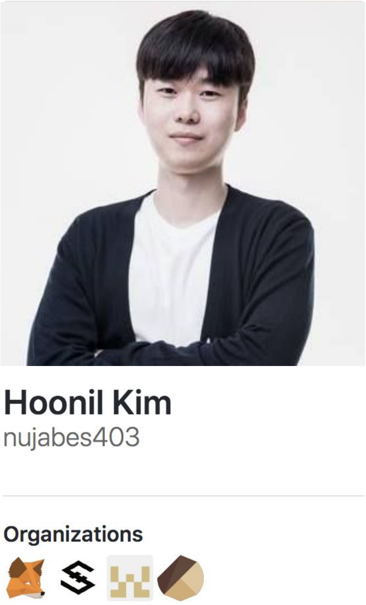
Reasons for choosing IOST from the developer's perspective

Hoonil Kim (IOST-Q.E.D)

About Me



Hoonil Kim
nujabes403

Organizations

Hoonil Kim

- Ex. Streami Inc. (GOPAX)
- Ex. Kakao Ground X (Klaytn)
- Open Source Contributor(web3.js, Metamask contributor)
- IOST-Q.E.D Node

Past and present of blockchain services

Past

Present

TRON
EOS™
IOST

“**Blockchain**" Application

Slow UX - Because it's blockchain ...

Blockchain “**Application**"

UX without knowing blockchain

Important things while making a blockchain service

Fast UX

The response should be fast.

Cheap fee

The fee must be low.

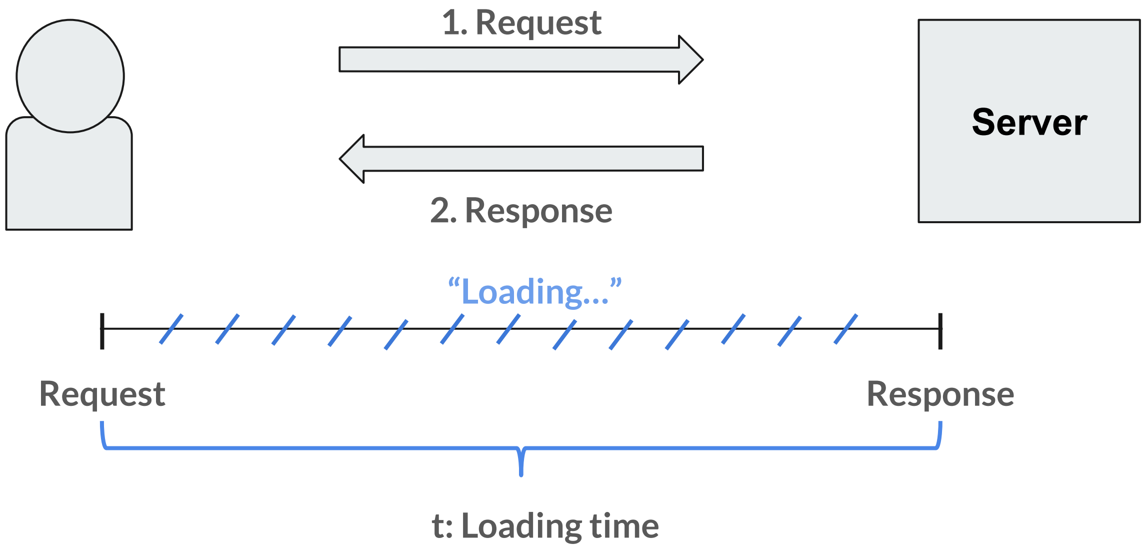
cf) Fee component:
- Contract calculation cost
- Contract storage cost
- Network cost

Developer Experience

It should not be difficult to develop.

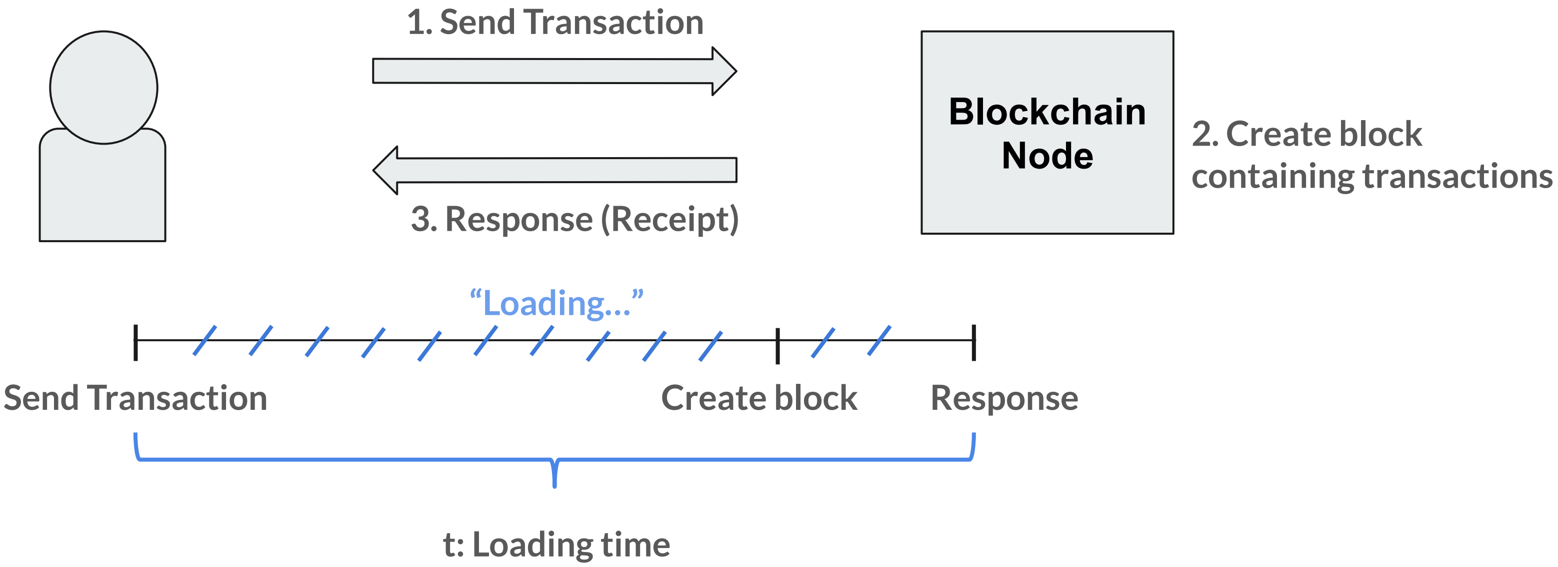
The response should be fast.

What does it mean to be quick to respond to traditional services?



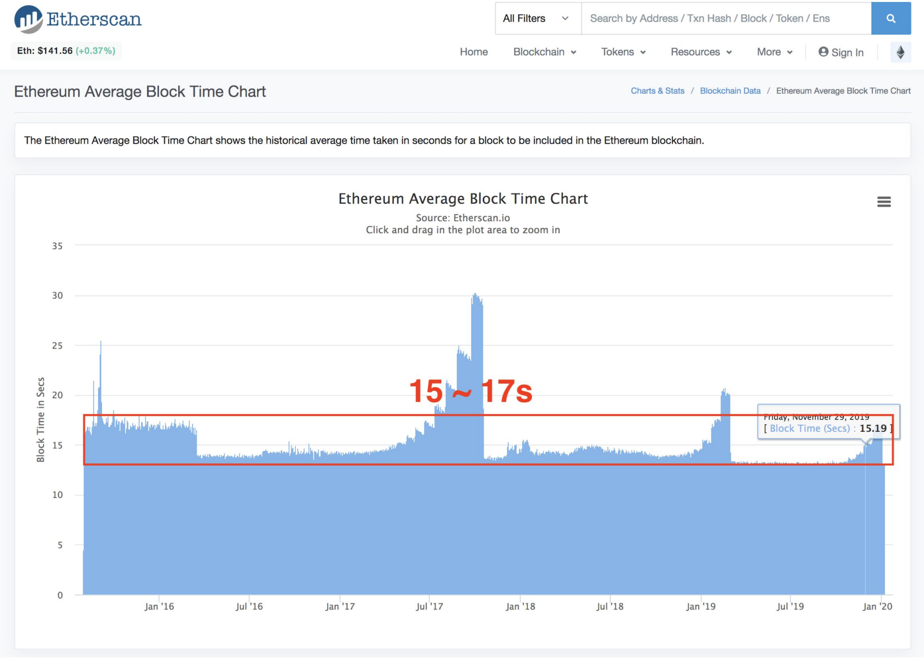
The response should be fast.

What does it mean to be quick to respond in a blockchain service?



The response should be fast.

15 ~ 17s



Ethereum Average Block Time Chart

Charts & Stats / Blockchain Data / Ethereum Average Block Time Chart

The Ethereum Average Block Time Chart shows the historical average time taken in seconds for a block to be included in the Ethereum blockchain.

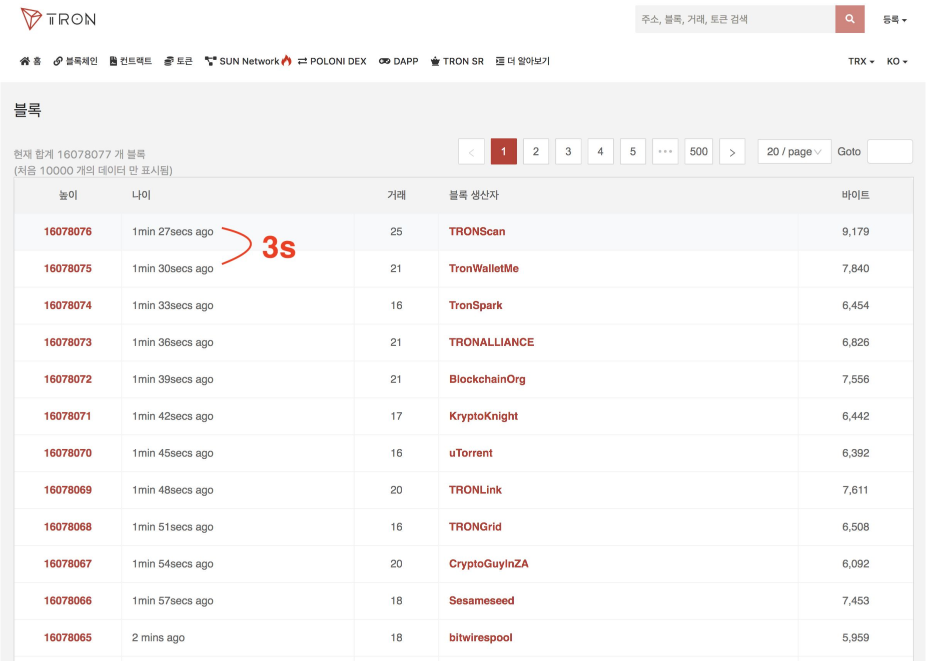
Ethereum Average Block Time Chart

Source: Etherscan.io

Click and drag in the plot area to zoom in

The response should be fast.

3s



블록

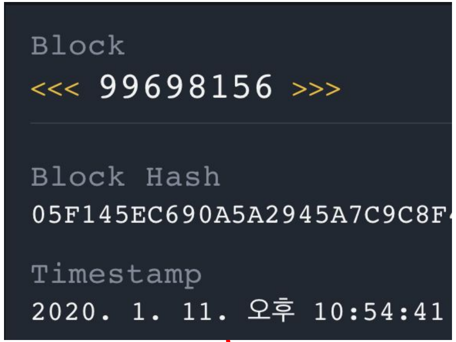
현재 합계 16078077 개 블록
(처음 10000 개의 데이터 만 표시됨)

높이	나이	거래	블록 생산자	바이트
16078076	1min 27secs ago	25	TRONScan	9,179
16078075	1min 30secs ago	21	TronWalletMe	7,840
16078074	1min 33secs ago	16	TronSpark	6,454
16078073	1min 36secs ago	21	TRONALLIANCE	6,826
16078072	1min 39secs ago	21	BlockchainOrg	7,556
16078071	1min 42secs ago	17	KryptoKnight	6,442
16078070	1min 45secs ago	16	uTorrent	6,392
16078069	1min 48secs ago	20	TRONLink	7,611
16078068	1min 51secs ago	16	TRONGrid	6,508
16078067	1min 54secs ago	20	CryptoGuyInZA	6,092
16078066	1min 57secs ago	18	Sesameseed	7,453
16078065	2 mins ago	18	bitwirespool	5,959

The response should be fast.

EOS™

0.5s



Block

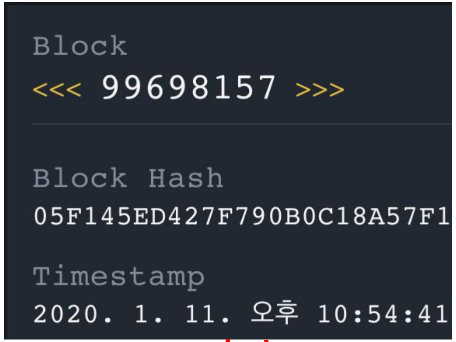
<<< 99698156 >>>

Block Hash

05F145EC690A5A2945A7C9C8F

Timestamp

2020. 1. 11. 오후 10:54:41



Block

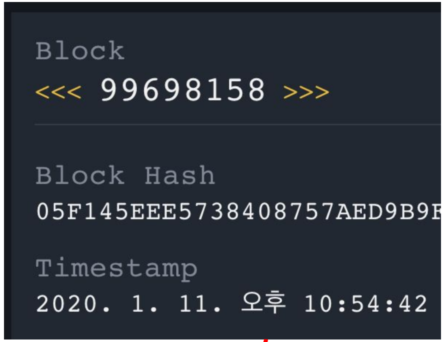
<<< 99698157 >>>

Block Hash

05F145ED427F790B0C18A57F1

Timestamp

2020. 1. 11. 오후 10:54:41



Block

<<< 99698158 >>>

Block Hash

05F145EEE5738408757AED9B9F

Timestamp

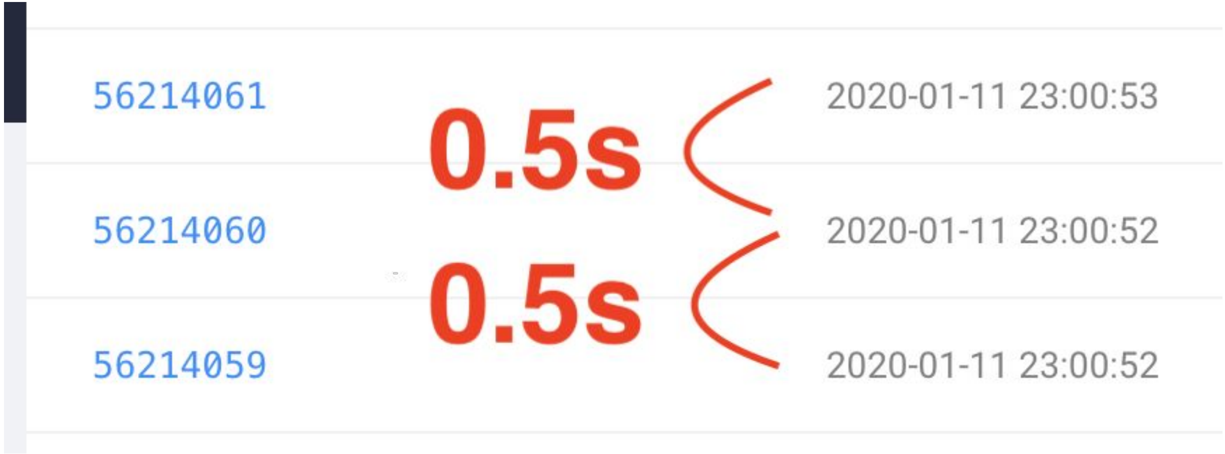
2020. 1. 11. 오후 10:54:42

0.5s

0.5s

The response should be fast.

0.5s



56214061	2020-01-11 23:00:53
56214060	2020-01-11 23:00:52
56214059	2020-01-11 23:00:52

The response should be fast.

Block Creation Time Comparison by Blockchain Platform

	EOS™	TRON	IOST
15 ~ 17s	0.5s	3s	0.5s

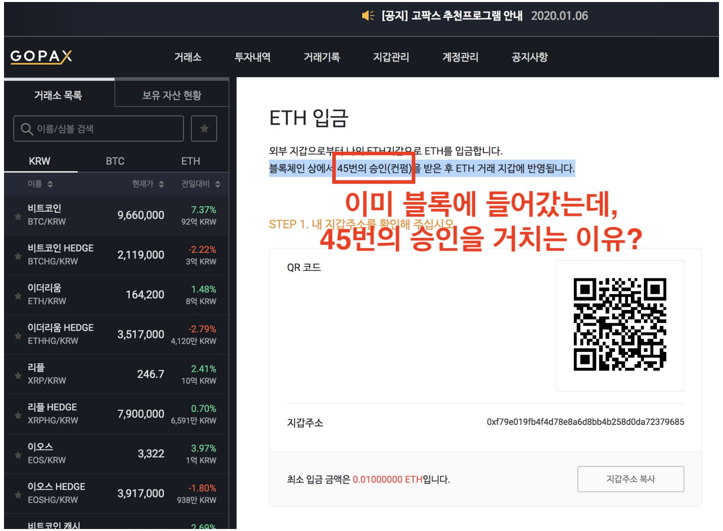
The response should be fast.

What if my transaction fails to enter this block and then to the next block?
=> You have to wait once more for block time. (Twice)

	EOS™	TRON	IOST
30 ~ 34s	1s	6s	1s

The response should be fast.

Block Finality



이름	현재가	전일대비
비트코인 BTC/KRW	9,660,000	7.37% 92억 KRW
비트코인 HEDGE BTCHG/KRW	2,119,000	-2.22% 3억 KRW
이더리움 ETH/KRW	164,200	1.48% 8억 KRW
이더리움 HEDGE ETHHG/KRW	3,517,000	-2.79% 4,120만 KRW
리플 XRP/KRW	246.7	2.41% 10억 KRW
리플 HEDGE XRPHG/KRW	7,900,000	0.70% 6,591만 KRW
이오스 EOS/KRW	3,322	3.97% 1억 KRW
이오스 HEDGE EOSHG/KRW	3,917,000	-1.80% 938만 KRW
비트코인 캐시		2.69%

ETH 입금

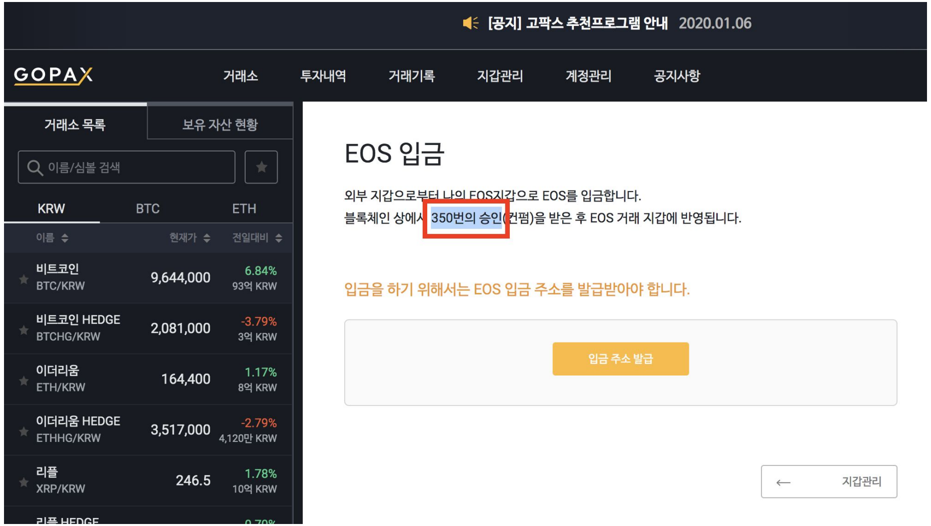
외부 지갑으로부터 나의 ETH지갑으로 ETH를 입금합니다.
블록체인 상에서 45번의 승인(컨펌)을 받은 후 ETH 거래 지갑에 반영됩니다.

STEP 1. 내 지갑주소를 확인해 주십시오.

지갑주소 0xf79e019fb4f4d78e8a6d8bb4b258d0da72379685

최소 입금 금액은 0.01000000 ETH입니다.

이미 블록에 들어갔는데,
45번의 승인을 거치는 이유?



이름	현재가	전일대비
비트코인 BTC/KRW	9,644,000	6.84% 93억 KRW
비트코인 HEDGE BTCHG/KRW	2,081,000	-3.79% 3억 KRW
이더리움 ETH/KRW	164,400	1.17% 8억 KRW
이더리움 HEDGE ETHHG/KRW	3,517,000	-2.79% 4,120만 KRW
리플 XRP/KRW	246.5	1.78% 10억 KRW
리플 HEDGE		0.70%

EOS 입금

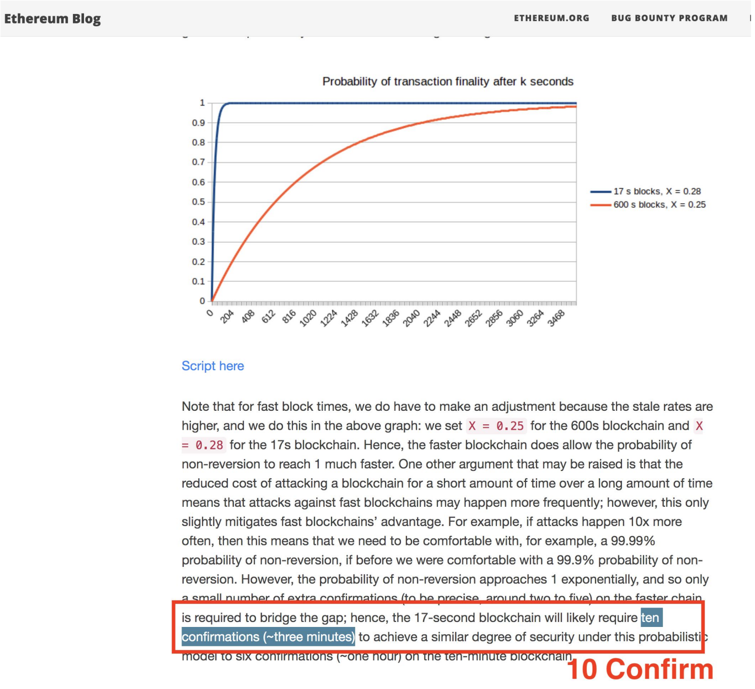
외부 지갑으로부터 나의 EOS지갑으로 EOS를 입금합니다.
블록체인 상에서 350번의 승인(컨펌)을 받은 후 EOS 거래 지갑에 반영됩니다.

입금을 하기 위해서는 EOS 입금 주소를 발급받아야 합니다.

The response should be fast.

Block Finality Comparison by Blockchain Platform

10 confirm



Script here

Note that for fast block times, we do have to make an adjustment because the stale rates are higher, and we do this in the above graph: we set `X = 0.25` for the 600s blockchain and `X = 0.28` for the 17s blockchain. Hence, the faster blockchain does allow the probability of non-reversion to reach 1 much faster. One other argument that may be raised is that the reduced cost of attacking a blockchain for a short amount of time over a long amount of time means that attacks against fast blockchains may happen more frequently; however, this only slightly mitigates fast blockchains' advantage. For example, if attacks happen 10x more often, then this means that we need to be comfortable with, for example, a 99.99% probability of non-reversion, if before we were comfortable with a 99.9% probability of non-reversion. However, the probability of non-reversion approaches 1 exponentially, and so only a small number of extra confirmations (to be precise, around two to five) on the faster chain is required to bridge the gap; hence, the 17-second blockchain will likely require ten confirmations (~three minutes) to achieve a similar degree of security under this probabilistic model to six confirmations (~one hour) on the ten-minute blockchain.

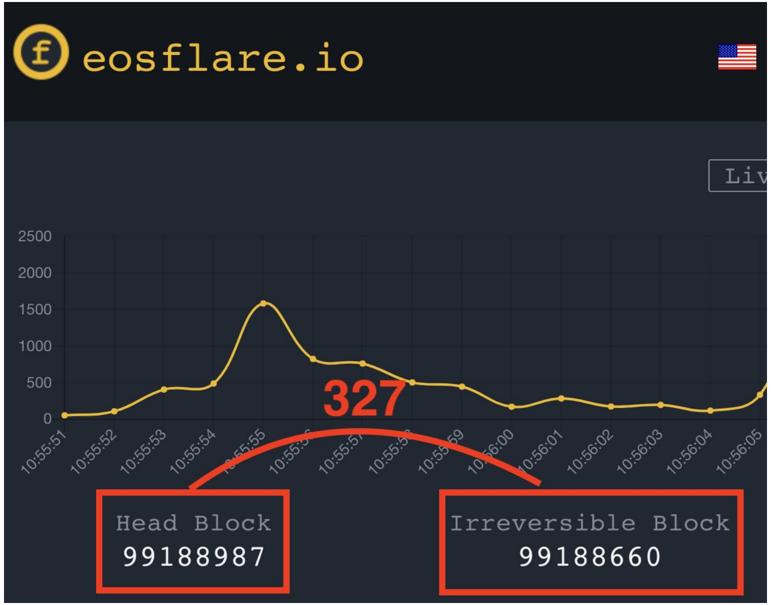
10 Confirm

The response should be fast.

Block Finality Comparison by Blockchain Platform

EOS™

327 confirm



eosflare.io

Head Block
99188987

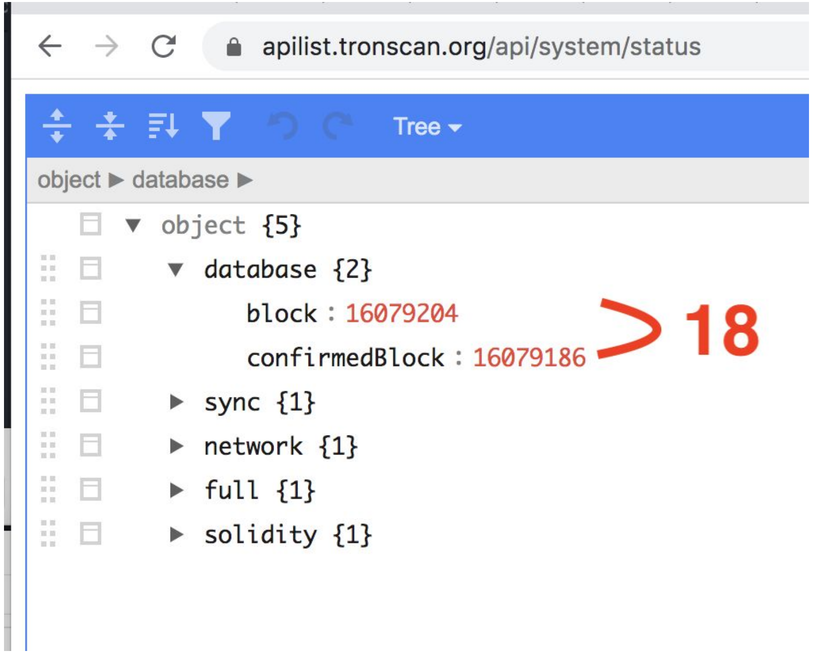
Irreversible Block
99188660

The response should be fast.

Block Finality Comparison by Blockchain Platform

TRON

18 confirm



apilist.tronscan.org/api/system/status

```
▼ object {5}
    ▼ database {2}
        block : 16079204
        confirmedBlock : 16079186
    ▶ sync {1}
    ▶ network {1}
    ▶ full {1}
    ▶ solidity {1}
```

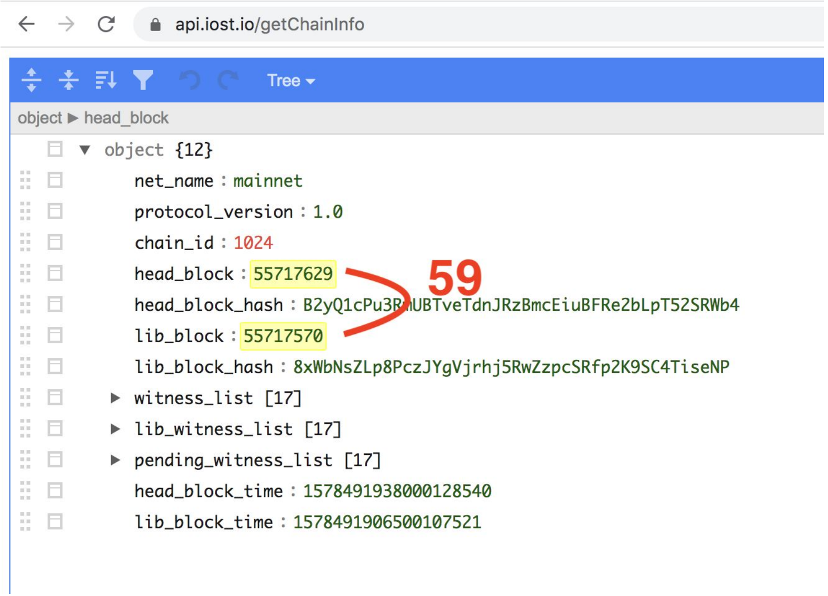
> 18

The response should be fast.

Block Finality Comparison by Blockchain Platform

IOST

59 confirm



```
▼ object {12}
    net_name : mainnet
    protocol_version : 1.0
    chain_id : 1024
    head_block : 55717629
    head_block_hash : B2yQ1cPu3RmUBTveTdnJRzBmcEiuBFRe2bLpT52SRWb4
    lib_block : 55717570
    lib_block_hash : 8xWbNsZLp8PczJYgVjrhj5RwZzpcSRfp2K9SC4TiseNP
  ▶ witness_list [17]
  ▶ lib_witness_list [17]
  ▶ pending_witness_list [17]
    head_block_time : 1578491938000128540
    lib_block_time : 1578491906500107521
```

The response should be fast.

Block Finality Comparison by Blockchain Platform

	EOS™	TRON	IOST
10 confirm	327 confirm	18 confirm	59 confirm
x 17s	x 0.5s	x 3s	x 0.5s
= 170s	= 163s	= 54s	= 29.5s

The response should be fast.

When should you care about Block Finality?

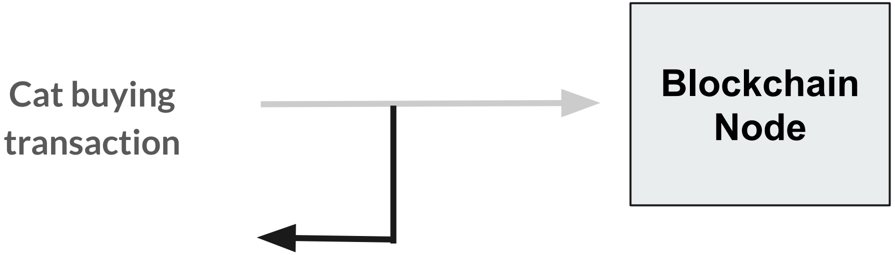
Services that only work in the blockchain world

Service with connection between blockchain world + external world (DB)

The response should be fast.

When should you care about Block Finality?

Services that only work in the blockchain world



Cat buying transaction

Blockchain Node

What if "roll-back" occurs because of bad luck?

=> You can buy it again.

The response should be fast.

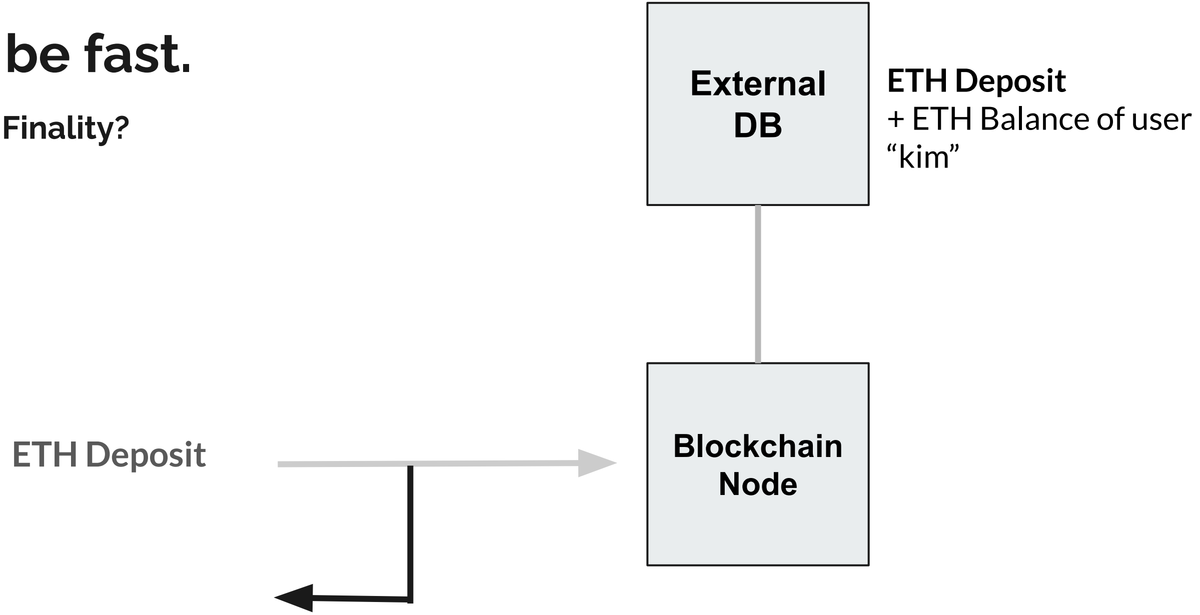
When should you care about Block Finality?

Service with connection between blockchain world + external world

UPbit

bithumb

GOPAX



External DB

ETH Deposit
+ ETH Balance of user "kim"

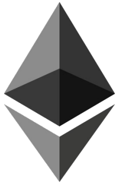
Blockchain Node

ETH Deposit

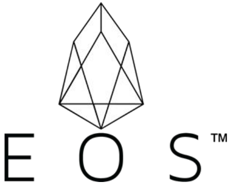
What if "roll-back" occurs because of bad luck?

The fee must be low.

Pay per transaction.



Pay by staking



The fee must be low.

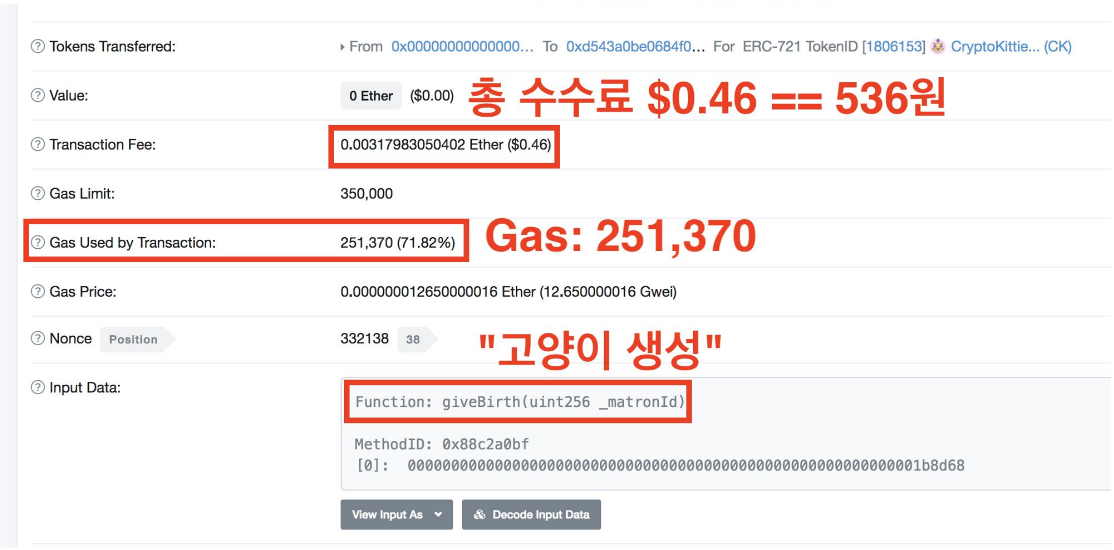
Max payload: 9,000,000 gas

Create a cat (weight:250,000 gas)

Cat-Generated Transactions on One Bus:
9,000,000 / 250,000 = 36 cats

The fee must be low.

Create a cat (weight:250,000 gas)



Tokens Transferred:	‣ From 0x00000000000000... To 0xd543a0be0684f0... For ERC-721 TokenID [1806153] CryptoKittie... (CK)
Value:	0 Ether ($0.00)
Transaction Fee:	0.00317983050402 Ether ($0.46)
Gas Limit:	350,000
Gas Used by Transaction:	251,370 (71.82%)
Gas Price:	0.000000012650000016 Ether (12.650000016 Gwei)
Nonce Position	332138 38
Input Data:	Function: giveBirth(uint256 _matronId) MethodID: 0x88c2a0bf [0]: 00000000000000000000000000000000000000000000000000000000001b8d68

총 수수료 $0.46 == 536원

Gas: 251,370

"고양이 생성"

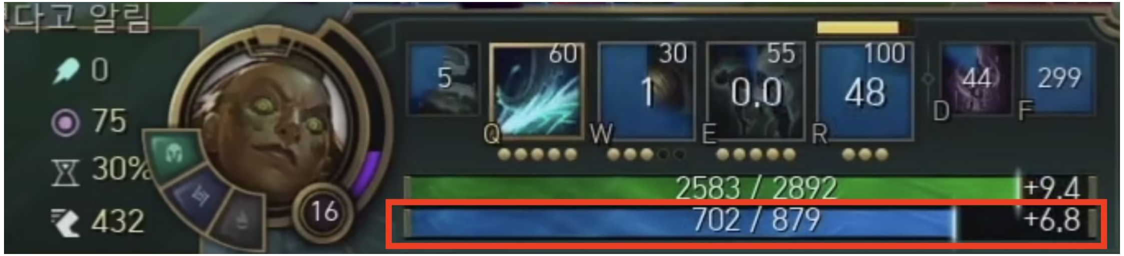
The fee must be low.

Pay by staking

EOS™

TRON

IOST



The fee must be low.

Pay by staking

EOS™

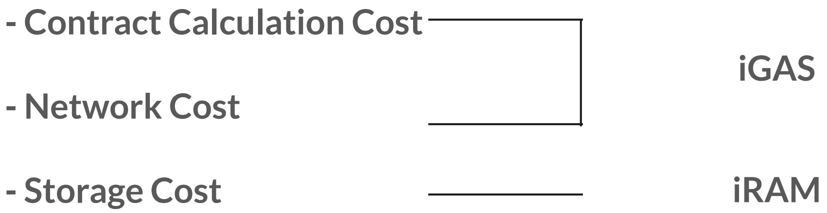
TRON

IOST

The fee must be low.

Pay by staking

IOST



- Contract Calculation Cost
- Network Cost
- Storage Cost

iGAS

iRAM

(The developer can pay for you.)

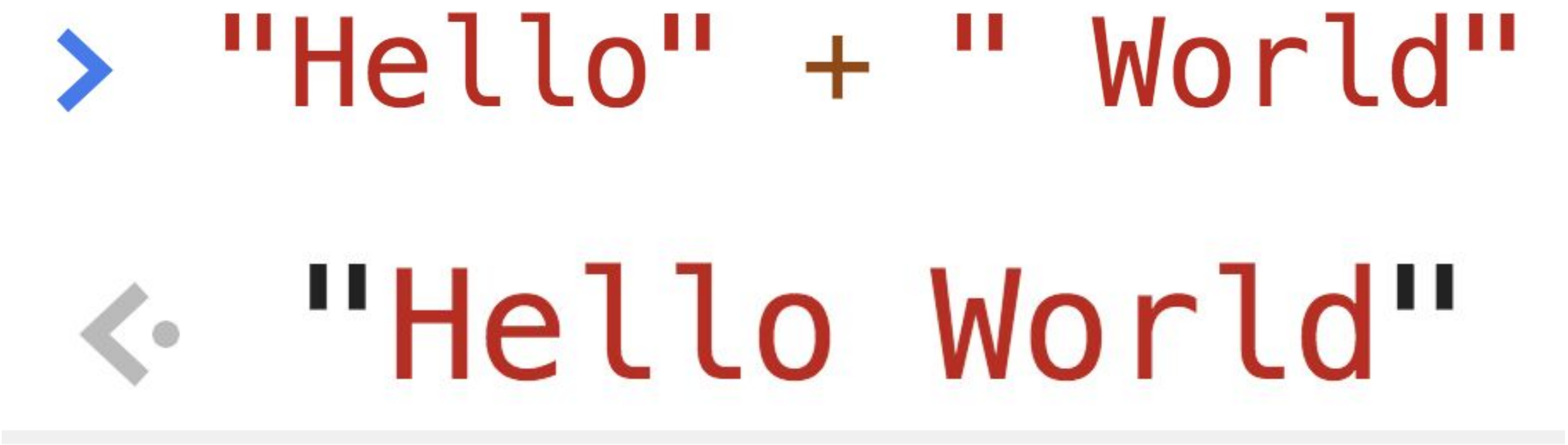
It should not be difficult to develop.

“Difficult to develop”?

1) Developing in a programming language you are not familiar with

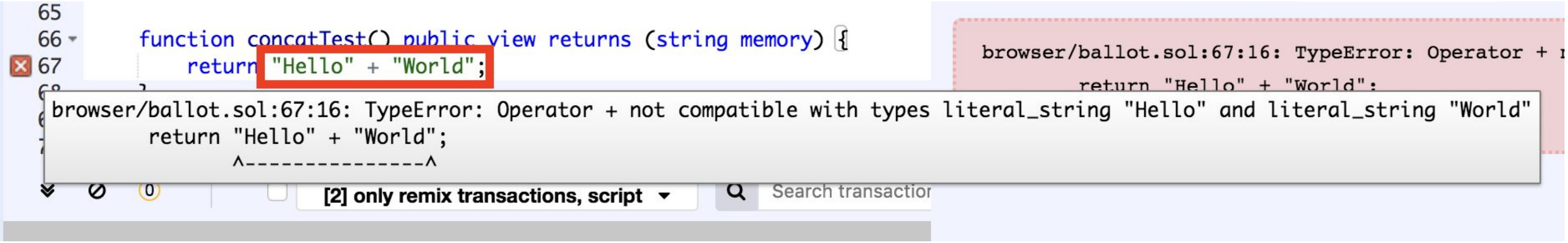
   => There is a lack of data, and it is difficult to answer questions. Why is this an error?

It should not be difficult to develop.



```
> "Hello" + " World"
< "Hello World"
```

It should not be difficult to develop.

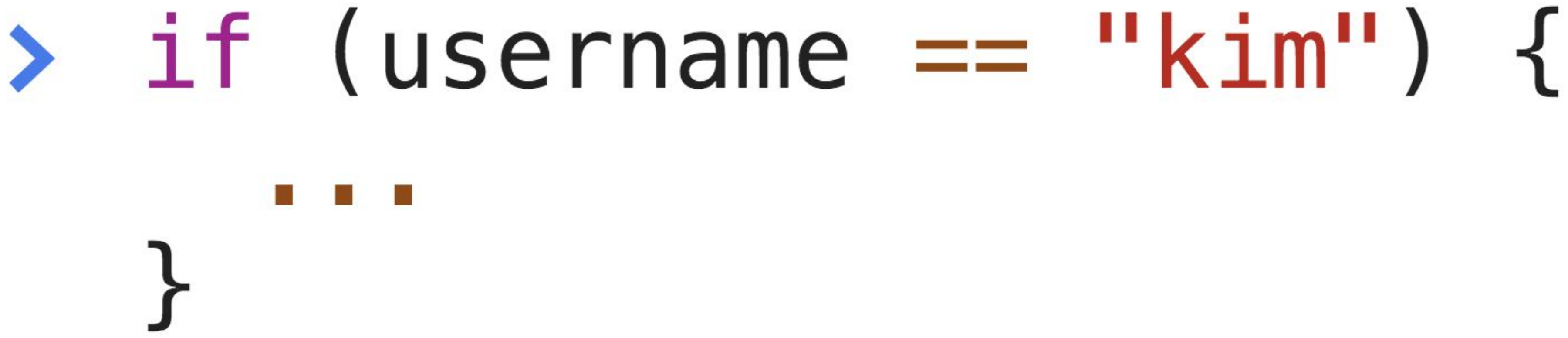


```
function concatTest() public view returns (string memory) {
    return "Hello" + "World";
}
```

```
browser/ballot.sol:67:16: TypeError: Operator + not compatible with types literal_string "Hello" and literal_string "World"
        return "Hello" + "World";
               ^---------------^
```

```
browser/ballot.sol:67:16: TypeError: Operator + r
        return "Hello" + "World";
```

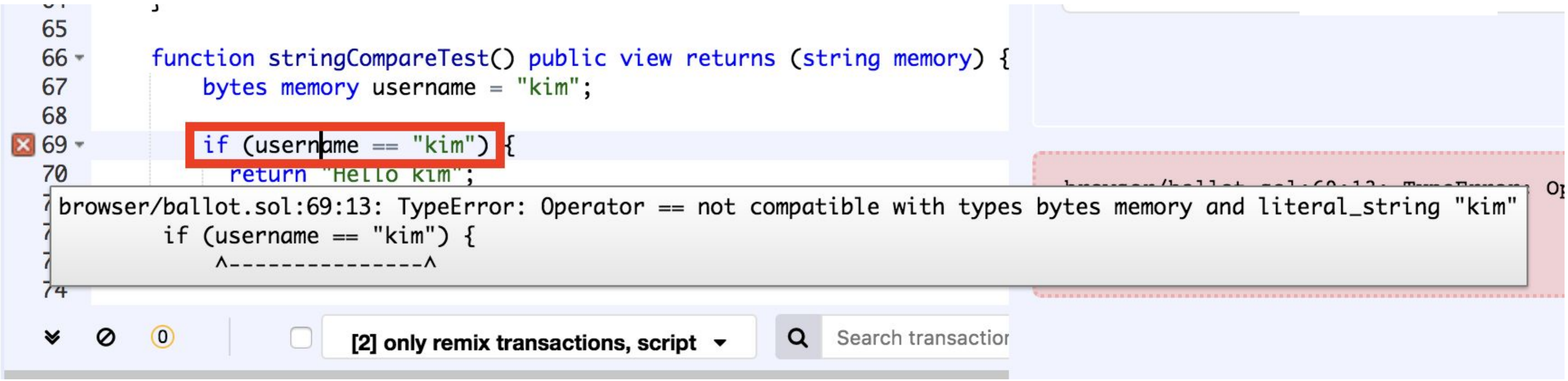
It should not be difficult to develop.



```
> if (username == "kim") {
      ...
  }
```

It should not be difficult to develop.

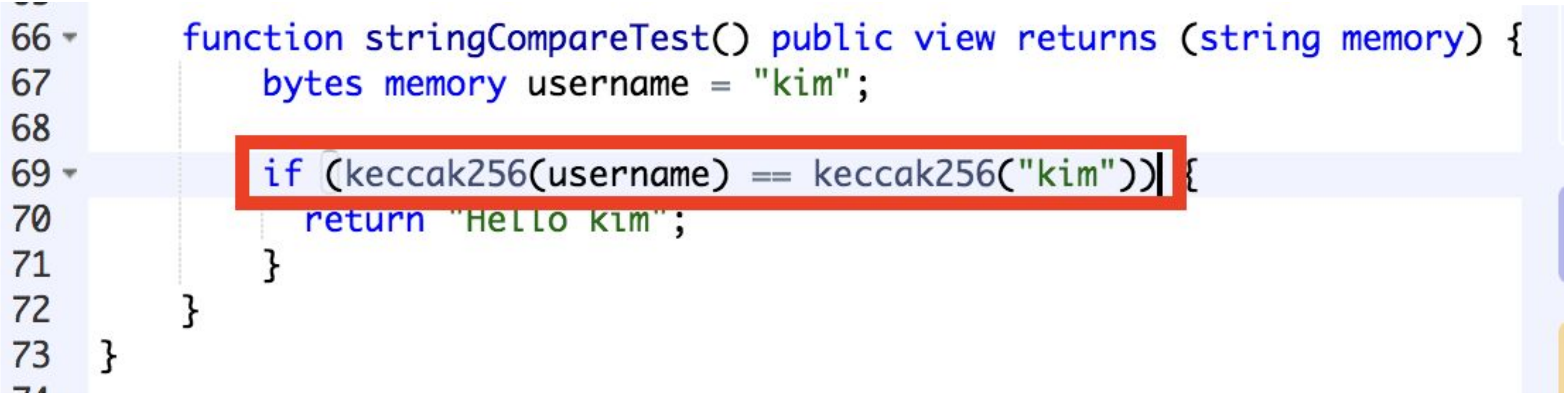
TRON



```
function stringCompareTest() public view returns (string memory) {
    bytes memory username = "kim";

    if (username == "kim") {
        return "Hello kim";
```

```
browser/ballot.sol:69:13: TypeError: Operator == not compatible with types bytes memory and literal_string "kim"
        if (username == "kim") {
            ^---------------^
```

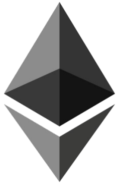
It should not be difficult to develop.



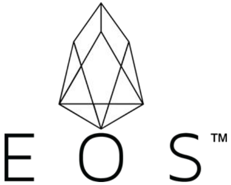
```
    function stringCompareTest() public view returns (string memory) {
        bytes memory username = "kim";

        if (keccak256(username) == keccak256("kim")) {
            return "Hello kim";
        }
    }
}
```

It should not be difficult to develop.

Smart Contract Language by Blockchain Platform



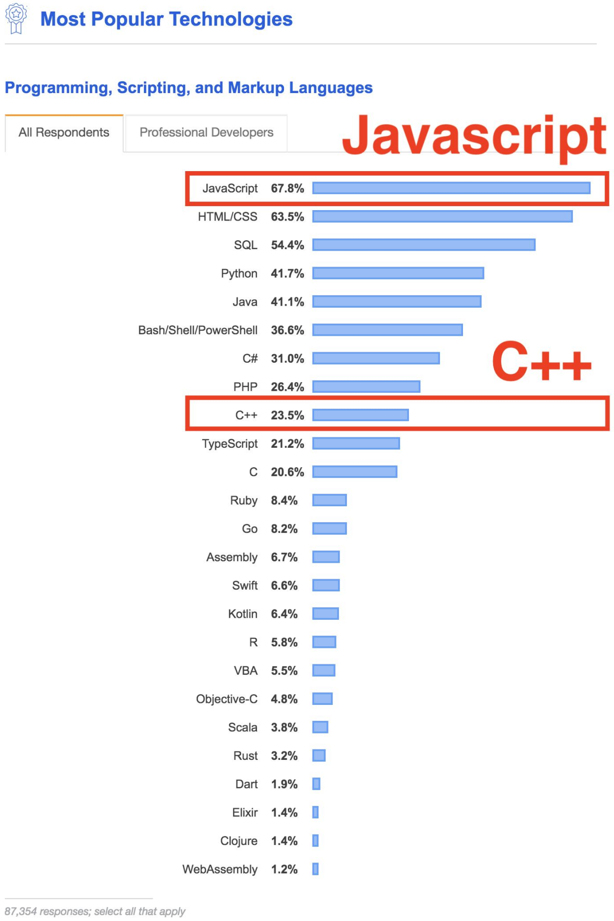
Solidity



C++

Solidity

Javascript



Most Popular Technologies

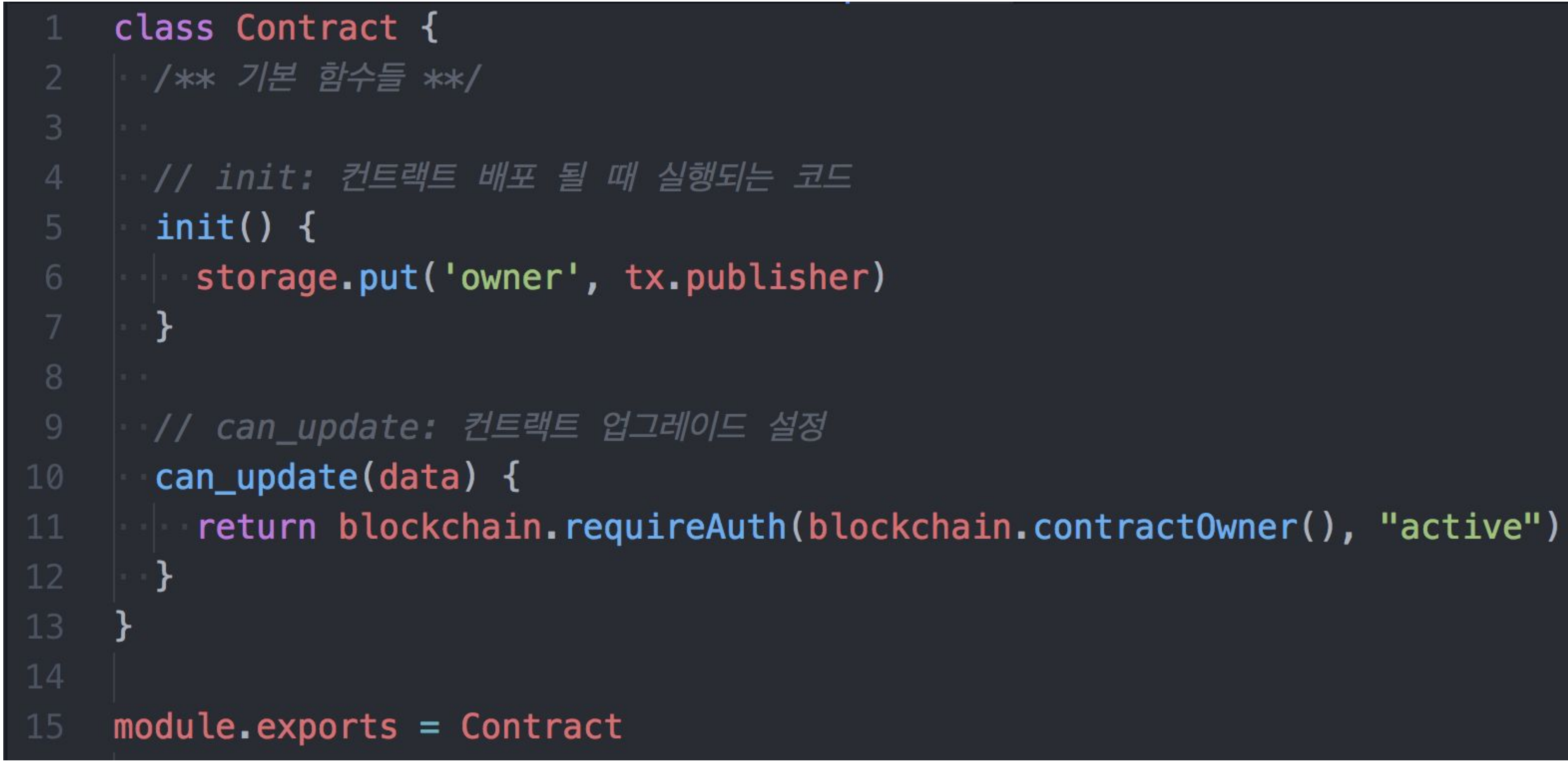
Programming, Scripting, and Markup Languages

JavaScript	67.8%
HTML/CSS	63.5%
SQL	54.4%
Python	41.7%
Java	41.1%
Bash/Shell/PowerShell	36.6%
C#	31.0%
PHP	26.4%
C++	23.5%
TypeScript	21.2%
C	20.6%
Ruby	8.4%
Go	8.2%
Assembly	6.7%
Swift	6.6%
Kotlin	6.4%
R	5.8%
VBA	5.5%
Objective-C	4.8%
Scala	3.8%
Rust	3.2%
Dart	1.9%
Elixir	1.4%
Clojure	1.4%
WebAssembly	1.2%

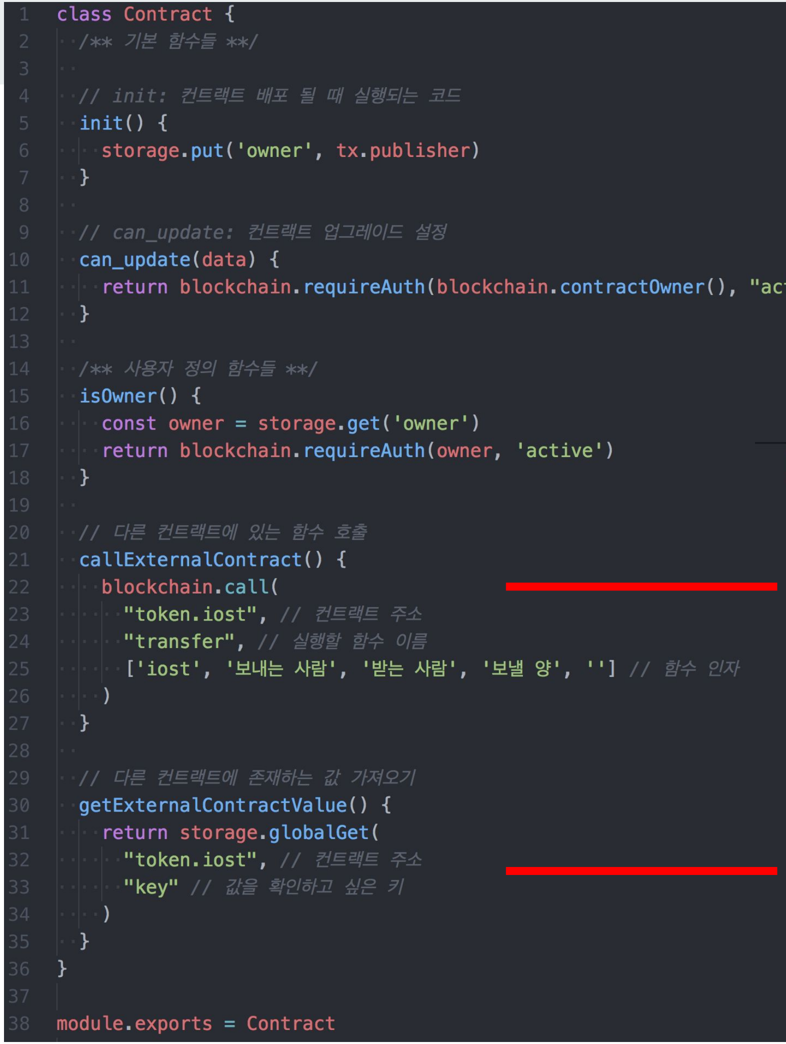
87,354 responses; select all that apply

Javascript

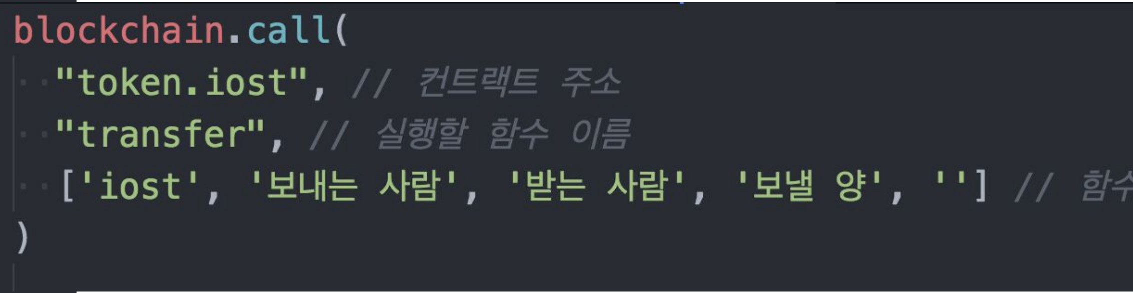
C++



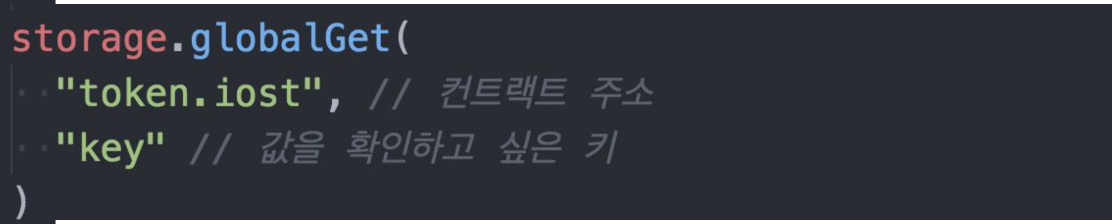
```
class Contract {
  /** 기본 함수들 **/

  // init: 컨트랙트 배포 될 때 실행되는 코드
  init() {
    storage.put('owner', tx.publisher)
  }

  // can_update: 컨트랙트 업그레이드 설정
  can_update(data) {
    return blockchain.requireAuth(blockchain.contractOwner(), "active")
  }
}

module.exports = Contract
```



```
class Contract {
  /** 기본 함수들 **/

  // init: 컨트랙트 배포 될 때 실행되는 코드
  init() {
    storage.put('owner', tx.publisher)
  }

  // can_update: 컨트랙트 업그레이드 설정
  can_update(data) {
    return blockchain.requireAuth(blockchain.contractOwner(), "active")
  }

  /** 사용자 정의 함수들 **/
  isOwner() {
    const owner = storage.get('owner')
    return blockchain.requireAuth(owner, 'active')
  }

  // 다른 컨트랙트에 있는 함수 호출
  callExternalContract() {
    blockchain.call(
      "token.iost", // 컨트랙트 주소
      "transfer", // 실행할 함수 이름
      ['iost', '보내는 사람', '받는 사람', '보낼 양', ''] // 함수 인자
    )
  }

  // 다른 컨트랙트에 존재하는 값 가져오기
  getExternalContractValue() {
    return storage.globalGet(
      "token.iost", // 컨트랙트 주소
      "key" // 값을 확인하고 싶은 키
    )
  }
}

module.exports = Contract
```

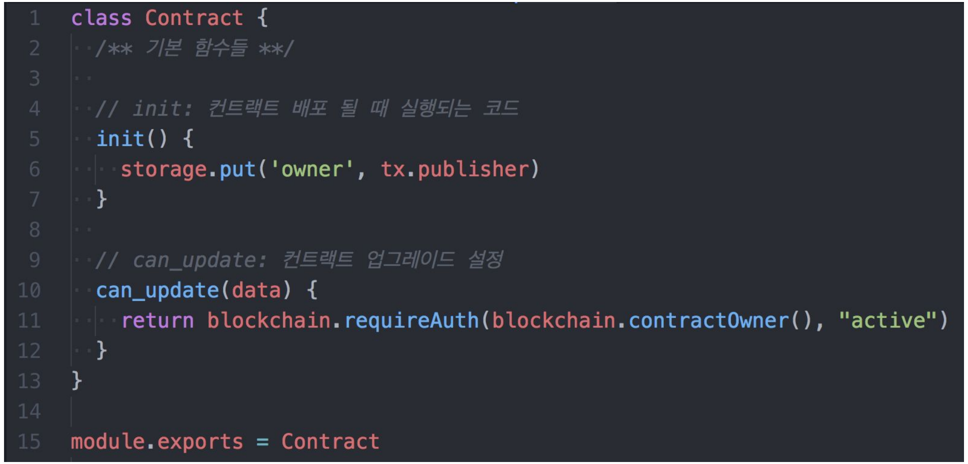


```
blockchain.call(
  "token.iost", // 컨트랙트 주소
  "transfer", // 실행할 함수 이름
  ['iost', '보내는 사람', '받는 사람', '보낼 양', ''] // 함수
)
```



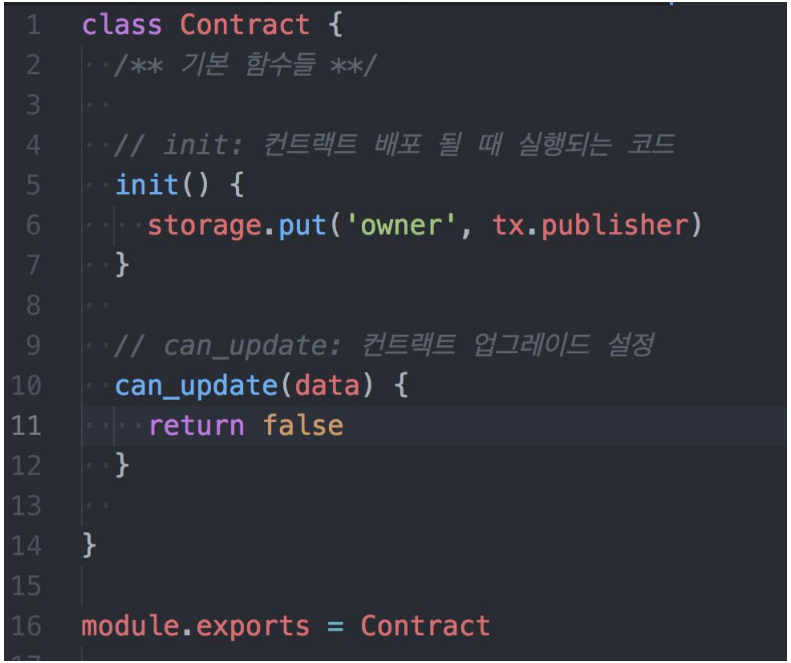
```
storage.globalGet(
  "token.iost", // 컨트랙트 주소
  "key" // 값을 확인하고 싶은 키
)
```

It should not be difficult to develop.



```
class Contract {
  /** 기본 함수들 **/
  
  // init: 컨트랙트 배포 될 때 실행되는 코드
  init() {
    storage.put('owner', tx.publisher)
  }
  
  // can_update: 컨트랙트 업그레이드 설정
  can_update(data) {
    return blockchain.requireAuth(blockchain.contractOwner(), "active")
  }
}

module.exports = Contract
```

컨트랙트 주인이 업그레이드 가능하게 설정



```
class Contract {
  /** 기본 함수들 **/
  
  // init: 컨트랙트 배포 될 때 실행되는 코드
  init() {
    storage.put('owner', tx.publisher)
  }
  
  // can_update: 컨트랙트 업그레이드 설정
  can_update(data) {
    return false
  }
  
}

module.exports = Contract
```

업그레이드 절대 불가능하게 설정