# Dive into ICON - DApp

ICON foundation

# Dive into ICON - DApp

Step 1. What is a DApp?

Step 2. Review

Step 3. How to build a DApp

Step 4. Hands-on Exercise

# Dive into ICON - DApp

Step 1. What is a DApp?

1. Definition
2. Why should we build a DApp on ICON?

Step 2. Review

Step 3. How to build a DApp

Step 4. Hands-on Exercise

# Dive into ICON - DApp

Step 1. What is a DApp?

Step 2. Review

1. Dive into ICON - Tools
2. Dive into ICON - SCORE

Step 3. How to build a DApp

Step 4. Hands-on Exercise

# Dive into ICON - DApp

Step 1. What is a DApp?

Step 2. Review

Step 3. How to build a DApp

1. How to send Tx with SDK
2. How to use ICONex connect

Step 4. Hands-on Exercise

# Dive into ICON - DApp

Step 1. What is a DApp?

Step 2. Review

Step 3. How to build a DApp

Step 4. Hands-on Exercise

1. Make complete example page, Welcome & Scrooge.

# Set IDE

## IDE Setting

# IDE Setting

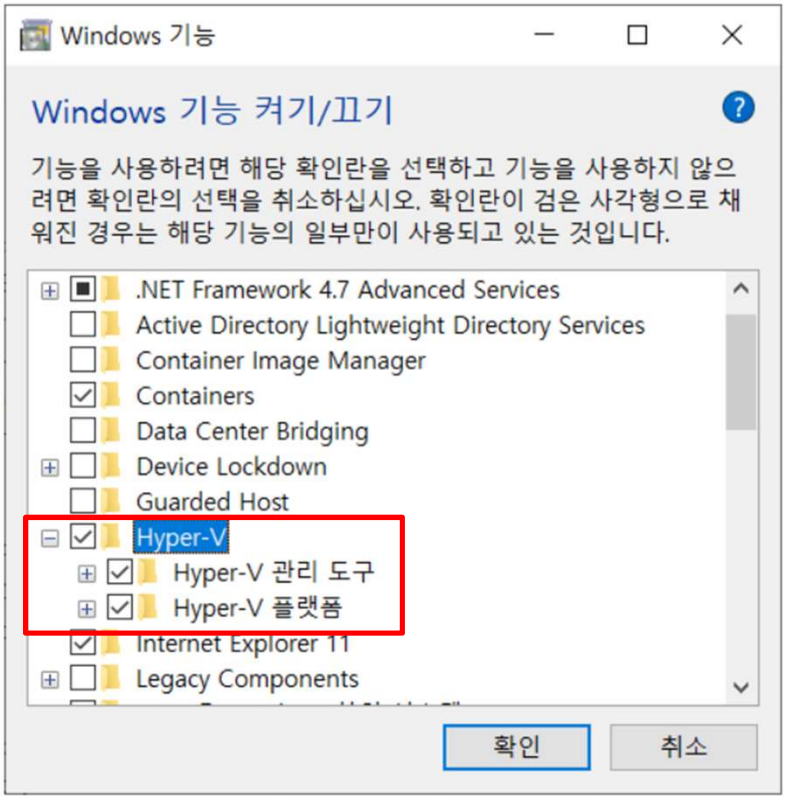
# IDE Setting – Based “윈도우에서 ICON 개발환경 구축하기-2”

# IDE Setting – Step

1. Install Docker Desktop.
2. [Windows]Turn on “Hyper-V”.
3. Install VSCode insider.
4. Pull ICONloop/tbears image in Docker.
5. Install plugin “Remote Development”
6. Connect between Container and VSCode.

# IDE Setting – Install Docker Desktop.

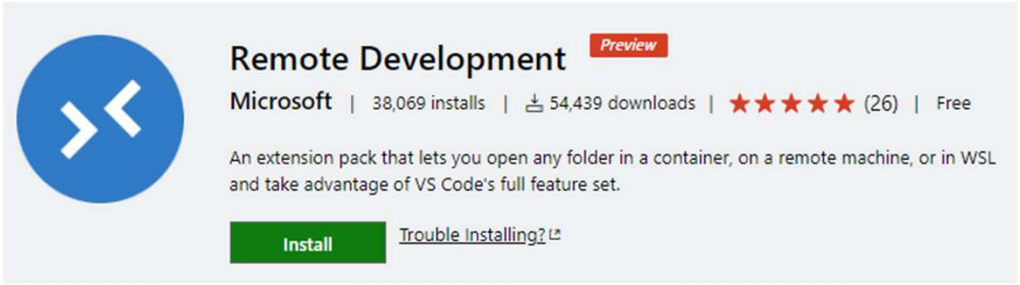
# IDE Setting – [Windows]Turn on “Hyper-V”.



# IDE Setting – Install VSCode insider.

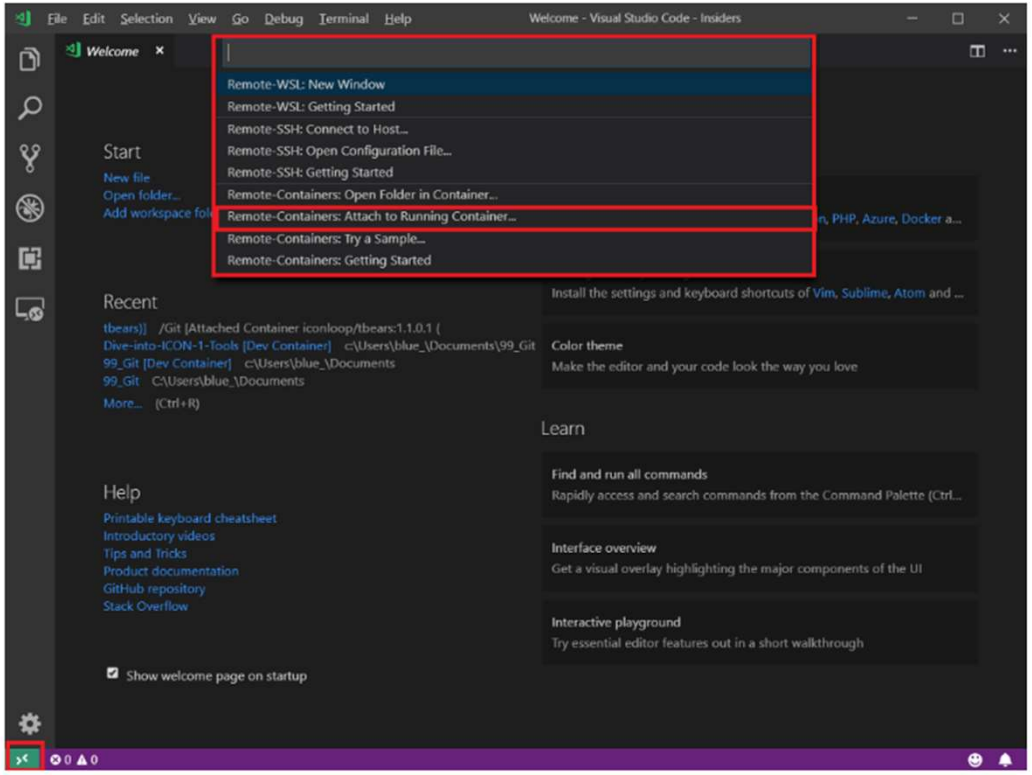
[Site Link]

# IDE Setting – Install plugin “Remote Development”

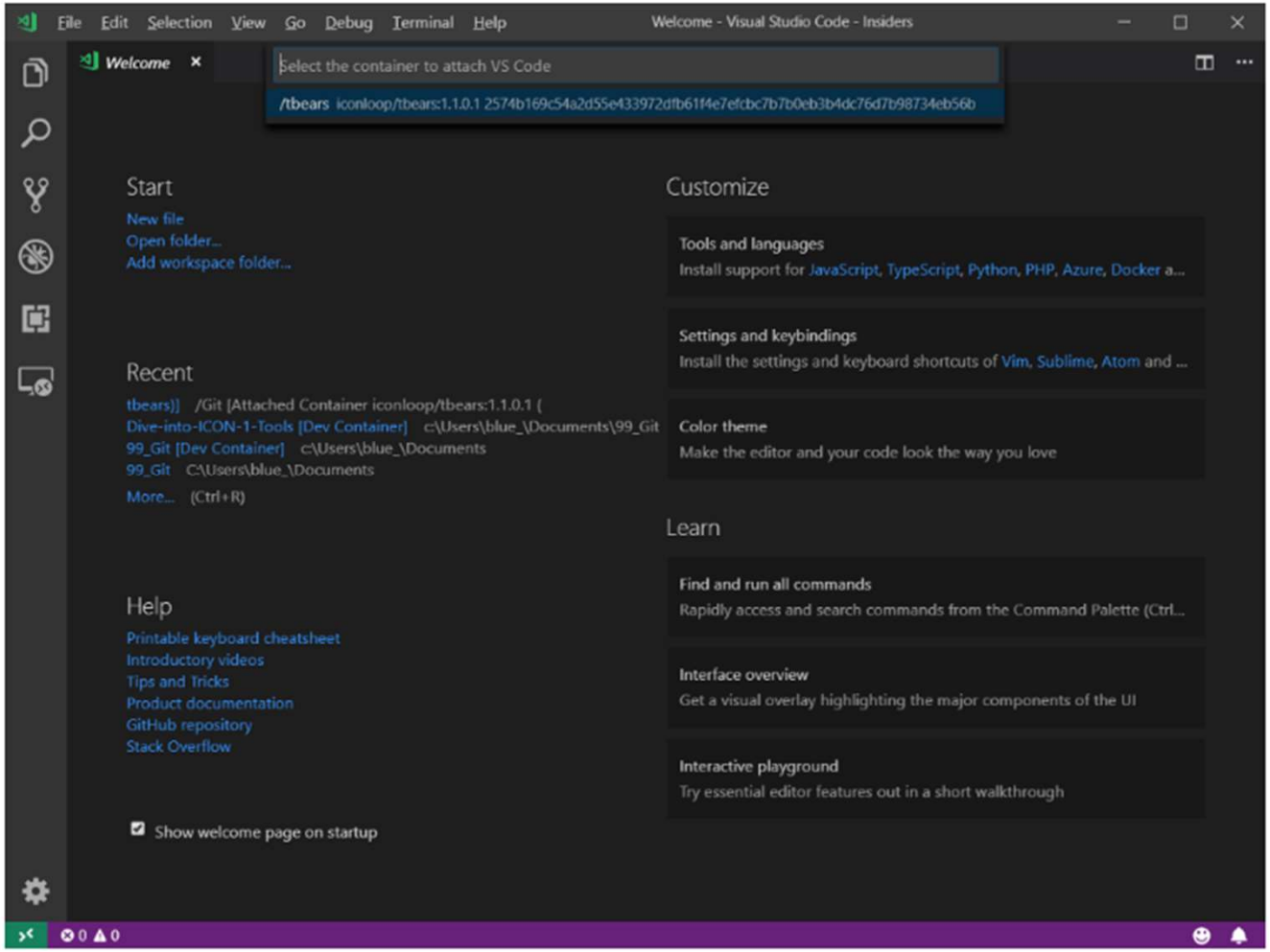



[Site Link]

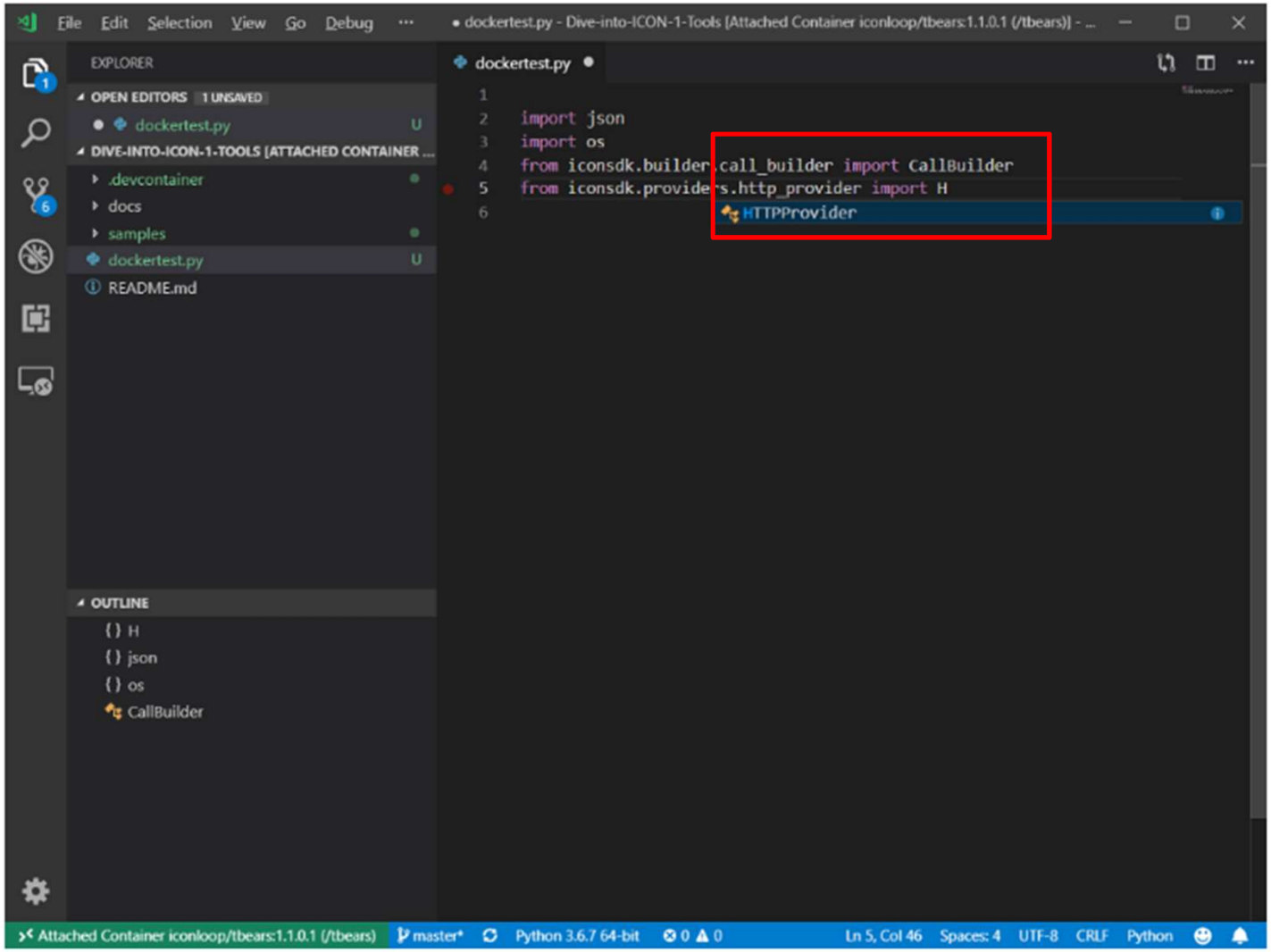
# IDE Setting –
# Connect between Container and VSCode.



# IDE Setting – Connect between Container and VSCode.



# IDE Setting


File Edit Selection View Go Debug
• dockertest.py - Dive-into-ICON-1-Tools [Attached Container iconloop/tbears:1.1.0.1 (/tbears)] - ...
EXPLORER
OPEN EDITORS 1 UNSAVED
dockertest.py U
DIVE-INTO-ICON-1-TOOLS [ATTACHED CONTAINER ...
.devcontainer
docs
samples
dockertest.py U
README.md
OUTLINE
H
json
os
CallBuilder
dockertest.py
1
2 import json
3 import os
4 from iconsdk.builder.call_builder import CallBuilder
5 from iconsdk.provider.http_provider import H
6
HTTPProvider
Attached Container iconloop/tbears:1.1.0.1 (/tbears)
master*
Python 3.6.7 64-bit
0 0
Ln 5, Col 46
Spaces: 4
UTF-8
CRLF
Python

# Dive into ICON - DApp

Step 1. What is a DApp?

1. Decentralized Application
2. Why should we build a DApp on ICON?

Step 2. Review

Step 3. How to build a DApp

Step 4. Hands-on Exercise

## What is a DApp?

# Decentralized Application

# Application

# DApp

## What is CryptoKitties?

CryptoKitties is a game centered around breedable, collectible, and oh-so-adorable creatures we call CryptoKitties! Each cat is one-of-a-kind and 100% owned by you; it cannot be replicated, taken away, or destroyed.

# Definition DApp


1. The application must be **completely open-source**, it must operate autonomously, and with no entity controlling the majority of its tokens. The application may adapt its protocol in response to proposed improvements and market feedback but all changes must be decided by consensus of its users.

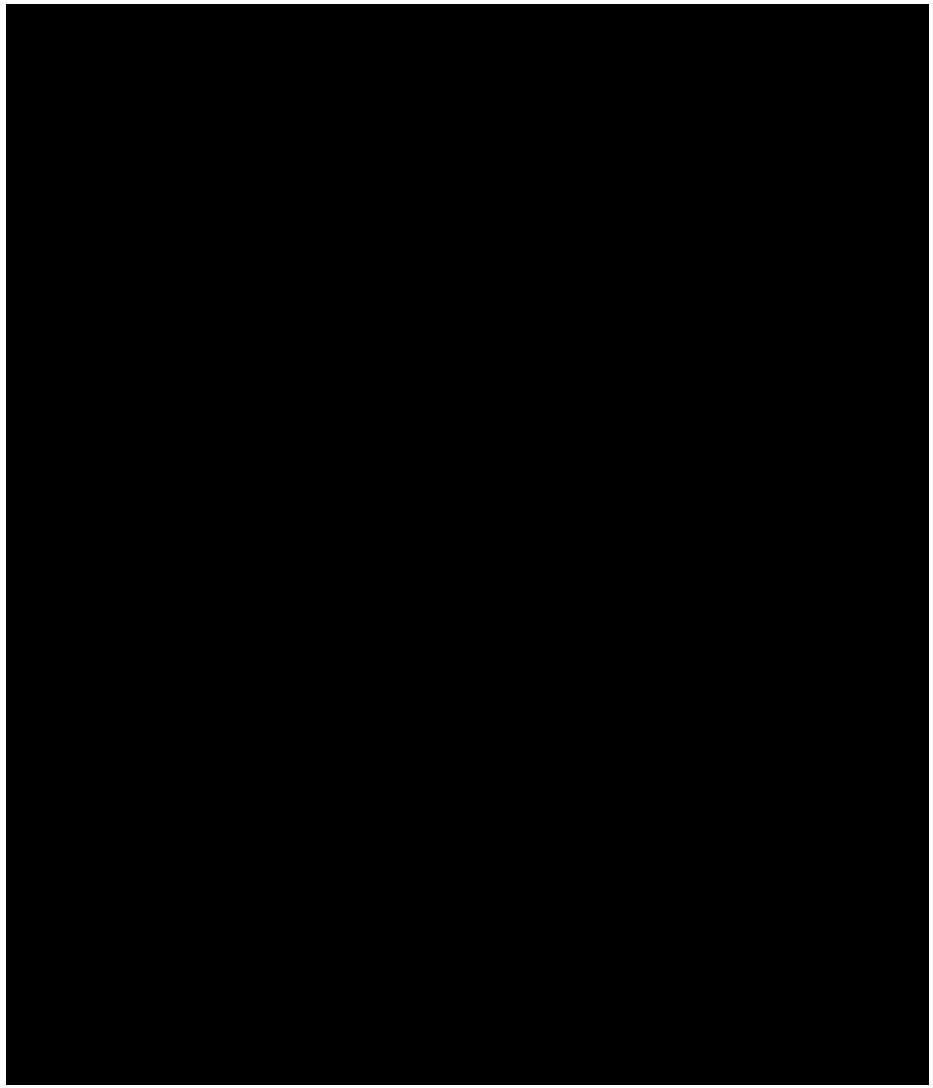
2. The application's data and records of operation must be **cryptographically stored in a public, decentralized blockchain** in order to avoid any central points of failure.

3. The application must use a **cryptographic token** (bitcoin or a token native to its system) which is necessary for access to the application and any contribution of value from (miners / farmers) should be rewarded in the application's tokens.

4. The application must generate tokens according to a standard **crytptographic algorithm** acting as a proof of the value nodes are contributing to the application (Bitcoin uses the Proof of Work Algorithm).

## DApp

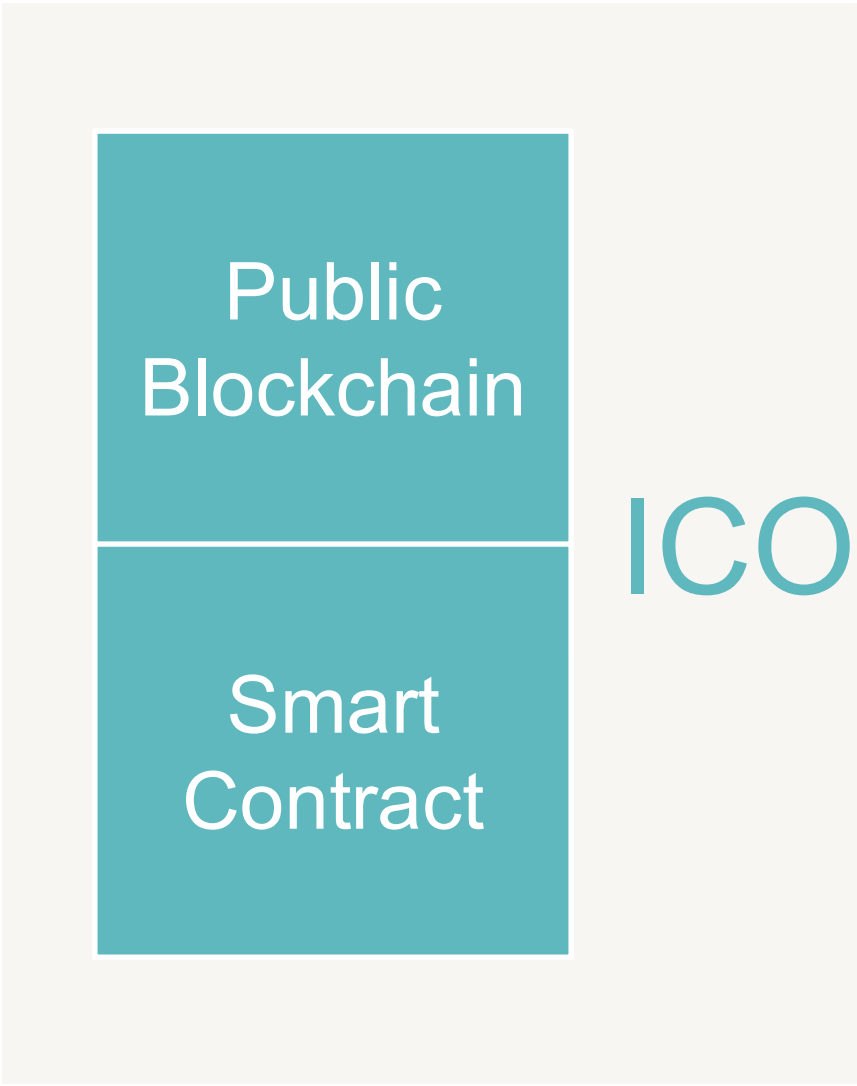
1. Contract-only
2. [Hybrid]
   Offchain Service +
   Onchain Service
3. [Save Data Only]
   Offchain Service +
   Block chain



## DApp

Contract-only

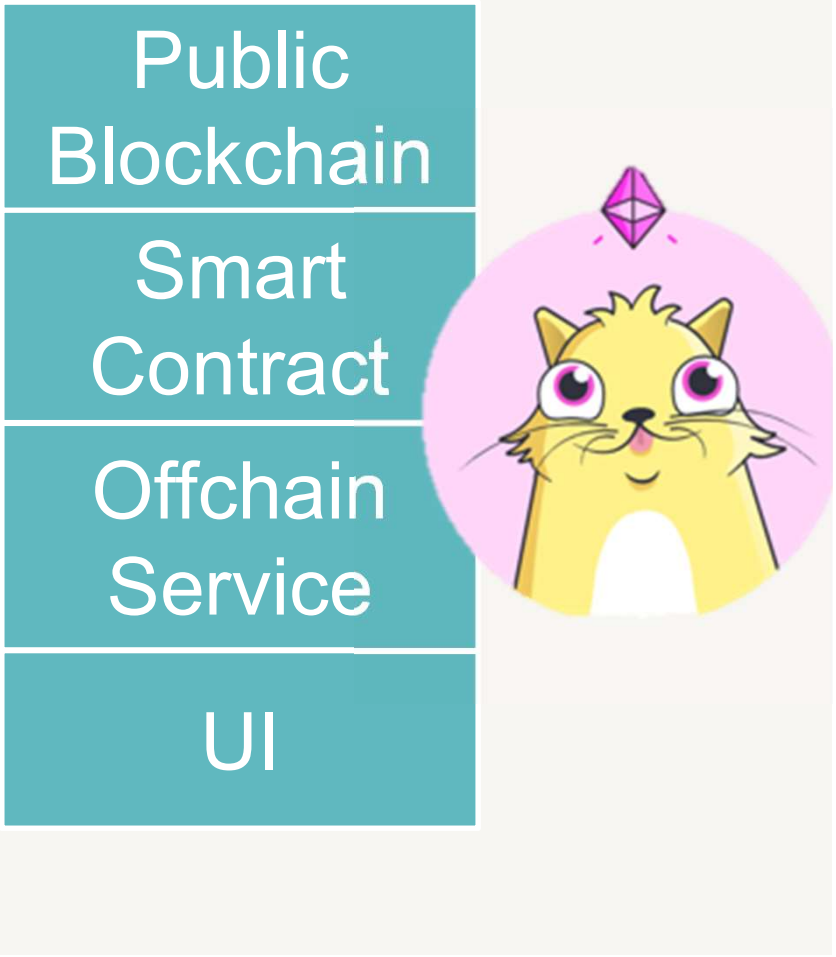
- Algorithm based Token
- Open Source
- Autonomous operation
- Save Data
  on Public Block Chain
- Smart Contract



## DApp

Service Logic(Hybrid)

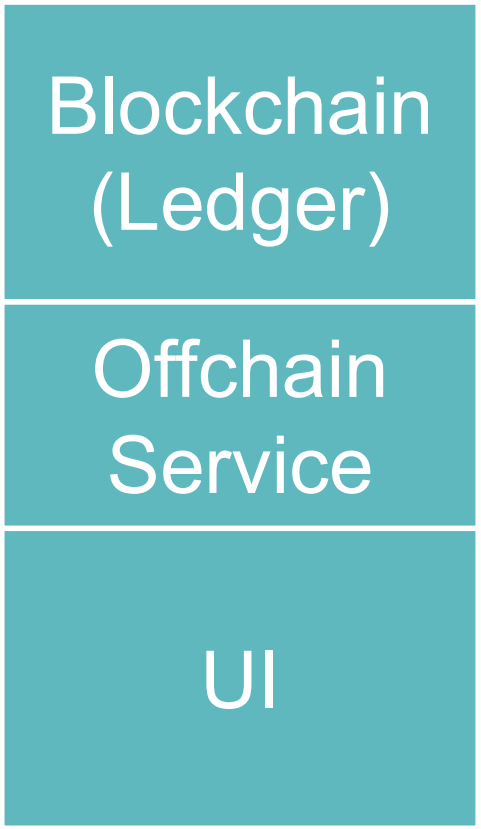
- Onchain Service + Offchain Service
- Algorithm based Token
- Open Source
- Autonomous operation
- Save Data on Public Block Chain



## DApp

Save Data Only

- Offchain Service + Block chain [to save data]
- Open Source
- Save Data on Block Chain



Unfortunately,
The Block chain
was **cursed**.

# The Curse of Blockchain: Transaction fee

하지만 한편으론 이더리움의 각종 불편함과 한계를
극명하게 보여줬습니다.

고양이를 살때도 수수료
고양이를 교배할때도 수수료
고양이를 판매할때도 수수료
고양이를 쓰다듬어도 수수료
고양이를 바라봐도 수수료
고양이를 고양이라고 불러도 수수료

이런 **수수료 지옥**은 처음 봤습니다.

뭔가 버튼 누르는 작업으로 이더리움과 연결되는 것은
하나도 빠짐없이 전부 수수료가 필요하며
그렇다고 신속하게 처리되는 것도 아니고
처리시 실패하면 **수수료가 전부 날아갑니다.**

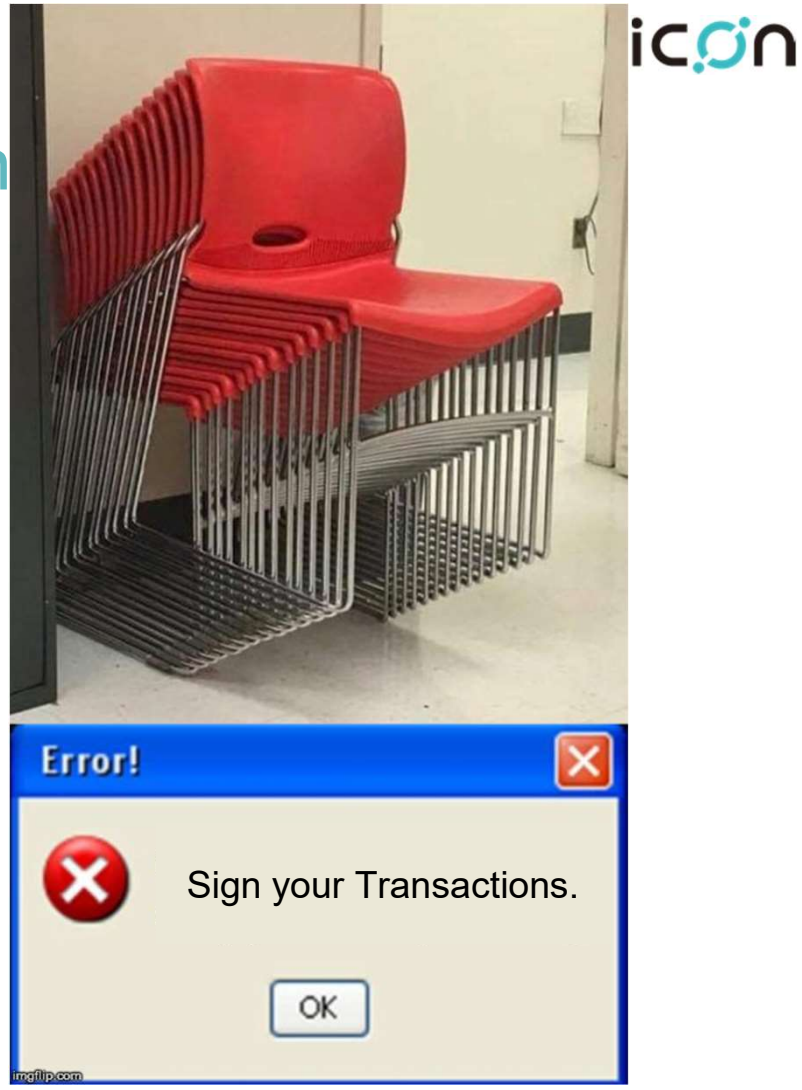
steemit / twinbraid

# The Curse of Blockchain : Transaction fee

# One More Thing!

Sign,
Sign,
Sign!

# The Curse of Blockchain : Sign



## 3 Ways to break the curses of Blockchain

1. DID
   Decentralized IDentifiers
2. Hybrid(OAuth … )
3. Use Blockchain to save data only

# How to Sign a Message

1. Traditional method (Wallet …)

2. Load User’s Keystore file

## 2 Ways to Sign

1. Traditional method (Wallet …)

2. Load User's Keystore file

Vulnerable

1.2

# Why should we build DApp on ICON?


B!ock.Chain
STUDY GROUP


WE WANT U
JOIN US
imgflip.com

Medium

# Your stories

Import a story | Write a story

Drafts 1 Published 1

## ICON에서 DApp을 만들어야 하는 이유

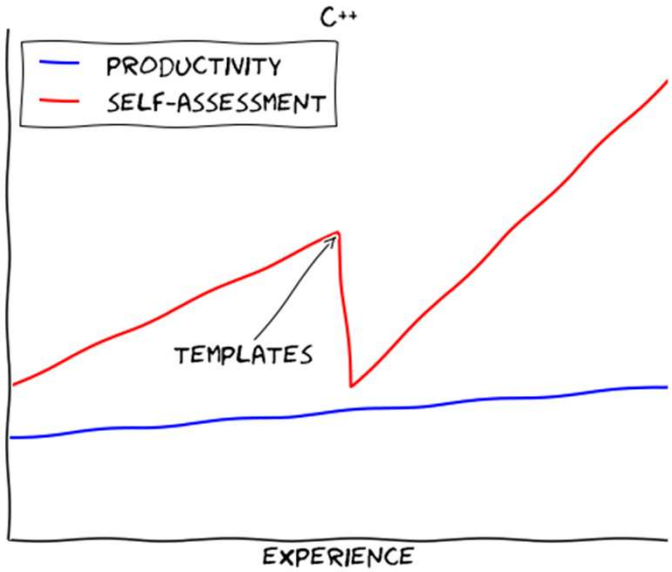
2년만에 연락해온 선배는 저의 이전글을 보았다며 곧장 물어왔습니다. 형, 봤으면 박수좀...

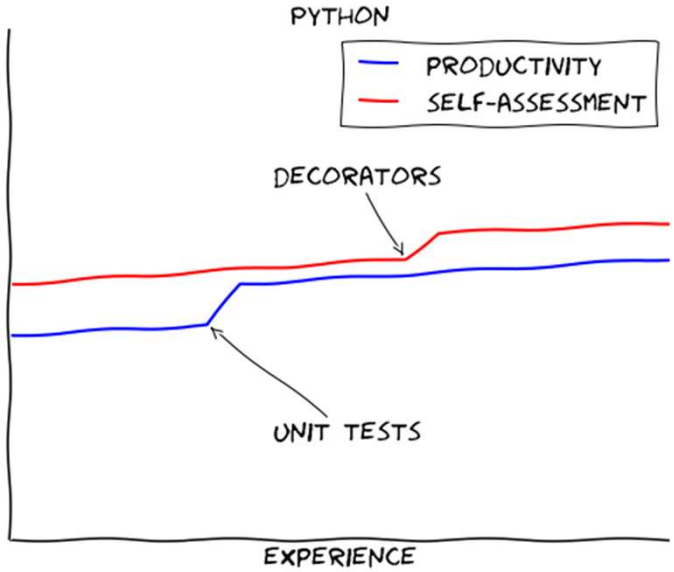
Last edited 3 days ago · 9 min read (1874 words) so far

# Why should we build a **DApp** on **ICON?**

1. Easy to learn
2. High TPS
3. Flexible Transaction Fee Policy
4. Interchain
5. Healthy network

# Easy to learn



## Easy to learn

# Workshop

# High TPS

20 TPS

6
Confirms

1000TPS

1
Confirm

## Low Transaction Fee

**0.00000001 ICX,**
**0.001USD**

0.00021 ETH,
0.03USD

글 작성 시점(2019. 03) 기준입니다.

# FEE 2.0

## Flexible Transaction Fee

Traditional Tx Fee

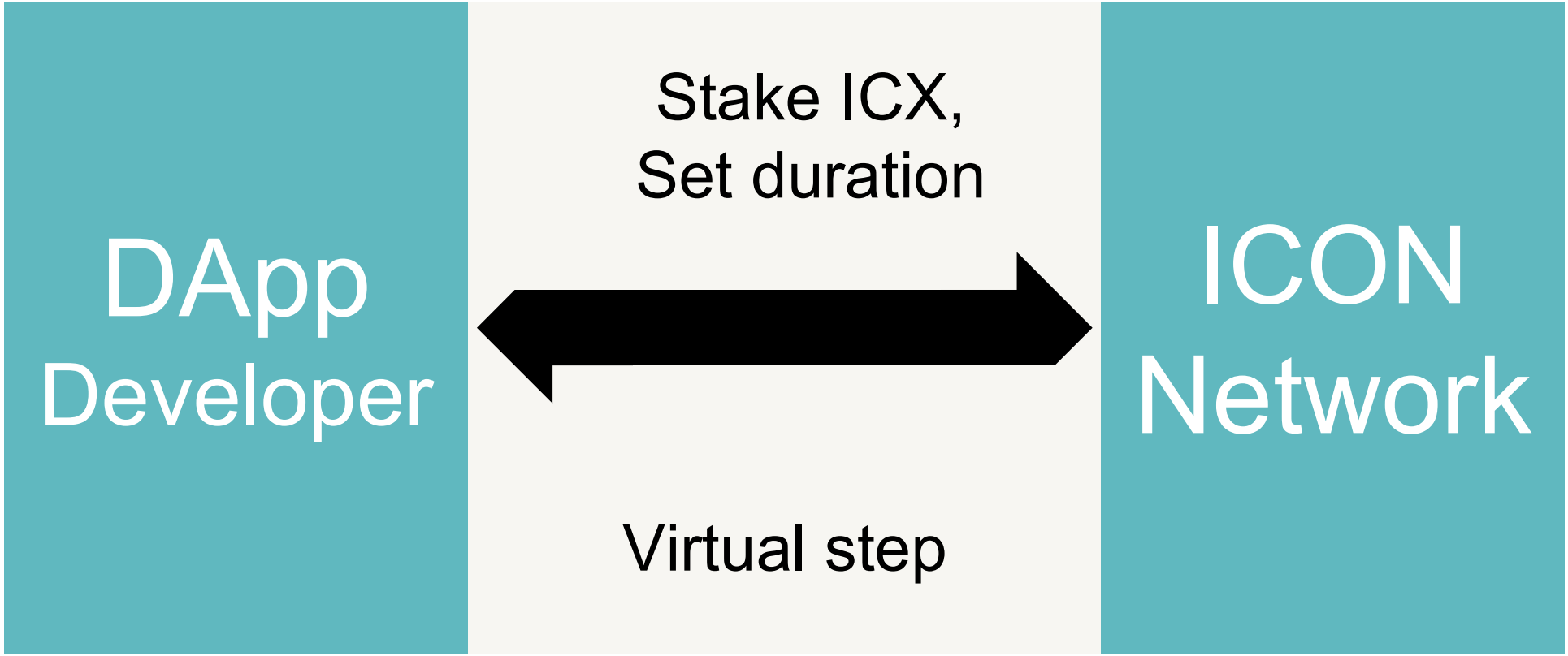
**User pays 100%**

Policy

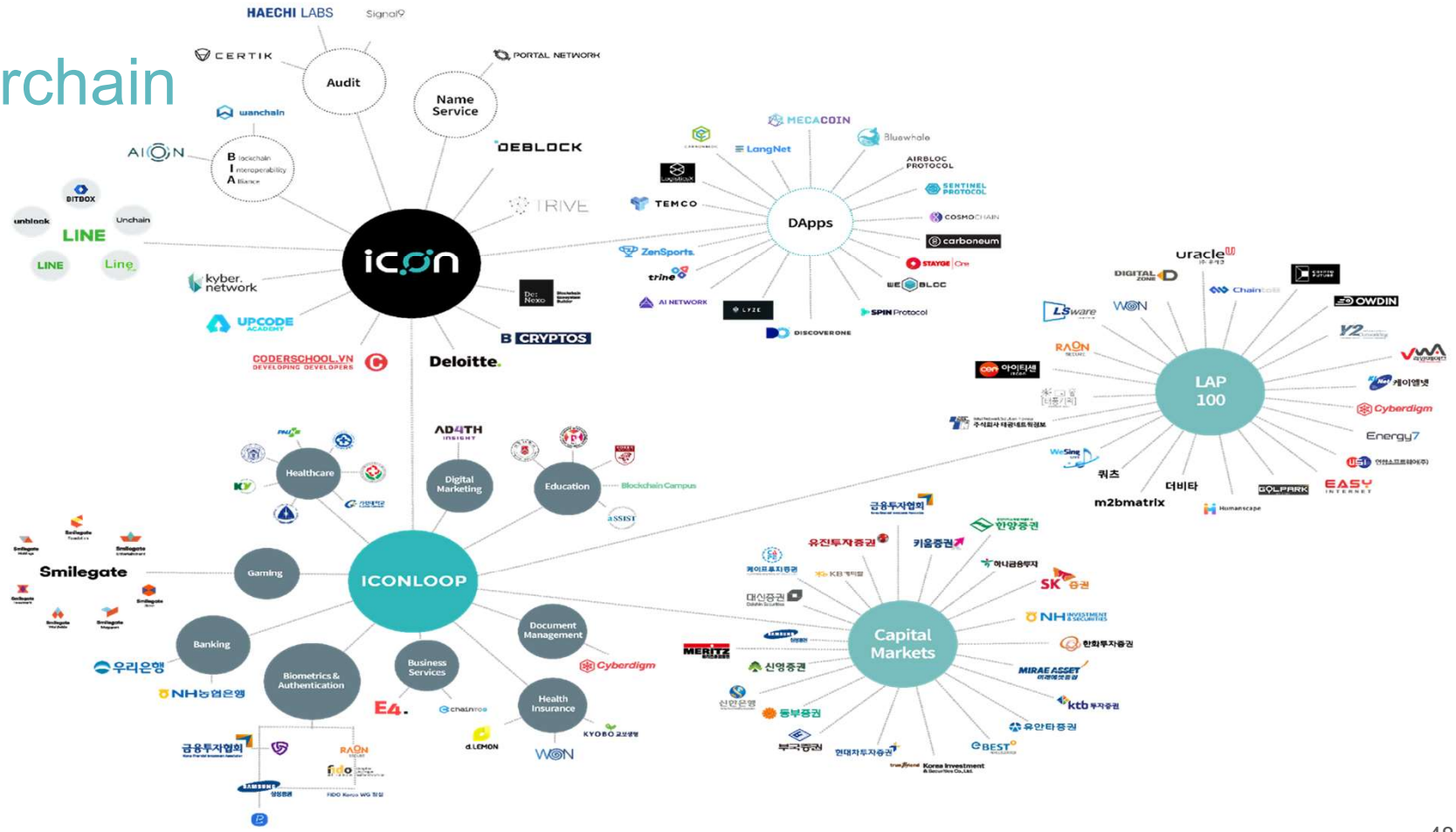
In ICON,
Developer can SET

**Fee sharing rate**

# Flexible Transaction Fee Policy – Virtual step



# Interchain


ICON
Audit
Name Service
Deloitte.
DApps
LAP 100
ICONLOOP
Healthcare
Digital Marketing
Education
Gaming
Banking
Biometrics & Authentication
Business Services
Document Management
Health Insurance
Smilegate
Capital Markets
m2bmatrix

Healthy Network

# DPoC
# Delegate Proof of Contribution

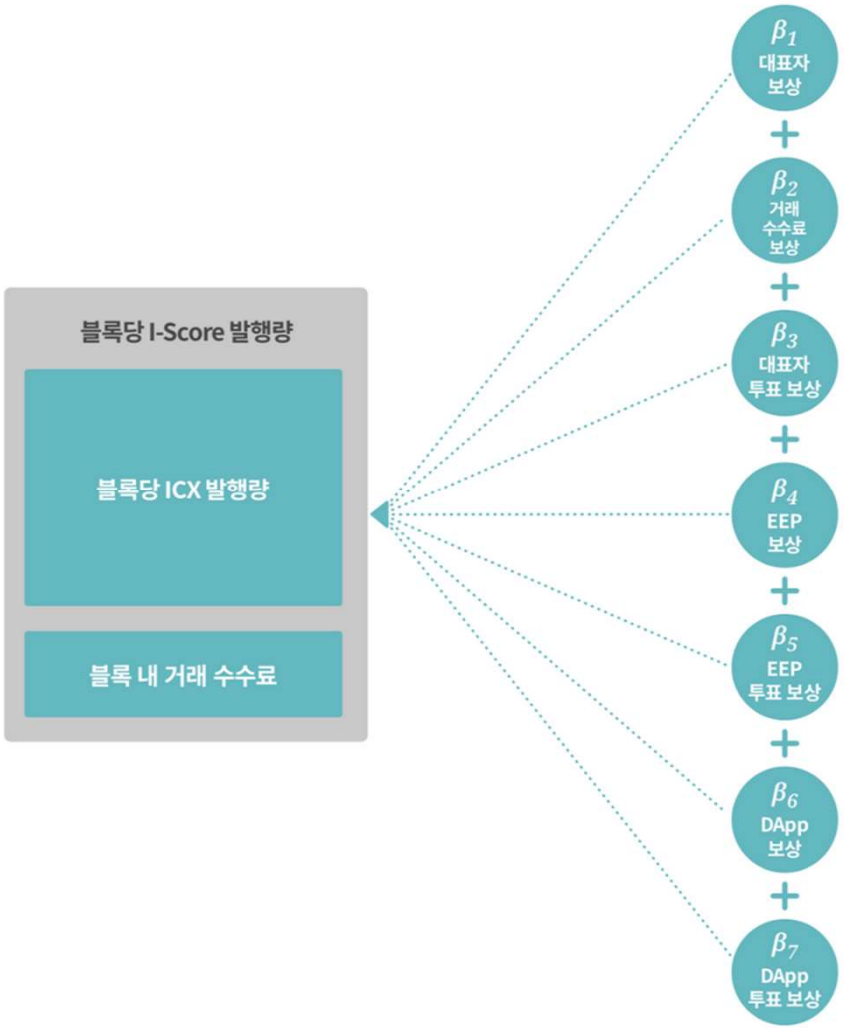
Healthy Network

# IISS

## ICON Incentive Scoreing System

## Healthy Network

# β6, β7
# DApp reward



# 2. Review

# Dive into ICON - DApp

Step 1. What is a DApp?

Step 2. Review

1. Dive into ICON - Tools
2. Dive into ICON - SCORE

Step 3. Make DApp

Step 4. Hands-on Exercise

Review

# Dive into ICON - Tools

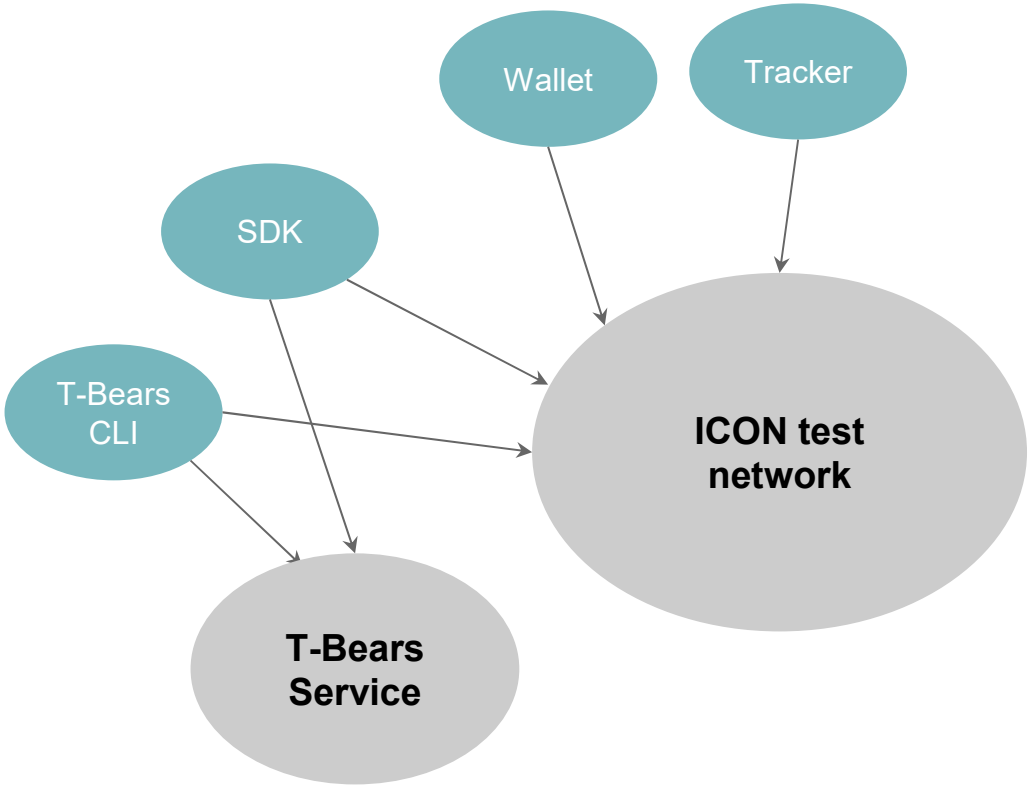
What we learned in “Dive into ICON – Tools”.

# T-Bears

# Python SDK

# ICON Tools

- T-Bears Dev Suite
  - SCORE library
  - Service (Node emulator)
  - Test framework
  - CLI
- Client SDK
  - Java, Python, JavaScript
- ICONex
- Tracker

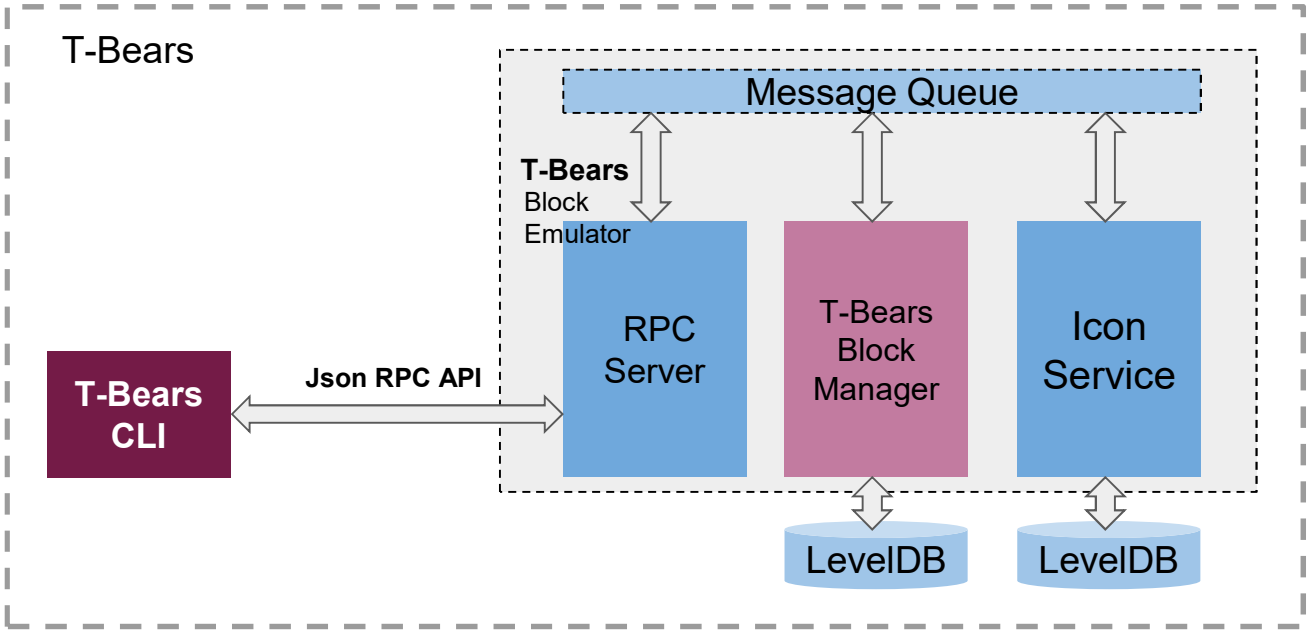


# Basic Commands Of T-Bears

| No | Command | Description |
|---|---|---|
| 1 | init | Initialize T-Bears project |
| 2 | deploy | Deploy SCORE |
| 3 | txresult | Get transaction result by hash |
| 4 | call | Request icx_call with user input json file. |
| 5 | sendtx | Request icx_sendTransaction with user input json file. |
| 6 | scoreapi | Get SCORE's API using given SCORE address. |

# How does T-Bears CLI work with RPC Server?

- T-Bears CLI interacts with RPC Server using JSON RPC API protocol



# ICON JSON-RPC API V3

- icx_sendTransaction
  - Transfer designated amount of ICX coins from 'from' address to 'to' address.
  - Install a new SCORE.
  - Update the SCORE in the 'to' address.
  - Invoke a function of the SCORE in the 'to' address.
  - Transfer a message.

# ICON JSON-RPC API V3

- JSON-RPC also provides various APIs like
  - icx_getLastBlock
  - icx_getBlockByHeight
  - icx_getBlockByHash
  - icx_getBalance
  - icx_getScoreApi
  - icx_getTotalSupply
  - icx_getTransactionResult
  - icx_getTransactionByHash

# SCORE Implementation Guide

- ICON Developers Portal
  - https://www.icondev.io/docs/overview

- iconservice API references
  - https://iconservice.readthedocs.io/en/latest/

Review

# Dive into ICON - SCORE

## What We learned in “Dive into ICON – SCORE”.

# SCORE

**Smart Contract On Reliable Environment**

## What We learned in “Dive into ICON – SCORE”.

# Audit

**Must check checklist!**

# Check Audit checklist

## 1. **Visit audit checklist**

- https://www.icondev.io/docs/audit-checklist#section-critical

## 2. **Check**
## Critical & Warnning List

### Critical

#### Timeout

SCORE function must return fairly immediately. Blockchain is not for any long-running operation.
For example, if you implement token airdrop to many users, do not iterate over all users in a single function. Handle each or partial airdrop(s) one by one instead.

```
# Bad
@external
def airdrop_token(self, _value: int, _data: bytes = None):
    for target in self._very_large_targets:
        self._transfer(self.msg.sender, target, _value, _data)

# Good
@external
def airdrop_token(self, _to: Address, _value: int, _data: bytes = None):
    if self._airdrop_sent_address[_to]:
        self.revert(f"Token was dropped already: {_to}")
    self._airdrop_sent_address[_to] = True
    self._transfer(self.msg.sender, _to, _value, _data)
```

#### Unfinishing loop

# Check Audit checklist

## Critical

Timeout
Unfinishing loop
Package import
System call
Randomness
Outbound network call
IRC2 Token Standard compliance
IRC2 Token parameter name
Eventlog on Token Transfer
Eventlog without Token Transfer
ICXTransfer Eventlog
Big Number Operation
Instance Variable
Super Class
StateDB write operation
Temporary Limitation

## Warnning

External Function Parameter Check
Internal Function Parameter Check
Predictable arbitrarity
Unchecked Low Level Calls
Underflow/Overflow
Vault
Reentrancy

# Check Audit checklist

## Critical

### Timeout

Unfinishing loop
Package import
System call
Randomness
Outbound network call
IRC2 Token Standard compliance
IRC2 Token parameter name
Eventlog on Token Transfer
Eventlog without Token Transfer
ICXTransfer Eventlog
Big Number Operation
Instance Variable
Super Class
StateDB write operation
Temporary Limitation

```python
# Bad
@external
def airdrop_token(self, _value: int, _data: bytes = None):
    for target in self._very_large_targets:
        self._transfer(self.msg.sender, target, _value, _data)

# Good
@external
def airdrop_token(self, _to: Address, _value: int, _data: bytes = None):
    if self._airdrop_sent_address[_to]:
        self.revert(f"Token was dropped already: {_to}")
    self._airdrop_sent_address[_to] = True
    self._transfer(self.msg.sender, _to, _value, _data)
```

# Check Audit checklist

## Critical

Timeout
Unfinishing loop
Package import
System call
Randomness
Outbound network call
IRC2 Token Standard compliance
IRC2 Token parameter name
**Eventlog on Token Transfer**
**Eventlog without Token Transfer**
ICXTransfer Eventlog
Big Number Operation
Instance Variable
Super Class
StateDB write operation
Temporary Limitation

### Eventlog on Token Transfer

Token transfer must trigger Eventlog.

```
# Good
@eventlog(indexed=3)
def Transfer(self, _from: Address, _to: Address, _value: int, _data: bytes):
    pass

@external
def transfer(self, _to: Address, _value: int, _data: bytes = None):
    self._balances[self.msg.sender] -= _value
    self._balances[_to] += _value
    self.Transfer(self.msg.sender, _to, _value, _data)
```

### Eventlog without Token Transfer

Do not trigger Transfer Eventlog without token transfer.

```
# Bad
@eventlog(indexed=3)
def Transfer(self, _from: Address, _to: Address, _value: int, _data: bytes):
    pass

@external
def doSomething(self, _to: Address, _value: int):
    #  no token transfer occurred
    self.Transfer(self.msg.sender, _to, _value, None)
```

# Check Audit checklist

## Critical

Timeout
Unfinishing loop
Package import
System call
Randomness
Outbound network call
IRC2 Token Standard compliance
IRC2 Token parameter name
Eventlog on Token Transfer
Eventlog without Token Transfer
ICXTransfer Eventlog
Big Number Operation
Instance Variable

## Super Class

StateDB write operation
Temporary Limitation

```python
# Bad
class MyClass(IconScoreBase):
    def __init__(self, db: IconScoreDatabase) -> None:
        self._context__name = VarDB('context.name', db, str)
        self._context__cap = VarDB('context.cap', db, int)

    def on_install(self, name: str, cap: str) -> None:
        # doSomething

    def on_update(self) -> None:
        # doSomething

# Good
class MyClass(IconScoreBase):
    def __init__(self, db: IconScoreDatabase) -> None:
        super().__init__(db)
        self._context__name = VarDB('context.name', db, str)
        self._context__cap = VarDB('context.cap', db, int)

    def on_install(self, name: str, cap: str) -> None:
        super().on_install()
        # doSomething

    def on_update(self) -> None:
        super().on_update()
        # doSomething
```

# Check Audit checklist

## Critical

Timeout
Unfinishing loop
Package import
System call
Randomness
Outbound network call
IRC2 Token Standard compliance
IRC2 Token parameter name
Eventlog on Token Transfer
Eventlog without Token Transfer
ICXTransfer Eventlog
Big Number Operation
Instance Variable
Super Class
StateDB write operation

## Temporary Limitation

### Temporary Limitation

Due to the known issue of ArrayDB, declaring ArrayDB as a class member variable in __init__() may not work as intended. Following workaround is needed. ArrayDB instance must be initialized every time it is used.

```
# Problematic (Original Usage)
def __init__(self, db: IconScoreDatabase) -> None:
    super().__init__(db)
    self.test_array = ArrayDB('test_array', db, value_type=int)

def func(self) -> None:
    self.test_array.put(0)


# Good (Temporary)
@property
def test_array(self) -> ArrayDB:
    return ArrayDB('test_array', db, value_type=int)

def __init__(self, db: IconScoreDatabase) -> None:
    super().__init__(db)
    # no declaration

def func(self) -> None:
    self.test_array.put(0)
```

# Check Audit checklist

## Critical

Timeout
Unfinishing loop
Package import
System call
Randomness
Outbound network call
IRC2 Token Standard compliance
IRC2 Token parameter name
Eventlog on Token Transfer
Eventlog without Token Transfer
ICXTransfer Eventlog
Big Number Operation
Instance Variable
Super Class
StateDB write operation

## **Temporary Limitation**

### Temporary Limitation

Due to the known issue of ArrayDB, declaring ArrayDB as a class member variable in __init__() may not work as intended. Following workaround is needed. ArrayDB instance must be initialized every time it is used.

```
# Problematic (Original Usage)
def __init__(self, db: IconScoreDatabase) -> None:
    super().__init__(db)
    self.test_array = ArrayDB('test_array', db, value_type=int)

def func(self) -> None:
    self.test_array.put(0)
```

Updated

# What We learned in “Dive into ICON – SCORE”.

## IIPs

**ICON I**mprovement **P**roposals

# Check IIPs

## 1. **Visit IIPs**

- https://github.com/icon-project/IIPs

## 2. **Check**

**Specification of proposal**

# 3. How to build a DApp ?

# Dive into ICON - DApp

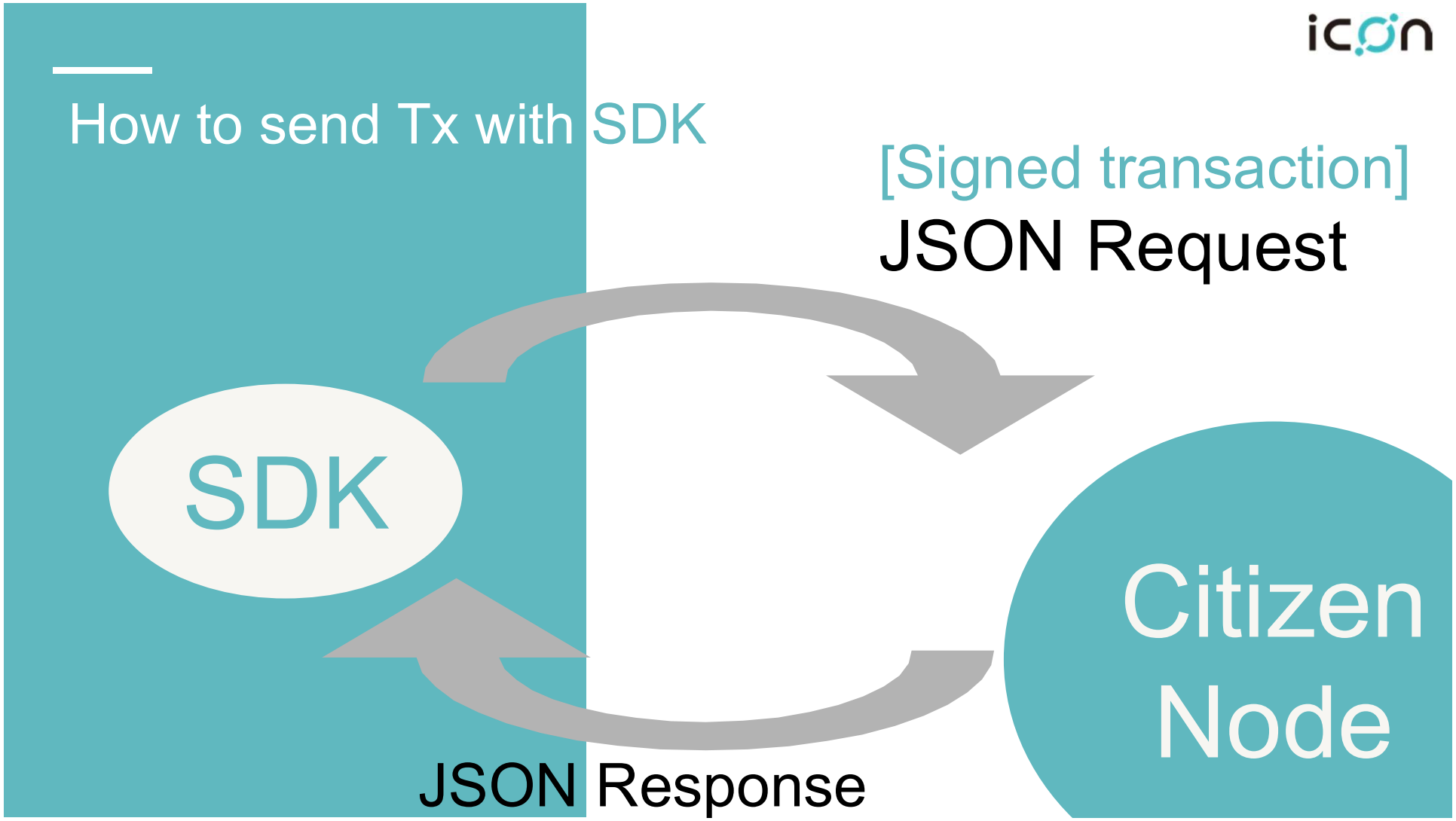
Step 1. What is a DApp?

Step 2. Review

Step 3. How to build a DApp

1. How to send Tx with SDK
2. How to use ICONex connect

Step 4. Hands-on Exercise


How to send Tx with SDK
[Signed transaction]
JSON Request
SDK
Citizen Node
JSON Response

# How to send Tx with SDK

SDK code

Connect to URI

Load / create Wallet

**Build Transaction**

Sign Transaction

Send Transaction

[option] Transaction Result print

# How to send Tx with SDK

## Python SDK

```
icon_service = IconService(HTTPProvider('https://bicon.net.solidwallet.io/api/v3'))

wallet = KeyWallet.create()

transaction = CallTransactionBuilder()\
    .from_(wallet.get_address())\
    .to("cxbff5fa7adc97f515070f2490d5a47aa927859549") \
    .nid(3) \
    .step_limit(1000000)\
    .value(10000000)\
    .version(3)\
    .method("scrooge")\
    .params({
        "_to": "hx9505040fc8883f9d4b287d1dbcd49bb2cd80748a",
        "_ratio": 2
    })\
    .build()

signed_transaction = SignedTransaction(transaction, wallet)

tx_hash = icon_service.send_transaction(signed_transaction)
```

## JavaScript SDK

```
var IconService = window['icon-sdk-js']
var provider = new IconService.HttpProvider('https://bicon.net.solidwallet.io/api/v3')
var iconService = new IconService(provider)
var IconConverter = IconService.IconConverter
var IconBuilder = IconService.IconBuilder
var IconAmount = IconService.IconAmount
```

```
requestScore.onclick = function() {
  var callTransactionBuilder = new IconBuilder.CallTransactionBuilder;
  var callTransactionData = callTransactionBuilder
            .from(fromAddress)
            .to("cxbff5fa7adc97f515070f2490d5a47aa927859549")
            .nid(IconConverter.toBigNumber(3))
            .timestamp((new Date()).getTime() * 1000)
            .stepLimit(IconConverter.toBigNumber(10000000))
            .value(IconAmount.of(amount_loop.value, IconAmount.Unit.ICX).toLoop())
            .version(IconConverter.toBigNumber(3))
            .method('scrooge')
            .params({
                "_to": "hx9505040fc8883f9d4b287d1dbcd49bb2cd80748a",
                "_ratio": IconConverter.toHex(2)
            })
            .build()

        scoreData.value = JSON.stringify({
            "jsonrpc": "2.0",
            "method": "icx_sendTransaction",
            "params": IconConverter.toRawTransaction(callTransactionData),
            "id": 8015
        })
```

# How to send Tx with SDK

## Python SDK

```python
icon_service = IconService(HTTPProvider('https://bicon.net.solidwallet.io/api/v3'))

wallet = KeyWallet.create()

transaction = CallTransactionBuilder()\
    .from_(wallet.get_address())\
    .to("cxbff5fa7adc97f515070f2490d5a47aa927859549") \
    .nid(3) \
    .step_limit(1000000)\
    .value(10000000)\
    .version(3)\
    .method("scrooge")\
    .params({
        "_to": "hx9505040fc8883f9d4b287d1dbcd49bb2cd80748a",
        "_ratio": 2
    })\
    .build()

signed_transaction = SignedTransaction(transaction, wallet)

tx_hash = icon_service.send_transaction(signed_transaction)
```

## JavaScript SDK

```javascript
var IconService = window['icon-sdk-js']
var provider = new IconService.HttpProvider('https://bicon.net.solidwallet.io/api/v3')
var iconService = new IconService(provider)
var IconConverter = IconService.IconConverter
var IconBuilder = IconService.IconBuilder
var IconAmount = IconService.IconAmount
```

```javascript
requestScore.onclick = function() {
  var callTransactionBuilder = new IconBuilder.CallTransactionBuilder;
  var callTransactionData = callTransactionBuilder
            .from(fromAddress)
            .to("cxbff5fa7adc97f515070f2490d5a47aa927859549")
            .nid(IconConverter.toBigNumber(3))
            .timestamp((new Date()).getTime() * 1000)
            .stepLimit(IconConverter.toBigNumber(10000000))
            .value(IconAmount.of(amount_loop.value, IconAmount.Unit.ICX).toLoop())
            .version(IconConverter.toBigNumber(3))
            .method('scrooge')
            .params({
                "_to": "hx9505040fc8883f9d4b287d1dbcd49bb2cd80748a",
                "_ratio": IconConverter.toHex(2)
            })
            .build()

        scoreData.value = JSON.stringify({
            "jsonrpc": "2.0",
            "method": "icx_sendTransaction",
            "params": IconConverter.toRawTransaction(callTransactionData),
            "id": 8015
        })
```

# How to send Tx with SDK

## Python SDK

```
icon_service = IconService(HTTPProvider('https://bicon.net.solidwallet.io/api/v3'))

wallet = KeyWallet.create()

transaction = CallTransactionBuilder()\
    .from_(wallet.get_address())\
    .to("cxbff5fa7adc97f515070f2490d5a47aa927859549") \
    .nid(3) \
    .step_limit(1000000)\
    .value(10000000)\
    .version(3)\
    .method("scrooge")\
    .params({
        "_to": "hx9505040fc8883f9d4b287d1dbcd49bb2cd80748a",
        "_ratio": 2
    })\
    .build()

signed_transaction = SignedTransaction(transaction, wallet)

tx_hash = icon_service.send_transaction(signed_transaction)
```

## JavaScript SDK

```
var IconService = window['icon-sdk-js']
var provider = new IconService.HttpProvider('https://bicon.net.solidwallet.io/api/v3')
var iconService = new IconService(provider)
var IconConverter = IconService.IconConverter
var IconBuilder = IconService.IconBuilder
var IconAmount = IconService.IconAmount
```

```
requestScore.onclick = function() {
    var callTransactionBuilder = new IconBuilder.CallTransactionBuilder;
    var callTransactionData = callTransactionBuilder
            .from(fromAddress)
            .to("cxbff5fa7adc97f515070f2490d5a47aa927859549")
            .nid(IconConverter.toBigNumber(3))
            .timestamp((new Date()).getTime() * 1000)
            .stepLimit(IconConverter.toBigNumber(10000000))
            .value(IconAmount.of(amount_loop.value, IconAmount.Unit.ICX).toLoop())
            .version(IconConverter.toBigNumber(3))
            .method('scrooge')
            .params({
                "_to": "hx9505040fc8883f9d4b287d1dbcd49bb2cd80748a",
                "_ratio": IconConverter.toHex(2)
            })
            .build()

        scoreData.value = JSON.stringify({
            "jsonrpc": "2.0",
            "method": "icx_sendTransaction",
            "params": IconConverter.toRawTransaction(callTransactionData),
            "id": 8015
        })
```

# How to send Tx with SDK

## Python SDK

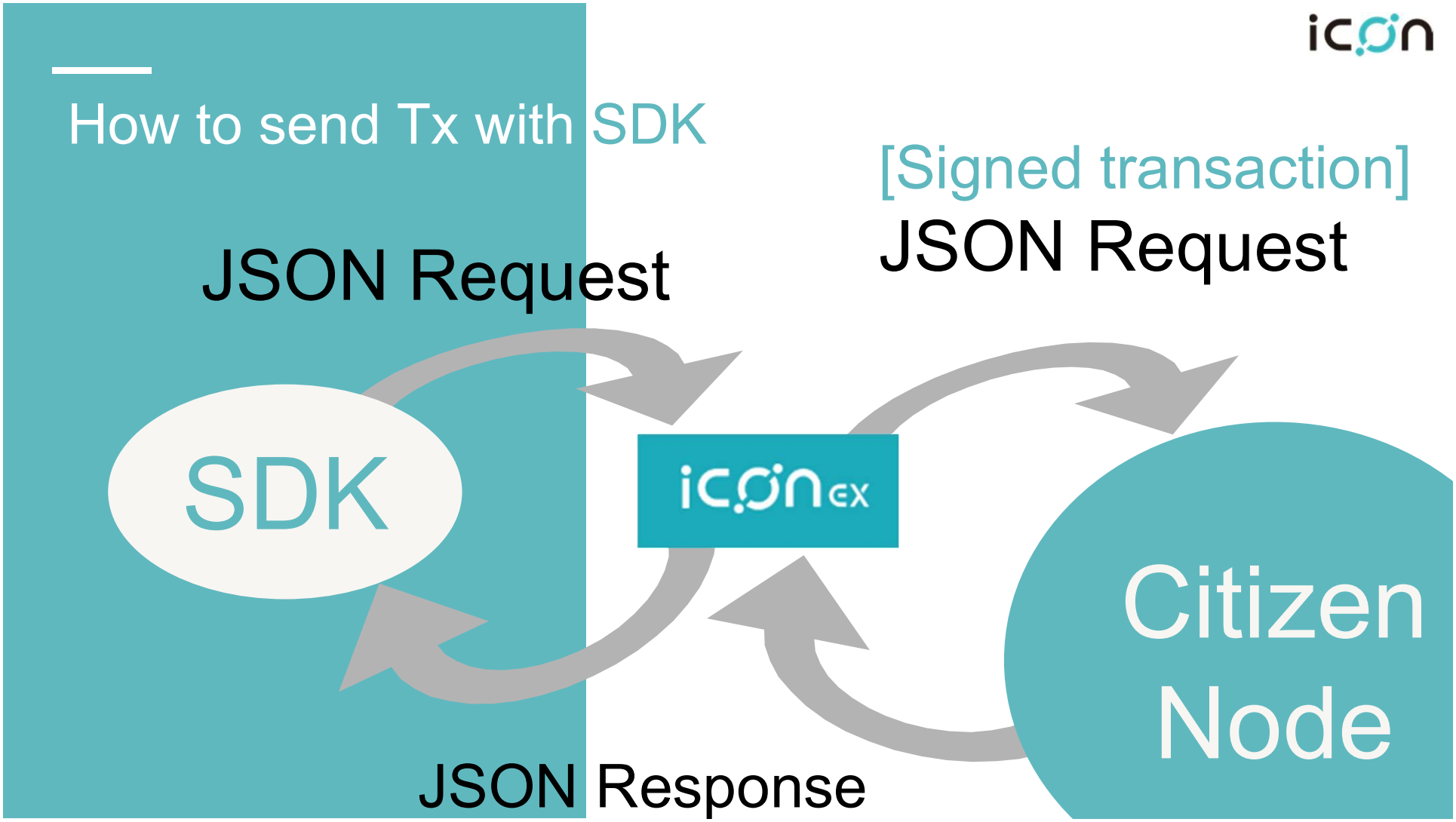
```
icon_service = IconService(HTTPProvider('https://bicon.net.solidwallet.io/api/v3'))

wallet = KeyWallet.create()

transaction = CallTransactionBuilder()\
    .from_(wallet.get_address())\
    .to("cxbff5fa7adc97f515070f2490d5a47aa927859549") \
    .nid(3) \
    .step_limit(1000000)\
    .value(10000000)\
    .version(3)\
    .method("scrooge")\
    .params({
        "_to": "hx9505040fc8883f9d4b287d1dbcd49bb2cd80748a",
        "_ratio": 2
    })\
    .build()

signed_transaction = SignedTransaction(transaction, wallet)

tx_hash = icon_service.send_transaction(signed_transaction)
```

## JavaScript SDK

```
var IconService = window['icon-sdk-js']
var provider = new IconService.HttpProvider('https://bicon.net.solidwallet.io/api/v3')
var iconService = new IconService(provider)
var IconConverter = IconService.IconConverter
var IconBuilder = IconService.IconBuilder
var IconAmount = IconService.IconAmount
```

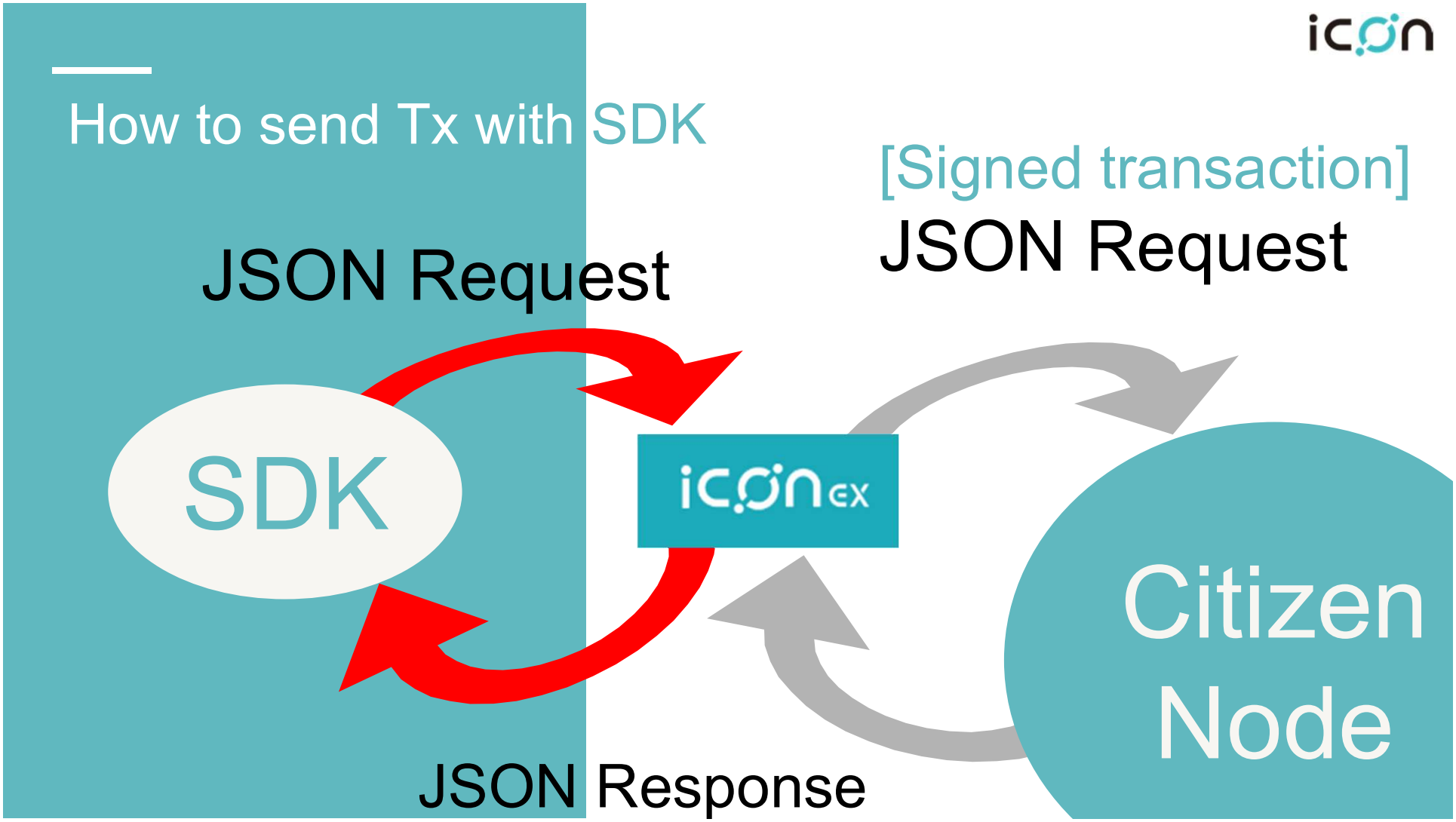
```
requestScore.onclick = function() {
  var callTransactionBuilder = new IconBuilder.CallTransactionBuilder;
  var callTransactionData = callTransactionBuilder
            .from(fromAddress)
            .to("cxbff5fa7adc97f5[illegible]aa927859549")
            .nid(IconConverter.to[illegible]Number(3))
            .timestamp((new Date()).getTime()[illegible]1000)
            .stepLimit(IconConverter.toBig[illegible]r(10000000))
            .value(IconAmount.of(amount_[illegible].value, IconAmount.Unit.ICX).toLoop())
            .version(IconConverter.toBi[illegible]mber(3))
            .method('scrooge')
            .params({
                "_to": "hx9505040fc888[illegible]4b287d1dbcd49bb2cd80748a",
                "_ratio": IconConverter.toHex(2)
            })
            .build()

        scoreData.value = JSON.stringify({
            "jsonrpc": "2.0",
            "method": "icx_sendTransaction",
            "params": IconConverter.toRawTransaction(callTransactionData),
            "id": 8015
        })
```

# How to send Tx with SDK




icon ex

# How to send Tx with SDK



# How to send Tx with SDK

**REQUEST_HAS_ACCOUNT**
**REQUEST_HAS_ADDRESS**
**REQUEST_ADDRESS**
**REQUEST_JSON-RPC**
**REQUEST_SIGNING**

SDK


**RESPONSE_HAS_ACCOUNT**
**RESPONSE_HAS_ADDRESS**
**RESPONSE_ADDRESS**
**RESPONSE_JSON-RPC**
**RESPONSE_SIGNING**

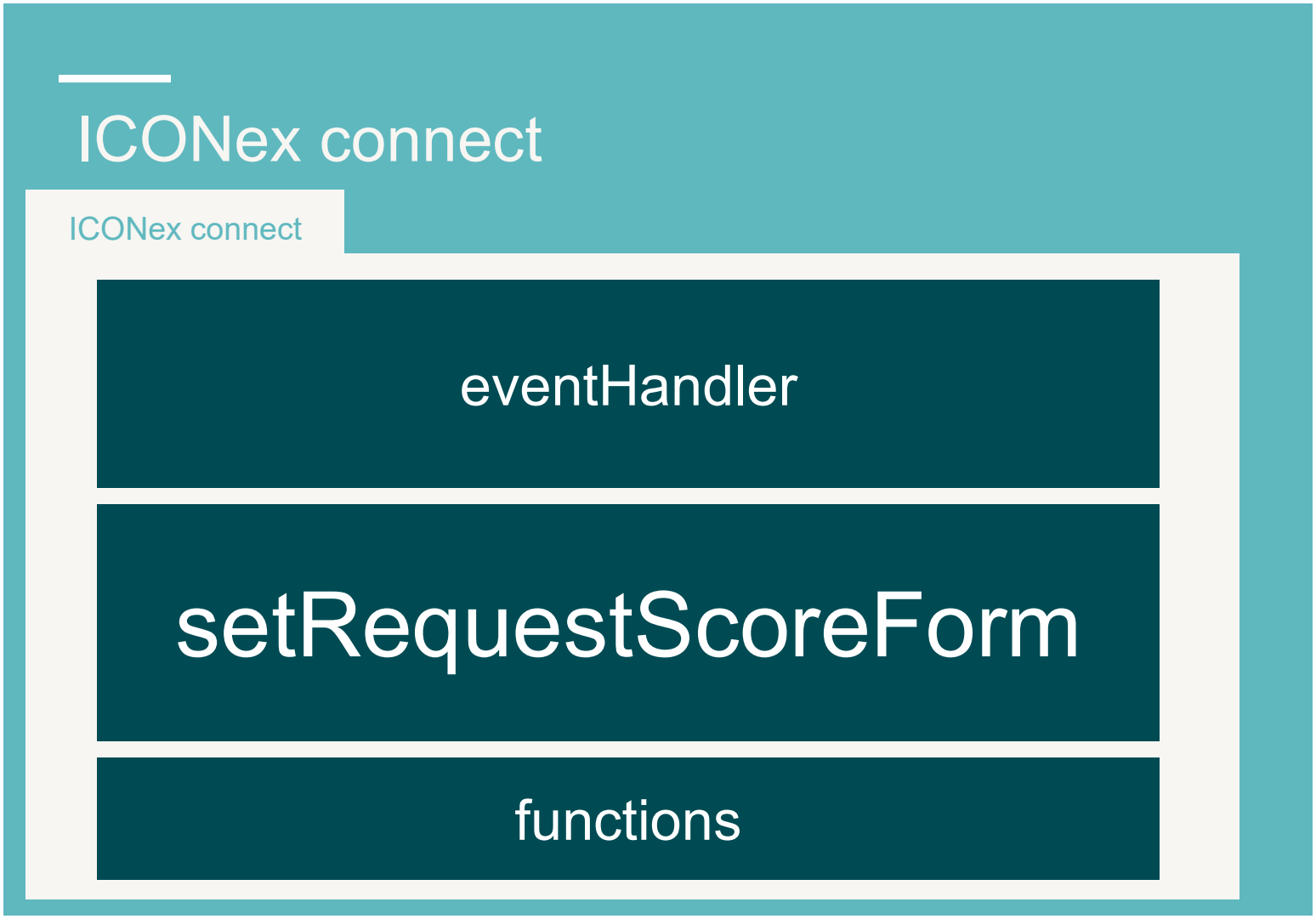
# How to send Tx with SDK

**REQUEST_HAS_ACCOUNT**
**REQUEST_HAS_ADDRESS**
**REQUEST_ADDRESS**
**REQUEST_JSON-RPC**
**REQUEST_SIGNING**

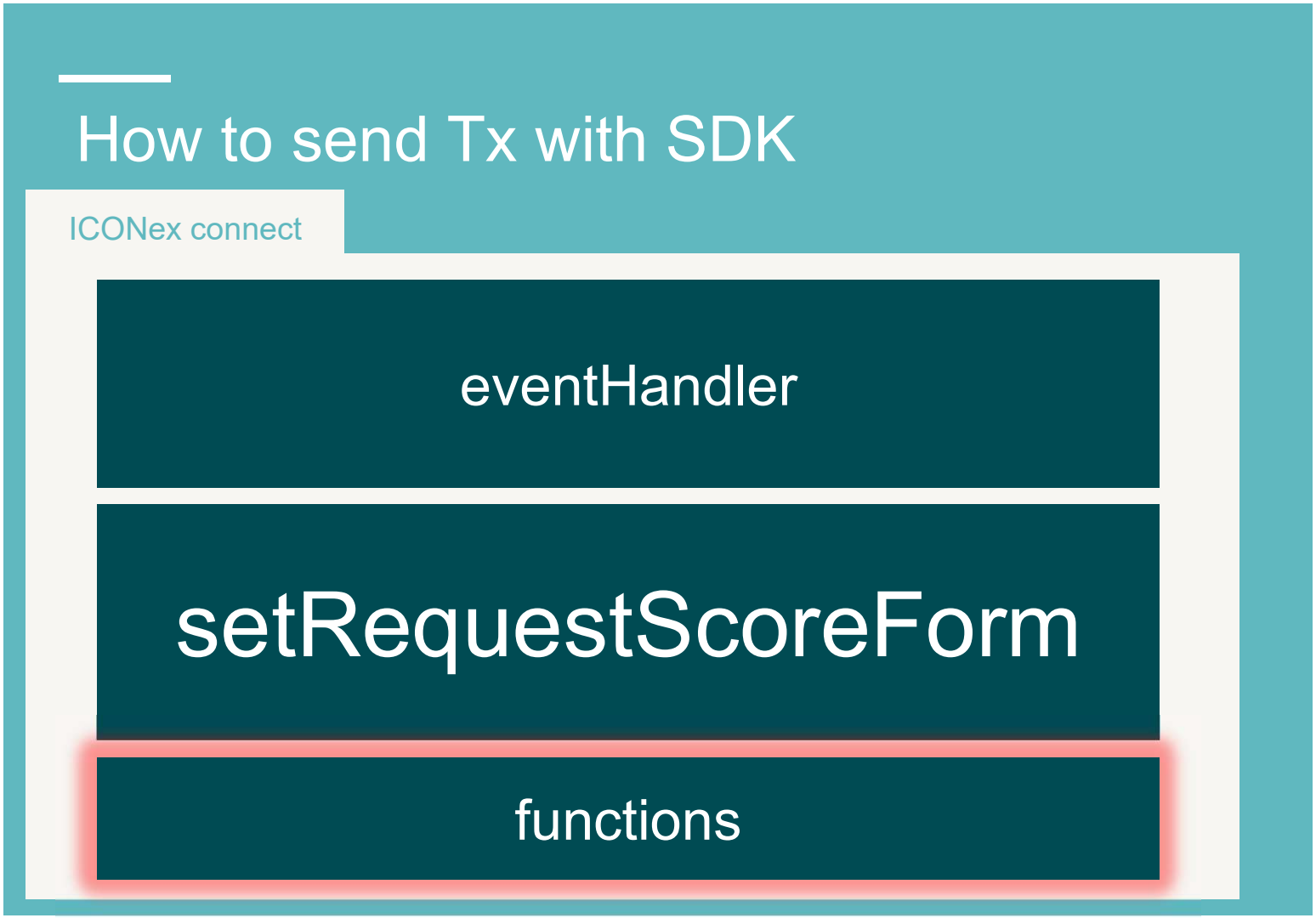
SDK


**RESPONSE_HAS_ACCOUNT**
**RESPONSE_HAS_ADDRESS**
**RESPONSE_ADDRESS**
**RESPONSE_JSON-RPC**
**RESPONSE_SIGNING**

# ICONex connect



# How to send Tx with SDK



# How to send Tx with SDK

## Functions(request)

```
requestHasAccount.onclick = function () {
    window.dispatchEvent(new CustomEvent('ICONEX_RELAY_REQUEST', {
        detail: {
            type: "REQUEST_HAS_ACCOUNT"
        }
    }))
}

requestHasAddress.onclick = function () {
    window.dispatchEvent(new CustomEvent('ICONEX_RELAY_REQUEST', {
        detail: {
            type: 'REQUEST_HAS_ADDRESS',
            payload: requestHasAddressData.value || requestHasAddressData.placeholder
        }
    }))
}

requestAddress.onclick = function () {
    window.dispatchEvent(new CustomEvent('ICONEX_RELAY_REQUEST', {
        detail: {
            type: 'REQUEST_ADDRESS'
        }
    }))
}
```

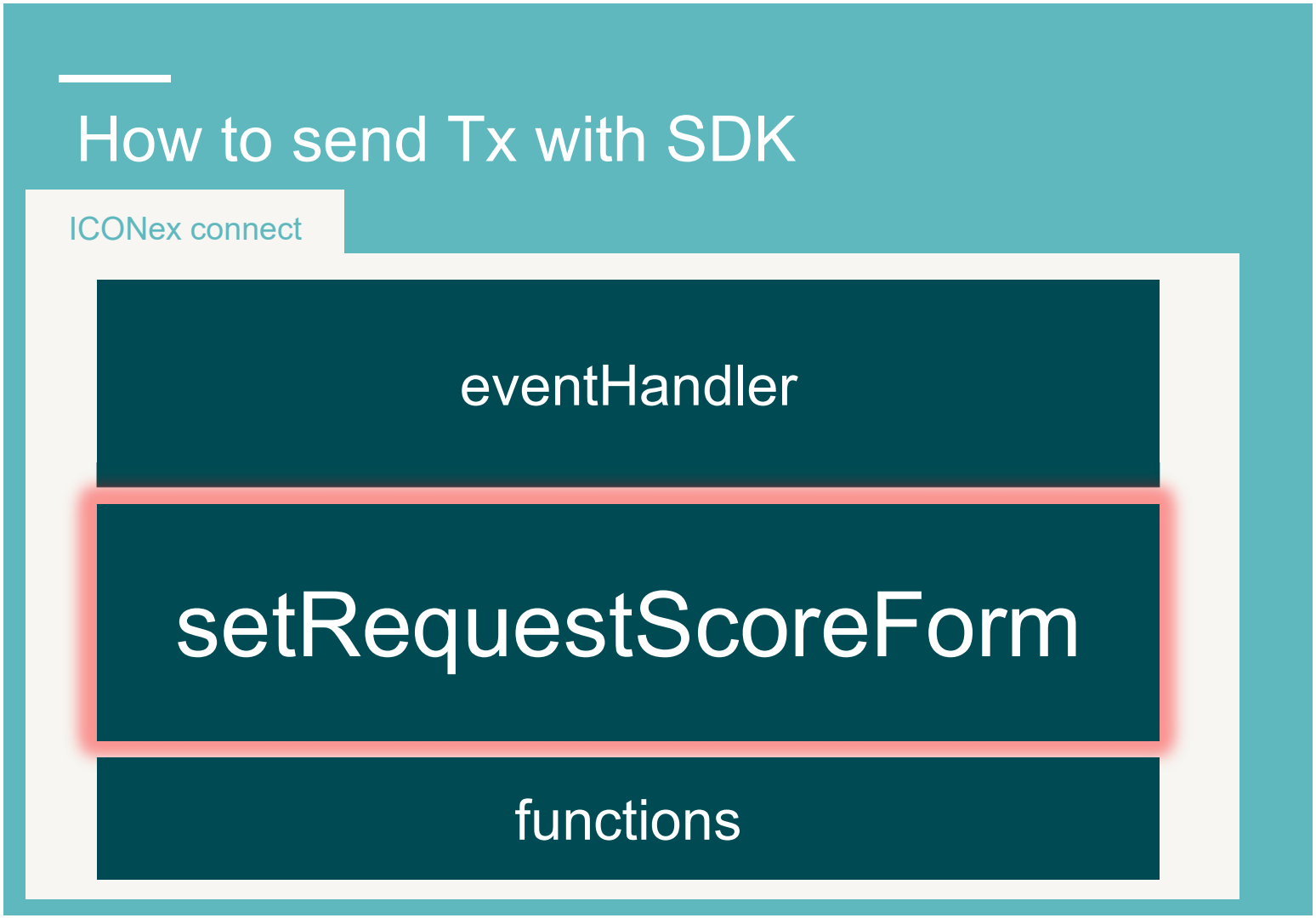
# How to send Tx with SDK

**REQUEST_HAS_ACCOUNT**
**REQUEST_HAS_ADDRESS**
**REQUEST_ADDRESS**
**REQUEST_JSON-RPC**
**REQUEST_SIGNING**

SDK


**RESPONSE_HAS_ACCOUNT**
**RESPONSE_HAS_ADDRESS**
**RESPONSE_ADDRESS**
**RESPONSE_JSON-RPC**
**RESPONSE_SIGNING**

# How to send Tx with SDK

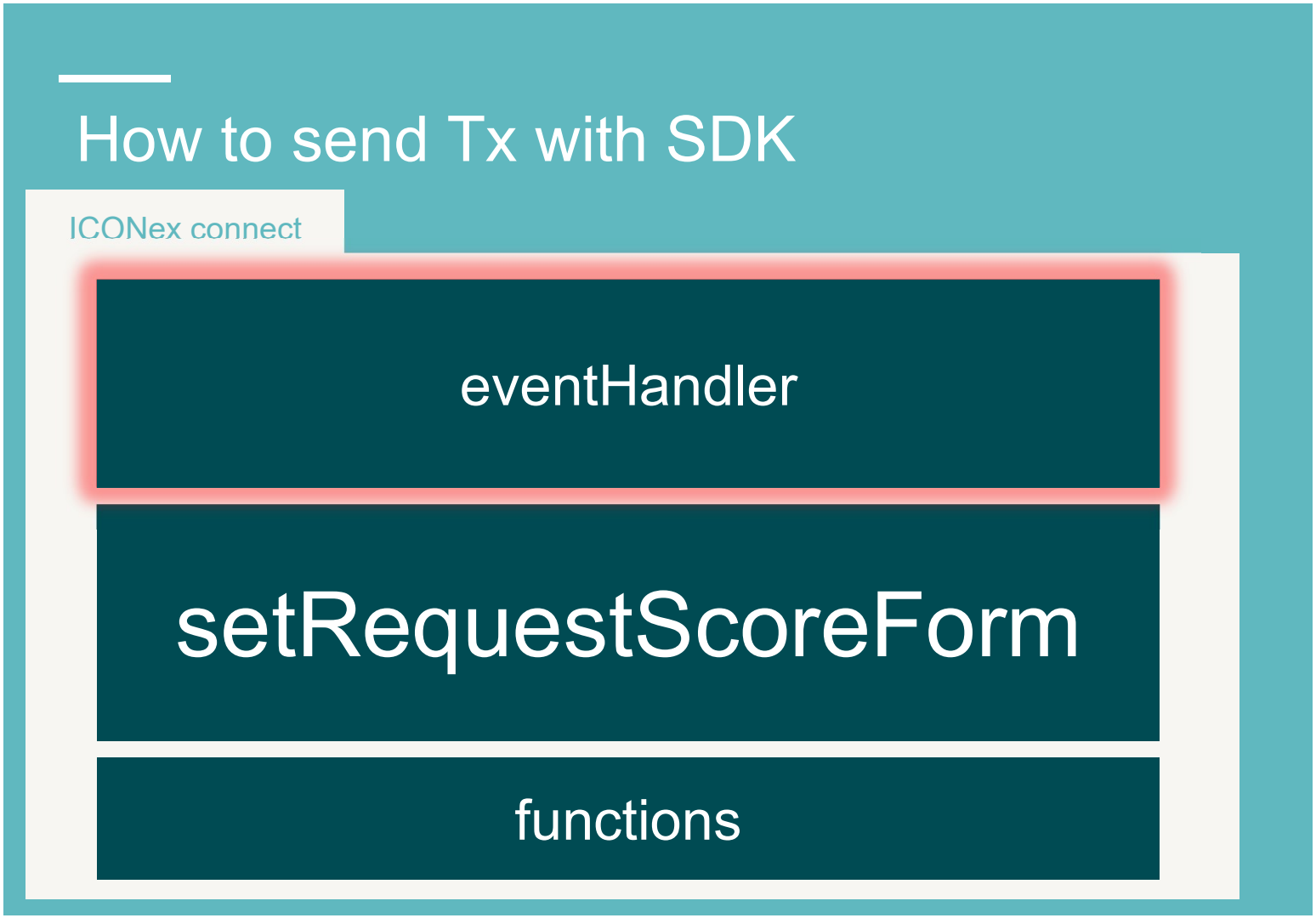


# How to send Tx with SDK

## setRequestScoreForm

```
function setRequestScoreForm() {
    var data = new FormData(requestScoreForm);
    var type = '';
    for (const entry of data) { type = entry[1] };
    switch (type) {
        case 'read-only':
            var callBuilder = new IconBuilder.CallBuilder;
            var readOnlyData = callBuilder
                .from(fromAddress)
                .to('cx43f59485bd34d0c7e9312835d65cb399f6d29651')
                .method("hello")
                .build()
            scoreData.value = JSON.stringify({
                "jsonrpc": "2.0",
                "method": "icx_call",
                "params": readOnlyData,
                "id": 50889
            })
            break;

        case 'send-transaction':
            var callTransactionBuilder = new IconBuilder.CallTransactionBuilder;
            var callTransactionData = callTransactionBuilder
                .from(fromAddress)
                .to("cxb20b5ff06ba50aef42c7832958af59f9ae0651e7")
                .nid(IconConverter.toBigNumber(3))
                .timestamp((new Date()).getTime() * 1000)
```

# How to send Tx with SDK



# How to send Tx with SDK

## eventHandler

```
window.addEventListener("ICONEX_RELAY_RESPONSE", eventHandler, false);

function eventHandler(event) {
    var type = event.detail.type
    var payload = event.detail.payload
    switch (type) {
        case "RESPONSE_HAS_ACCOUNT":
            responseHasAccount.innerHTML = "> Result : " + payload.hasAccount + " (" + typeof payload.hasAccount + ")";
            break
        case "RESPONSE_HAS_ADDRESS":
            responseHasAddress.innerHTML = "> Result : " + payload.hasAddress + " (" + typeof payload.hasAddress + ")";
            break
        case "RESPONSE_ADDRESS":
            fromAddress = payload
            responseAddress.innerHTML = "> Selected ICX Address : " + payload;
            jsonRpc0.disabled = false
            jsonRpc1.disabled = false
            jsonRpc2.disabled = false
            jsonRpc3.disabled = false
            break
        case "RESPONSE_JSON-RPC":
            responseScore.value = JSON.stringify(payload);
            break
        case "CANCEL_JSON-RPC":
            responseScore.value = null;
            break
        case "RESPONSE_SIGNING":
            signingData.value = null
            responseSigning.innerHTML = "> Signature : " + JSON.stringify(payload);
            break
        case "CANCEL_SIGNING":
            signingData.value = null
            responseSigning.value = "> Signature : ";
            break
        default:
    }
}
```

# How to send Tx with SDK

**REQUEST_HAS_ACCOUNT**
**REQUEST_HAS_ADDRESS**
**REQUEST_ADDRESS**
**REQUEST_JSON-RPC**
**REQUEST_SIGNING**

SDK


**RESPONSE_HAS_ACCOUNT**
**RESPONSE_HAS_ADDRESS**
**RESPONSE_ADDRESS**
**RESPONSE_JSON-RPC**
**RESPONSE_SIGNING**

# Dive into ICON – SCORE

## Blackjack

```
$ cd ./samplepage
```

```
$ python manage.py runserver 0.0.0.0:8000
```

# 4. Hands on Exercise

# Dive into ICON - DApp

Step 1. What is a DApp?

Step 2. Review

Step 3. How to build a DApp

Step 4. Hands-on Exercise

1. Make complete example page, Welcome & Scrooge.

## Make complete example SCORE, Scrooge

### Scrooge SCORE

```
$ cd ./exercise/sampleSCORE
```

Select easy or hard

Guide
./exercise/sampleSCORE/README.md

## Make complete example SCORE, Scrooge

Scrooge page

```
$ cd ./exercisepage
```

```
$ python manage.py runserver 0.0.0.0:8000
```

Guide
./exercisepage/README.md