasm/exec/env_func.go:405 printi(): VM: from contract: example, method:test_method, func printi: 99
```

2.4. Storage and reading of contract data

In order to facilitate unified access to data, we require that the data to be saved by the contract be serialized with Messagepack, and that the read data be deserialized accordingly to obtain the original data.

Here we define a structure for testing:

```
struct TestStruct {
    uint32_t valueA;
    uint32_t valueB;
};
```

We pass param to a parameter of type teststruct, where the values of the fields are filled in 1 and 2 respectively, first serialized, obtained [220,0,2,206,0,0,0,1,206,0,0,0,2], and converted into a 16-binary string: " Dc0002ce00000001ce00000002 ", we fill in the Param in the transaction with this value:

```
{"version":1, "cursor_num":28,"cursor_label":3745260307,"lifetime":15270819998,"sender":"example", "contract":"example", "method":"test_method", "param":"dc0002ce00000001ce00000002", "sig_alg":1, "signature":""}
```

We are in the contract to reverse the incoming parameters:

```
#include "contractcomm.hpp"
#define ERROR_PARAM (-1)
#define ERROR_UNPACK (-2)

struct TestStruct {
    uint32_t valueA;
    uint32_t valueB;
};


static bool unpack_struct(MsgPackCtx *ctx, TestStruct *info)
{
    uint32_t size = 0;

    if (!unpack_array(ctx, &size)) return false;
    if (2 != size) return false;

    if (!unpack_u32(ctx, &info->valueA)) return false;
    if (!unpack_u32(ctx, &info->valueB)) return false;

    return true;
}

static bool pack_struct(MsgPackCtx *ctx, TestStruct *info)
{
    if (!pack_array16(ctx, 2)) return false;

    if (!pack_u32(ctx, info->valueA)) return false;
    if (!pack_u32(ctx, info->valueB)) return false;

    return true;
}

int start(char* method)
{
    char param[PARAM_MAX_LEN];
    uint32_t paramLen = 0;

    paramLen = getParam(param, PARAM_MAX_LEN);

    if (0 == paramLen)
    {
        myprints("paramLen is 0, error");
        return ERROR_PARAM;
    }

    MsgPackCtx ctx;
    msgpack_init(&ctx, (char*)param, paramLen);
```

```
    TestStruct testStruct;
    bool suc = unpack_struct(&ctx, &testStruct);
    if (!suc)
    {
        myprints("unpack struct error");
        return ERROR_UNPACK;
    }

    myprints("data from input param:");
    printi(testStruct.valueA);
    printi(testStruct.valueB);

    return 0;
}
```

By calling this contract, we can see that the structural body inside correctly gets the parameters set when called, 1 and 2, respectively:

```
2018-08-08 18:20:00 [INF] vm/wasm/exec/env_func.go:431 prints(): VM: func prints: d
ata from input param:

2018-08-08 18:20:00 [INF] vm/wasm/exec/env_func.go:405 printi(): VM: from contract:
example, method:test_method, func printi: 1

2018-08-08 18:20:00 [INF] vm/wasm/exec/env_func.go:405 printi(): VM: from contract:
example, method:test_method, func printi: 2
```

We rewrite this data and save it through the Setbinvalue interface, then read it out through Getbinvalue, and then deserialize to get the real data:

```
#include "contractcomm.hpp"
#define ERROR_PARAM (-1)
#define ERROR_UNPACK (-2)
#define ERROR_SAVE_DB (-3)
#define ERROR_READ_DB (-4)
#define ERROR_CONTRACT_NAME (-5)

struct TestStruct {
    uint32_t valueA;
    uint32_t valueB;
};

static bool unpack_struct(MsgPackCtx *ctx, TestStruct *info)
{
    uint32_t size = 0;

    if (!unpack_array(ctx, &size)) return false;
    if (2 != size) return false;
```

```
    if (!unpack_u32(ctx, &info->valueA)) return false;
    if (!unpack_u32(ctx, &info->valueB)) return false;

    return true;
}

static bool pack_struct(MsgPackCtx *ctx, TestStruct *info)
{
    if (!pack_array16(ctx, 2)) return false;

    if (!pack_u32(ctx, info->valueA)) return false;
    if (!pack_u32(ctx, info->valueB)) return false;

    return true;
}

int start(char* method)
{
    char param[PARAM_MAX_LEN];
    uint32_t paramLen = 0;

    paramLen = getParam(param, PARAM_MAX_LEN);
    if (0 == paramLen)
    {
        myprints("paramLen is 0, error");
        return ERROR_PARAM;
    }

    /* Reverse the param of the order */
    MsgPackCtx ctx;
    msgpack_init(&ctx, (char*)param, paramLen);

    TestStruct testStruct;
    bool suc = unpack_struct(&ctx, &testStruct);
    if (!suc)
    {
        myprints("unpack struct error");

        return ERROR_UNPACK;
    }

    myprints("data from input param:");
    printi(testStruct.valueA);
    printi(testStruct.valueB);

    /* Rewrite fields to 3, 4 */
    testStruct.valueA = 3;
    testStruct.valueB = 4;

    /* Serialization of operations */
```

```
    msgpack_init(&ctx, (char*)param, PARAM_MAX_LEN);
    suc = pack_struct(&ctx, &testStruct);
    if (!suc)
    {
        myprints("pack struct error");
        return ERROR_UNPACK;
    }

    char tableName[] = "testTableName";
    char keyName[] = "testKeyName";

    /* Saves the calling interface */
    uint32_t handleResult = setBinValue(tableName, strlen(tableName), keyName, strl
en(keyName), ctx.buf, ctx.pos);
    if (0 == handleResult)
    {
        myprints("save to db error");
        return ERROR_SAVE_DB;
    }

    char contractname[STR_ARRAY_LEN(USER_NAME_MAX_LEN)];
    uint32_t contractNameLen = getCtxName(contractname, STR_ARRAY_LEN(USER_NAME_MAX
_LEN));
    if (0 == contractNameLen)
    {
        myprints("contract name error");
        return ERROR_CONTRACT_NAME;
    }

    /* The calling interface gets the value that was just saved */
    handleResult = getBinValue(contractname, contractNameLen, tableName, strlen(tab
leName), keyName, strlen(keyName), param, PARAM_MAX_LEN);
    if (0 == handleResult)
    {
        myprints("read from db error");
        return ERROR_READ_DB;
    }

    /* Reverse serialization of operations */
    msgpack_init(&ctx, (char*)param, handleResult);
    suc = unpack_struct(&ctx, &testStruct);
    if (!suc)
    {
        myprints("unpack struct error");
        return ERROR_UNPACK;
    }

    myprints("data from db:");
    printi(testStruct.valueA);
    printi(testStruct.valueB);
```

```
    return 0;
}
```

Call this contract again with the following trades (that is, input structures, fields 1, 2, respectively):

```
{"version":1, "cursor_num":28,"cursor_label":3745260307,"lifetime":15270819998,"sen
der":"example", "contract":"example", "method":"test_method", "param":"dc0002ce0000
0001ce00000002", "sig_alg":1, "signature":""}
```

As you can see from the log, the fields obtained to the input are 1, 2, which are overwritten to 3, 4, and then saved, and the expected values can still be obtained after the last read is deserialized, that is, the fields are 3 and 4, respectively:

```
2018-08-08 18:38:34 [INF] vm/wasm/exec/env_func.go:431 prints(): VM: func prints: d
ata from input param:

2018-08-08 18:38:34 [INF] vm/wasm/exec/env_func.go:405 printi(): VM: from contract:
example, method:test_method, func printi: 1

2018-08-08 18:38:34 [INF] vm/wasm/exec/env_func.go:405 printi(): VM: from contract:
example, method:test_method, func printi: 2

......

2018-08-08 18:38:34 [INF] vm/wasm/exec/env_func.go:431 prints(): VM: func prints: d
ata from db:

2018-08-08 18:38:34 [INF] vm/wasm/exec/env_func.go:405 printi(): VM: from contract:
example, method:test_method, func printi: 3

2018-08-08 18:38:34 [INF] vm/wasm/exec/env_func.go:405 printi(): VM: from contract:
example, method:test_method, func printi: 4
```

2.5. Invoking other contracts

The following is a compute contract that currently implements the Add method, which calculates the and of two parameters:

```
#include "contractcomm.hpp"
#include "string.hpp"

#define ERROR_PARAM (-1)
#define ERROR_UNPACK (-2)
#define ERROR_INVALID_METHOD (-3)

struct TestStruct {
    uint32_t valueA;
```

```
    uint32_t valueB;
};

static bool unpack_struct(MsgPackCtx *ctx, TestStruct *info)
{
    uint32_t size = 0;

    if (!unpack_array(ctx, &size)) return false;
    if (2 != size) return false;

    if (!unpack_u32(ctx, &info->valueA)) return false;
    if (!unpack_u32(ctx, &info->valueB)) return false;

    return true;
}

static bool pack_struct(MsgPackCtx *ctx, TestStruct *info)
{
    if (!pack_array16(ctx, 2)) return false;

    if (!pack_u32(ctx, info->valueA)) return false;
    if (!pack_u32(ctx, info->valueB)) return false;

    return true;
}

int start(char* method)
{
    if (0 == strcmp("add", method))
    {
        char param[PARAM_MAX_LEN];
        uint32_t paramLen = 0;

        /* Get the parameters from input */
        paramLen = getParam(param, PARAM_MAX_LEN);

        if (0 == paramLen)
        {
            myprints("paramLen is 0, error");
            return ERROR_PARAM;
        }

        /* Deserialize parameters to get raw data */
        MsgPackCtx ctx;
        msgpack_init(&ctx, (char*)param, paramLen);
        TestStruct testStruct;
        bool suc = unpack_struct(&ctx, &testStruct);
        if (!suc)
        {
            myprints("unpack struct error");
```

```
            return ERROR_UNPACK;
        }

        printi(testStruct.valueA);
        printi(testStruct.valueB);

        /* Print the and of two parameters */
        myprints("add result is:");
        printi(testStruct.valueA + testStruct.valueB);
    }
    else
    {
        myprints("invalid method");

        return ERROR_INVALID_METHOD;
    }

    return 0;
}
```

Then we write a contract in which the above calculation contract is called, which is passed to the above calculation contract two parameters, in the calculation contract to calculate the and of these two parameters: (assuming that the above calculation contract has been deployed on the Calccontract account)

```
#include "contractcomm.hpp"
#define ERROR_UNPACK  (-1)
#define ERROR_CALLTRX (-2)

struct TestStruct {
    uint32_t valueA;
    uint32_t valueB;
};

static bool unpack_struct(MsgPackCtx *ctx, TestStruct *info)
{
    uint32_t size = 0;

    if (!unpack_array(ctx, &size)) return false;
    if (2 != size) return false;

    if (!unpack_u32(ctx, &info->valueA)) return false;
    if (!unpack_u32(ctx, &info->valueB)) return false;

    return true;
}

static bool pack_struct(MsgPackCtx *ctx, TestStruct *info)
{
```

```
    if (!pack_array16(ctx, 2)) return false;

    if (!pack_u32(ctx, info->valueA)) return false;
    if (!pack_u32(ctx, info->valueB)) return false;

    return true;
}

int start(char* method)
{
    char param[PARAM_MAX_LEN];
    TestStruct testStruct;

    testStruct.valueA = 3;
    testStruct.valueB = 4;

    /* Serialization of operations */
    MsgPackCtx ctx;
    msgpack_init(&ctx, (char*)param, PARAM_MAX_LEN);
    bool suc = pack_struct(&ctx, &testStruct);
    if (!suc)
    {
        myprints("pack struct error");
        return ERROR_UNPACK;
    }

    /* Define the name of the contract to call, and the method name */
    char *callContractName = "calccontract";
    char *callMethod = "add";

    /* Invoke the contract */
    uint32_t callResult = callTrx(callContractName , strlen(callContractName), call
Method, strlen(callMethod), ctx.buf , ctx.pos);
    if (0 != callResult)
    {
        myprints("call trx error");
        return ERROR_CALLTRX;
    }

    myprints("call trx succed");

    return 0;
}
```

Let's call this contract above and there will be the following log:

```
2018-08-09 12:00:34 [INF] vm/wasm/exec/env_func.go:431 prints(): VM: func prints: c
all trx succed

2018-08-09 12:00:34 [INF] vm/wasm/exec/env_func.go:405 printi(): VM: from contract:
```

```
calccontract, method:add, func printi: 3

2018-08-09 12:00:34 [INF] vm/wasm/exec/env_func.go:405 printi(): VM: from contract:
calccontract, method:add, func printi: 4

2018-08-09 12:00:34 [INF] vm/wasm/exec/env_func.go:431 prints(): VM: func prints: a
dd result is:

2018-08-09 12:00:34 [INF] vm/wasm/exec/env_func.go:405 printi(): VM: from contract:
calccontract, method:add, func printi: 7
```

2.6. Attachment 1: interface function for shared invocation

- void prints(char *str, uint32_t len);

  Function: Print string

  Parameter description：

  | Parameter | Type | Description | | --------- | -------- | ---------- | | str | char* | Character Pointer to print | | len | uint32_t | The length of the string to print |

- void printi(uint64_t param);

  | Parameter | Type | Description | | --------- | -------- | ---------- | | param | uint64_t | The integer to print |

- uint32_t setBinValue(char *object, uint32_t objLen, char* key, uint32_t keyLen, char *value, uint32_t valLen);

  Function: Save data to the chain

  Parameter description：

  | Parameter | Type | Description | | --------- | -------- | ---------- | | object | char | *The table name corresponding to the saved dat* | | *objLen* | *uint32_t* | *The length of the table name corresponding to the saved data* | | *key* | *char* | Key value name of the saved data | | keyLen | uint32_t | The key value name length of the saved data | | value | char* | Data to be saved | | valLen | uint32_t | Length of data to be saved |

  Return value: Saved data length

- uint32_t getBinValue(char *contract, uint32_t contractLen, char* object, uint32_t objLen, char *key, uint32_t keyLen, char* valueBuf, uint32_t valueBufLen);

  Function: Get data from the chain

  Parameter description：

| Parameter | Type | Description | | ---------- | -------- | ---------- | | contract | char | *The contract name corresponding to the obtained data* | | *contractLen* | *uint32_t* | *The length of the contract name corresponding to the obtained data* | | *object* | *char* | Gets the table name corresponding to the data | | objLen | uint32_t | Gets the length of the table name corresponding to the data | | key | char | *Gets the key value name of the data* | | *keyLen* | *uint32_t* | *Gets the length of the key value name of the data* | | *valueBuf* | *char* | Buffers used to store data | | valueBufLen | uint32_t | The length of the buffer used to store the data |

Return value: The length of the obtained data

- uint32_t removeBinValue(char *object, uint32_t objLen, char* key, uint32_t keyLen);

  Function: Remove data from the chain

  Parameter description：

  | Parameter | Type | Description | | --------- | -------- | ---------- | | object | char | *The table name corresponding to the deleted data* | | *objLen* | *uint32_t* | *The length of the table name corresponding to the deleted dat* | | *key* | *char* | The key value name of the deleted data | | keyLen | uint32_t | The length of the key value name of the deleted data |

  Return value: The length of the deleted data

- uint32_t getParam(char *param, uint32_t bufLen);

  Function: Gets the parameter when the contract is invoked

  Parameter description：

  | Parameter | Type | Description | | --------- | -------- | ---------- | | param | char* | Buffer for storing parameter | | bufLen | uint32_t | Buffer for storing parameter |

  Return value: The length of the parameter obtained

- bool callTrx(char *contract , uint32_t contractLen, char* method , uint32_t methodLen, char *buf , uint32_t bufLen);

  Function: Invoke other contracts (asynchronous mode)

  Parameter description：

  | Parameter | Type | Description | | ---------- | -------- | ---------- | | contract | char | *The name of the contract to invoke* | | *contractLen* | | *contractLen* | *uint32_t* | *The length of the name of the contract to invoke* | | *method* | *char* | The method name to invoke the contract | | methodLen | uint32_t | The length of the method name to invoke the contract | | buf | char* | Call the contract incoming parameter | | bufLen | uint32_t | The length of the parameter passed by the calling contract |

  Return value：true: success，false: failure

- uint32_t getCtxName(char *str , uint32_t len);

  Function: Get contract name

Parameter description：

| Parameter | Type | Description | | --------- | -------- | ---------- | | str | char* | Buffer for storing contract name | | len | uint32_t | The length of the buffer that stores the contract name |

Return value: The length of the account name obtained

- uint32_t getSender(char *str , uint32_t len);

  Function: Gets the name of the account that invokes the current contract

  Parameter description：

  | Parameter | Type | Description | | --------- | -------- | ---------- | | str | char* | Buffer for storing account names | | len | uint32_t | The length of the buffer that stores the account name |

  Return value: The length of the account name obtained

- bool isAccountExist(char *name， uint32_t nameLen);

  function: Check if the account exists

  Parameter description：

  | Parameter | Type | Description | | --------- | -------- | ---------- | | name | char* | The account you want to check | | nameLen | uint32_t | The length of the account name to check |

  Return value：true: exists，false: not exists

2.7. Serialization of the attached 2:messagepack

2.7.1 Overview

In order to facilitate the parameter transmission when the contract is invoked, and to read the persistent data of the contract, we have selected Messagepack this lightweight codec method, detailed specification reference:

https://msgpack.org/

https://github.com/msgpack/msgpack/blob/master/spec.md

In addition, we have made some tailoring to the characteristics of the contract data.

1. Basic Type：Supports uint8、uint16、uint32、uint64、array16、bin16、str16Type；
2. Variable length data: For example strtype,messagepack the original specification according to the length of the string fill in the length of bytes, there are 1, 2, 4 byte length of the difference, after the transformation of the default use of 2 bytes (str16), Bintype and ArrayType is also, that is, only support Bin16, ARRAY16 this type;

3. Structure: The structure body is encapsulated in the form of array, the ARRAY16 head is written first, and then the fields are encoded in turn.

Sample(C)：

```
struct user_login {
    char user_name[USER_NAME_MAX_LEN];
    uint32_t random_num;
};

user_login login;
strcpy(login.user_name, "testuser");
login.random_num = 99;

pack_array16(&ctx, 2);
pack_str16(&ctx, login->user_name, strlen(login->user_name));
pack_u32(&ctx, login->random_num);
```

Encoding result：

```
0xdc, 0x00, 0x02, 0xda, 0x00, 0x08, 0x74, 0x65, 0x73, 0x74, 0x75, 0x73, 0x65, 0x72,
 0xce, 0x00, 0x00, 0x00, 0x63
```

2.7.2 Coding specification

Types

- Integer represents an integer
- Raw
  - String extending Raw type represents a UTF-8 string
  - Binary extending Raw type represents a byte array
- Array represents a sequence of objects

Format

format name	first byte (in binary)	first byte (in hex)
bin16	11000101	0xc5
uint8	11001100	0xcc
uint16	11001101	0xcd
uint32	11001110	0xce
uint64	11001111	0xcf
str16	11011010	0xda
array16	11011100	0xdc

Notation in diagrams

```
one byte:
+--------+
|        |
+--------+

a variable number of bytes:
+========+
|        |
+========+

variable number of objects stored in MessagePack format:
+~~~~~~~~~~~~~~~~~+
|                 |
+~~~~~~~~~~~~~~~~~+
```

int format family

Int format family stores an integer in 2, 3, 5, or 9 bytes.

```
uint 32 stores a 32-bit big-endian unsigned integer
+--------+--------+--------+--------+--------+
|  0xce  |ZZZZZZZZ|ZZZZZZZZ|ZZZZZZZZ|ZZZZZZZZ|
+--------+--------+--------+--------+--------+

uint 64 stores a 64-bit big-endian unsigned integer
+--------+--------+--------+--------+--------+--------+--------+--------+--------+
|  0xcf  |ZZZZZZZZ|ZZZZZZZZ|ZZZZZZZZ|ZZZZZZZZ|ZZZZZZZZ|ZZZZZZZZ|ZZZZZZZZ|ZZZZZZZZ|
+--------+--------+--------+--------+--------+--------+--------+--------+--------+
```

str format family

Str format family stores a byte array in 3 bytes of extra bytes in addition to the size of the byte array.

```
str 16 stores a byte array whose length is upto (2^16)-1 bytes:
+--------+--------+--------+========+
|  0xda  |ZZZZZZZZ|ZZZZZZZZ|  data  |
+--------+--------+--------+========+

where
* ZZZZZZZZ_ZZZZZZZZ is a 16-bit big-endian unsigned integer which represents N
* N is the length of data
```

bin format family

Bin format family stores a byte array in 3 bytes of extra bytes in addition to the size of the byte array.

```
bin 16 stores a byte array whose length is upto (2^16)-1 bytes:
+--------+--------+--------+========+
```

```
|  0xc5  |YYYYYYYY|YYYYYYYY|  data  |
+--------+--------+--------+========+

where
* YYYYYYYY_YYYYYYYY is a 16-bit big-endian unsigned integer which represents N
* N is the length of data
```

array format family

Array format family stores a sequence of elements in 3 bytes of extra bytes in addition to the elements.

```
array 16 stores an array whose length is upto (2^16)-1 elements:
+--------+--------+--------+~~~~~~~~~~~~~~~~~+
|  0xdc  |YYYYYYYY|YYYYYYYY|    N objects    |
+--------+--------+--------+~~~~~~~~~~~~~~~~~+

where
* YYYYYYYY_YYYYYYYY is a 16-bit big-endian unsigned integer which represents N
    N is the size of a array
```

Serialization: type to format conversion

MessagePack serializers convert MessagePack types into formats as following:

source types	output format
Integer	int format family (positive fixint uint 8/16/32/64)
String	str format family (str16)
Binary	bin format family (bin16)
Array	array format family (array16)

Deserialization: format to type conversion

MessagePack deserializers convert MessagePack formats into types as following:

source formats	output type
positive fixint,uint 8/16/32/64	Integer
str16	String
bin16	Binary
array16	Array

Deploying and Testing

1. Deploy a contract

Prerequisites for Deploying a contract: Compile the Wasm file generated by the contract, scan the generated ABI file and an account number, the command format is：

```
./bcli contract deploycode --name testhelloworld --code testHelloWorld.wasm  --abi
testHelloWorld.abi
```

For more details please refer to BCLI command description

2. Invoke contract

The invoking contract function is essentially sending a transaction to the chain, as detailed in the chain interface's description of sending transaction information.

Common REST Interface

Get the block information

API Functions

> API： Get block information
>
> API Address URL: /v1/block/detail
>
> Response Format
>
> JSON
>
> Request Method
>
> POST
>
> Request Parameters：

Parameter	Mandatory	Type	Default Value	Description
block_num	FALSE	uint64	Null	Block number
block_hash	FALSE	string	Null	Block hash value

Note：block_num、block_hash can only choose one of them; If not given in either，it will returns block information of "block 0" in default。

Response Fields：

Parameter	Type	Description
errcode	uint32	Error code，0-succeed，others please refer to the error code chapter.
msg	string	response description
result	jsonObject	response result
block_num	uint64	Block number
prev_block_hash	string	Previous Block hash value
block_hash	string	Current Block hash value
cursor_block_label	uint32	Current block identity
block_time	uint64	Generate time of the block
trx_merkle_root	string	Merkle root value，The hash value of all transactions packaged by the block
delegate	string	Producer name
delegate_sign	string	Producer signature

trxs	jsonArray	The list of transactions packaged in the current block, as described in the response field in the "Query Transaction information" section

Field changes

- Null

  API Smple

> Address：http://127.0.0.1:8689/v1/block/detail

- Request:

```
  {
      "block_num": 32,
      "block_hash": "405a6fb8b91a055a7a4cf007451ce0b31ea6626cb2d56ec050b126701fbf09
3d"
  }
```

- Response:

```
  HTTP/1.1 200 OK

  {
      "errcode": 0,
      "msg": "success",
      "result": {
          "prev_block_hash": "1b345401d8f859c05e37b5ccacd39ff5b4a21615b63f9b96edb59
352a3e54e82",
          "block_num": 32,
          "block_hash": "405a6fb8b91a055a7a4cf007451ce0b31ea6626cb2d56ec050b126701f
bf093d",
          "cursor_block_label": 532613437,
          "block_time": 1537259127,
          "trx_merkle_root": "85f9d2fbe1d3c0a217e10932899b6f73b24fafe59a006406ed65d
7e4a39a7416",
          "delegate": "bottos",
          "delegate_sign": "41c25998e7d432527f07ae5c10c5c30a4873e9519c29cb02e7f46e3
a8e238baf47748efe3f24db3ff3630305441b1d24c5d0ba796b3332a8b8fd3fca4efc4cfb",
          "trxs": [
              {
                  "version": 1,
                  "cursor_num": 31,
                  "cursor_label": 2749714050,
                  "lifetime": 1537259224,
                  "sender": "bottos",
                  "contract": "bottos",
                  "method": "newaccount",
                  "param": {
                      "name": "testtest",
```

```
                    "pubkey": "0454f1c2223d553aa6ee53ea1ccea8b7bf78b8ca99f3ff622a
3bb3e62dedc712089033d6091d77296547bc071022ca2838c9e86dec29667cf740e5c9e654b6127f"
                },
                "sig_alg": 1,
                "signature": "c85fd25af493cbb6a79870ce0fc602acc892664ca17e2c646af
f0332ca6db7787beeb7e5d8553de8e4b83bdf7b227762fedf9e3674888893f18bf31f0b05d622"
            }
        ]
    }
}
```


Get Block Header Information

API Function

> API Description： Get block header information
>
> APIAddress
>
> URL: /v1/block/height
>
> Response Format
>
> JSON
>
> Request Format
>
> GET
>
> Request Parameter：

Parameter	Mandatory	Type	Default Value	Description
Null				

Response Fields：

Parameter	Type	Description
errcode	uint32	Error code，0-Succeed，others refer to error code chapter
msg	string	response description
result	jsonObject	response result
head_block_num	uint64	Block number
head_block_hash	string	Previous Block hash value
head_block_time	uint64	Generating time of block
head_block_delegate	string	Producer of block
cursor_label	uint32	Block identity
last_consensus_block_num	uint32	Irreversible Block's number

chain_id	string	Chain ID，The chain_id must be the same for all nodes in the same chain

Fields Changes

- Null

  API Sample

> Address：http://127.0.0.1:8689/v1/block/height

- Request:

```
Null
```

- Response:

```
HTTP/1.1 200 OK
{
    "errcode": 0,
    "msg": "",
    "result": {
        "head_block_num": 87,
        "head_block_hash": "b34806eefc77b88743ab447f43658bf229fd4e5cd9452340e21f3
995a5d2054b",
        "head_block_time": 1534213225,
        "head_block_delegate": "alsephina",
        "cursor_label": 2782004555,
        "last_consensus_block_num": 64,
        "chain_id": "4b97b92d2c78bcffe95ebd3067565c73a2931b39d5eb7234b11816dcec54
761a"
    }
}
```

Send the transaction information

API Function

> API Description： Send the transaction information
>
> APIAddress
>
> URL: /v1/transaction/send
>
> Response Format
>
> JSON
>
> Request Format
>
> POST

Request Parameter：

Parameter	Mandatory	Type	Default Value	Description
version	TRUE	uint32	1	Version number of block chain
cursor_num	TRUE	uint64	Null	Latest block number，getting by querying header block
cursor_label	TRUE	uint32	Null	Latest block identity，get by querying the block header
lifetime	TRUE	uint64	Null	Transaction exipired time，by ways of getting block header, plus a certain delay，it is recommanded to delay by 100 seconds
sender	TRUE	string	Null	Sender
contract	TRUE	string	Null	Contract name
method	TRUE	string	Null	Contract method
param	TRUE	string	Null	Business parameter, hexadecimal string
sig_alg	TRUE	uint32	1	Sign algorism
signature	TRUE	string	Null	Signature value

Response Fields：

Parameter	Type	Description
errcode	uint32	Error code，0-Succeed，others refer to error code chapter
msg	string	response description
result	jsonObject	response result
trx_hash	string	Hash value of each transaction
trx	jsonObject	Transaction details
version	uint32	Version number of block chain
cursor_num	uint64	Latest block number，get by querying the block header
cursor_label	uint32	Latest block label，get by querying the block header
lifetime	uint64	Transaction exipired time，by ways of getting block header, plus a certain delay
sender	string	Sender
contract	string	Contract name
method	string	Contract method
param	string	Business parameter，hexadecimal string
sig_alg	uint32	Sign algorism
signature	string	Signature value

Fields Changes

- Null

  API Sample

> Address：http://127.0.0.1:8689/v1/transaction/send

- Request:


```
  {
      "version": 1,
      "cursor_num": 719,
      "cursor_label": 2997806499,
      "lifetime": 1534143531,
      "sender": "bottos",
      "contract": "bottos",
      "method": "newaccount",
      "param": "dc0002da0009757365727465737431da0082303464303735383830303536343838613938646133656432346237666132656330616238643834643437646235343663336631383161373634626133666131653832373966373634343039633431646530316230303830656231616165656239353039663739323335353230613735653334323434323931343462343363313034623",
      "sig_alg": 1,
      "signature": "f0069bc363a55dc22207c75d15cc75524bf4950159130c6bf385f6f1ca877177362ad5ab51108e7f396043e3aee7058f1ca6a40fd6c79a8483e439d2e2bccf2c"
  }
```


- Response:


```
  HTTP/1.1 200 OK
  {
      "errcode": 0,
      "msg": "trx receive succ",
      "result": {
          "trx": {
              "version": 1,
              "cursor_num": 719,
              "cursor_label": 2997806499,
              "lifetime": 1534143531,
              "sender": "delta",
              "contract": "bottos",
              "method": "newaccount",
              "param": "dc0002da0009757365727465737431da0082303464303735383830303536343838613938646133656432346237666132656330616238643834643437646235343663336631383161373634626133666131653832373966373634343039633431646530316230303830656231616165656239353039663739323335353230613735653334323434323931343462343363313034623",
              "sig_alg": 1,
              "signature": "f0069bc363a55dc22207c75d15cc75524bf4950159130c6bf385f6f1ca877177362ad5ab51108e7f396043e3aee7058f1ca6a40fd6c79a8483e439d2e2bccf2c"
          },
```

```
          "trx_hash": "1815f4d4dfb52b88fb445efc255a5be6275fc3ad694f802c01c40644f09b
651f"
      }
  }
```

Query Transaction Information

API Function

> API Description： Query transaction information
>
> APIAddress
>
> URL: /v1/transaction/get
>
> Response Format
>
> JSON
>
> Request Format
>
> POST

Request Parameter：

Parameter	Mandatory	Type	Default Value	Description
trx_hash	TRUE	string	Null	Hash value of each transaction

Response Fields：

Parameter	Type	Description
errcode	uint32	Error code，0-Succeed，others refer to error code chapter
msg	string	response description
result	jsonObject	response result
version	uint32	Version number of block chain
cursor_num	uint64	Latest block number，get by querying the block header
cursor_lab	uint32	Latest label of block，get by querying the block header
lifetime	uint64	Transaction exipired time，by ways of getting block header, plus a certain delay
sender	string	Sender
contract	string	Contract name
method	string	Contract method
param	jsonObject	Business Parameter
sig_alg	uint32	Sign algorism

signature	string	Signature value

Fields Changes

- Null

  API Sample

> Address：http://127.0.0.1:8689/v1/transaction/get

- Request:

```
  {
      "trx_hash": "85f9d2fbe1d3c0a217e10932899b6f73b24fafe59a006406ed65d7e4a39a7416
"
  }
```

- Response:

```
   HTTP/1.1 200 OK
  {
      "errcode": 0,
      "msg": "success",
      "result": {
          "version": 1,
          "cursor_num": 31,
          "cursor_label": 2749714050,
          "lifetime": 1537259224,
          "sender": "bottos",
          "contract": "bottos",
          "method": "newaccount",
          "param": {
              "name": "testtest",
              "pubkey": "0454f1c2223d553aa6ee53ea1ccea8b7bf78b8ca99f3ff622a3bb3e62d
edc712089033d6091d77296547bc071022ca2838c9e86dec29667cf740e5c9e654b6127f"
          },
          "sig_alg": 1,
          "signature": "c85fd25af493cbb6a79870ce0fc602acc892664ca17e2c646aff0332ca6
db7787beeb7e5d8553de8e4b83bdf7b227762fedf9e3674888893f18bf31f0b05d622"
      }
  }
```

Query Data Based on Keywords

API Function

> API Description： Query data based on keywords
>
> APIAddress

> URL: /v1/common/query
>
> Response Format
>
> JSON
>
> Request Format
>
> POST

Request Parameter：

Parameter	Mandatory	Type	Default Value	Description
contract	TRUE	string	Null	Contract name
object	TRUE	string	Null	Contract table name
key	TRUE	string	Null	The keyword for querying

Response Fields：

Parameter	Type	Description
errcode	uint32	Error code，0-Succeed，others refer to error code chapter
msg	string	response description
result	jsonObject	response result
contract	string	Contract name
object	string	Contract table name
key	string	The keyword for querying
value	uint32	Queried value

Fields Changes

- Null

  API Sample

> Address：http://127.0.0.1:8689/v1/common/query

- Request:

```
{
    "contract":"bottostoken",
    "object":"DTO",
    "key":"aaa"
}
```

- Response:

```
 HTTP/1.1 200 OK
```

```
{
  "errcode": 0,
  "msg": "",
  "result": {
        "contract": "bottostoken",
        "object": "DTO",
        "key": "aaa",
        "value": "dc0001cf000000ba43b74000"
  }
}
```

Serialization Of Business Data

API Function

> API Description: Serialize the business data, and the JSON converts to a hexadecimal string as the Parm field value in the "Send the transaction information request".
>
> APIAddress
>
> URL: /v1/common/jsontobin
>
> Response Format
>
> JSON
>
> Request Format
>
> POST

Request Parameter：

Parameter	Mandatory	Type	Default Value	Description
contract	TRUE	string	Null	Contract name
object	TRUE	string	Null	Contract table name
key	TRUE	string	Null	The keyword for querying

Response Fields：

Parameter	Type	Description
errcode	uint32	Error code，0-Succeed，others refer to error code chapter
msg	string	response description
result	jsonObject	response result
contract	string	Contract name
object	string	Contract table name
key	string	The keyword for querying

value	uint32	Queried value

Fields Changes

- Null

  API Sample

> Address：<http://127.0.0.1:8689/v1/common/jsontobin

- Request:

```
  {
      "account_name": "testtest",
      "public_key": "0454f1c2223d553aa6ee53ea1ccea8b7bf78b8ca99f3ff622a3bb3e62dedc7
12089033d6091d77296547bc071022ca2838c9e86dec29667cf740e5c9e654b6127f"
  }
```

- Response:

```
   HTTP/1.1 200 OK
  {
    "errcode": 0,
    "msg": "",
    "result": {
          "contract": "bottostoken",
          "object": "DTO",
          "key": "aaa",
          "value": "dc0001cf000000ba43b74000"
    }
  }
```

Query Accounts Information**

API Function

> API Description： Query accounts information
>
> APIAddress
>
> URL: /v1/account/info
>
> Response Format
>
> JSON
>
> Request Format
>
> POST

Request Parameter：

Parameter	Mandatory	Type	Default Value	Description
account_name	TRUE	string	Null	Account name

Response Fields：

Parameter	Type	Description
errcode	uint32	Error code，0-Succeed，others refer to error code chapter
msg	string	response description
result	jsonObject	response result
account_name	string	Account name
pubkey	string	Public key
balance	string	Token value，
staked_balance	string	Staked token value
unStaking_balance	string	Under unstakeing token value
unStaking_time	uint64	Unstake token timestamp（Unix timestamp）

Note：balance、staked_balance、unStaking_balance, the summary value of that three is the total token value of the change account.

Fields Changes

- Null

  API Sample

> Address：http://127.0.0.1:8689/v1/account/info

- Request:

```
{
    "account_name":"delta"
}
```

- Response:

```
 HTTP/1.1 200 OK
{
    "errcode": 0,
    "msg": "success",
    "result": {
        "account_name": "testtest",
        "pubkey": "0454f1c2223d553aa6ee53ea1ccea8b7bf78b8ca99f3ff622a3bb3e62dedc7
12089033d6091d77296547bc071022ca2838c9e86dec29667cf740e5c9e654b6127f",
        "balance": "0",
        "staked_balance": "0",
```

```
            "unStaking_balance": "0",
            "unStaking_time": 0
        }
    }
```

Query Contract ABI

API Function

> API Description： Query contract ABI
>
> APIAddress
>
> URL: /v1/contract/abi
>
> Response Format
>
> JSON
>
> Request Format
>
> POST

Request Parameter：

Parameter	Mandatory	Type	Default Value	Description
contract	TRUE	string	Null	Contract name

Response Fields：

Parameter	Type	Description
errcode	uint32	Error code，0-Succeed，others refer to error code chapter
msg	string	response description
result	jsonObject	response result
types	jsonObject	reserved field
structs	jsonArray	The struct list of method which included in the contract
name	string	struct name，typically the custom contract method name aliases
base	string	Father method，usually be Null
fields	jsonObject	method's inputting parameters' structs
name	string	Parameter-1，take a registered account as an example：Account name
pubkey	string	Parameter-2，take a registered account as an example：Public key
actions	jsonArray	The method included in the contract

action_name	string	Contract method name，relate to “Send the transaction information”'s request parameter's method value
type	string	Same as name, the struct name, typically the custom contract method name aliases
tables	jsonArray	Describe how the contract data is stored, defining the table name, Type,key value, Type,value Type. Built-in contract the field is null
table_name	string	Table name for contract storage
index_type	string	Index Type，default is string
key_names	jsonArray	Key value for contract data storage
key_types	jsonArray	key value's Type，default is string
type	string	Value values corresponding to contract data key, usually be structs

Fields Changes

- Null

  API Sample

> Address：<http://127.0.0.1:8689/v1/contract/abi >

- Request:

```
{
    "contract":"bottos"
}
```

- Response:

```
 HTTP/1.1 200 OK
{
    "errcode": 0,
    "msg": "success",
    "result": {
        "actions": [
            {
                "action_name": "newaccount",
                "type": "NewAccount"
            },
            {
                "action_name": "transfer",
                "type": "Transfer"
            },
            {
                "action_name": "grantcredit",
                "type": "GrantCredit"
            },
```

```
            {
                "action_name": "cancelcredit",
                "type": "CancelCredit"
            },
            {
                "action_name": "transferfrom",
                "type": "TransferFrom"
            },
            {
                "action_name": "deploycode",
                "type": "DeployCode"
            },
            {
                "action_name": "deployabi",
                "type": "DeployABI"
            },
            {
                "action_name": "regdelegate",
                "type": "RegDelegate"
            },
            {
                "action_name": "unregdelegate",
                "type": "UnregDelegate"
            },
            {
                "action_name": "votedelegate",
                "type": "VoteDelegate"
            },
            {
                "action_name": "stake",
                "type": "Stake"
            },
            {
                "action_name": "unstake",
                "type": "Unstake"
            },
            {
                "action_name": "claim",
                "type": "Claim"
            },
            {
                "action_name": "setdelegate",
                "type": "SetDelegate"
            },
            {
                "action_name": "blkprodtrans",
                "type": "BlkProdTrans"
            }
        ],
        "structs": [
            {
```

```
                "base": "",
                "fields": {
                    "name": "string",
                    "pubkey": "string"
                },
                "name": "NewAccount"
            },
            {
                "base": "",
                "fields": {
                    "from": "string",
                    "to": "string",
                    "value": "uint256"
                },
                "name": "Transfer"
            },
            {
                "base": "",
                "fields": {
                    "description": "string",
                    "location": "string",
                    "name": "string",
                    "pubkey": "string"
                },
                "name": "SetDelegate"
            },
            {
                "base": "",
                "fields": {
                    "limit": "uint256",
                    "name": "string",
                    "spender": "string"
                },
                "name": "GrantCredit"
            },
            {
                "base": "",
                "fields": {
                    "name": "string",
                    "spender": "string"
                },
                "name": "CancelCredit"
            },
            {
                "base": "",
                "fields": {
                    "from": "string",
                    "to": "string",
                    "value": "uint256"
                },
                "name": "TransferFrom"
```

```
            },
            {
                "base": "",
                "fields": {
                    "contract": "string",
                    "contract_code": "bytes",
                    "vm_type": "uint8",
                    "vm_version": "uint8"
                },
                "name": "DeployCode"
            },
            {
                "base": "",
                "fields": {
                    "contract": "string",
                    "contract_abi": "bytes"
                },
                "name": "DeployABI"
            },
            {
                "base": "",
                "fields": {
                    "description": "string",
                    "location": "string",
                    "name": "string",
                    "pubkey": "string"
                },
                "name": "RegDelegate"
            },
            {
                "base": "",
                "fields": {
                    "name": "string"
                },
                "name": "UnregDelegate"
            },
            {
                "base": "",
                "fields": {
                    "delegate": "string",
                    "voteop": "uint8",
                    "voter": "string"
                },
                "name": "VoteDelegate"
            },
            {
                "base": "",
                "fields": {
                    "amount": "uint256"
                },
                "name": "Stake"
```

```
            },
            {
                "base": "",
                "fields": {
                    "amount": "uint256"
                },
                "name": "Unstake"
            },
            {
                "base": "",
                "fields": {
                    "amount": "uint256"
                },
                "name": "Claim"
            },
            {
                "base": "",
                "fields": {
                    "actblknum": "uint64"
                },
                "name": "BlkProdTrans"
            }
        ],
        "tables": null,
        "types": null
    }
}
```


Query contract code

API Function

> API Description： Query contract code
>
> APIAddress
>
> URL: /v1/contract/code
>
> Response Format
>
> JSON
>
> Request Format
>
> POST

Request Parameter：

Parameter	Mandatory	Type	Default Value	Description
contract	TRUE	string	Null	Contract name

Response Fields：

Parameter	Type	Description
errcode	uint32	Error code，0-succeed，others refer to error code chapters
msg	string	response description
result	jsonObject	response result

Fields Changes

- Null

  API sample

> Address：<http://127.0.0.1:8689/v1/contract/code >

- Request:

```
{
    "contract":"usermng"
}
```

- Response:

```
 HTTP/1.1 200 OK
{
    "errcode": 0,
    "msg": "",
    "result": "7b0a20202274797065732223a205b5d2c0a20202273747275637473223a205b7b0a
202020202020226e616d65223a202255736572496e666f222c0a2020202020202262617365223a20222
22c0a2020202020202266696656c6473223a207b0a2020202020202020202022646964964223a202273
7472696e67222c0a2020202020202020202022696e666f223a2022737472696e67220a2020202020202
0207d0a202020207d2c7b0a202020202020226e616d65223a2022557365724c6f67696e222c0a202020
2020202022626173652223a2022222c0a2020202020202266696656c6473223a207b0a2020202020202
02022757365724e616d65223a2022737472696e67222c0a2020202020202020202022272616e646f6d4e
756d223a202275696e743332220a2020202020202020207d0a202020207d2c7b0a202020202020226e616
d65223a202255736572426173652496e666f222c0a2020202020202262617365223a2022222c0a202020
202020226669656c6473223a207b0a2020202020202020202022696e666f223a2022737472696e67220
a2020202020202020207d0a202020207d0a20205d2c0a2020226163746963f6e73223a205b7b0a20202020
202022616374696f6e5f6e616d65223a2022726567757365722222c0a2020202020202274797065223a2
02255736572496e666f220a202020207d2c7b0a2020202020202262163746963f6e5f6e616d65223a2022
757365726c6f67696e222c0a2020202020202274797065223a2022557365724c6f67696e220a2020202
07d0a20205d2c0a2020227461626c6573223a205b7b0a2020202020202274616263655f6e616d65223a
202275736572726567696e666f222c0a2020202020202022696e6465785f74797065223a2022737472696
e67222c0a202020202020226b65795f6e616d6573223a205b0a2020202020202020202264696469642220a
2020202020205d2c0a202020202020226b65795f7479706573223a205b0a2020202020202020202737473
2696e67220a2020202020205d2c0a2020202020202274797065223a2022557365724261736496e666f
220a202020207d0a20205d0a7d0a"
}
```

Query All Producers

API Functions

> API Description： Query all producers
>
> APIAddress
>
> URL: /v1/delegate/getall
>
> Response Format
>
> JSON
>
> Request Format
>
> POST

Request Parameter：

Parameter	Mandatory	Type	Default Value	Description
limit	TRUE	uint32	Null	Query numbers
start	TRUE	uint32	0	Query start number

Response Fields：

Parameter	Type	Description
errcode	uint32	Error code，0-Succeed，others refer to error code chapter
msg	string	response description
result	jsonArray	response result
account_name	string	Account name
public_key	string	Public key
location	string	Node geolocation information, such as "shanghai,china"
desc	string	Node descriptions

Fields Changes

- Null

  API Sample

> Address：<http://127.0.0.1:8689/v1/delegate/getall >

- Request:

```
{
    "limit": 10,
    "start": 0
}
```

- Response:

```
   HTTP/1.1 200 OK
  {
      "errcode": 0,
      "msg": "success",
      "result":  [
          {
              "account_name": "adhil",
              "report_key": "0454f1c2223d553aa6ee53ea1ccea8b7bf78b8ca99f3ff622a3bb3
e62dedc712089033d6091d77296547bc071022ca2838c9e86dec29667cf740e5c9e654b6127f",
              "location": "shanghai,china",
              "desc": "",
              "last_slot": 1029945,
              "total_missed": 0,
              "last_confirmed_block_num": 32,
              "active": true
          },
          {
              "account_name": "albireo",
              "report_key": "0454f1c2223d553aa6ee53ea1ccea8b7bf78b8ca99f3ff622a3bb3
e62dedc712089033d6091d77296547bc071022ca2838c9e86dec29667cf740e5c9e654b6127f",
              "location": "",
              "desc": "",
              "last_slot": 1026356,
              "total_missed": 123,
              "last_confirmed_block_num": 10,
              "active": false
          }
      ]
  }
```

Wallet REST Interface

One-Step to create wallet

API FUNCTION

> API Description: One step to create the wallet that is stored by default under directory of Linux： /home/bottos/bot/
>
> API Address
>
> URL: /v1/wallet/createwallet
>
> Result Format
>
> JSON
>
> Request Format
>
> POST

Request Parameter：

Parameter	Mandatory	Type	Default Value	Description
account_name	TRUE	string	Null	Account name
passwd	TRUE	string	Null	Password

Response Field:

Parameter	Type	Description
errcode	uint32	Error code, 0-succeed, others refer to error code chapter
msg	string	Response description
result	jsonObject	Response result
wallet_name	string	Generated wallet file name, default is account name+".keystore" in tail

Field change

- Null

  API Sample

> Address : <http://127.0.0.1:6869/v1/wallet/createwallet >

- Request :

  ```
  ```

```
  "account_name": "testtest1",
  "passwd": "123456"
}
```

- Response :

HTTP/1.1 200 OK { "errcode": 0, "msg": "success", "result": { "wallet_name": "testtest1.keystore" } }

```
## Generate Public Private Key Pairs

**API Function**

> API Description: Generate public private key pairs
>
> **API Address**
>
> URL:  /v1/wallet/generatekeypair
>
> **Result Format**
>
> JSON
>
> **Request Format**
>
> GET

**Request Parameter: **

| Parameter | Mandatory | Type | Default Value | Description |
| ---- | ---- | ---- | ------ | ---- |
| Null   |      |      |        |      |

**Response Field:**

| Parameter        | Type       | Description                                |
| ---------- | ---------- | ---------------------------------- |
| errcode     | uint32     | Error code, 0-succeed, others refer to error code chap
ter |
| msg         | string     | Response description                           |
| result      | jsonObject | Response result                              |
| public_key  | string     | Public key                                  |
```

```
| private_key | string     | Private Key                                  |

**Field change**

- Null

  **API Sample**

> Address : <http://127.0.0.1:6869/v1/wallet/generatekeypair  >

- Request :
```

Null

```
- Response :
```

HTTP/1.1 200 OK { "errcode": 0, "msg": "success", "result": { "public_key": "043680ae1b81f87274d7051e6101afc8f8da3cd978bb75b22f24becb98afb456f110151644330ff6c742e44f8e9652f1ab5ea1cd3997bebe5a23a4389ff0cb6493", "private_key": "8a068d821ad8d58fe8caaa87f32ea69f889f6b8ce8c6bc3bc67ce2c68c43da3b" } }

```
## Create Account

**API Function**

> API Description: Create account
>
> **API Address**
>
> URL:  /v1/wallet/createaccount
>
> **Result Format**
>
> JSON
>
> **Request Format**
>
> POST

**Request Parameter: **

| Parameter         | Mandatory  | Type   | Default Value | Description
                        |
| ------------ | ----- | ------ | ------ | ------------------------------------ |
| account_name | TRUE  | string | Null     | Account name
     |
| public_key   | TRUE  | string | Null     | Public key
```

```
       |
| referrer     | FALSE | string | bottos | Referrer account, if is set to null, it
is bottos account by default |

**Response Field:**

| Parameter         | Type       | Description                                |
| ------------ | ---------- | ---------------------------------- |
| errcode      | uint32     | Error code, 0-succeed, others refer to error code cha
pter |
| msg          | string     | Response description                            |
| result       | jsonObject | Response result and data details               |
| account_name | string     | New registered account name                     |

**Field change**

- Null

  **API Sample**

> Address : <http://127.0.0.1:6869/v1/wallet/createaccount >

- Request :
```

{ "account_name": "testtest", "public_key": "0454f1c2223d553aa6ee53ea1ccea8b7bf78b8ca99f3ff622a3bb3e62dedc712089033d6091d77296547bc071022ca2838c9e86dec29667cf740e5c9e654b6127f", "referrer": "bottos" }

```
- Response :
```

HTTP/1.1 200 OK { "errcode": 0, "msg": "success", "result": { "account_name":"testtest" } }

```
## Create Wallet by Manual

**API Function**

> API Description: Create wallet manually, wallet file is stored by default in Linu
x directory:  /home/bottos/bot/, and the generated wallet file name is account name
 +".keystore" in tail by default.
>
> **API Address**
>
> URL:  /v1/wallet/createwalletmanual
>
```

```
> **Result Format**
>
> JSON
>
> **Request Format**
>
> POST

**Request Parameter: **

| Parameter         | Mandatory | Type   | Default Value | Description     |
| ------------ | ---- | ------ | ------ | -------- |
| account_name | TRUE | string | Null     | Account name |
| private_key  | TRUE | string | Null     | Private Key     |
| passwd       | TRUE | string | Null     | Password     |

**Response Field:**

| Parameter        | Type       | Description
         |
| ----------- | ---------- | ------------------------------------------------- |
| errcode     | uint32     | Error code, 0-succeed, others refer to error code chap
ter               |
| msg         | string     | Response description
         |
| result      | jsonObject | Response result
    |
| wallet_name | string     | the generated wallet file name, it is account name +".
keystore" in tail by default |

**Field change**

- Null

  **API Sample**

> Address : <http://127.0.0.1:6869/v1/wallet/createwalletmanual  >

- Request :
```

{ "account_name": "testtest1", "private_key": "b799ef616830cd7b8599ae7958fbee56d4c8168ffd5421a16025a398b8a4be45", "passwd": "123456" }

```
- Response :
```

HTTP/1.1 200 OK { "errcode": 0, "msg": "success", "result": { "wallet_name": "testtest1.keystore" } }

```
## Open/Unlock wallet

**API Function**

> API Description: Open/Unlock wallet
>
> **API Address**
>
> URL:  /v1/wallet/unlockaccount
>
> **Result Format**
>
> JSON
>
> **Request Format**
>
> POST

**Request Parameter: **

| Parameter         | Mandatory | Type   | Default Value | Description     |
| ------------ | ---- | ------ | ------ | -------- |
| account_name | TRUE | string | Null     | Account name |
| passwd       | TRUE | string | Null     | Passsword     |

**Response Field:**

| Parameter    | Type       | Description                                  |
| ------- | ---------- | ---------------------------------- |
| errcode | uint32     | Error code, 0-succeed, others refer to error code chapter
|
| msg     | string     | Response description                         |
| result  | jsonObject | Response result                            |
| unlock  | bool       | Unlock Result                            |

**Field change**

- Null

  **API Sample**

> Address : <http://127.0.0.1:6869/v1/wallet/unlockaccount  >

- Request :
```

{ "account_name": "testtest1", "passwd": "123456" }

```
- Response :
```

HTTP/1.1 200 OK { "errcode": 0, "msg": "success", "result": { "unlock": true } }

```
## Close/Lock Wallet

**API Function**

> API Description: Close/Lock wallet
>
> **API Address**
>
> URL:  /v1/wallet/lockaccount
>
> **Result Format**
>
> JSON
>
> **Request Format**
>
> POST

**Request Parameter: **

| Parameter         | Mandatory | Type   | Default Value | Description     |
| ------------ | ---- | ------ | ------ | -------- |
| account_name | TRUE | string | Null     | Account name |

**Response Field:**

| Parameter    | Type       | Description                                  |
| ------- | ---------- | ---------------------------------- |
| errcode | uint32     | Error code, 0-succeed, others refer to error code chapter
|
| msg     | string     | Response description                           |
| result  | jsonObject | Response result                              |
| lock    | bool       | Unlock Result                               |

**Field change**

- Null

  **API Sample**

> Address : <http://127.0.0.1:6869/v1/wallet/lockaccount  >
```

```
- Request :
```

{ "account_name": "testtest1" }

```
- Response :
```

HTTP/1.1 200 OK { "errcode": 0, "msg": "success", "result": { "lock": true } }

```
## Query wallets' list

**API Function**

> API Description: Query wallets' list
>
> **API Address**
>
> URL:  /v1/wallet/list
>
> **Result Format**
>
> JSON
>
> **Request Format**
>
> GET

**Request Parameter: **

| Parameter | Mandatory | Type | Default Value | Description |
| ---- | ---- | ---- | ------ | ---- |
| Null   |      |      |        |      |

**Response Field:**

| Parameter         | Type       | Description
  |
| ------------ | ---------- | ---------------------------------------- |
| errcode      | uint32     | Error code, 0-succeed, others refer to error code cha
pter       |
| msg          | string     | Response description
  |
| result       | jsonObject | Response result                                |
| wallet_path  | string     | Wallet path+file name                          |
| account_name | string     | Account name                                   |
| public_key   | string     | Public key value. It could not be found in chain if t
```

```
he account hasn't been registered to chain. |

**Field change**

- Null

  **API Sample**

> Address : <http://127.0.0.1:6869/v1/wallet/list >

- Request :
```

Null

```
- Response :
```

HTTP/1.1 200 OK { "errcode": 0, "msg": "success", "result": [{ "wallet_path": "C:\Users\jim\AppData\Roaming\bot\testtest12.keystore", "account_name": "testtest12", "public_key": "0454f1c2223d553aa6ee53ea1ccea8b7bf78b8ca99f3ff622a3bb3e62dedc712089033d6091d772965 47bc071022ca2838c9e86dec29667cf740e5c9e654b6127f" }, { "wallet_path": "C:\Users\jim\AppData\Roaming\bot\testtest2.keystore", "account_name": "testtest2", "public_key": "not found on Chain" }] }

```
## Query Public and Private Key Pair

**API Function**

> API Description: Query public and private Key pair, the precondition to invoke th
is API: the account must be in opened or unlocked status.
>
> **API Address**
>
> URL:  /v1/wallet/getkeypair
>
> **Result Format**
>
> JSON
>
> **Request Format**
>
> POST

**Request Parameter: **
```

```
| Parameter         | Mandatory | Type   | Default Value | Description     |
| ------------ | ---- | ------ | ------ | -------- |
| account_name | TRUE | string | Null     | Account name |

**Response Field:**

| Parameter         | Type       | Description                                 |
| ------------ | ---------- | ----------------------------------- |
| errcode      | uint32     | Error code, 0-succeed, others refer to error code cha
pter |
| msg          | string     | Response description                          |
| result       | jsonObject | Response result                            |
| account_name | string     | Account name                              |
| private_key  | string     | Private Key value                                |
| public_key   | string     | Public Key value                               |

**Field change**

- Null

  **API Sample**

> Address : <http://127.0.0.1:6869/v1/wallet/getkeypair>

- Request :
```

{ "account_name":"testtest" }

```
- Response :
```

{ "errcode": 0, "msg": "success", "result": { "account_name": "testtest", "private_key": "8a068d821ad8d58fe8caaa87f32ea69f889f6b8ce8c6bc3bc67ce2c68c43da3b", "public_key": "043680ae1b81f87274d7051e6101afc8f8da3cd978bb75b22f24becb98afb456f110151644330ff6c742e44f8e9652f1ab5ea1cd3997bebe5a23a4389ff0cb6493" } }

```
## Sign for Transaction

**API Function**

> API Description: Sign for transaction
>
> **API Address**
>
> URL:  /v1/wallet/signtransaction
>
```

```
> **Result Format**
>
> JSON
>
> **Request Format**
>
> POST

**Request Parameter: **

| Parameter     | Mandatory | Type   | Default Value | Description
                            |
| -------- | ---- | ------ | ------ | ----------------------------------------
-- |
| sender   | TRUE | string | Null     | Account name who send the transaction
   |
| contract | TRUE | string | Null     | Contract name
   |
| method   | TRUE | string | Null     | Contract method
          |
| param    | TRUE | string | Null     | Business structured data, after the bpl ser
ialization, it converts to hexadecimal string |

**Response Field:**

| Parameter    | Type       | Description                                      |
| ------- | ---------- | ------------------------------------------ |
| errcode | uint32     | Error code, 0-succeed, others refer to error code chapter
      |
| msg     | string     | Response description                                     |
| result  | jsonObject | Response result, See the "Send Transaction information" re
quest field specifically |

**Field change**

- Null

  **API Sample**

> Address : <http://127.0.0.1:6869/v1/wallet/signtransaction>

- Request :
```

{ "sender": "bottos", "contract": "bottos", "method": "newaccount", "param": "" }

```
- Response :
```

HTTP/1.1 200 OK { "errcode": 0, "msg": "success", "result": { "version": 1, "cursor_num": 26, "cursor_label": 1948301132, "lifetime": 1537330126, "sender": "bottos", "contract": "bottos", "method": "newaccount", "param": "", "sig_alg": 1, "signature": "aa249f7c0e5a5564b48ada5e9f0ffa4665d955b08e08c43906a156fa5bf30272547f2e4cde31fa8d5d303f36a4de0718ec3284b2e82f93fa7da50a54cbdcf86a" } }

```
## Sign for data

**API Function**

> API Description: sign for data
>
> **API Address**
>
> URL:  /v1/wallet/signhash
>
> **Result Format**
>
> JSON
>
> **Request Format**
>
> POST

**Request Parameter: **

| Parameter         | Mandatory | Type   | Default Value | Description         |
| ------------ | ---- | ------ | ------ | ------------ |
| account_name | TRUE | string | Null     | Account name     |
| param        | TRUE | []byte | Null     | Arrays to be signed |

**Response Field:**

| Parameter       | Type       | Description                                  |
| ---------- | ---------- | ------------------------------------ |
| errcode    | uint32     | Error code, 0-succeed, others refer to error code chapter |
| msg        | string     | Response description                         |
| result     | jsonObject | Response result                            |
| hash_value | string     | Signature value                              |

**Field change**

- Null

  **API Sample**

> Address : <http://127.0.0.1:6869/v1/wallet/signhash
```

```
- Request :
```

{ "account_name": "bottos", "param": [11] }

```
- Response :
```

HTTP/1.1 200 OK { "errcode": 0, "msg": "success", "result": { "hash_value": "ed6b57fa91ee369af925e4c0545c06f5829ce3d2825949931a2b8816039f53f47160701c779b528abe84f5d6acdb2d9c46de6958e42606d2e4d7b9b072991729" } } ```

Wallet SDK

Classes

BottosWalletSDK

Objects

Api : **`object`**

Contract : **`object`**

Tool : **`object`**

Wallet : **`object`**

Typedefs

functionCallback : **`function`**

This callback is displayed as a global member.

originFetchTemplate : **`Object`**

BottosWalletSDK

Kind: global class

new BottosWalletSDK(config)

Represents the BottosWalletSDK.

Param	Type	Default
config	`Object`	
[config.baseUrl]	`string`	`"http://localhost:8689/v1&quot;`
[config.version]	`number`	`1`

[config.crypto]	`Object`	`BTCryptTool`

bottosWalletSDK.Api

See Api.

Kind: instance property of `BottosWalletSDK`

bottosWalletSDK.Tool

See Tool.

Kind: instance property of `BottosWalletSDK`

bottosWalletSDK.Wallet

See Wallet.

Kind: instance property of `BottosWalletSDK`

bottosWalletSDK.Contract

See Contract.

Kind: instance property of `BottosWalletSDK`

bottosWalletSDK.setBaseUrl(baseUrl)

Kind: instance method of `BottosWalletSDK`

Param	Type	Description
baseUrl	`string`	The baseUrl used in request.

Api : `object`

Kind: global namespace

- Api : `object`
  - .chain_id
  - .request(url, params, [method])
  - .getAbi(contract, [callback]) ⇒ `Promise.<Object>` | `undefined`
  - .getBlockHeader([callback]) ⇒ `Promise.<Object>` | `undefined`

Api.chain_id

Documented as Api.chain_id

Kind: static property of `Api`

Api.request(url, params, [method])

Kind: static method of `Api`

Param	Type	Default
url	`string`	
params	`Object`	
[method]	`string`	`"POST"`

Api.getAbi(contract, [callback]) ⇒ `Promise.<Object>` | `undefined`

Returns the abi. If callback is undefined, this function will return a promise.

Kind: static method of `Api`

Param	Type	Description
contract	`string`	The contract name.
[callback]	`functionCallback`	The optional callback.

Api.getBlockHeader([callback]) ⇒ `Promise.<Object>` | `undefined`

Documented as Api.getBlockHeader.

Kind: static method of `Api`
Returns: `Promise.<Object>` | `undefined` - If callback is undefined, a promise will be returned.

Param	Type
[callback]	`functionCallback`

Contract : `object`

Kind: global namespace

- Contract : `object`
  - .deployContract(param, senderInfo) ⇒ `Promise.<Object>`
  - .deployABI(param, senderInfo) ⇒ `Promise.<Object>`
  - .callContract(originFetchTemplate, privateKey) ⇒ `Promise.<Object>`

Contract.deployContract(param, senderInfo) ⇒ `Promise.<Object>`

Deploy a contract.

Kind: static method of `Contract`

Param	Type	Default	Description	
param	`Object`			
[param.vm_type]	`number`	`1`	vm_type, now is 1.	
[param.vm_version]	`number`	`1`	vm_version, now is 1.	
param.contract_code	`Uint8Array \`	`ArrayBuffer`		wasm file buffer.
senderInfo	`Object`		The sender	
senderInfo.account	`string`		sender's account	
senderInfo.privateKey	`string \`	`Uint8Array`		sender's privateKey

Contract.deployABI(param, senderInfo) ⇒ `Promise.<Object>`

Deploy an abi.

Kind: static method of `Contract`

Param	Type	Description		
param	`Object`			
param.contract_abi	`string \`	`Uint8Array \`	`ArrayBuffer`	ABI content or file buffer.
senderInfo	`Object`	The sender		
senderInfo.account	`string`	sender's account		
senderInfo.privateKey	`string \`	`Uint8Array`	sender's privateKey	

Contract.callContract(originFetchTemplate, privateKey) ⇒ `Promise.<Object>`

Kind: static method of `Contract`

Param	Type	
originFetchTemplate	`Object`	
privateKey	`string \`	`Uint8Array`

Tool : `object`

Kind: global namespace

- Tool : `object`
  - .buf2hex(buffer) ⇒ `string`
  - .getRequestParams(originFetchTemplate, privateKey) ⇒ `Promise.<Object>`

Tool.buf2hex(buffer) ⇒ `string`

Kind: static method of `Tool`
Returns: `string` - Hexadecimal string.

Param	Type	Description
buffer	`Uint8Array`	Uint8Array buffer.

Tool.getRequestParams(originFetchTemplate, privateKey) ⇒ `Promise.<Object>`

Kind: static method of `Tool`

Param	Type	Description	
originFetchTemplate	`originFetchTemplate`		
privateKey	`string \`	`Uint8Array`	Your private key.

Wallet : `object`

Kind: global namespace

- Wallet : `object`
  - .createAccountByIntro(params) ⇒ `Object`
  - .createKeys() ⇒ `Object`
  - .createAccountWithIntro(params, referrerInfo) ⇒ `Promise.<Object>`
  - .recover(password, keyObject) ⇒ `Uint8Array`
  - .getAccountInfo(account_name) ⇒ `Promise.<Object>`
  - .sendTransaction(params, privateKey) ⇒ `Promise.<Object>`
  - .stake(amount, privateKey) ⇒ `Promise.<Object>`
  - .unstake(amount, privateKey) ⇒ `Promise.<Object>`
  - .claim(amount, privateKey) ⇒ `Promise.<Object>`

Wallet.createAccountByIntro(params) ⇒ `Object`

Kind: static method of `Wallet`
Returns: `Object` - keystore

Param	Type	Description

params	`Object`	the params required for create keystore	
params.account	`string`	account	
params.password	`string`	password	
params.privateKey	`string \`	`Uint8Array`	privateKey

Wallet.createKeys() ⇒ `Object`

Create public and private key pair

Kind: static method of `Wallet`
Returns: `Object` - keys

Wallet.createAccountWithIntro(params, referrerInfo) ⇒ `Promise.<Object>`

register account on chain

Kind: static method of `Wallet`

Param	Type	Description	
params	`Object`	The params	
params.account	`string`	The new user's account	
params.publicKey	`string \`	`Uint8Array`	The publicKey provided by the new user
referrerInfo	`Object`	The referrer	
referrerInfo.account	`string`	referrer's account	
referrerInfo.privateKey	`string \`	`Uint8Array`	referrer's privateKey

Wallet.recover(password, keyObject) ⇒ `Uint8Array`

Kind: static method of `Wallet`
Returns: `Uint8Array` - privateKey

Param	Type	Description
password	`string`	password
keyObject	`Object`	the keystore

Wallet.getAccountInfo(account_name) ⇒ `Promise.<Object>`

Kind: static method of `Wallet`

Param	Type
account_name	`string`

Wallet.sendTransaction(params, privateKey) ⇒ `Promise.<Object>`

Kind: static method of `Wallet`

Param	Type	
params	`Object`	
params.from	`string`	
params.to	`string`	
params.value	`string \`	`number`
privateKey	`string`	

Wallet.stake(amount, privateKey) ⇒ `Promise.<Object>`

Kind: static method of `Wallet`

Param	Type	
amount	`number`	
privateKey	`string \`	`Uint8Array`

Wallet.unstake(amount, privateKey) ⇒ `Promise.<Object>`

Kind: static method of `Wallet`

Param	Type	
amount	`number`	
privateKey	`string \`	`Uint8Array`

Wallet.claim(amount, privateKey) ⇒ `Promise.<Object>`

Kind: static method of `Wallet`

Param	Type	
amount	`number`	
privateKey	`string \`	`Uint8Array`

functionCallback : `function`

This callback is displayed as a global member.

Kind: global typedef

Param	Type	Description
err	`*`	The callback error.
result	*	The success result.

originFetchTemplate : `Object`

Kind: global typedef
Properties

Name	Type	Default	Description
[originFetchTemplate.version]	`number`	`1`	Default value is 1.
originFetchTemplate.sender	`string`		Default value is bottos.
[originFetchTemplate.contract]	`string`	`"bottos"`	The contract. Default value is bottos.
originFetchTemplate.method	`string`		
originFetchTemplate.param	`Object`		
[originFetchTemplate.sig_alg]	`number`	`1`	Default value is 1.

Building a development environment

Golang environment Installation

Note: In this specification we use the version of Ubuntu Server 18.04.1 LTS

- Download golang source code

```
wget https://dl.google.com/go/go1.11.linux-amd64.tar.gz
```

- Extract golang package into directory of /usr/local

```
sudo tar -zxvf go1.11.linux-amd64.tar.gz -C /usr/local
```

- Add the directory `/usr/local/go/bin` into system PATH environment variable

```
export PATH=$PATH:/usr/local/go/bin
```

- Check if the golang was installed successfully

```
go env
```

If the following results occur, the configuration is successful

```
GOARCH="amd64"
GOBIN="/home/bottos/go/bin"
GOCACHE="/home/bottos/.cache/go-build"
GOEXE=""
GOFLAGS=""
GOHOSTARCH="amd64"
GOHOSTOS="linux"
GOOS="linux"
GOPATH="/home/bottos/go"
GOPROXY=""
GORACE=""
GOROOT="/usr/lib/go1.10"
GOTMPDIR=""
GOTOOLDIR="/usr/lib/go1.10/pkg/tool/linux_amd64"
GCCGO="gccgo"
CC="gcc"
CXX="g++"
CGO_ENABLED="1"
GOMOD=""
CGO_CFLAGS="-g -O2"
CGO_CPPFLAGS=""
```

```
CGO_CXXFLAGS="-g -O2"
CGO_FFLAGS="-g -O2"
CGO_LDFLAGS="-g -O2"
PKG_CONFIG="pkg-config"
GOGCCFLAGS="-fPIC -m64 -pthread -fmessage-length=0 -fdebug-prefix-map=/tmp/go-build
892316758=/tmp/go-build -gno-record-gcc-switches"
```

- Configure the `GOPATH` working directory

Enter the directory corresponding to `GOPATH` in the return results of the ' go env ' above and create the appropriate working directory

Note: `The exact catalogue here depends on the real situation of the individual.`

```
mkdir /home/bottos/go
cd /home/bottos/go
mkdir bin pkg src src/github.com src/golang.org
```

At this point, the golang environment has been successfully set up.

Install and Run Bottos

> Difference: Single-node is generally used to start a single node locally for contract development and testing, and is not connected to other nodes, generally can retain the default configuration. Multiple nodes are connected to the Bottos main network or to the test network. It can also be used to form a small private network of its own. Generally need to do some basic configuration.

Start a single node

- Come into `GOPATH` directory

```
cd  ~/go/src/github.com/
```

- Download and extract the file bottos.tar.gz

```
wget https://github.com/bottos-project/bottos/releases/download/tag_bottos3.2/botto
s.tar.gz
tar zxvf bottos
cd bottos
```

- Start a Bottos Single Node

```
./bottos --delegate=bottos
```

- `--delegate` : The designated block producer's account

If the following information is returned, the node starts successfully

```
CommitBlock by p2p: lib: 1
InsertBlock: number:1, delegate:bottos, trxn:0, time=1537948299, hash: 566fb29ab982
c128bf6c71297bc4e7d558e0f86ae89a7f3955ea46b04689fb5a, prevHash=caf2bae84f7041235421
1dd5028142eca6901b06b9a65dfbe9df065bcf56e291
CommitBlock by p2p: lib: 2
InsertBlock: number:2, delegate:bottos, trxn:0, time=1537948302, hash: 8abe6aef2224
9ab58d6c7cd3970f909571c4e818f3757d9de7d86870bfc7465b, prevHash=566fb29ab982c128bf6c
71297bc4e7d558e0f86ae89a7f3955ea46b04689fb5a
```

Connect to a test network

Connect the single node initiated above to the Bottos test network. The following configuration is required.

Edit the `config-testnet.toml` file

- P2PServAddr: Change this into you node's external IP

```
P2PServAddr = "192.168.1.1"   //Change this into you node's external IP
```

Then run the following command to connect the current node to the test network

' Note ': If there is a ' datadir ' cache directory under the project directory, we first need to run the following command to remove the cache.

```
rm -rf datadir
```

Start node and connect it to test network

```
./bottos --config="./config-testnet.toml" --genesis="./genesis-testnet.toml"
```

Wait a while and if a large number of the following print information appears, the block is automatically synchronized. That is to say it has been successfully connected to the test network

```
CommitBlock by p2p: lib: 1
InsertBlock: number:1, delegate:bottos, trxn:0, time=1537888767, hash: 03f6c7aa7231
4be76902b6c2d4b86b7afbb07d2b4b4dec67caf6fc51e125e9ed, prevHash=98128aa21d634eda9cb0
152314b06480d4c51b0bf18ea6d39f5189388e1bf4ee
CommitBlock by p2p: lib: 2
InsertBlock: number:2, delegate:bottos, trxn:0, time=1537888770, hash: c87a1c59aaa8
7f890169a1016931b3a9e539e72e475c0861623ed36fbd00c1b4, prevHash=03f6c7aa72314be76902
b6c2d4b86b7afbb07d2b4b4dec67caf6fc51e125e9ed
CommitBlock by p2p: lib: 3
InsertBlock: number:3, delegate:bottos, trxn:0, time=1537888773, hash: 3bcf9ecf1168
91b226b2c6b31578d5f1ee867a75b667752286eeaf3d237e684b, prevHash=c87a1c59aaa87f890169
a1016931b3a9e539e72e475c0861623ed36fbd00c1b4
CommitBlock by p2p: lib: 4
```

Development and deployment of Smart Contracts

Now, We use ' C + + ' to develop a smart contract for a mobile phone contact, writing the user information to the chain before reading the information out.

Sample of How to Develop a Smart Contract

The code is as follows

> The functions of the following contract are the storage and reading of `userName` and `userinfo`

- testRegUser.hpp

```
#define USER_NAME_LEN (20)
#define USER_INFO_LEN (20)


//@abi action reguser
struct UserInfo {
    char userName[USER_NAME_LEN];
    char userInfo[USER_INFO_LEN];
};


//@abi table userinfo:[index_type:string, key_names:userName, key_types:string]
struct UserBaseInfo {
    char userInfo[USER_INFO_LEN];
};
```

- testRegUser.cpp

```
#include "contractcomm.hpp"
#include "string.hpp"
#include "testRegUser.hpp"


#define PARAM_MAX_LEN (2048)

#define ERROR_PACK_FAIL (-1)
#define ERROR_UNPACK_FAIL (-2)
#define ERROR_SAVE_DB_FAIL (-3)
#define ERROR_METHOD_INVALID (-4)



static bool unpack_struct(MsgPackCtx *ctx, UserInfo *info)
```

```
{
    uint32_t size = 0;
    UNPACK_ARRAY(2)

    UNPACK_STR(info, userName, (USER_NAME_LEN+1))
    UNPACK_STR(info, userInfo, (USER_INFO_LEN+1))

    return 1;
}

static bool pack_struct(MsgPackCtx *ctx, UserBaseInfo *info)
{
    PACK_ARRAY16(1)
    PACK_STR16(info, userInfo )

    return 1;
}

void init()
{
}

int start(char* method)
{
    if (0 == strcmp("reguser", method))
    {
        char param[PARAM_MAX_LEN] = {0};
        uint32_t paramLen = 0;
        UserInfo userinfo = {{0}};

        paramLen = getParam(param, PARAM_MAX_LEN);

        MsgPackCtx ctx;
        msgpack_init(&ctx, (char*)param, paramLen);
        bool unpackSuc = unpack_struct(&ctx, &userinfo);
        if (!unpackSuc)
        {
            char pstr[] = "unpack userinfo failed";
            myprints(pstr);

            return ERROR_UNPACK_FAIL;
        }

        UserBaseInfo userBaseInfo = {{0}};

        strcpy(userBaseInfo.userInfo, userinfo.userInfo);

        msgpack_init(&ctx, (char*)param, paramLen);

        bool packSuc = pack_struct(&ctx, &userBaseInfo);
        if (!packSuc)
```

```
        {
            char pstr[] = "pack userbaseinfo failed";
            myprints(pstr);

            return ERROR_PACK_FAIL;
        }

        char objname[] = "userreginfo";

        uint32_t saveLen = setBinValue(objname, strlen(objname), userinfo.userName,
 strlen(userinfo.userName), ctx.buf, ctx.pos);
        if (0 == saveLen)
        {
            char pstr[] = "save db failed";
            myprints(pstr);

            return ERROR_SAVE_DB_FAIL;
        }

        return 0;

    }
    else
    {
        char pstr[] = "invalid method";
        prints(pstr, strlen(pstr));
        return ERROR_METHOD_INVALID;
    }
}
```

Build a smart contract

Our smart contracts need to be compiled into a `. Wasm` format file for normal calls to be executed. So, the first step is to compile our smart contract code into a `. Wasm` file

- Download the compilation tool

```
git clone https://github.com/bottos-project/contract-tool-cpp.git
```

- Compile the contract
- Enter the contract directory `Testreguser` and then run the following command to compile the contract

```
python ../gentool.py wasm testRegUser.cpp
```

At this point, `Testreguser.wast` and `testreguser.wasm` files are generated in the contract directory

2. Then run the following command to generate file `Testreguser.abi`

```
python ../gentool.py testRegUser.hpp
```

At this point, the contract directory generates a `Testreguser.abi` file

Deploy Smart Contract and ABI file

Once we have compiled the smart contract, we need the `bcli` tool to deploy the `. Wasm` file to the `Bottos` chain

BCLI Tool Usage

- Go to the `Bottos` project and deploy the compiled `wasm` file using the `bcli` tool

Copy the `BCLI` tool to the directory where the `wasm` and `Abi` files are located. Execute the following command

See how `bcli` is used

```
bcli --help
```

- Create a public-private key pair

```
./bcli --servaddr 192.168.52.130:8689 wallet generatekey
```

The above command returns the public-private key pair.

```
{
    "private_key": "773df18f6d7e1fa3fda1f2c36806bc40b20bbdd61ab59d00a2b47ecc4f9718e
6",
    "public_key": "04daec4f6b166ea21f5a3c4f8d50ef638ad0312cb01def711ba0ab350c10abeb
a8a137e5b5c4fffcbfdae4d747595bb33f24fa2174abf8f631610d253977dfed23"
}
```

- Create an account by public key

```
./bcli --servaddr 192.168.52.130:8689 account create --username john --pubkey 04dae
c4f6b166ea21f5a3c4f8d50ef638ad0312cb01def711ba0ab350c10abeba8a137e5b5c4fffcbfdae4d7
47595bb33f24fa2174abf8f631610d253977dfed23
```

If the following information is returned, the account creation is successful.

```
Create account: john Succeed
Trx:
{
    "version": 1,
    "cursor_num": 1071,
    "cursor_label": 1597958457,
    "lifetime": 1537171186,
    "sender": "bottos",
    "contract": "bottos",
    "method": "newaccount",
    "param": {
        "name": "john",
        "pubkey": "04daec4f6b166ea21f5a3c4f8d50ef638ad0312cb01def711ba0ab350c10abeb
a8a137e5b5c4fffcbfdae4d747595bb33f24fa2174abf8f631610d253977dfed23"
    },
    "param_bin": "dc0002da00046a6f686eda008230346461656334663662313636656132316635
6133633466386435306566363338616430333132636230316465663731316261306162333530633130
6165626138613133376535623563346666666362666461653464373437353935626233366323466613
2313734616266386636331363130643235333937376466656432 33",
    "sig_alg": 1,
    "signature": "fe62ac036c193a870ef1667f264b62c6c17c9d09f00958a9f6971931a187bbbf6
b5c6129a162765cfcc02b5da23d602bf4e19b84036bb88e58ce29ad38715117"
}
```

- See if your account creates results

```
./bcli --servaddr 192.168.52.130:8689 account get --username john
```

If you return a description, the creation succeeds.

```
Account: john
Balance: 0.00000000 BTO
```

- Deploy the `wasm` file to the chain

```
./bcli --servaddr 192.168.52.130:8689 contract deploy --name john --code ./testRegU
ser.wasm -abi testRegUser.abi
```

`Note` : The name of the contract here is the name of the account we created above

If the following information is returned, the contract was successfully deployed.

```
Deploy contract: john Succeed
Trx:
{
    "version": 1,
    "cursor_num": 45,
```

```
    "cursor_label": 3016045512,
    "lifetime": 1537497253,
    "sender": "john",
    "contract": "bottos",
    "method": "deploycode",
    "param": {
        "name": "john",
        "vm_type": 1,
        "vm_version": 1,
        "contract_code": "0061736d01000000012506600027f7f017f60027f7f0060067f7f7f7f7
f7f017f60037f7f7f017f60000060017f017f023c0403656e7608676574506172616d000003656e7606
6d656d736574000303656e76067072696e7473000103656e760b73657442..."
    },
    "param_bin": "dc0004da00046a6f686ecc01cc01c50ea40061736d01000000012506600027f7f0
17f60027f7f0060067f7f7f7f7f017f60037f7f7f017f60000060017f017f023c0403656e76086765
74506172616d000003656e76066d656d736574000303656e7606...",
    "sig_alg": 1,
    "signature": "f2f96610b5ede1975359fd962b447264da4b1fa492869350a2a93540122dfd1d2
b10f1bfe722ce484ea054b59e2f8bd7120fa3ce17cc2e0de893997a2dcc18a5"
}
TrxHash: 7d3bea7da59d3a74b2d9625505bd64099efd2a509c20387ff28c924a12619250
signTrx: [183 153 239 97 104 48 205 123 133 153 174 121 88 251 238 86 212 200 22 14
3 253 84 33 161 96 37 163 152 184 164 190 69]
trx :%!(EXTRA *api.Transaction=version:1 cursor_num:45 cursor_label:3016045512 life
time:1537497253 sender:"john" contract:"bottos" method:"deployabi" param:"dc0002da0
0046a6f686ec503867b0a09227479706573223a205b5d2c0a092273747275637473223a205b0a202020
20202020202020202020207b0a20202020202020202020202020202009226e616d65223a2022557365724
96e666f222c0a20202020202020202020202020202009226261736 5223a2022222c0a2020202020202020
20202020202020092266696 56c6473223a207b0a20202020202020202020202020202009092275736572496e6
16d65223a2022737472696e67222c0a20202020202020202020202020202009092275736572496e666f22
3a2022737472696e67220a20202020202020202020202020202020202020207d0a20202020202020202
02020202020097d2c0a20202020202020202020202020207b0a20202020202020202020202020200922
6e616d65223a2022557365724261736 5496e666f222c0a20202020202020202020202020200922626 17
365223a2022222c0a20202020202020202020202020202009226669656c6473223a207b0a202020202020
20202020202020200909227573657 2496e666f223a2022737472696e67220a20202020202020202020202
02020202020202020207d0a20202020202020202020202020202097d0a202020202020205d2c0a092261
6374696f6e73223a205b0a20202020202020202020202020207b0a20202020202020202020202020200
922616374696f6e5f6e616d65223a2022726567 5736572222c0a20202020202020202020202020202009
2274797065223a2022557365724 96e666f220a20202020202020202020202020207d0a2020202020202
05d2c0a09227461626c6573223a205b0a20202020202020202020202020207b0a202020202020202020
20202020209227461626c655f6e616d65223a2022757365724 96e666f222c0a20202020202020202020
02020202009226 96e6465785f74797065223a2022737472696e67222c0a20202020202020202020202020
202009226b65795f6e616d6573223a20205b0a20202020202020202020202020202009092275736572496e6
16d65220a20202020202020202020202020202009205d2c0a20202020202020202020202020202009226b65
795f7479706573223a20205b0a20202020202020202020202020202009092273747269 6e67220a2020202
02020202020202020202009205d2c0a20202020202020202020202020202009227479706573223a20225573
657242617365496e666f220a202020202020202020202020207d0a202020202020205d0a7d0a" sig
_alg:1 ){
    "errcode": 0,
    "msg": "trx receive succ",
    "result": {
```

```
        "trx": {
            "version": 1,
            "cursor_num": 45,
            "cursor_label": 3016045512,
            "lifetime": 1537497253,
            "sender": "john",
            "contract": "bottos",
            "method": "deployabi",
            "param": "dc0002da00046a6f686ec503867b0a09227479706573223a205b5d2c0a092
27374727563747322 3a205b0a20202020202020202020202020207b0a20202020202020202020202020
2009226e616d65223a202255736572496e666f222c0a20202020202020202020202020009226261736
5223a2022222c0a20202020202020202020202020202009226669656c6473223a207b0a20202020202020
20202020202020090922757365724e616d65223a2022737472696e67222c0a20202020202020202020202
02020200909227573657 2496e666f223a2022737472696e67220a20202020202020202020202020202020
20202020207d0a20202020202020202020202020202020097d2c0a20202020202020202020202020207b0a2
02020202020202020202020202009226e616d65223a2022557365724261736 5496e666f222c0a202020
20202020202020202020202009226261736 5223a2022222c0a20202020202020202020202020202009226666
9656c6473223a207b0a20202020202020202020202020202009092275736572496e666f223a2022737472
696e67220a20202020202020202020202020202020202020207d0a20202020202020202020202020200
97d0a202020202020205d2c0a0922616374696f6e73223a205b0a20202020202020202020202020207b
0a20202020202020202020202020202009226163746 96f6e5f6e616d65223a2022726567757365 72222c0
a20202020202020202020202020202009227479706 5223a202255736572496e666f220a20202020202020
202020202020207d0a202020202020205d2c0a09227461626c6573223a205b0a20202020202020202020
0202020207b0a20202020202020202020202020202009227461626c655f6e616d65223a20227573657269
6e666f222c0a20202020202020202020202020202009226696e6465785f74797065223a2022737472696e6
7222c0a20202020202020202020202020202009226b65795f6e616d6573223a20205b0a20202020202020
20202020202020090922757365724e616d65220a20202020202020202020202020202009205d2c0a20202
02020202020202020202009226b65795f7479706573223a20205b0a20202020202020202020202020202
0090922737472696e67220a20202020202020202020202020202009205d2c0a20202020202020202020202
02020200922747970 65223a2022557365724261736 5496e666f220a20202020202020202020202020202020
7d0a202020202020205d0a7d0a",
            "sig_alg": 1,
            "signature": "7d7ab2051ac3e2c419614849e036678e996a65999be556d5259087ff9
30c1c1a716d529c7062371e57a0fd44894247b8e183b6cb1273dc517d8f5d8e64eaf8c5"
        },
        "trx_hash": "bfbeed13120730d77a9c3ea8aa0bb1f6bc8b4df78dc8fd306581ca8f598536
17"
    }
}
```

Dapp Development and commissioning

Download and introduce the ' SDK ' library file

Download and import ' bottos-sdk-js.min.js ' library files

`bottos-sdk-js.min.js` file download address :https://github.com/bottos-project/bottos-sdk-js

Create a basic project structure

```
Project Catalog
    - index.html
    - index.js
    - bottos-sdk-js.min.js
```

- index.html

By injecting the ' Index.js ' and ' bottos-sdk-js.min.js ' in index.html, the code is as follows

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=375px, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Bottos Demo</title>
    <script src="./bottos-sdk-js.min.js"></script>
    <script src="./index.js"></script>
</head>
<body>
    <div style="margin-top:50px">
        <button style="width:100px;height:30px;background-color:coral" onClick="cre
ateAccount()">创建账户</button>
        <button style="width:100px;height:30px;background-color:coral" onClick="pus
hTransaction()">发布联系人</button>
        <button style="width:100px;height:30px;background-color:bisque" onClick="ge
tTransaction()" >获取联系人</button>
    </div>


    <div style="margin-top:100px">
        <p>Create account excution result</p>
        <div id="createAccount" style="min-height:150px;background-color:#ccc">

        </div>
```

```
        <p>Publish the contact people execution result</p>
        <div id="pushTransaction" style="min-height:150px;background-color:#ccc">

        </div>

        <p>Get the contact people execution result</p>
        <div id="getTransaction" style="min-height:150px;background-color:#ccc">

        </div>
    </div>

</body>
</html>
```

- index.js

' Index.js ' implements the write and read operation of the smart contract on the chain, the code is as follows

```
var BottosWalletSDK = window.BottosWalletSDK
const config = {
    baseUrl:'http://192.168.52.130:8689/v1',
    version:1
}
var SDK = new BottosWalletSDK(config)
var Tool = SDK.Tool
var Wallet = SDK.Wallet
var Contract = SDK.Contract
var Api = SDK.Api

let account = 'testaccount'
let password = 'testpassword'
let keystore = null

// Create account
function createAccount(){
    console.log("createAccount")
    let params = {account:account,password:password}
    Wallet.createAccount(params)
        .then(response=>{
            keystore = response
            document.getElementById('createAccount').innerHTML = JSON.stringify(res
ponse)
        }).catch(error=>{
            console.log({error})
            document.getElementById('createAccount').innerHTML = JSON.stringify(err
or)

        })
}
```

```
// Invoke the contract
function callContract(requestParam){
    let params = {
        method:'reguser',
        contract:'john',
        sender:account,
        param:{
            userName:'john',
            userInfo:JSON.stringify({phone:'110120',age:18})
        }
    }

    if(keystore == null){
        alert('Please create the account at first')
        return
    }
    let privateKey = Wallet.recover(password,keystore)
    let privateKeyStr = Tool.buf2hex(privateKey)

    Contract.callContract(params,privateKeyStr)
        .then(response=>{
            console.log({response})
            document.getElementById('pushTransaction').innerHTML = JSON.stringify(r
esponse)
        }).catch(error=>{
            console.log({error})
            document.getElementById('pushTransaction').innerHTML = JSON.stringify(e
rror)
        })
}

// publish the cotact people
function pushTransaction(){
    console.log("pushTransaction")
    callContract()
}

// Read the contact people
function getTransaction(){
    console.log("getTransaction")
    // callContract()
    let url =  config.baseUrl + '/common/query'
    let params = {
        contract:'john',
        object:'userreginfo',
        key:'john'

    }

    fetch(url,{
```

```
        method:'POST',
        body:JSON.stringify(params)
    }).then(function(response){return response.json()})
    .then(function(response){
        document.getElementById('getTransaction').innerHTML = JSON.stringify(respon
se)
    }).catch(function(error){
        document.getElementById('getTransaction').innerHTML = JSON.stringify(error)
    })
}
```

Verify the Contract Execution Results

- Create an account as a first

Click ' Create account ' and if the following structure is returned, the account creation is successful and the return value is the user's ' KeyStore ' file and needs to be securely saved.

```
{"account":"testaccount","crypto":{"cipher":"aes-128-ctr","ciphertext":"4cd010eba51
ff45d9a752f38d64a4def7697a142570253cc381afa2b04288e68","cipherparams":{"iv":"46b469
bb1c87751852857114b928e6ad"},"mac":"04051cce0379362c28d8027a19c8b79f03621602c2eff68
d8b793f94ae6f58fb","kdf":"scrypt","kdfparams":{"dklen":32,"n":1024,"r":1,"p":8,"sal
t":"cf9dbed2413764c640018b9fccb6629ed38dad72b333c7d12e89994523ec7857"}},"id":"48a92
3c2-1c40-46aa-b0c1-8f24d8e1bc81","version":3}
```

- Writing a contract

Click the ' Publish Contacts ' button to indicate that the write contract was executed successfully if the following results are returned

```
{"errcode":0,"msg":"trx receive succ","result":{"trx":{"version":1,"cursor_num":286
,"cursor_label":423110953,"lifetime":1537498176,"sender":"testaccount","contract":"
john","method":"reguser","param":"dc0002da00046a6f686eda001b7b2270686f6e65223a22313
130313230222c22616765223a31387d","sig_alg":1,"signature":"4149cb35eab48415082081aee
0ab3766b7a6730b1a2213c96638540227a8616f17cdf773ba90e3c515183c3fc5631fa53b65dcf93eed
320d53ec78a672738675"},"trx_hash":"eb79dc2d8a8461451ea55257fcbaa5897a9b9732e7cc3b0c
8a8322ab4909789b"}}
```

- Reading a contract

Click the ' Read Contacts ' button and if the following results are returned, the read contract execution is successful

```
{"errcode":0,"msg":"success","result":{"contract":"john","object":"userreginfo","ke
y":"john","value":"dc0001da001b7b2270686f6e65223a22313130313230222c22616765223a3138
7d"}}
```

At this point, one of our simple and complete dapp has been developed, and if you want to develop a more powerful dapp, you can view detailed interface documentation