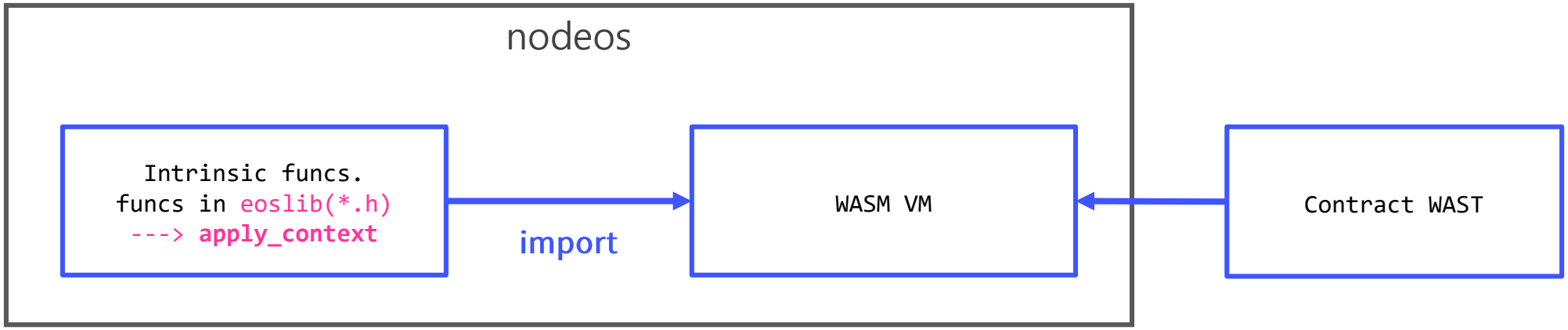
EOSIO
SMART CONTRACT DEV.

2018 2nd EOSIO Developer Meetup, Seoul
plactal.io

Eric song
CTO at plactal.io

Table of contents

- WebAssembly
- No need to learn web assembly, just use it.



- eoscpp (compile tool for EOS smart contract)
  - shell script
  - uses clang(C++14), llvm, llc, s2wasm

- **No numerical address on EOS -> AccountName**
  - Human readable
  - Base32 encoding
    - ".12345abcdefghijklmnopqrstuvwxyz"
    - Max 13 자리, 마지막은 ".12345abcdefghi" 만 (Name type)
    - Account 용은 '.' 제외, 최대 12자리. Account auction 구현 전까지.
  - uint64_t 로 packing (fast !)
  - Sort 가 쉬운 구조 (Persistence api 내부에서 key 로 사용됨)

- **ToString**
  - `eosio::name{}.to_string(), N()`

Contract Skeleton

```
eoscpp -n 컨트랙트이름
```

```
/**
*  The apply() method must have C calling convention so that the blockchain can lookup and call this
method.
*/
extern "C" {

   /**
    * This method is called once when the contract is published or updated.
    */
   void init()  {
      eosio::print( "Init World!\n" );
   }

   /// The apply method implements the dispatch of events to this contract
   /// receiver: 현재 코드 account_name,
   /// code: action이 시작된 account_name
   /// action : action name
   void apply( uint64_t receiver, uint64_t code, uint64_t action ) {
      eosio::print( "Hello World: ", eosio::name{code}.to_string(), "->",
eosio::name{action}.to_string(), "\n" );
   }

} // extern "C"
```

Contract 정의방법

- implement apply() in extern “C”
  - `if...else / switch..case` 로 message filtering
  - `eosio::dispatch()` 사용 (metaprogramming)
    - `static Contract::on( ActionType )` 으로 dispatch
    - `bancor` 예제 (converter.hpp)

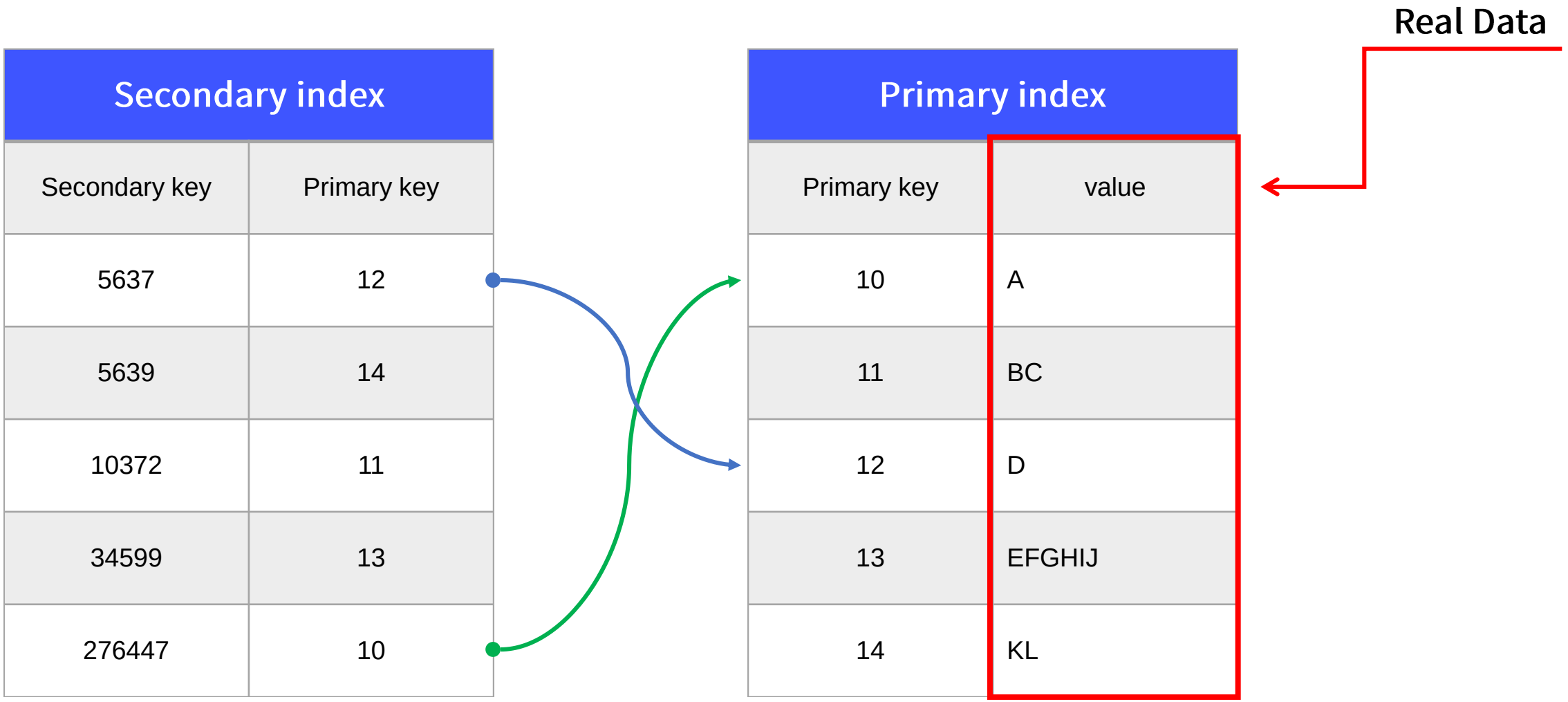
- EOSIO_ABI() macro
  - `EOSIO_ABI( class_name, (func_name1)(func_name2)..)`
  - `class` 는 `eosio::contract` 상속
  - `receiver == code` 인 경우만 filtering 됨. 주의!
    - external code 에 의한 inline action 처리 안됨.
    - 예) transfer EOS to contract

db.h

- **primary index(key) : db_xxx_i64()**
  - 64-bit key 로 data insert, read, update, delete, find
  - int32_t 타입 리턴값은 iterator.임의 크기의 데이터 저장. (table)
  - 64-bit key-value database 로 생각하면 됨.
  - key 는 주로 account_name 이 됨. eos 에서 account_name 이 실제로 uint64_t 인 이유중 하나.
  - account_name 은 uint64_t 로 표현하면, alphabetic sorting 이 numeric sorting 과 같음(고속)

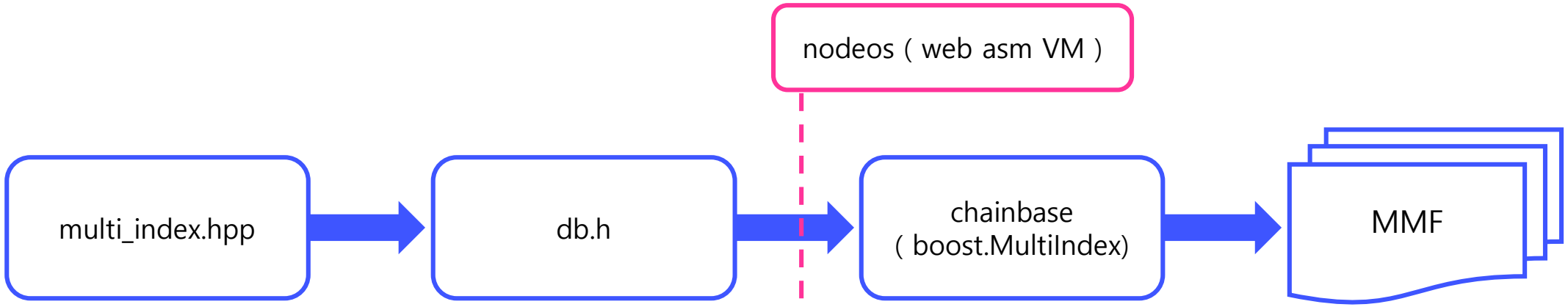
- **secondary index : db_idxYYY_xxx()**
  - 위의 primary key 에 연관 시키는 secondary key 라고 생각하면 됨.
  - YYY : 64, 128, 256 uint, double(64-bit), long_double(128-bit)
    - db_idx64_store, db_idx128_store, db_idx_long_double_find_primary

db.h


Real Data
Secondary index
Secondary key
Primary key
5637
12
5639
14
10372
11
34599
13
276447
10
Primary index
Primary key
value
10
A
11
BC
12
D
13
EFGHIJ
14
KL

multi_index

- **Boost lib 의 MultiIndex 처럼 eosio 용으로 만든것**
  - Boost.MultiIndex: 다수의 interface 를 지원하는 container
  - 실제 data access/modify 는 db.h 의 api 를 사용
  - dh.h 의 api 는 chainbase 사용 -> chainbase 의 storage 는 MMF(Memory Mapped File)
  - primary key + secondary keys -> relational database 개념적 유사.
  - 간단하게 simple db 라고 생각.



chainbase

- **Chainbase**
  - eos 의 내부 database
  - eos_src/libraries/chainbase
  - Boost.managed_mapped_file 에 Boost.MultiIndex container 의 object이름을 key로 해서 container 저장
  - EOS 시스템 정보 + 각 contract DB 가 MMF 로 access 되는 것임.
  - steemit 도 사용. EOS 와는 조금 다른 코드

table

- Table 구조 정의하기
  - primary key 로 사용할 variable 추가(uint64_t 혹은 그 보다 작은 size 변수)
    - uint64_t primary_key() const 함수 반드시 구현!
  - 실제 data access/modify 는 db.h 의 api 를 사용
  - multi_index 정의하기

```
eosio::multi_index< N(테이블명), 테이블 struct, typename… secondary_Indcies>
```

```
struct currency_stats
{
  asset supply;
  asset max_supply;
  account_name issuer;

  uint64_t primary_key() const { return supply.symbol.name();}
};

typedef eosio::multi_index<N(stat), currency_stats> stats;
```

특징 및 주의 사항

- EOS Contract 은 매 msg call 마다(외부에서의 contract 함수 call) 새로운 context 로 call 됨
  - msg(action) 마다 contract 멤버 변수 값이 유지되는 것이 아님. (differ from ETH)
    - action 간의 데이터 전달은 db(multi_index table) 이용할 것.
    - 변수 정의 형태의 간단한 데이터 전달은 singleton.hpp 의 singleton 이용할 것.
      - `singleton< N(athprice), uint64_t> ath(code,scope);`
- secondary index 는 16개 까지만 가능
- iterator(forward / reverse), find 제공됨.

1. 개발자용 문서는 어디있나요?

https://github.com/EOSIO/eos/wiki

2. eosio 빌드후 뭘 해야 하나요?

nodeos, cleos, keosd 역할과 실행 스위치 학습. p2p 없이 단일 노드 띄우기 위주로 공부하세요. single node private network 구성되면 준비완료.

3. nodeos 를 드디어 띄웠습니다.

참 잘했어요. https://github.com/EOSIO/eos/wiki/Tutorial-Comprehensive-Accounts-and-Wallets 로 eosio 에서의 계정과 지갑을 공부하세요.

4. 이제 컨트랙 작성할까요?

eos 는 bios boot 절차가 있습니다. 공부하세요. https://github.com/EOSIO/eos/wiki/Tutorial-Bios-Boot-Sequence

5. 스테이킹, ram 마켓.., 영어니 더 어렵습니다. 한글 없나요?

http://koreos.io 에 비 개발자도 이해하는 문서가 풍부하게 있습니다. 한국의 BP 후보들도 양질의 자료를 만들어 게시합니다.

6. eos 가 제공하는 api 목록은 어디있나요?

eosio_src/contracts/eosiolib 의 *.h 파일을 보세요.

7. 얼마만큼의 EOS 를 가져야 서비스를 할 수 있을까요?

다음 session인 EOSEOUL 발표를 잘 들으시고,
eosio_src/contracts/delegate_bandwidth.cpp 참조. mainnet 이후 BP 들이 제공하는 정보나 툴을 이용하세요.

8. 권한 확인은 어떤 api?

`action.h : require_auth(), require_auth2(), has_auth()`

9. ICO 컨트랙 작성 중인데, EOS 들어오는 건 어떻게 아나요?

`apply()`에 **`if ((code==N(eosio.token)) && (action==N(transfer)) {}`** 처리 추가.

10. 9번 처럼 내 컨트랙의 action 이 발생할 때 다른 contract 에 알리려면?

`action.h: require_recipient()`

11. contract 업로드 중 unresolvable.. 에러가 납니다.

함수 body 가 없는 함수를 call 했습니다. 일반 windows/linux 에서 C++ 코딩 처럼 생각하고, 함수 call 할 때 주로 나타납니다.
즉 eos 에서 제공하지 않는 api 사용한 것이 없는지 보세요. (예) network function.)
간혹, nodeos 와 cleos 버전이 맞지 않는 경우에도 발생합니다.

12. 값 리턴은 어떻게 하죠?

console 출력을 이용할 수 있습니다. rest api call 의 response 로 console 출력을 보내줍니다.

13. 특정 event가 발생했는지 어떻게 monitoring 하죠?

현재는 pub/sub 기능이 없습니다. polling 으로 table 을 조회하세요.

14. javascript 로 dApp 개발?

eosio 공식 client lib. 는 https://github.com/EOSIO/eosjs 가 2018.05. 현재 유일합니다.

15. java / android 개발?

https://github.com/plactal/EosCommander 의 data package 를 가져다 쓰시면 됩니다.

16. go lang 개발?

https://github.com/eoscanada/eos-go

17. dApp 개발 하려니 막막합니다. 데이터를 체인 올리려니 너무 복잡합니다.

블럭체인은 비싼 자원입니다. dApp 이라고 해서, full 로 모든 데이터를 체인에 올릴 필요는 없습니다.
사용자에게 신뢰를 줄 수 있는 데이터+로직의 최소화된 부분만 체인을 이용하세요. 모든 데이터/로직이 체인에 있을 필요는 없습니다.

Resources

- **Github : https://github.com/EOSIO/eos**
  - See wiki pages for tutorials and docs.
  - binary : https://github.com/EOSIO/eos/releases
- **EOSIO medium**
  - https://medium.com/eosio
- **eosio 공식 채널 (dApp vc 투자등)**
  - https://eos.io/contact-us
- **Community**
  - https://forums.eosgo.io/
  - Telegram
    - https://t.me/joinchat/EaEnSUPktgfoI-XPfMYtcQ (general dev.)
    - https://t.me/EosGameDevelopers (game dev.)
  - Koreos
    - http://koreos.io/KOREOS

      특히, EOS Ground Zero BPC 메뉴를 보시면 BP 의 공약과 활동을 보실 수 있습니다.