

# Blockchain 입문

2019.6.12

백명숙



## 과정개요

■ 이 과정에서는 1) 블록체인의 기본개념 및 용어 2) 이더리움 플랫폼의 아키텍처 3) 이더리움 클라이언트 프로그램인 Geth(go-ethereum)를 이용하며 Private 네트워크를 구축 4) 이더리움 스마트 컨트랙트 코드 작성 IDE 인 Remix소개 5) 스마트 컨트랙트를 실행 및 배포를 위한 Ganache와 Truffle 소개 등 이더리움 기반 DApp 개발에 필요한 내용들을 학습합니다.

# 과정목표

- 비트코인과 이더리움의 탄생 배경과 작동 원리를 이해합니다.
- 비트코인, 이더리움 코어 마스터를 기반으로 블록체인의 기본 개념들을 이해 합니다.
- 이더리움 DApp 개발에 필요한 내용들을 이해합니다.

# 러닝 맵



# 강사 프로필

성명	백명숙
소속 및 직함	휴클라우드 이사
주요 경력	일은시스템, Sun Micro Systems 교육서비스, 인터넷커머스코리아
강의/관심 분야	Java, Framework, Python, Data Science, Machine Learning, Deep Learning
자격/저서/대외활동	Java기반 오픈소스 프레임워크/NCS 개발위원

## Learning Object(학습모듈) 및 커리큘럼

LO	커리큘럼
블록체인 개요	- 블록체인의 기본 개념 및 용어들
이더리움 개요	- 이더리움 플랫폼의 아키텍처 및 용어들
Geth 사용	- 이더리움 클라이언트 (Geth)을 사용하여 Private 네트워크 구축 - 이더리움 Dashboard 구축, 네트워크 status 모니터링
DApp 개발 사전지식	- Solidity, Remix, Ganache 등 이더리움 DApp개발 준비에 필요한 내용들

# 1. 블록체인 개요



# 돈(錢, Money)이란?

---

## 질문) 돈이란 무엇이라고 생각하는가?

- 돈은 물건의 **가치 표현 단위**이자 필요한 물건과의 **교환 수단**
- 쉽게 구하기 어려운 **희소성**이 있어야 하고 **똑같은 모양과 형상**을 갖고 있어야 한다.
- 가벼워서 이동 자체가 쉽고 , 쉽게 변하거나 손상되지 않도록 **내구성** 보유해야 한다.
- 돈은 교환을 쉽게 하기 위한 매개수단이기 때문에 그 **가치는 사회적 합의와 신뢰를 통해 인정되고 유지**되어야 한다. (화폐는 ‘믿음’이다.)
- 실물화폐(Physical Currency)  
조개, 돌, 금/은/동전, 지폐
- 전자화폐(Digital Currency)
  - 1) 가상화폐(virtual currency) : 네이버페이, 포인트, 게임머니 등 지폐나 동전과 같은 실물이 없이 네트워크로 연결된 가상 공간에서 전자적 형태로 사용되는 디지털 화폐 또는 전자 화폐를 말한다.
  - 2) 암호화폐(crypto currency) : 암호화폐는 안전한 거래와 통화발행을 조정하기 위해 암호학을 사용하여 교환의 수단으로 만들어진 디지털 자산(digital asset)으로 비트코인, 이더리움, 리플, 모네로등이 있다.



# 현실화된 암호화폐, 비트코인

---

- 비트코인(bitcoin)은 블록체인 기술을 기반으로 만들어진 온라인 암호화폐이다.
- 2008년 10월 사토시 나카모토라는 가명을 쓰는 프로그래머가 개발하여, 2009년 1월 프로그램 소스를 배포했다.
- 비트코인은 중개인 없이 개인간의 거래를 가능하게 해주는 전자 화폐이다.
- 중앙은행이 없이 전 세계적 범위에서 P2P 방식으로 개인들 간에 자유롭게 송금 등의 금융거래를 할 수 있게 설계되어 있다. 거래 장부는 블록체인 기술을 바탕으로 전 세계적인 범위에서 여러 사용자들의 서버에 분산하여 저장하기 때문에 해킹이 사실상 불가능하다. SHA-256 기반의 암호 해시 함수를 사용한다. – 위키피디아

(A purely Peer-to-Peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution)

– Satoshi paper, <https://bitcoin.org/bitcoin.pdf>



# 금융위기와 비트코인

---

질문) 블록체인/비트코인 기술은 왜 각광받기 시작했는가?(2008년 금융위기와 연결 지어 설명하시오)

- 달러의 유일한 공급기관인 미국 연방준비은행(Federal Reserve)이 모든 권한을 갖고 있기 때문에 2008년 미국 연준의 대규모 양적완화가 있으면서, '탈중앙화된 화폐가 필요하다.' 라는 생각이 나오기 시작했다. 기존의 통화 시스템이 구조적인 문제점을 드러내기 시작하면서 비트코인(암호화폐)이 가지고 있는 잠재력에 사람들이 관심을 갖기 시작했다.
- 블록체인 기술은 2008년 사토시 나카모토가 쓴 8페이지 짜리 논문으로 부터 시작됐다. 거래의 기록과 관리 권한을 P2P 네트워크로 분산해 블록으로 기록하고 관리할 수 있도록 했다. 매 10분마다 새로운 거래 정보를 담은 블록이 시간 순으로 계속 연결되기 때문에 블록체인 이라는 이름이 붙었다. 중간 관리자 없이 거래 당사자 간 직접 거래가 가능해 비용 절감, 효율적인 거래가 가능하다는 것이 강점으로 꼽힌다.

# 기존의 돈과 비트코인 차이점

---

질문) 기존의 돈과 비트코인은 어떻게 다른가?(제3 신뢰기관을 이용해서 설명하시오)

- 기존의 돈의 특징은 정부가 추가적인 지폐를 찍어낼 수 있다는 것이다. 이로 인하여 돈의 공급량이 늘어나기 때문에, 유통되는 모든 돈의 가치는 떨어지게 됩니다. 반면에 대부분 암호화폐의 경우, 중앙은행이 발행하는 돈과는 달리 총 발행량은 공개되어 있습니다.
- 탈중앙화 화폐인 비트코인은 최대 발행량이 2100만개로 한정되어 있습니다. 이후로 단 한개의 비트코인도 발행할 수 없기 때문에, 수요와 공급의 법칙에 따라 비트코인이 언제나 어느 정도의 가치가 존재할 수 밖에 없습니다. 또한 시간이 지남에 따라 그 수요가 늘어날 수 있으므로 그 가치가 오를 수 있는 잠재력 또한 가지고 있습니다.

# 블록체인의 분산 공개 원장

- 기존의 금융거래에서는 '제3자'가 있습니다. (중앙 집중 원장)

원화를 계좌이체 할 때 먼저 보내는 사람 계좌에서 받는 사람 계좌로 돈을 계좌이체 '요청'을 합니다.

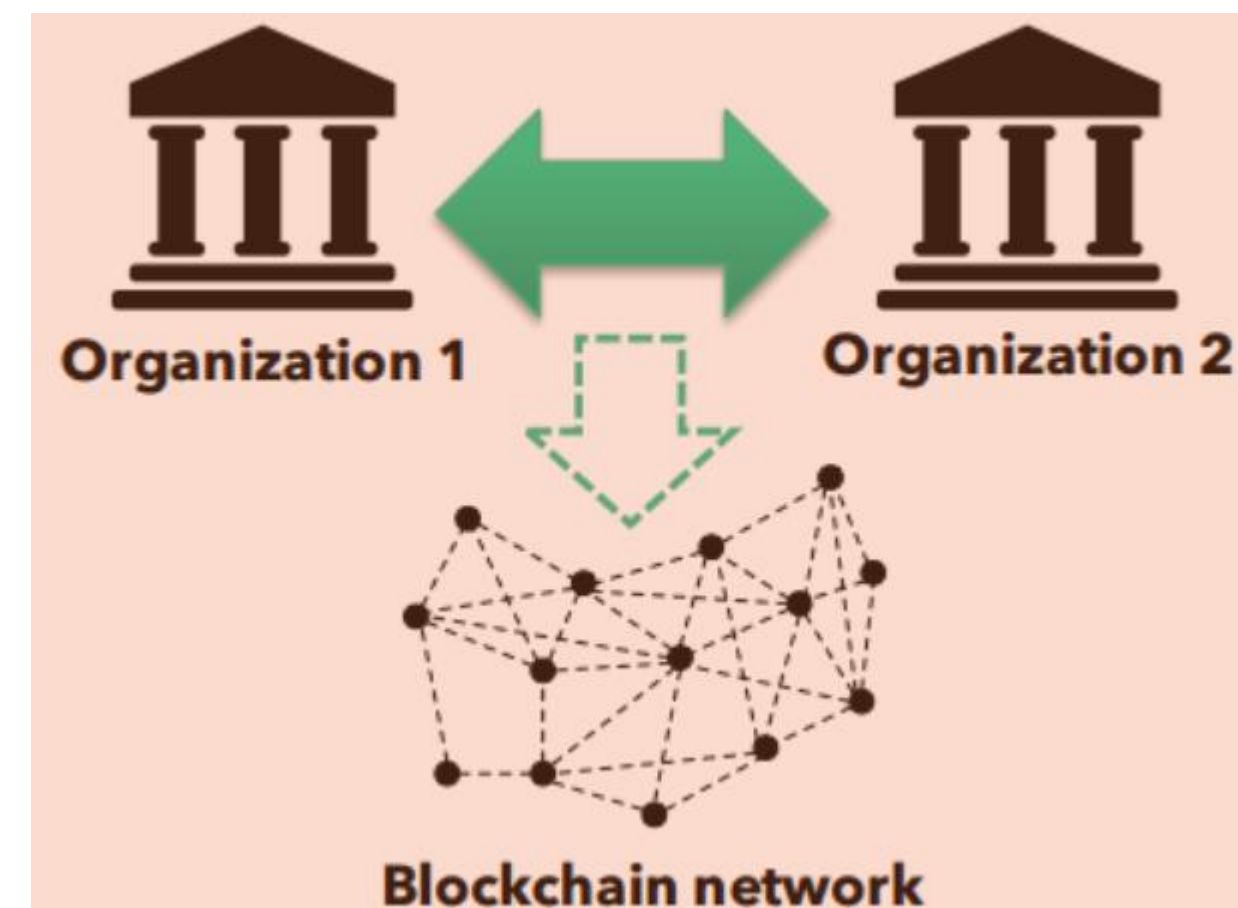
그러면 '제3자'인 은행이 해당 거래를 검토하여 수신자 계좌에 돈을 입금하게 되는 것입니다.

제3자인 'Centralized Trusted Authority(중앙 신뢰기관)'이 거래를 인증하고 관리합니다.

- 블록체인에서는 개인과 개인의 거래에서 '제3자'가 없습니다. (분산 공개 원장)

즉, 블록체인은 “제3자” 없이 안전한 거래를 할 수 있게 해주는 기발한 기술입니다. 거래 정보를 감추지 않고 모두에게 공개하고 누구나 거래 정보를 생성할 수 있으며, 거래 정보를 모두에게 복사해서 사본을 저장하고 그 사본끼리 동기화 시킨다.

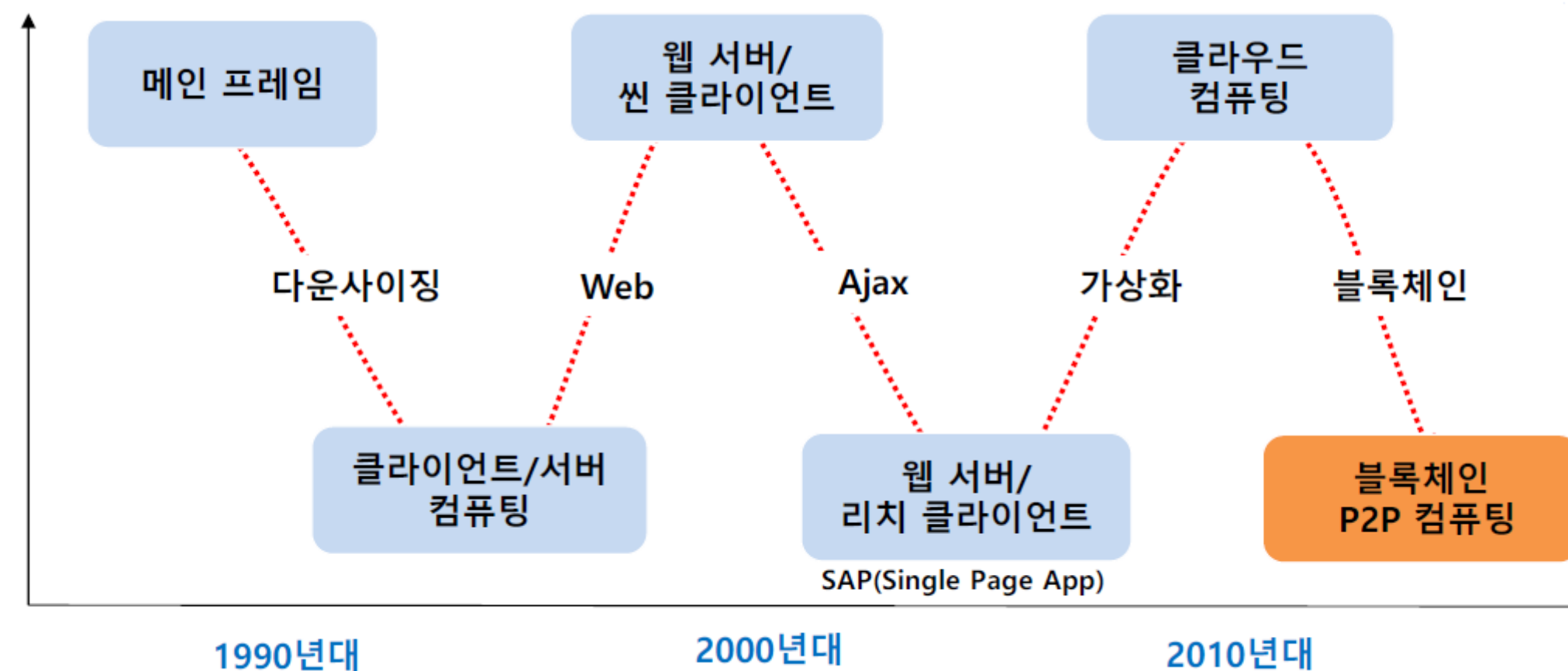
이중화, 삼중화가 아니라 수만중화 처리를 해서 기록이 사라지는 일을 원천적으로 막아버린다.



# P2P 컴퓨팅

기존의 가상화폐는 은행과 같은 중앙 서버에 거래내역을 저장하지만, **블록체인 기반 코인은 P2P(Peer to Peer) 형식**으로 거래가 되고 거래내역을 은행이 아닌 블록에 저장합니다. 그리고 이 블록에 한번 저장되면 누구도 임의로 수정할 수 없으며, 누구나 인터넷이 연결된 어느 곳이든 데이터를 열람할 수 있게끔 기술적으로 설계되어 있습니다.

블록체인 기술은 국제 송금, 소액결제 같은 금융 뿐 만 아니라 의료 데이터, 정부 행정서비스, IoT(사물인터넷)까지 그 활용 범위를 넓혀가고 있습니다.



# 블록체인과 비트코인 관계

---

## 질문) 블록체인과 비트코인의 관계는 무엇입니까?

- 블록 체인이 바탕이고, 비트코인은 블록 체인 바탕 위에서 구현된 하나의 서비스 또는 상품이다. 블록 체인은 비트코인 뿐만 아니라 다른 코인의 바탕이 될 수도 있으며, 코인 뿐만 아니라 다른 서비스나 상품의 바탕이 될 수도 있다. 비트코인을 만들기 위해 고민하던 중에 블록 체인이라는 기술이 탄생하였다.
- 거래가 일어날 때마다 새로운 장부(블록)를 봉인해서 기존의 블록체인에 추가하는 작업을 '작업증명'이라고 하는데, 이 과정에서 참여자들의 컴퓨팅 파워와 전기가 이용됩니다.
- 블록체인이 안전해 지려면 많은 참여자가 필요한데, 아무런 보상없이 내 컴퓨팅 파워와 전기를 들여서 작업증명에 나설 리 없겠죠? 보상이 필요한 순간입니다. 바로 비트코인 같은 암호화폐입니다. 보상이 없다면 참여자도 없고, 블록체인 역시 존재할 수 없을 겁니다. 암호화폐는 블록체인을 움직이는 연료라고도 볼 수 있습니다. 이 때문에 블록체인은 육성하고 암호화폐를 금지하겠다는 것은 현재 시스템 아래에서는 모순되는 발언이 라고도 볼 수 있지 않을까요?

# 블록체인의 장단점

---

- 장점

1. 모든 사용자가 장부를 가지고 있기 때문에 신뢰성을 보장할 제3자가 필요 없다는 점입니다.
2. 해킹을 쉽게 차단할 수 있습니다. 일부 네트워크가 해킹 당해도 타격이 없으며 분산 구조이므로 디도스 공격에도 문제가 없습니다.
3. 모든 거래내역을 공개하기 때문에 기존 금융 서비스보다 확실하고 거래내역이 투명하게 보관됩니다.
4. 블록체인은 중앙 관리자가 따로 필요가 없기 때문에 유지 보수, 보안 유지, 거래 중계자 등에 필요한 비용이 절감되어 매우 경제적입니다.

- 단점

1. 일단은 속도가 느립니다. 중앙으로만 보내면 되던 정보가 블록체인은 개인과 개인이 진행하고 정보교류가 필요하기 때문에 속도면에서 느립니다.
2. 기술적 오류나 업그레이드 진행 시 사용자의 과반수가 동의해야 하고 의사결정을 지연할 수 있기 때문에 신속한 업데이트가 어렵습니다.



# 블록체인의 종류

## 1. 퍼블릭 블록체인(public blockchain)

: 공개형 블록체인이며 거래 내역뿐만 아니라 네트워크에서 이루어지는 여러 행동(Actions)이 모두 공유되어 추적이 가능하다. 퍼블릭 블록체인 네트워크에 참여할 수 있는 조건(암호화폐 수량, 서버 사양 등)만 갖추면 누구나 블록을 생성할 수 있다. 대표적인 예로 비트코인, 이더리움 등이 있다.

## 2. 프라이빗 블록체인(private blockchain)

: 폐쇄형 블록체인이며 허가된 참여자 외 거래 내역과 여러 행동(Actions)은 공유되지 않고 추적이 불가능하다. 프라이빗 블록체인 네트워크에 참여하기 위해 한 명의 주체로부터 허가된 참여자만 참여하여 블록을 생성할 수 있다. 대표적인 예로 세계 최대 금융 컨소시엄 R3의 블록체인 코다([Corda](#)), IBM과 리눅스 재단의 하이퍼렛저 패브릭([Hyperledger Fabric](#))

	퍼블릭 블록체인	프라이빗 블록체인
읽기 권한	누구나 가능	허가된 기관만
검증	누구나 참여하면 검증	승인된 기관과 감독만
트랜잭션 생성자	누구나 생성	책임자만 참여
속도	7~20TPS	1000 TPS 이상
권한 관리	누구나	허가된 기관만
예시	비트코인, 이더리움, 라이트코인 등	IBM Fabric, 각종 금융 기관

# Blockchain 용어들

---

- 비트코인의 3가지 기술  
: 1) 51% 공격 2) 작업증명(POW) 3) 해쉬(Hash)와 논스(Nonce)
- 블록(Block) : 블록해쉬, 해쉬, 논스
- 노드(Node)
- 채굴(Mining)
- 작업증명(Proof of Work)
- 합의 메커니즘
- 비잔틴 장군의 딜레마
- 이중지불문제
- 작업증명(Proof of Work)과 지분증명(Proof of Stake)
- UTXO(Unspent Transaction Output)

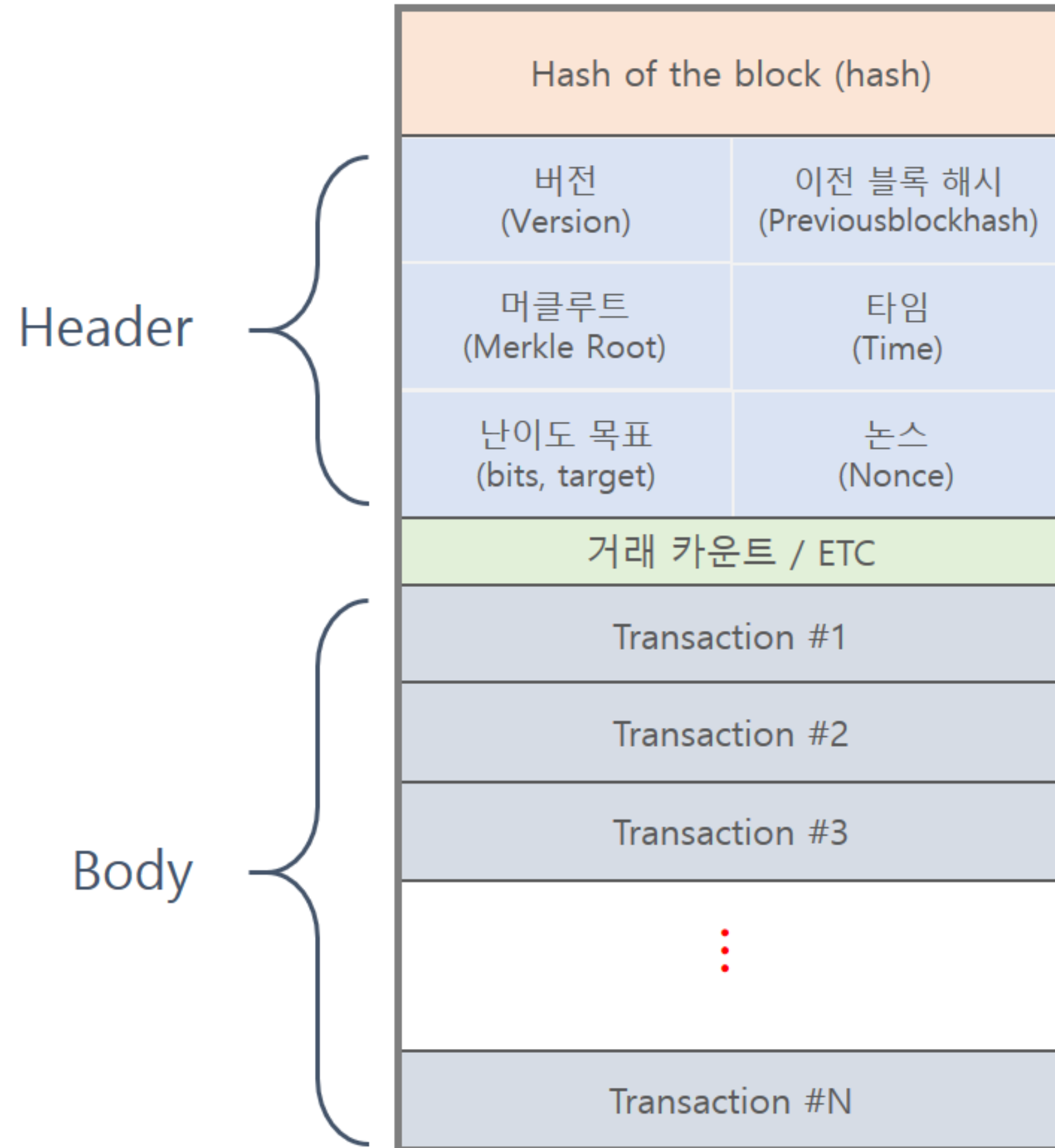
# 블록이란?

---

## 질문) 블록체인에서 ‘블록’이란 무엇입니까?

- 블록은 유효한 거래 정보의 묶음이다.
- 한 예로, 비트코인의 블록은 모든 암호화 화폐에 있어 필수적인 정보인 트랜잭션 정보를 저장한다.
- 비트코인의 블록 하나에는 평균 1,800개의 거래 정보가 포함될 수 있으며, 블록 하나의 물리적인 크기는 평균 1.12Mbyte이다. (출처: <https://bitnodes.earn.com/> )
- 블록은 블록헤더와 거래정보, 기타 정보로 구성된다.
  - ✓ 블록헤더는 version, previousblockhash, merklehash, time, bits, nonce 6개의 정보로 구성된다.
  - ✓ 거래정보는 입출금과 관련한 여러가지 정보를 가지고 있다.
  - ✓ 기타정보는 블록내에 있는 정보 중에서 블록헤더와 거래정보에 해당하지 않는 정보를 말하며, 블록해쉬계산에 사용되지 않는다.

# 블록헤더



• 블록헤더는 다음의 6가지 정보로 구성

1. Version : 소프트웨어/프로토콜 버전

2. Previous blockhash : 블록 체인에서 바로 앞에 위치하는 블록의 블록 해시

3. Merklehash : 개별 거래 정보의 거래 해시를 2진 트리 형태로 구성할 때, 트리 루트에 위치하는 해시값

4. Time : 블록이 생성된 시간

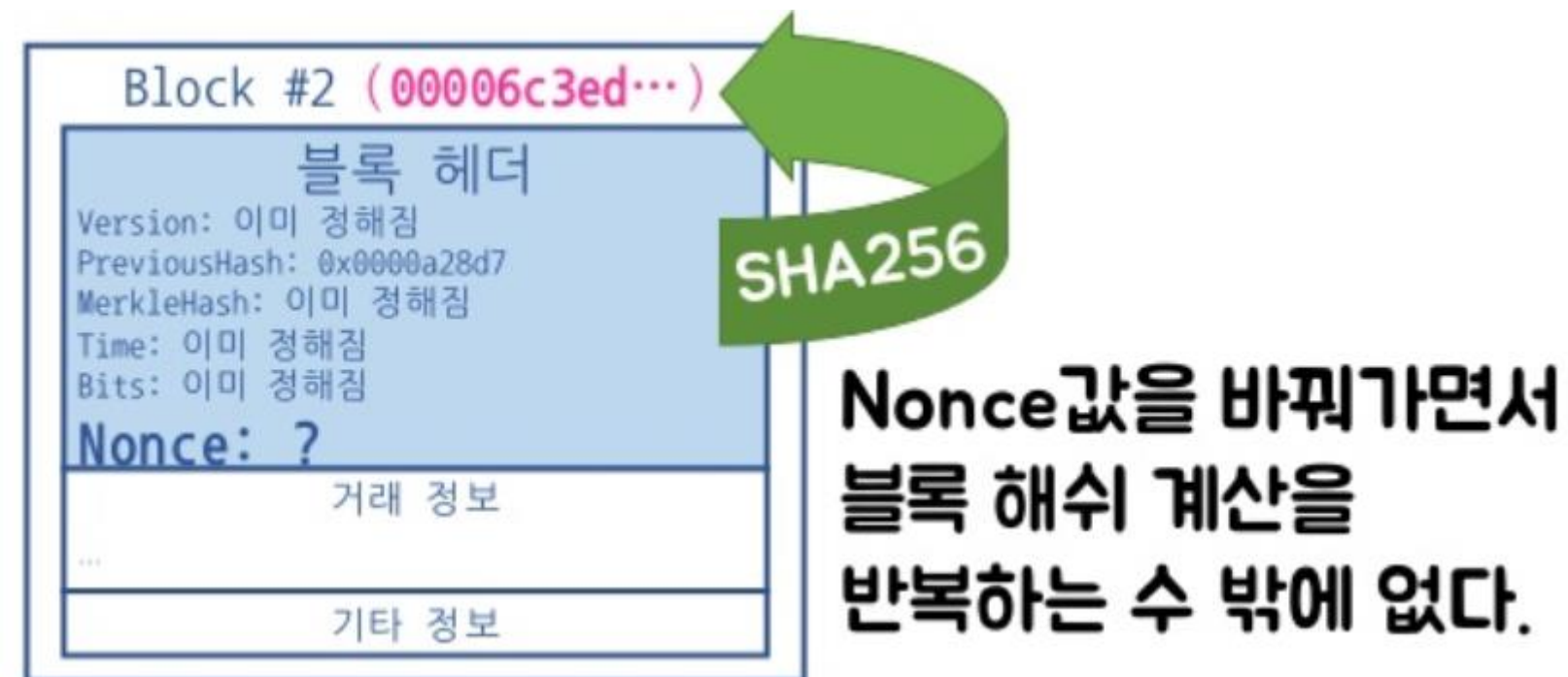
5. Bits : 난이도 조절용 수치

6. Nonce : 최초 0에서 시작하여 조건을 만족하는 해시값을 찾아낼 때까지 1씩 증가하는 계산 횟수

블록 헤더가 중요한 이유는 블록의 식별자 역할을 하는 블록 해시가 이 블록 헤더의 6가지 정보를 입력값으로 해서 구해지기 때문이다.

# 블록해쉬

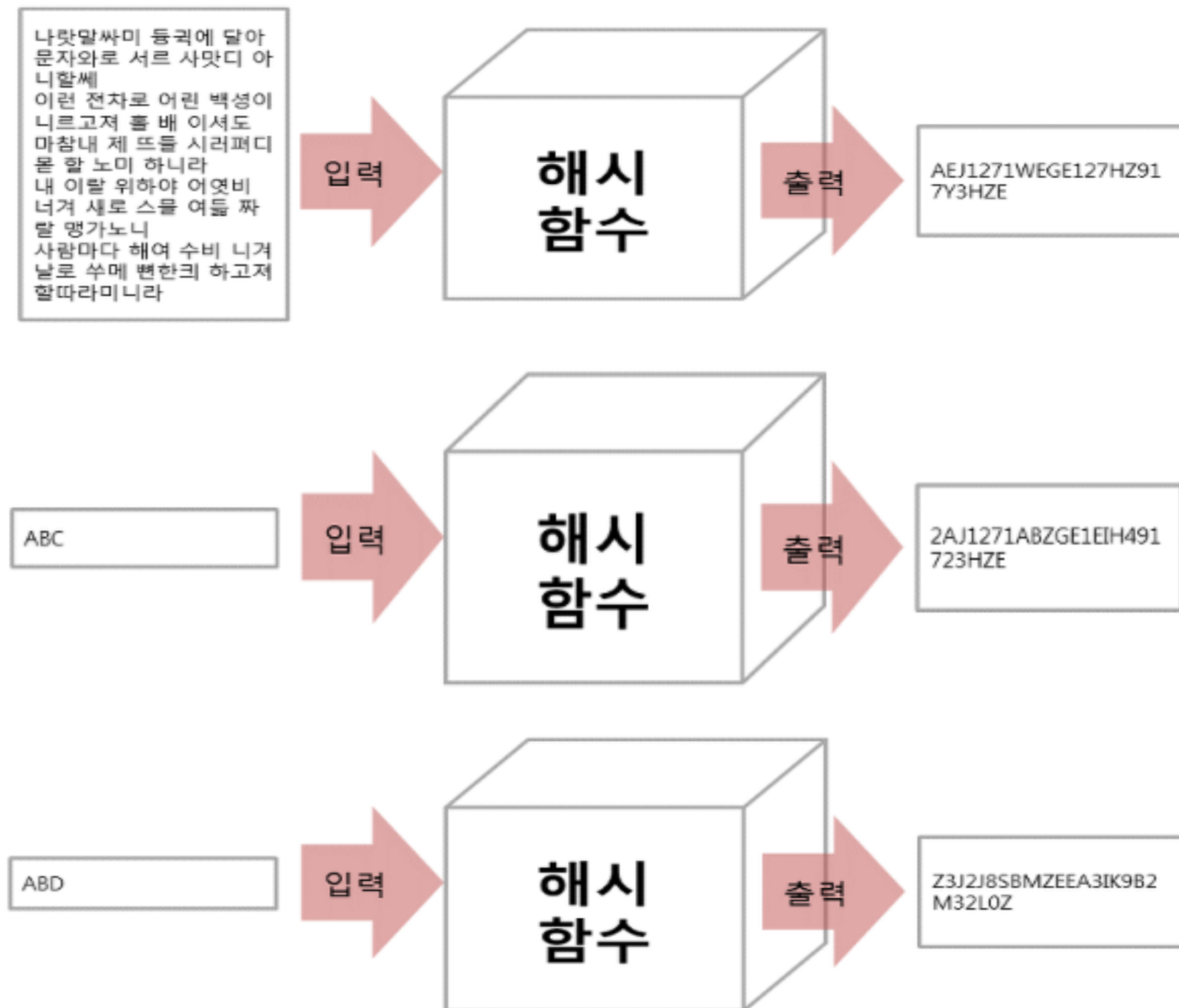
- 블록해쉬는 블록헤더를 해쉬함수로 계산한 값이며, 블록의 식별자 역할을 한다.
- 블록해쉬는 6가지 블록 헤더 정보를 입력값으로 하고, SHA256(Secure Hash Algorithm) 해쉬 함수를 적용해서 계산되는 32바이트의 숫자값이다.
- 블록헤더의 version, previousblockhash, merklehash, time, bits 5가지는 블록해쉬를 만드는 시점에서 확정되어 있는 값이지만 , nonce는 새로 구해야 하는 값이다. 이 nonce 값을 구해서 최종적으로 블록해쉬값을 구하고, 이 블록 해쉬값을 식별자로 가지는 유효한 블록을 만들어 내는 것이 바로 작업증명(Proof of Work), 즉 채굴이다.



# 해쉬(Hash)

## 질문) 해쉬(Hash)란 무엇입니까?

- 해쉬함수 체험해 보기 <http://www.convertstring.com/ko/Hash/SHA256>



- 긴 문장을 해쉬함수에 입력을 시켜보니, 약 20글자 정도의 의미를 알 수 없는 문자+숫자로 결과를 출력하였습니다. 값을 축약시켜 준다.
- 매우 짧은 단어를 입력을 시켜도, 약 20글자 정도의 의미를 알 수 없는 문자 + 숫자의 조합이 결과로 출력됨. 값을 변경해 준다.
- ABC와 ABD는 한 글자 차이가 나지만 결과는 완전히 다른 글자가 출력됨 . 약간 다른 입력 값으로 출력도 비슷해지겠지 생각을 하지만 **완전히 예상을 못하는 난수 값을 출력하는 것이 바로 해쉬입니다.**

# 해쉬(Hash)

---

- 해쉬의 특징

1. 어떤 입력 값에도 항상 고정된 길이의 해쉬 값을 출력한다.
2. 해쉬를 사용하는 목적은 데이터의 무결성을 제공하기 위해 사용한다.
3. 입력 값의 아주 일부만 변경 되어도 전혀 다른 결과 값을 출력한다. (눈사태 효과)
4. 출력값으로 입력값을 예측할 수 없다. 역 추적이 안 된다는 것은 " 단방향 "으로 만 되어 있다는 것이다.
5. 같은 입력 값은 항상 동일한 해쉬 값을 출력한다.
6. 해쉬 때문에 블록체인에서 적은 양의 데이터 값으로 수많은 양의 데이터 비교가 가능해졌다.

- **블록체인에서 해쉬함수를 사용하는 이유는 입력 메시지에 대한 변경할 수 없는 증거값을 뽑아냄 으로서 메시지의 오류나 변조를 탐지할 수 있는 무결성을 제공하기 위해서 입니다.**



# 논스(Nonce)

---

## 질문) 논스(Nonce)란 무엇입니까?

- 블록을 대표하는 해쉬값인 블록해쉬를 생성하려면 일정한 조건을 만족해야 한다. 일정한 조건이란 블록의 난이도에 따라 자동으로 설정된 '목표값' 보다 작은 블록 해쉬값을 찾아야 한다는 제약조건이다. 해쉬는 랜덤하게 생성되기 때문에, 수없이 많은 연산을 반복해서 미리 정해진 목표값 이하의 해쉬값이 나오도록 해야 한다. 이때 랜덤한 해쉬값을 생성할 수 있도록 매번 임시값을 사용해야 하는데, 그 임시값이 바로 논스(Nonce)이다.
- 논스 값을 수없이 바꾸어 가면서 하나씩 대입하다가 새로 생성된 해쉬값이 일정한 목표값보다 더 작을 경우에 새로운 블록이 성공적으로 생성된다.
- 특정한 블록에 대해 목표값 이하의 크기를 가진 해쉬값을 생성하는 논스값을 찾음으로써 새로운 블록을 생성하는 행위를 작업증명(PoW)이라고 한다. 작업 증명의 대가로 일정한 개수의 암호 화폐를 지급 받는 것을 채굴 또는 마이닝이라고 한다.
- 결국 채굴을 통한 작업증명 과정은 목표값 이하의 블록해쉬를 생성하는 '논스값' 을 찾는 행위이다.

# 논스(Nonce)

## 질문) 논스(Nonce)란 무엇입니까?

- 블록을 대표하는 해시값인 블록해시를 생성하려면 일정한 조건을 만족해야 한다. 일정한 조건이란 블록의 난이도에 따라 자동으로 설정된 '목표값' 보다 작은 블록 해시값을 찾아야 한다는 제약조건이다. 해시는 랜덤하게 생성되기 때문에, 수없이 많은 연산을 반복해서 미리 정해진 목표값 이하의 해시값이 나오도록 해야 한다. 이때 랜덤한 해시값을 생성할 수 있도록 매번 임시값을 사용해야 하는데, 그 임시값이 바로 논스(Nonce)이다.

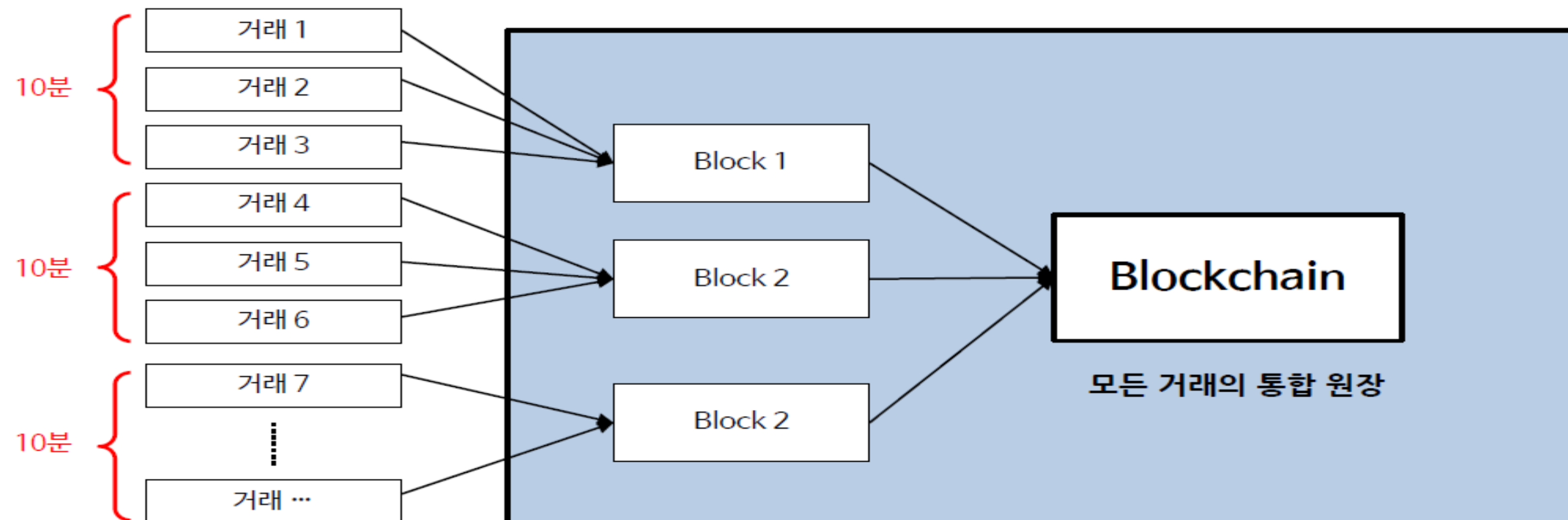
- 블록해시가 00000a84... 라는 특정값 보다 작게 나오는 하는 nonce 값을 구하는 과정을 나타내고 있다.



# 블록체인이란?

질문) 블록체인이란 무엇인가? 왜 "체인"이라는 단어를 사용하는가?

- 블록체인은 데이터 분산처리 기술입니다. 즉, 네트워크에 참여하는 모든 사용자가 모든 거래 내역 등의 데이터를 분산, 저장하는 기술을 지칭하는 말입니다. 블록들을 체인 형태로 묶은 형태이기 때문에 블록체인이라는 이름이 붙었죠. 블록체인에서 ‘블록’은 개인과 개인의 거래(P2P)의 데이터가 기록되는 장부가 됩니다. 모든 사용자가 거래내역을 보유하고 있어 거래 내역을 확인할 때는 모든 사용자가 보유한 장부를 대조하고 확인해야 합니다. 이 때문에 블록체인은 ‘공공거래장부’ 또는 ‘분산거래장부’로도 불리기도 합니다. 즉, 블록체인은 블록으로 이루어진 링크드 리스트이다.



# 노드

---

## 질문) 노드는 무엇입니까?

- 암호화 화폐의 거래에 있어서는 누구나 금융기관의 역할을 할 수 있다. 블록체인 기록을 다운로드 받으면 블록체인 P2P 네트워크의 일원으로서 이 기록을 가진 사람들이 ‘노드(Node)’라고 불리며, 각자가 금융기관의 역할을 하게 된다. 이 노드의 과반수 이상이 동의했을 때 기록으로서 영구적으로 장부에 기록된다.
- 블록이 생성되는 합의과정에서는 가장 많은 ‘작업(Work)’을 한 노드가 어느 기록이 참인지 결정할 수 있는 권한을 더 많이 가지게 된다. 이것이 작업증명(Proof of Work)이라는, 비트코인 창시자 나카모토 사토시가 고안한 블록 생성 방식이다.
- Global Bitcoin Nodes Distribution (<https://bitnodes.earn.com/>)

# 채굴

## 질문) 채굴(Mining)이란 무엇입니까?

- 작업증명을 통해 블록에 거래 내역을 정리해 주고, 그 보상으로 **코인**과 **거래 수수료**를 받게 되는데 이 과정을 “채굴(Mining)”이라고 합니다. 블록체인이 유지되기 위해 필요한 리소스를 제공하면, 이에 따르는 보상을 받는 개념이다.



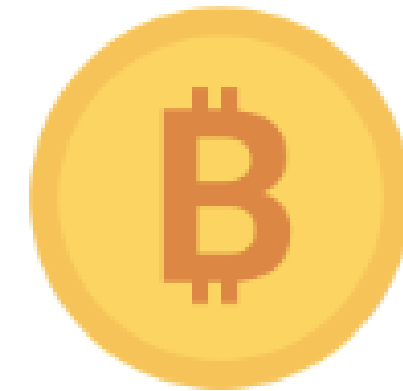
채굴 (Mining)

=



작업증명 (Pow)

+



보상 (Reward)

- 채굴기 AsicMinerMarket ( <https://asicminermarket.com/> )
- 채굴 소프트 웨어 5선 : ( <http://www.itworld.co.kr/news/108568> )

# 채굴

질문) 채굴(Mining)이란 무엇입니까?

- 나는 사토시 나카모토이다0 -----> 1a3523ba3510adbc2b3szc  
해쉬(hash)연산
- 나는 사토시 나카모토이다1 -----> 93151abed2bc15f3cca118
- 나는 사토시 나카모토이다2 -----> bc170d25bf9uac115db12d
- 나는 사토시 나카모토이다32151 -----> 00000d25bf9uac115db..

논스(nonce)

앞의 n 5자리가 0일 때 까지 논스(nonce)를

지속적으로 바꿔가면서 해쉬연산 수행

# 합의 알고리즘

---

질문) 합의 알고리즘(consensus algorism)은 무엇입니까?

- 중앙(centralized) 데이터 베이스  
: 신뢰 받는 중앙기관이 데이터의 정당성을 검증하고 이에 따라 데이터베이스를 업데이트
- 비허가성(permission-less) 분산화(distributed) 데이터 베이스  
: 누구나 허가 받지 않고 네트워크에 참가하게 되면, 도대체 누가 데이터베이스를 업데이트를 하지? 그리고 악의적인 공격으로 부터 데이터베이스를 어떻게 보호하지?
- 합의 알고리즘  
: 합의 알고리즘(consensus algorism)이란 다수의 참여자들이 통일된 의사결정을 하기 위해 사용하는 알고리즘이다.  
블록체인 시스템의 경우 네트워크에 참여하는 모든 참여자들이 동일한 데이터를 복사하여 분산 저장하기 때문에 원본과 사본의 구별이 없으며, 통일된 의사결정을 내릴 수 있는 권위 있는 중앙(center)이 존재하지 않는다. 이런 상황에서 합리적이고 효율적인 의사결정을 내릴 수 있는 다양한 알고리즘이 개발되었다.  
: 합의알고리즘 종류



# 작업증명

---

## 질문) 작업증명(POW Proof Of Work)이란 무엇입니까?

- 모든 블록체인을 이용한 암호화폐는 (거래)증명이 필요합니다. 장부가 곧 돈이기 때문에 장부상 거래를 확인 시켜주는 작업은 필수적입니다. 블록체인 시스템 내에서는 수 많은 노드들이 그 증명의 역할을 하고 있습니다.
- 작업증명은 암호 해독 능력인 컴퓨팅 파워를 이용해 함께 블록에 담기는 해쉬를 생성하기 위한 숫자값을 찾는 문제를 풀어 거래를 증명해주는 일입니다.
- 결국 작업증명이 블록체인과 비트코인이 가치 있고 안전 하도록 도와주는 역할을 합니다.

### ❖ 장점

- 1) 안전성이 검증되었다. 2) 높은 시장가치를 형성한다.(채굴하는데 드는 비용이 커지면서 자연스럽게 암호화폐의 가격이 올라간다) 3) 낮은 전송 수수료

### ❖ 단점

- 1) 채굴 난이도가 높아지면서 연산에 필요한 고사양 장비가 많이 필요하고, 과도한 전력 소모로 인한 에너지 낭비가 커진다. 2) 채굴난이도가 높아지면서 개인채굴자는 채굴에 참여하기 어렵다. (초기 장비 구매비용이 많이 들어서 진정한 민주주의 방식이라고 보기 어렵다.)

# 작업증명

---

## 질문) 작업증명(POW, Proof of Work)이란 무엇입니까?

- PoW는 해시함수에서 나온 출력값을 채굴자 들이 하드웨어 장비(GPU, CPU와 같은 컴퓨팅 파워)를 통해 결과를 도출하는 것입니다. 해시는 단방향 암호화 기술이므로 결과값을 가지고 입력값을 찾아낼 수 없습니다. 따라서 무차별 대입으로 출력값과 똑같은 결과가 나올 때 까지 실행하는 방법 밖에 없습니다. 이러한 방식으로 가장 빨리 채굴된 블록만 인정을 받고 나머지는 버려지게 되므로 이중지불 문제가 해결 됩니다.
- 의문점 : 거대 자본가가 슈퍼 컴퓨터를 구입하여 연산을 수행한다면 분산된 장부작성 방식이 아닌 중앙집권적 방식이 되지 않을까?  
=> 과반수의 해시파워를 가진 컴퓨터 (51% 공격)를 구매하는 건 엄청난 돈(약 2500억원)이 들어갑니다. 만약 천문학적인 돈을 투자하여 구매했다 하더라도, 거래가 위조되고 부당한 장부라고 느낀다면 블록체인의 가치가 급락할 것이기 때문에 정당한 방식으로 네트워크를 운영하는 것이 훨씬 큰 이득입니다.

# 51% 공격

---

## 질문) 51% 공격이란 무엇입니까?

- 51% 공격은 블록체인에 참여하는 51% 이상의 노드가 동시에 블록체인 거래내역을 조작하는 행위를 의미합니다.
- 비트코인 개념이 처음 등장할 때부터 이론적으로 존재했던 공격 방식이다. 다만 세계적으로 구축된 방대한 마이닝 풀로 인해 사실상 불가능한 기술로 여겨졌다. 당초 비트코인 창시자로 알려진 사토시 나카모토가 암호화폐를 통한 경제적 보상 시스템을 고안한 것도 51% 공격으로부터 블록체인 네트워크를 보호하기 위해서다.
- 하지만 지난 5월 비트코인골드, 라이트코인캐시, 모나코코인 등 일부 후발 암호화폐에 51% 공격으로 인한 피해가 발생하면서 위협이 현실화됐다. 해외 전문가에 의한 51% 공격 실증 연구도 이어지는 추세이다.
- 크립토51이 시가총액 상위 암호화폐를 대상으로 51% 공격 진행하는데 들어가는 비용을 공개했다.

( <https://www.crypto51.app/> )

# 지분증명

## 질문) 지분증명(POS, Proof of Stake)이란 무엇입니까?

- PoS는 채굴기 없이 본인이 소유한 코인의 지분으로 채굴되는 방식입니다. POW의 단점을 극복하기 위해 등장하였습니다.
- 해당 코인을 가지고 있는 소유자가 현재 보유하고 있는 자산(Stake) 양에 비례하여 블록을 생성할 권한을 더 많이 부여하는 방식입니다. 참여에 대한 보상은 이자와 같은 방식으로 코인이 지급된다.
- 시스템이 임의의 한 명을 random 선택하여 블록 생성 권한을 준다. (블록 생성을 위해 참여자가 임의의 값을 찾는 POW 알고리즘과의 큰 차이점이다.)
- 임의의 한명이 되기 위해서는(POS 채굴을 하기 위해서) 일정량 이상의 암호화 화폐를 소유하고 보증금처럼 Lock을 걸어둔 상태로 코인을 보관하고 있는 지갑(Wallet)을 블록체인 네트워크에 연결시켜 놓는다.

$$\frac{\text{내가 채굴에 참여한 암호화 화폐 개수}}{\text{채굴에 참여한 암호화 화폐 총 개수}} \times 100 = \text{채굴할 확률}$$

$$\frac{2,000}{10,000,000} \times 100 = 0.02\%$$

채굴에 총 참여한 총 개수가 10,000,000 개 일 때 한 참여자가 2000개로 채굴에 참여하면 2분마다(Qtum 기준) 0.02% 확률로 블록을 생성할 권한이 주어진다.

# 지분증명

---

## 질문) 지분증명(POS, Proof of Stake)이란 무엇입니까?

- POS 알고리즘 채굴자는 그 블록체인의 암호화 화폐를 소유한 사람이기 때문에 악의적인 블록을 생성해서 화폐의 가치를 하락시키지 않는다는 개념이다. 즉, 블록생성자와 지분 생성자의 이해관계를 일치 시킴으로서 블록을 나쁜 의도로 생성할 동기부여를 없애며 잘못 생성할 경우 패널티를 부여합니다.
- 이더리움은 PoW에서 PoS로 컨센서스 알고리즘(코인을 보유한 지분율에 따라 새롭게 생성하는 코인을 분배 받는 방식)을 변경할 예정 입니다.

- POW 보다 POS 가 더 안전하다고 하는 이유
  - ✓ PoW에서 51%의 해시파워를 가지는 비용 = 약 2500억원
  - ✓ PoS에서 전 세계 자산의 51% = 약 25조원

이렇게 100배 가량의 차이가 나타나기 때문에 PoS가 중앙 집권화가 더 어렵고 코인을 가진 노드 누구나 네트워크에 허가 없이 참여하기 때문에 오히려 더 분산화가 더 잘 된다고 볼 수 있고 더 많은 사람들이 의사결정 과정에 쉽게 참여할 수 있다.

# 지분증명

---

질문) 지분증명(POS, Proof of Stake)이란 무엇입니까?

❖ 장점

- 해시파워가 많이 필요하지 않아 경제적이며 친환경적이다.
- POW 알고리즘의 채굴 Pool 처럼 블록생성의 독점을 막을 수 있다.
- 블록을 생성하기 위해서는 지분을 담보로 잡아야 하기 때문에 덤핑을 방지한다.

❖ 단점

- 모두 이자를 받으려고 코인을 묶어 놓기 때문에 시중 코인의 유통량 감소로 이어질 수 있다.
- 기술에 검증이 확실하지 않기 때문에 안정성이 100% 검증되지 않았다.
- 코인을 많이 보유한 사람이 권력을 가지게 되는 구조 (부익부 빈익빈)

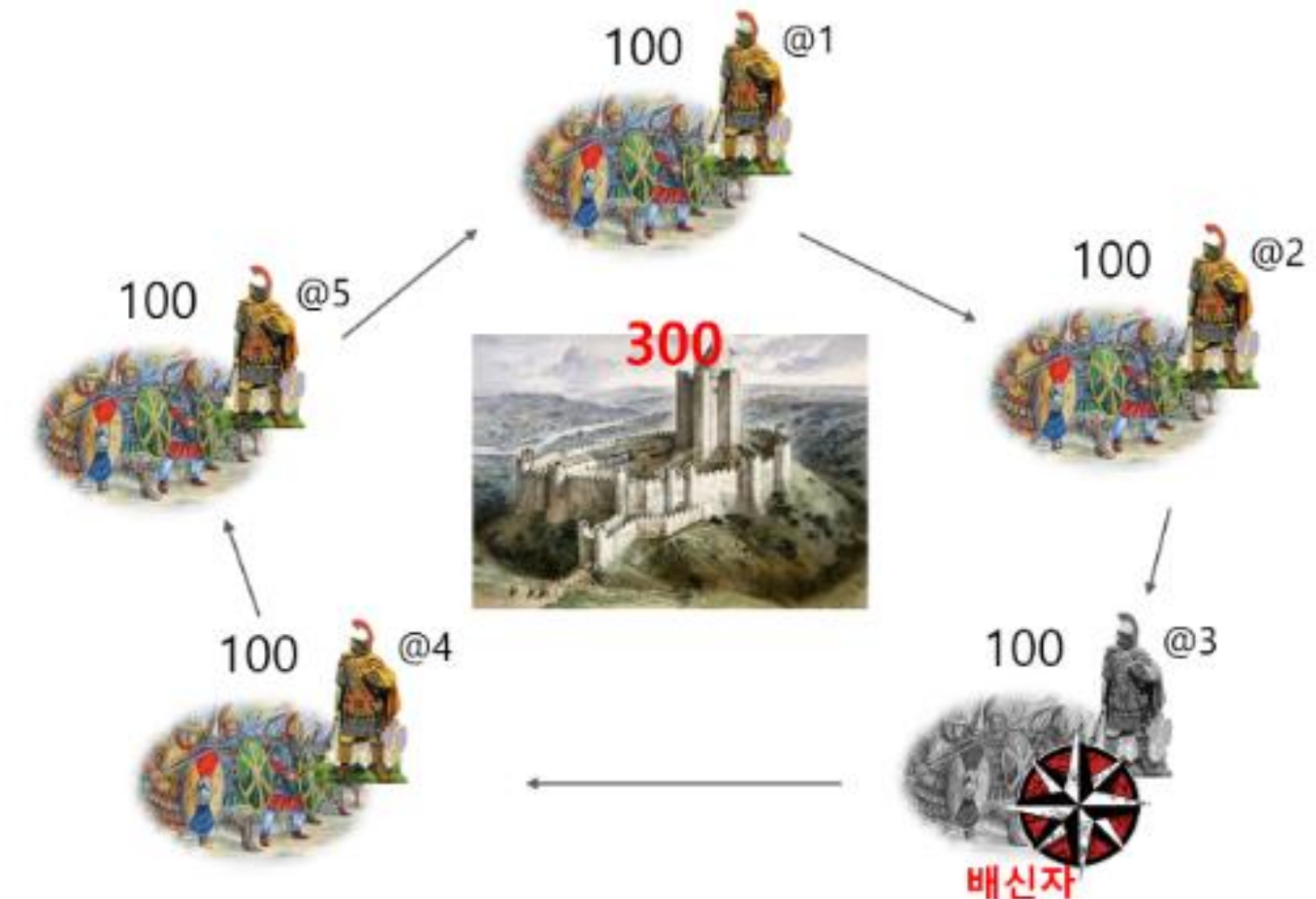
# 비잔틴 장군 문제

질문) 비잔틴 장군의 딜레마(Byzantine Generals Problem)는 무엇입니까?

비잔틴 장군 문제 = 분산 네트워크의 보안 문제

: 비잔틴 제국의 장군(general)들이 적군 도시를 에워싸고 공격하려 하는 상황. 장군들은 서로 메신저를 통해 의사 소통하며 적군 공격을 위해 공동의 계획에 (common plan of action)에 합의해야 한다.

: 하지만 장군들 중 몇 명은 배신자(traitor)일 수 있으며 계획의 합의를 방해할 수 있음. 따라서 비잔틴의 장군들은 배신자들이 어떠한 행동을 하든지 **공동의 계획에 합의 할 수 있는 알고리즘이 필요**하다.





# 비잔틴 장군 문제

---

## 질문) 비잔틴 장군의 딜레마는 무엇입니까?

비잔틴 장군의 문제를 PoW 를 통한 블록체인 기술로 해결 한 예시이다. 블록체인은 "이전 메시지를 포함" 시킨 새로운 메시지로 이해할 수 있으며, 포함된 이전 메시지를 변경한 경우 변경(위조)되었다는 사실을 바로 알 수 있는 장부이다. PoW 는 "10분의 시간을 들여 메시지를 만드는 과정" 이며, 정말 그러한 작업을 하였는지를 단번에 검증해낼 수 있는 방법을 제공한다.

그러나 10분의 시간을 들이지 않고는 절대 새로운 메시지에 포함된 이전 메시지를 다시 만들어낼 수는 없는 알고리즘이며, 비잔틴 장군의 문제 해결에 가장 핵심이 되는 메커니즘인 것이다.

# 이중 지불 문제

---

질문) 이중 지불 문제란 무엇인가? 비트코인은 이 문제를 어떻게 해결하는가?

- 이중 지불 문제는 말 그대로, 단일 화폐 단위가 두 번(이중) 결제되어 발생하는 문제입니다.
- 기존의 인터넷에서는 이중지불 문제를 해결할 수 없어, 인터넷으로 정보를 전달할 수는 있었지만 중개기관이 없이는 가치를 전달할 수 없었다. 하지만, 블록체인 기술을 이용할 경우 은행이라는 중개기관이 없이도 이중지불 문제 없이 가치를 전송할 수 있다. 즉, 블록체인은 이중지불 문제를 해결함으로써 암호화폐의 기반 기술이 되었다.
- 블록체인은 과거의 모든 거래내역이 담긴 장부를 해시함수로 변환하여 하나의 파일로 만들어 전송하고, 그 결과를 블록체인 네트워크에 참여한 다른 노드들이 검증하게 함으로써 이중지불 문제를 해결할 수 있게 되었다.

# 비트코인 획득

---

## 질문) 비트코인 획득 방법은?

- 비트코인 거래를 하기 위해서는 기본적으로 비트코인이 있어야 하며, 비트코인을 보유하고 있다는 의미는 누군가로 부터 비트코인을 수신 받았다는 의미가 됩니다.
- 비트코인을 획득할 수 있는 방법
  - 1) 비트코인 " 채굴 " 에 대한 보상으로 발행된 비트코인을 수령한다.
  - 2) 현물로 구입 후 유저에게 비트코인을 수령한다.
- 코인마켓캡 (<https://coinmarketcap.com/>)
- 국내 가상화폐 TOP3
  1. 업비트(<https://upbit.com/home>)
  2. 빗썸(<https://www.bithumb.com/>)
  3. 코인원(<https://coinone.co.kr/>)

# UTXO

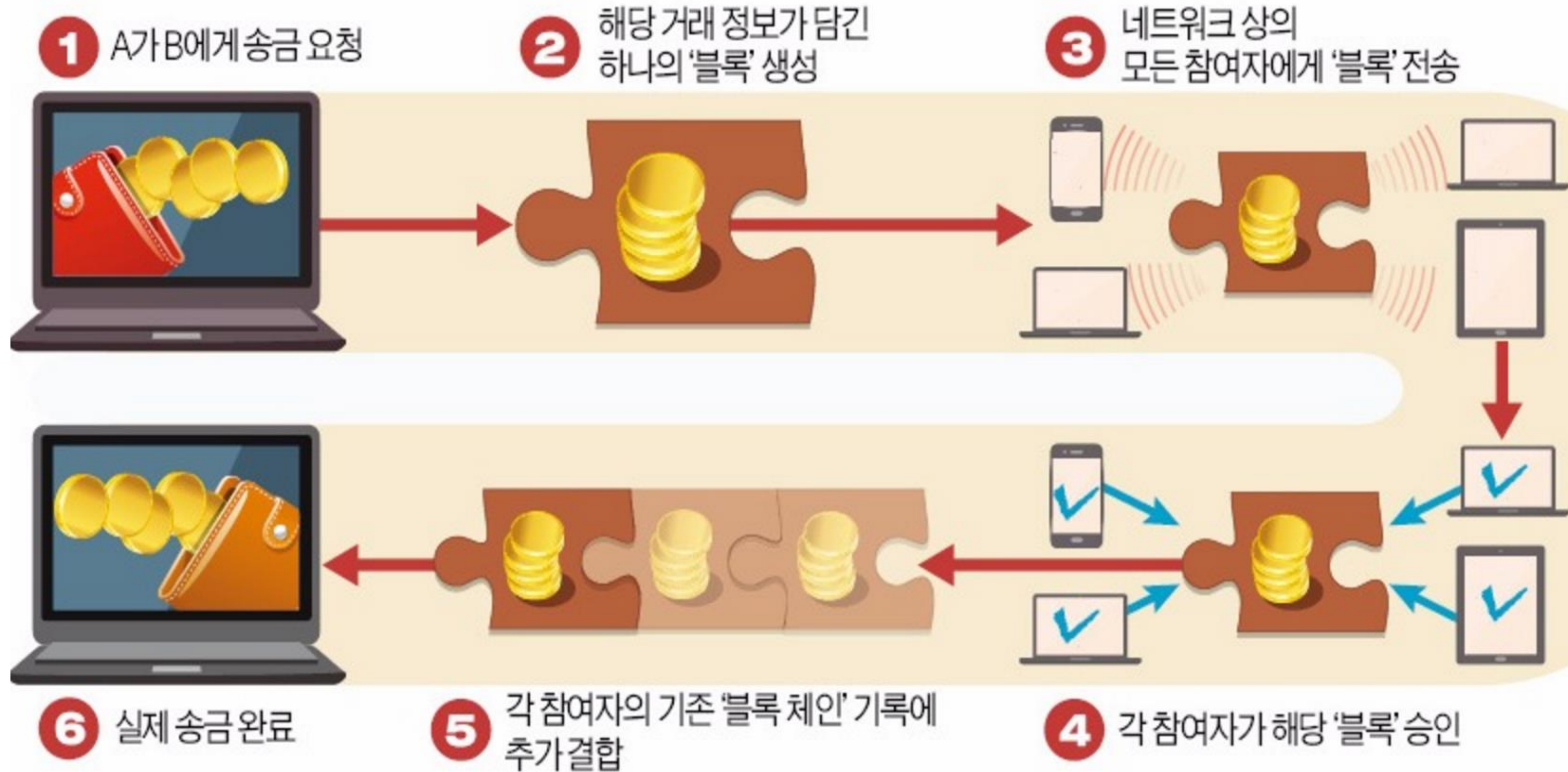
---

## 질문) UTXO란 무엇입니까?

- UTXO는 비트코인 거래의 기초 요소이다. UTXO(Unspent Transaction(Tx) Output)의 약자로 아직 사용하지 않는 거래의 출력값이다.
- 비트코인 거래에서 출력값이란 누구에게 얼마만큼의 비트코인을 보내는지에 대한 정보이다.
- UTXO는 결국 “받은 이가 아직 다른 이에게 보내지 않은 채 남겨둔 비트코인 뭉치”이다.
- 비트코인 이용자들은 따로 마련된 나의 계좌 또는 지갑에 비트코인이 예치되어 있다고 생각하지만 실제 비트코인 블록체인에서 나의 잔고는 별도의 계좌에 보관된 형태가 아니라 네트워크상에 UTXO 형태로 흩어져 있다.
- 수많은 UTXO 가운데 내 앞으로 보내진 UTXO를 찾아 모으면 나의 잔고가 된다

# 블록체인 기술의 적용

## ‘블록체인’ 기술의 적용 원리



## 2. 이더리움 플랫폼

<https://github.com/ethereum/wiki/wiki/%5BKorean%5D-White-Paper>

이더리움 한글 백서

# 플랫폼으로서의 이더리움

---

## 1. 플랫폼이란?

일반적으로 컴퓨팅 분야에서 플랫폼은 일련의 소프트웨어 프로그램을 작동 되게 하는 프레임워크를 말한다.  
보통 컴퓨터 아키텍처와 운영시스템 또는 개발 언어와 런타임 라이브러리 등으로 구성된다.

## 2. 이더리움 플랫폼

- 이더리움 플랫폼은 컨트랙트와 탈중앙화앱 (DApp)을 개발하고 이를 이더리움 네트워크상의 블록체인과 전용 브라우저를 통해 배포하고 활용함.
- 이더리움은 탈 중앙화 앱(DApp, Decentralized Applications)을 실행할 수 있는 플랫폼이다.
- P2P 네트워크 노드상의 EVM 운영체제
- 전용 브라우저와 각종 관리 및 유틸리티 서비스
- 블록체인을 통한 데이터와 코드 배포
- 암호화폐와 운영 토큰에 의한 생태계 운영 메카니즘 제공
- Ethereum (<https://www.ethereum.org/>)
- Ethereum Foundation(<https://www.ethereum.org/foundation>)





# 이더리움이란?

---

**블록체인이 “화폐”를 넘어서 무한한 가능성을 가지게 하자!**

- 이더리움은 2015년 출시된 차세대 스마트 계약 분산 응용프로그램 기술이다.
- 이더리움은 탈 중앙화 앱(DApp)을 실행할 수 있는 플랫폼이다
- 이더리움은 스마트 계약(Smart Contract)이라는 사용자가 제작한 어플리케이션을 구동할 수 있는 블록체인 기반의 플랫폼이다.
- 비트코인이 블록체인의 기술을 활용하여 화폐 거래를 위한 기능 위주로 구성이 되어 있는 반면, 이더리움 스마트 계약은 사용자의 요구에 따라 무한 확장 되는 어플리케이션 환경을 제공해 준다.
- 비트코인과 함께 대표적인 퍼블릭 블록체인 중 하나로 스위스의 비영리 단체인 이더리움 재단(Ethereum Foundation)에서 개발한 오픈소스 프로젝트이다.
- Solidity 등의 튜링 완전성(Turing-Completeness)을 갖춘 확장용 언어를 갖춰 스마트 계약을 쉽고 간단하게 프로그래밍 가능하다.
- 이더리움은 이더(Ether, ETH)라고 불리는 내부 화폐를 가지고 있습니다. 스마트 컨트랙트를 배포하거나 스마트 컨트랙트의 기능을 수행하기 위해서는 이더(ETH)가 필요합니다.



# 이더리움 관련 용어들

---

- 포크(fork)
- DApp(Decentralized Application)
- 스마트 컨트랙트 (Smart Contract)
- ICO(Initial Coin Offering)
- Ether
- ERC20(Ethereum Request for Comments)
- GAS
- 계정(Account)
- 트랜잭션(Transaction)
- EVM(Ethereum Virtual Machine)
- 지갑(Wallet)
- Solidity
- Remix

# 이더리움의 로드맵

---

## \* 이더리움의 로드맵

- 2013년 비탈릭 부테린 백서 발표
- 2014년 7~8월에 크라우드 세일(ICO. Initial Crowdfunding Offer)
- 2014년 7월 Frontier 출시 – 이더리움을 개발·채굴하고 네트워크를 형성하는 단계
- 2015년 8월 4일 프로토콜 업데이트
- 2016년 2월 Homestead 출시 – 이더리움 생태계를 구축하는 단계
- 2016년 11월 18일 Spurious Dragon 업데이트(내용 없는 블록제거)
- 2017년 10월 Metropolis 출시 – 이더리움 대중화를 위한 사회적 인프라가 형성되는 단계
- 2019년 2월 Serenity 출시 – 작업증명(POW)에서 지분증명(POS)로 바뀌게 됩니다.
  - 이더리움 2.0으로 부르기도 함
  - 이더리움 로드맵의 종착지
  - 샤드 추가
  - EVM을 웹 어셈블리 기반인 eWasm으로 변경할 계획

# 포크

---

## 질문) 소프트 포크와 하드포크란 무엇입니까?

: 소프트 포크와 하드포크는 모두 가상화폐(암호화폐)의 버전의 업그레이드를 말하는데 이전 버전과 호환이 가능하면 소프트 포크이고, 이전 버전과 호환이 불가능하면 하드 포크이다.

**소프트 포크**는 가상화폐의 버전이 업그레이드 되었을 때 개발자가 업데이트를 했어도 이용자는 업데이트를 하지 않아도 이용할 수 있다. 하지만 새로운 기능 등 기술적으로 여러 제한사항이 있어서 이용자 스스로가 업데이트를 하게 된다.

**하드포크**는 기존의 블록체인의 기능과 오류, 문제점을 수정하고 개선할 목적으로 기존의 블록체인과는 호환되지 않는 새로운 블록체인에서 다른 종류의 가상화폐를 만드는 것이다. 하드포크를 적용하게 된다면 이전 버전의 블록체인을 사용할 수 없게 되어서 개발 및 채굴하던 사용자의 다수가 업그레이드를 찬성해야 적용이 가능하다.

실제에는, 비트코인에서 하드포크가 된 비트코인 캐쉬, 이더리움에서 하드포크된 이더리움 클래식이 있다.

# 이더리움의 하드포크

---

## \* 이더리움의 하드포크 일대기

: 이더리움은 총 7번의 하드포크를 진행함. 4번은 로드맵에 포함된 하드포크였으며, 로드맵에 없지만 해킹, 서비스거부공격(DoS)과 같은 외부 상황에 대응하기 위해 3번의 추가적인 하드포크를 진행함.

### (1) 콘스탄티노플 하드포크(Constantinople hard fork)

: 원활한 거래를 위한 확장성 문제를 해결하고, 이더리움 채굴방식을 작업증명(POW)에서 지분증명(POS) 방식으로 전환하게 된다. 이더리움의 블록 보상을 3에서 2로 줄이기 위해 설정되어 있어 장기간 이더리움의 순환 공급을 줄이게 된다.

### (2) 비잔티움 하드포크(Byzantium hard fork)

: 프라이버시 보호를 위한 솔루션 도입(코인 거래자들의 주소와 금액을 모두 보이지 않게 함), EVM 성능 향상을 위한 업데이트(EVM이 트랜잭션을 병렬적으로 처리하도록 함), 난이도 폭탄 연기 및 채굴보상액 변경(블록당 채굴 보상을 5이더에서 3이더로 줄임)

# 이더리움의 하드포크

---

## \* 이더리움의 하드포크 일대기

### (3) 다오 포크 (DAO fork)

: 탈 중앙화 정신과 상통하는 분산형 자율 조직인 '다오(DAO, Decentralized Autonomous Organization)'를 출범시켜 1.5억 달러를 모으며 성공적인 펀딩을 하였지만, 해커들이 이더리움 환전 시 발생하는 취약점에 공격코드를 삽입해 DAO을 해킹, 이더리움을 탈취하는 사건이 일어납니다. 해킹 당한 물량과 피해자들을 보호하기 위하여 이더리움은 예정에 없던 포크를 실행하게 되었다.

### (4) 스푸리어드 드래곤 하드포크 (Spurious Dragon hard fork)

: 서비스 거부 공격(DoS)을 당하여 실시한 하드 포크입니다. 공격자는 스마트 계약 안에 특정 부호를 넣어 체인을 공격해, 다른 유저의 트랜잭션을 지연시키는 결과를 낳았습니다.

# 비트코인의 특징

---

## 1. 튜링 불완전성(Turing-incompleteness)

: 비트코인의 스크립트 언어는 '비교적' 단순해서 비트코인이 '**화폐**'로서만 작동하게 한다. 비트코인의 스크립팅 언어는 아주 단순하고 표현 방법이 많지 않은 언어이다.

이러한 특징을 비트코인의 '**튜링 불완전성**'라고 한다.

## 2. 상태표현제한(Lack of state)

: 비트코인의 UTXO(비트코인의 잔액 덩어리)가 표현할 수 있는 상태는 사용했거나 안 했거나 둘 중 하나이기 때문에 이 두 가지 상태 이외에 다른 어떤 조건에서 UTXO를 전부 사용하지 않고 나눠서 사용하는 계약을 할 수가 없다. 이를 '**상태표현제한**'이라고 한다.

## 3. Blockchain-blindness

: UTXO가 블록체인의 블록헤더 데이터들을 해독하지 못해서 화폐의 기능 이외의 다른 분야의 어플리케이션을 만드는 데 한계가 있는데 이를 'Blockchain-blindness'라고 한다.

**즉, 비트코인의 스크립트는 비트 코인이 전자계산기로서만 역할을 할 수 있는 정도로 제한적이고 단순하다.**

# 비트코인과 이더리움의 비교

질문) 비트코인과 이더리움의 가장 큰 차이점은 무엇입니까?

비트코인(Bitcoin, BTC)		이더리움(Ethereum, ETH)	
약어	BTC	약어	ETH
최초발행	2009년 1월	최초발행	2015년 7월
시가총액	167.9 조 (19.6.10기준)	시가총액	31.4 조 (19.6.10기준)
블록 생성주기	10분	블록 생성주기	약 12초
총 발행한도	21,000,000	총 발행한도	제한없음
합의 프로토콜	PoW	합의 프로토콜	PoW (PoS로 전환예정)

- 다른 다양한 코인들이 등장하기 전에는 비트코인이 사실상 유일한 암호화폐였으며, 그 이후에도 여전히 암호화폐 시장에서 독보적인 위치를 점하고 있다.
- 이더리움의 블록체인 네트워크에서 다양한 앱을 실행할 수 있다. 참여자들은 네트워크의 특정 주소에 앱을 올려둘 수 있고, 이 앱은 미리 설정해놓은 조건 하에서 특정한 명령어를 실행하게 된다. 그래서 비트코인에는 화폐를 보관하는 '지갑'이란 한 가지 주소만 있지만, 이더리움에는 화폐를 넣어놓는 '지갑'과 이런 앱을 저장하는 '계약서'라는 두 종류의 주소가 존재한다.

# DAPP

---

## 질문) DApp은 무엇입니까?

탈 중앙화 어플리케이션(Decentralized Application, DApp)

: 이더리움=OS , DAPP=어플 이라는 개념으로 생각하면 간단합니다.

이더리움은 DAPP를 위한 플랫폼입니다. 즉 DAPP는 이더리움 플랫폼 위에서 돌아가는 서비스 입니다.

현재 이더리움 플랫폼을 기반으로 만들어진 많은 DAPP이 있고, 아직도 수많은 DAPP가 개발되고 있습니다.

✓ Dapp 현황 (<https://www.stateofthedapps.com/>), DappRadar (<https://dappradar.com/>)

✓ 국내 아이콘 블록체인 플랫폼 (<https://icon.foundation/contents/dapp?lang=ko>)

: DApp의 종류

모든 정보를 블록체인에 기록하면 데이터를 안전하게 저장할 수 있습니다. 그러나 처리 속도가 느려질 수 있어 DApp은 데이터를 두 가지 경로로 나눠서 처리합니다.

✓ 블록체인 상에 데이터를 저장하는 온체인(On-Chain)

✓ 블록체인이 아닌 앱의 중앙 서버에 데이터를 저장하는 오프체인(off-chain)



## 질문) DApp이 실생활에 쓰일 수 있을까요?

- 아직까지는 우리에게 익숙하지 않은 DApp 들이 많습니다. 채굴, 거래소 등 대부분 개발자나 전문가들을 위한 DApp 들이었는데요. 아직 일반인이 실생활에 사용할 수 있는 다양한 DApp 들이 많이 출시된 상황은 아닙니다. 이더리움 기반 DApp 중 1위를 차지하고 있다는 아이덱스 조차 일일 사용자는 1500여 명에 불과합니다.
- 업계 관계자들은 구글 플레이 스토어나 애플 앱스토어처럼 사용자와 개발자를 이어질 수 있는 마땅한 통로가 없어 DApp 들이 초기 이용자 확보에 어려움을 겪고 있다고 지적합니다. 아직 DApp을 안정적으로 구동될 만큼 블록체인 기술이 충분히 개발되지 못했다는 점도 확장성이 떨어지는 데 일조합니다.
- 이를 위해 애플 앱 스토어처럼 DApp 스토어를 목표로 한 블록체인 프로젝트들도 등장하고 있는 추세입니다. 블록체인 네트워크 처리 속도를 높이기 위한 다양한 시도들도 이어지고 있습니다.

이러한 문제가 해결된다면 곧 DApp 전성시대가 도래하지 않을까요?

# 스마트 컨트랙트

---

## 질문) 스마트 컨트랙트(Smart Contract)는 무엇입니까?

- 스마트 컨트랙트란 블록체인 위에 “약속”을 올려 놓는 것이다. 어떤 조건을 만족하면 어떤 행동을 하라는 코드를 거래에 추가한다.
- 블록체인이 이 계약의 진위성과 실행을 보장한다. 단순한 약속이 아니라 “계약”이 된다. 블록체인에 기록되면 그 내용을 누구도 부정하거나, 위조할 수 없기 때문이다. 법이 아니라, 블록체인이 보증하는 계약이다.
- 블록체인 내부에 기록된 코드에 의해서 특정 조건을 만족할 경우, 정해진 행동이 자동으로 실행된다.
- Solidity언어로 스마트 컨트랙트를 작성하여 거래에 포함시키면, 블록체인 내부에 영구히 저장된다.
- 스마트 컨트랙트를 활용하면, 블록체인 위에서 단순히 돈을 주고 받는 것 이상의 수많은 응용이 가능해지며 제 3자의 개입이 필요 없는 서비스 어플리케이션을 만드는 기초가 된다.

## 질문) ICO(Initial Coin Offering)는 무엇입니까?

- 유가증권시장의 IPO(Initial Public Offering) 개념과 비슷하다고 할 수 있는 가상화폐의 ICO는 블록체인 기술과 가상화폐 기반 프로젝트를 추진하는 사업자가 초기 자금 조달을 목적으로 가상화폐 코인을 발행하고 투자자들에게 선 판매하여 자금을 확보하는 방식을 의미한다. 투자금은 비트코인이나 이더리움, 달러로만 받기 때문에 국적에 상관없이 전 세계 투자자들이 쉽게 참여가 가능한 것이 특징이다.
- DApp을 개발하는 데 자금이 필요하다. 이더리움은 각 DApp들이 자체적으로 사용하는 토큰(앱코인)을 발행할 수 있게 해준다. 토큰을 발행해서 판매하는 방식으로 자금문제를 해결한다. DApp이 토큰을 만들어서 공개적으로 판매하는 것을 ICO(Initial Coin Offering)이라고 한다.
- ICO 평가 (<https://icorating.com/ko/>)
- ICO Rating platform(<https://icobench.com/>)

## 질문) 이더(Ether)는 무엇입니까?





: 이더리움의 기초화폐이다. 이더(ETHER)는 서로 돈을 주고 받을 수 있는 화폐의 기능을 한다. 또한 이더는 채굴자들이 스마트 컨트랙트를 실행시키는 데 필요한 수수료의 지불 수단이 된다. 비트코인의 경우 채굴자들에게 네트워크를 유지하는 데 기여한 대가로 BTC를 준다. 이더리움도 PoW(Proof-of-Work)방식을 사용해 블록을 생성하기 때문에 채굴자들에게 보상을 제공하는데 이 수수료를 지불하는 수단이 이더(ETHER)이다.

: 원화가 십 원, 천 원과 같은 액면가로 나누어 집니다. 기존 화폐처럼 이더리움 네트워크의 통화인 이더도 액면가로 나누어 집니다.

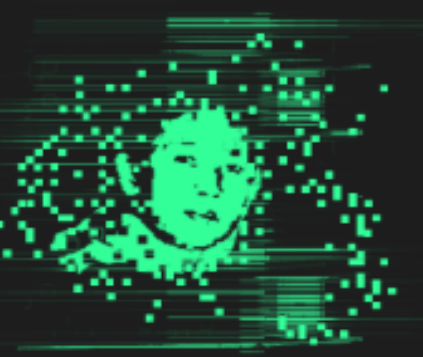



: 1원이 원화의 가장 작은 단위인 것 처럼 이더의 가장 작은 단위는 wei입니다. 1wei가 1이더가 되기 위해서는 엄청나게 많은 wei가 필요합니다. 정확하게  $10^{18}$  wei가 필요합니다.

# 이더리움 액면가

질문) 이더리움 액면가에는 어떤 것들이 있습니까? ( <https://gwei.io/kr/> )

			
<b>WEI</b>	<b>GWEI</b>	<b>PWEI</b>	<b>ETHER</b>
기본적으로 인식할 수 없을 정도의 디지털 먼지 같은 존재입니다. 보통 기술적인 경우나 코드 작성에만 사용됩니다.	가장 일반적으로 가스 (네트워크 거래 수수료)에 사용됩니다. 가장 많이 쓰는 단위 중 하나입니다.	나노보다는 크지만 ETH만큼 큰 단위는 아닙니다.	가장 일반적인 액면가입니다. 실질적인 거래의 대부분은 ETH의 관점에서 생각합니다.

			
<b>&lt;WEI&gt;</b> <b>WEI</b>	<b>&lt;SHANNON&gt;</b> <b>GWEI</b>	<b>&lt;FINNEY&gt;</b> <b>PWEI</b>	<b>&lt;BUTERIN&gt;</b> <b>Ether</b>
Wei Dai. 모든 현대 암호화폐 개념을 공식화한 사람	Claude Shannon. 정보 이론의 아버지. 코드 브레이커 및 암호 분석 전문가.	Hal Finney. 싸이어 펑크의 대부이며 사토시로부터 처음으로 비트코인을 받은 사람.	비탈릭 부테린. 이더리움 제작자

# ERC20

---

## 질문) ERC20 토큰은 무엇입니까?

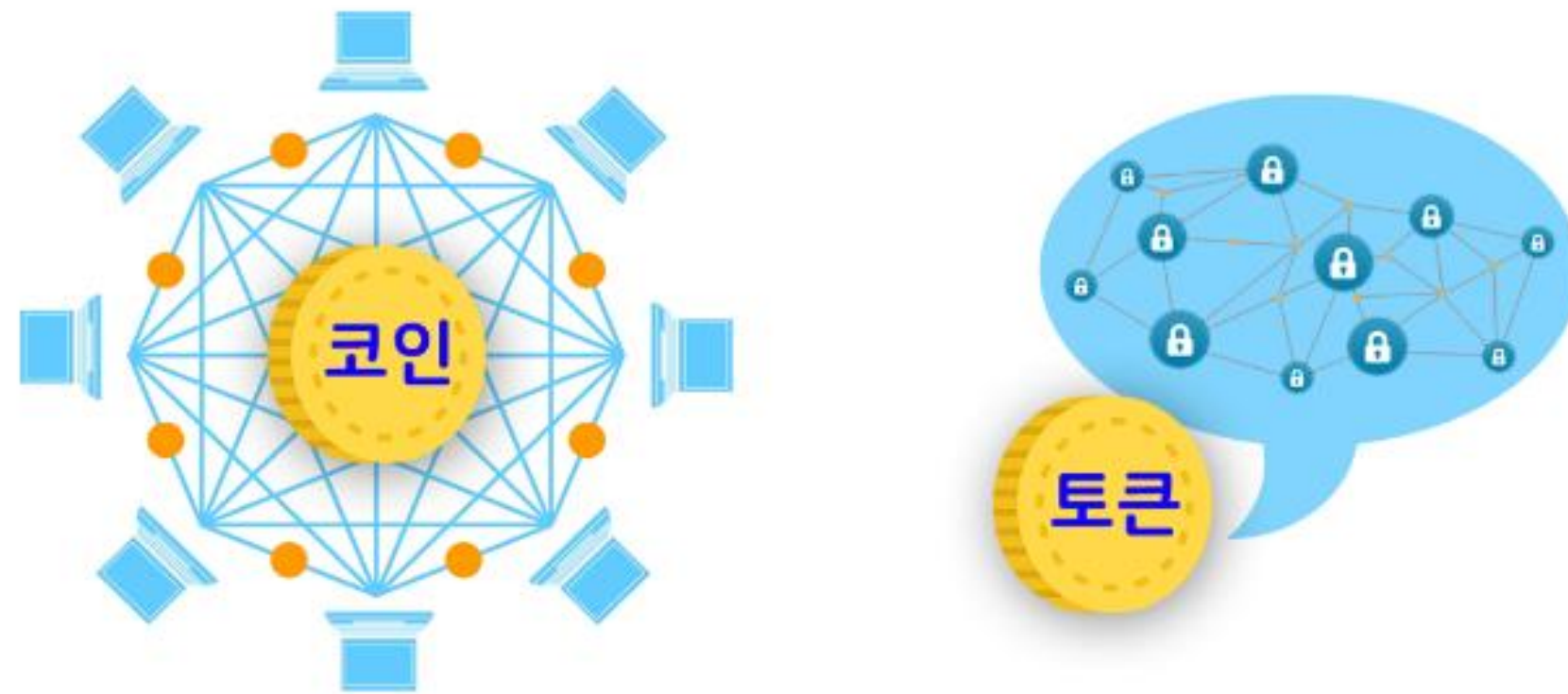
- 표준을 제안하고자 만든 것이 ERC20 입니다. 이더리움 플랫폼에서 표준을 제안하기 위해 개발자는 EIP(Ethereum Improvement Proposal 이더리움 개선 제안서)를 제출합니다. 이것이 승인을 받게 되면 ERC가 되는 것입니다. ERC20는 Ethereum Request for Comments의 줄임말입니다. 이더리움 제안서에 따른 답변이라고 보면 됩니다. 20은 요청에 부여된 번호입니다.
- ERC20는 누군가가 자신만의 토큰을 만들 때 기준 혹은 가이드라인이 됩니다. ERC-20 토큰은 개발자들이 작업하기에 상대적으로 쉽기 때문에 많은 암호화폐 프로젝트들의 ICO 표준으로 선택되었습니다.
- 2018년도 7월달 기준으로 ERC20를 표준으로 하는 토큰들은 100,000개나 있습니다. 대표적으로는 EOS, Filecoin, Bancor, Qash, Bankex 등이 있습니다. 그러므로 ERC20를 기준으로 삼았다고 하면 그만큼 상용성이 높아지는 것입니다.
- ERC20 Token Standard ( [https://theethereum.wiki/w/index.php/ERC20\\_Token\\_Standard](https://theethereum.wiki/w/index.php/ERC20_Token_Standard) )
- ERC20 Tokens List ( <https://eidoo.io/erc20-tokens-list/> )



# ERC20

## 질문) 토큰과 코인의 차이점은 무엇입니까?

- 토큰과 코인의 차이를 알기 위해서는 메인넷을 먼저 알아야 합니다.
- 메인넷은 기존에 존재하는 플랫폼에 종속되어 있지 않고, 독립적으로 생태계를 구성하는 것



- 자체 프로토콜인 메인넷을 보유하고 있다면 '코인'(비트코인, 이더리움, EOS, 리플, 스텔라 등),
- 다른 플랫폼에서 파생되어 만들어진 것을 '토큰'(Tether, OmiseGo, Augur 등)이라고 하죠.
- Top100 Coins <https://coinmarketcap.com/coins/>
- Top100 Tokens <https://coinmarketcap.com/tokens/>

# GAS

---

## 질문) 가스(GAS)는 무엇입니까?

- 이더는 변동성이 있기 때문에 트랜잭션 비용으로 직접 사용하지 않고 GAS를 사용한다. 거래(Transaction)을 생성하는 사람은 이더로 GAS를 사서 거래에 첨부해야 한다. 이 GAS는 이더리움에서 스마트 컨트랙트를 실행시키는 대가로 지불하는 연료이다. GAS는 스마트 컨트랙트의 코드가 복잡할 수록, 저장 공간을 많이 쓸수록 더 많이 지불해야 한다.
- GAS는 이더리움 블록체인의 적합성과 안정성을 유지시키기 위한 토큰입니다. 트랜잭션의 수행을 위해서는 Gas Limit \* Gas Price로 가격이 정해집니다. <https://ethgasstation.info/>
- 기본적으로 GAS 가격이 높아지면 트랜잭션이 빨리 처리가 됩니다. 이더리움 블록체인의 채굴자가 GAS 가격이 높은 것을 우선순위로 처리하기 때문입니다.
- 많은 트랜잭션이 일어나면 GAS 가격도 만만치 않게 되기 때문에 실제 DApp 개발 시 적정 GAS 가격으로 개발해야 합니다.
- 작업 코드에 따른 가스 비용 리스트 (<https://docs.google.com/spreadsheets/d/1m89CVujrQe5LAFJ8-YAUCcNK950dUzMQPMJBxRtGCqs/edit#gid=0>)



# GAS 비용

---

## 질문) 가스(GAS) 비용은 어떻게 계산하나요?

- 기본 GAS 비용 계산

기본적으로 21000Gas를 소모한다. 이는 Won으로 따지면 약 10원 정도이다.(ETH 50만원 기준)

가스(Gas) 라고 하는 것의 Ether의 단위 중 하나인 Gwei를 의미한다. 이더의 최소 단위는 소수점이 18인 wei이고, Gwei는 1,000,000wei 즉 백만 wei이다.

Ether랑 비교하자면,  $0.1 \times 10^{-9} \text{ ether} = 1 \text{ Gwei}$  이다. 21000 Gas는 21000 Gwei 이고, 즉, 0.000000021 이더이다. 여기에 현 시세를 곱하면 기본 가스비가 측정된다.

- 트랜잭션 실행자는 계산된 GAS로 트랜잭션을 실행 시키고 계산된 GAS는 채굴자에게 갑니다. 그리고 남은 가스는 다시 트랜잭션 실행자에게 되돌아갑니다. 즉 이더리움 블록체인은 위에 계산된 Gas 가격에 초과하지 않는 선에서 트랜잭션을 수행합니다.

- 현재 가스 적정 가격 및 현황 정보는 이 사이트에서 실시간으로 확인할 수 있습니다.

(<https://ethgasstation.info/>)

# GAS 비용

## GAS 비용 계산

- 가스 비용은 함수의 복잡도에 따라 결정된다.
- 연산에 소모되는 비용 == 오퍼코드(opcode)
- 네트워크 상태 컴퓨팅 자원에 따라 Gas Limit이 달라질 수 있음
- GAS 비용 = Gas Limit \* Gas Price (Ether)
- GWEI를 ETHER 로 Convert 하기

<https://www.etherchain.org/tools/unitConverter>

11 GWEI = 0.0000000011 Ether

- GAS 비용

$400455 * 0.0000000011 = 0.004405$

- ETH를 USD 로 환산 하기 <https://coinmarketcap.com/ko/converter/eth/usd/>
- USD를 KRW로 환산하기

<https://kr.investing.com/currencies/usd-krw-converter>

MetaMask Notification

CONFIRM TRANSACTION Ropsten Test Net

Account 2  
1E06F1...6211  
3.410 ETH  
1935.70 USD

83E565...a335

Amount 0.020000 ETH  
11.35 USD

Gas Limit 400455 UNITS

Gas Price 11 GWEI

Max Transaction Fee 0.004405 ETH  
2.50 USD

Max Total 0.024405 ETH  
13.85 USD

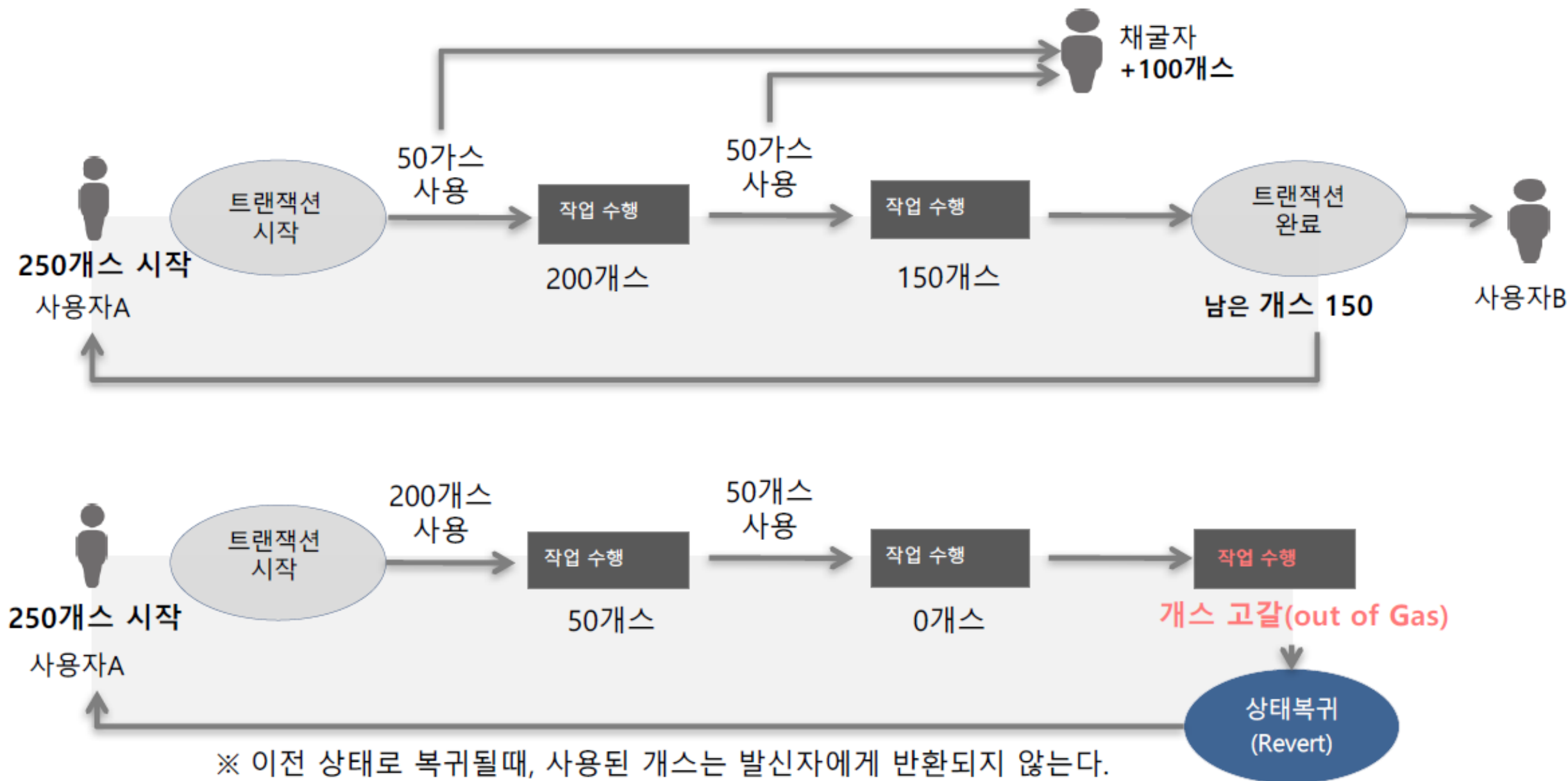
Data included: 132 bytes

RESET SUBMIT REJECT

# 트랜잭션 수행비용

질문) 트랜잭션 수행 비용은 어떻게 계산하나요?

- 트랜잭션 실행을 위한 기본 가스 비용 : 21,000 GAS 가 소모됨
- 한 블록의 최대 gasLimit 6,700,000



이더리움 = 이더 + 계좌 + 가스

# Account

## 질문) 계정(Account)란 무엇입니까?

외부소유계정 EOA (Externally Owned Accounts)	<ul style="list-style-type: none"><li>• 개인키(Private Key)로 관리되는 계정</li><li>• 이더를 주고 받을 수 있는 일반적인 은행 계좌와 비슷</li><li>• 트랜잭션을 전송할 수 있음(이더거래 혹은 스마트 코드 작동)</li><li>• 관련 코드를 가지고 있지 않음</li></ul>
계약계정 CA(Contracts Accounts)	<ul style="list-style-type: none"><li>• 개인키가 아니라 계약 코드에 의해서 관리되는 계정</li><li>• 계약계정은 외부 소유계정과 다르게 '계약코드'와 '계약에 필요한 정보들'을 저장하는 공간'을 가지고 있음 ( 관련 코드를 가지고 있음 )</li><li>• 코드실행은 다른 계약에 의해서 받는 메시지 또는 트랜잭션을 통해 작동이 시작됨</li></ul>

- 이더리움의 계약계정을 GAS를 구입해서 계약 코드를 걸어 놓으면 이 계약 코드의 조건이 만족될 때에만 계약 계정이 메시지를 받고 계약의 내용이 실행된다.
- <http://etherchain.org> 이나 <http://etherscan.io> 에 들어가면 계좌나 계약의 잔액을 알 수 있습니다.

# Transaction

---

## 질문) 트랜잭션의 개념과 구조는 어떻게 되나요?

- 트랜잭션(Transactions)은 외부 소유 계정(EOA)에서 생성되어 이더리움 블록체인에 기록된 서명된 메시지다. 트랜잭션을 통해서만 이더(Ether)를 전송하거나, 이더리움 가상 머신(EVM)에 있는 컨트랙트를 실행 할 수 있다.
- 트랜잭션의 구조
  - 1) nonce: EOA(Externally Owned Accounts)에 발급되는 트랜잭션 일련번호.
  - 2) From : 발신자 주소
  - 3) To : 수신자 주소
  - 4) Gas price: 가스 가격
  - 4) Gas limit: 가스의 최대 사용량
  - 6) value: 수신자에게 보내는 이더(ether) 개수
  - 7) data: 가변 길이의 바이너리 데이터(payload)
  - 8) v, r, s: ECDSA 서명 구성 요소

# Transaction

---

## 질문) 트랜잭션의 개념과 구조는 어떻게 되나요?

- 트랜잭션의 구조

1) nonce: EOA(Externally Owned Accounts)에 발급되는 트랜잭션 일련번호이며, 트랜잭션의 중복 전송을 방지하는데 사용된다.

2) gas price: 트랜잭션에 포함되는 gasPrice 필드는 트랜잭션 생성자가 가스 가격을 설정할 수 있다.

가스 가격은 GWei 단위를 사용한다. 이더리움 네트워크의 가스 정보는 [ethgasstation.info](https://ethgasstation.info)에서 확인

3) gas limit: gaslimit는 트랜잭션을 처리하는데 사용할 최대 가스 개수를 설정한다. 하나의 EOA에서 다른 EOA로 이더(ether)를 전송하는 트랜잭션에 필요한 가스량은 21,000개로 고정되어 있다. 따라서 이더 전송에 소비되는 수수료를 계산하려면 가스 가격에 21,000를 곱하면 된다.

트랜잭션 목적지가 컨트랙트 주소인 경우에는 필요한 가스량을 추정할 수는 있지만 정확하게 계산할 수는 없다. 컨트랙트 호출에 필요한 가스량은 컨트랙트 실행이 완료된 후에 결정된다.

4) to 필드: 수신자(도착지) 주소를 의미한다. 수신자 주소는 EOA 또는 컨트랙트 주소이다. 이더리움 네트워크에서는 주소를 검증하지 않는다. 따라서 잘못된 주소로 전송된 이더는 영원히 사용할 수 없게 된다.

# Transaction

---

## 질문) 트랜잭션의 개념과 구조는 어떻게 되나요?

- 트랜잭션의 구조

5) value와 data: 트랜잭션에서 가장 중요한 “payload”는 value와 data라는 두 개의 필드에 의해 사용된다. value를 포함하는 트랜잭션은 송금(payment)이라고 보면 된다. EOA 주소 또는 컨트랙트 주소로 value를 포함한 트랜잭션을 전송하면 해당 계정의 ether 잔액에 value가 추가된다.

트랜잭션에 data가 포함되어 있으면 이 트랜잭션은 컨트랙트 주소로 처리가 된다. 포함된 data는 EVM에 의해 함수 호출(function invocation)로 해석되고 data에 지정된 함수가 호출된다.

6) 이더리움은 디지털 서명에 ECDSA(Elliptic Curve Digital Signature Algorithm) 알고리즘을 사용한다.

ECDSA는 타원 곡선 개인/공개키를 기반으로 하는 디지털 서명에 사용하는 알고리즘이다.

디지털 서명은 이더 소유를 증명하고 부인 방지 하는데 사용된다.

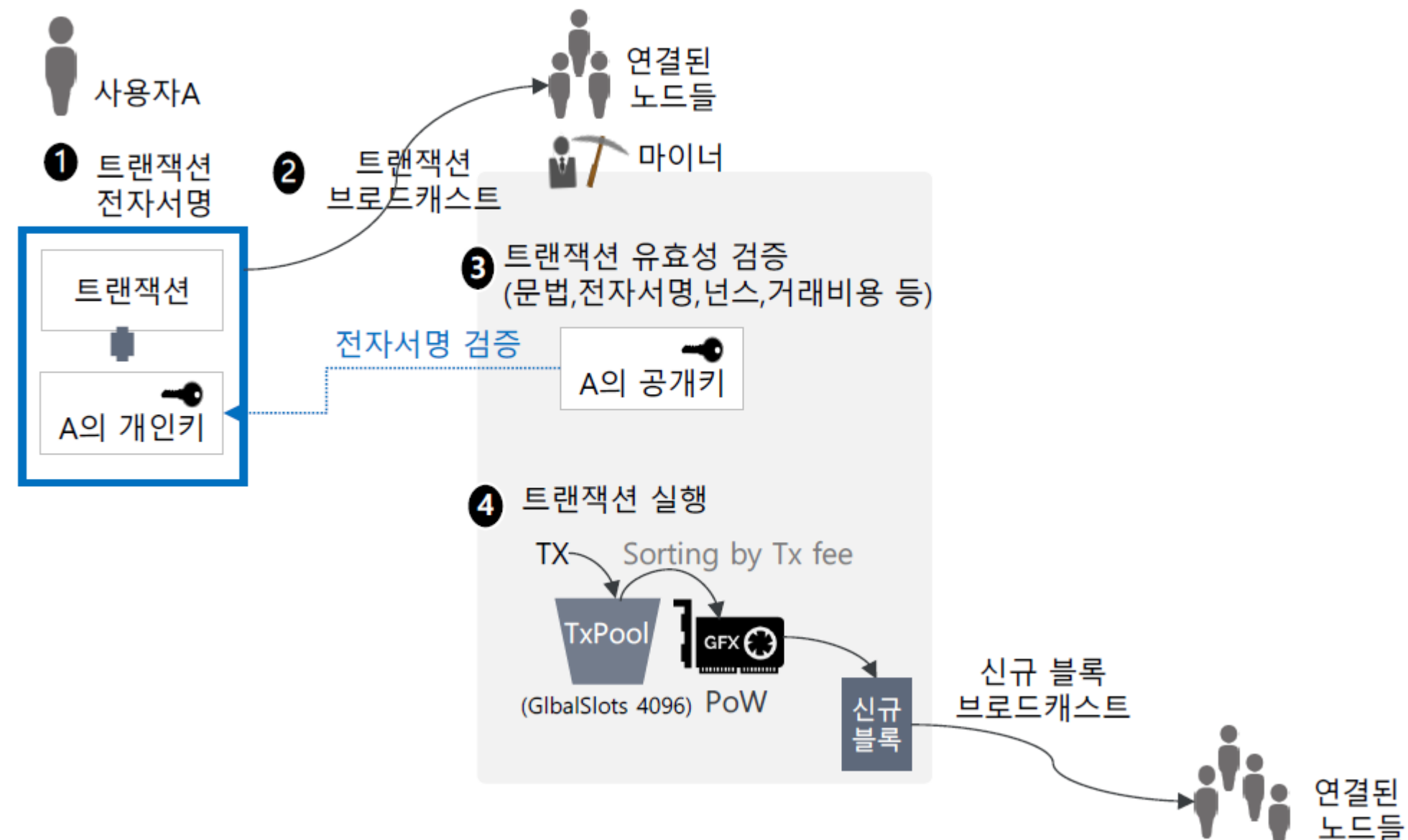
ECDSA 자세한 설명(<http://bit.ly/2r0HhGB>)



# Transaction

질문) 블록생성 할때 트랜잭션을 처리과정은 어떻게 되나요?

1. A는 트랜잭션 내역과 A의 공개 키를 가져와서 암호해싱을 한다. 그리고 자신(A)의 개인키를 사용해서 전자서명(Sign, ECDSA)을 한다.
2. 서명된 트랜잭션을 현재 연결되어 있는 전체 노드에 브로드캐스팅 한다.
3. 채굴자는 A의 공개키로 개인키를 해제하여 전자서명 검증 등 트랜잭션의 유효성을 검증하고 이상이 없으면 트랜잭션 풀에 이를 등록한다.
4. 이후 채굴자는 트랜잭션풀에서 가장 비싼 수행 대가의 트랜잭션을 꺼내서 EVM을 생성하여 실행하고 블록 생성 후 마이닝 작업을 하여 블록체인에 추가한다.

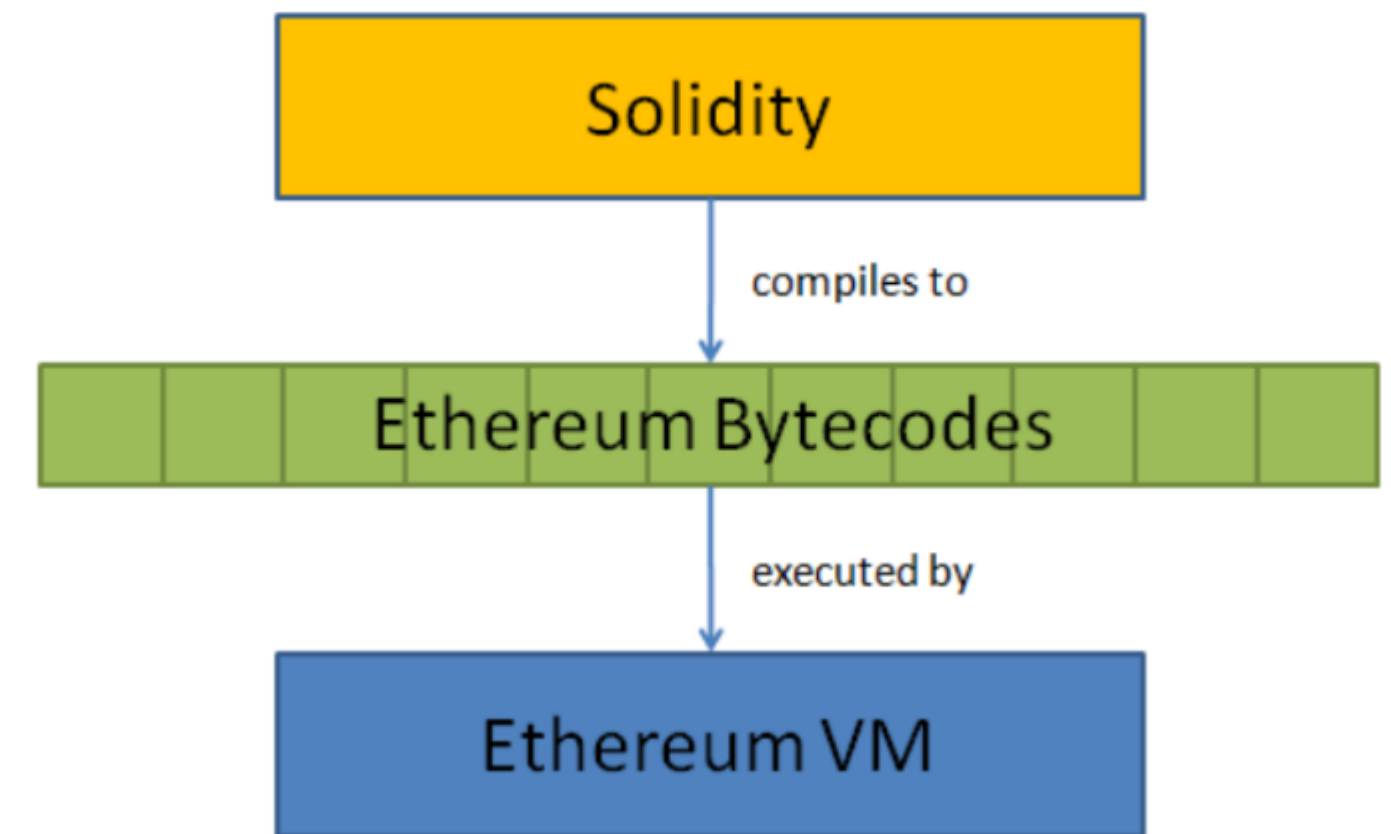




# EVM

## 질문) EVM(Ethereum Virtual Machine)란 무엇입니까?

- EVM은 Ethereum 블록체인 네트워크의 노드들이 공유하는 하나의 가상머신입니다.
- Solidity 코드가 컴파일 되어 바이트 코드가 만들어지고, 이 바이트 코드는 EVM이 실행합니다.
- EVM을 동작 시킬 수 있는 동력은 바로 이더리움 노드들입니다.  
즉, 채굴자 들이 EVM을 작동시키고 수수료를 취합니다.
- DApp 개발자는 코드를 작성 후 EVM에서 실행시킬 때 수수료를 지불하고, 채굴자는 완성된 코드를 EVM에 실행시켜 주면서 수수료를 받습니다.
- EVM은 GAS 개념 도입으로 트랜잭션을 생성할 때 GAS Limit을 정하고, EVM은 트랜잭션을 처리하면서 GAS를 소모합니다.  
GAS Limit으로 설정한 GAS가 모두 소모되면 EVM은 해당 트랜잭션을 종료 시키기 때문에 무한정 도는 Loop는 존재할 수 없습니다.



## 질문) EVM(Ethereum Virtual Machine)이 동작하는 순서는?

1. 트랜잭션이 올바른 형식인지 확인한다.
2. 트랜잭션 수수료를 계산( $\text{Gas Limit} * \text{Gas Price}$ )한다.
3. GAS 지불 초기화 이 시점부터 트랜잭션에서 처리된 바이트 만큼 특정량의 GAS를 차감한다.
4. 트랜잭션 금액을 수신 계정으로 보냄 (Smart Contract도 이 단계에서 실행됨)
5. 송신 계정에 트랜잭션을 완료할 수 있을 만큼 GAS Price가 충분하지 않으면 트랜잭션의 모든 변경 사항이 되돌려진다. 그러나, 트랜잭션 수수료는 채굴자에게 지불되고 환불되지 않음.
6. 5번의 경우와 다른 이유로 트랜잭션이 실패한 경우, 송신 계정에 GAS Price를 환불하고, 사용된 GAS와 관련된 비용은 채굴자에게 전달된다.

# 지갑(wallet)

---

## 질문) 암호화폐 지갑이란 무엇입니까?

: 암호화폐를 보관 및 관리를 할 수 있는 지갑은 암호화폐를 보내고 받을 수 있는 공개키와 개인키를 저장하고 있습니다. 공개키는 계좌번호이고 개인키는 그 계좌의 비밀번호라 생각하시면 됩니다.

현금 지갑에서는 현금이 지갑 안에 있지만, 암호화폐는 디지털 지갑에 저장되어 있지 않고 블록체인에 저장되어 있습니다.

- 핫 월렛(Hot Wallet)

핫 월렛이라는 온라인에 연결돼 있어 바로 입출금, 송금이 가능한 암호화폐 지갑을 의미한다. 언제나 연결되어 있어 편리하고 빠르지만, 해킹에 취약합니다.

- 콜드 월렛(Cold Wallet)

콜드 월렛은 오프라인에서 작동하는 지갑을 의미하고 콜드 월렛에는 하드웨어 지갑, USB보관, 종이 지갑등이 있다. 콜드 월렛은 인터넷에 연결되어 있지 않으므로, 해커가 그 지갑 자체를 소유하고 있지 않는 이상 해킹은 거의 불가능하다고 보면 된다. 다만 월렛에 해당되는 개인 키를 잃어버리면 복구할 방법이 없습니다.

# 지갑(wallet)

---

질문) 가장 인기있는 암호화폐 지갑들은 어떤 것들이 있습니까?

1. Ledger Nano S (Hardware Wallet)

Ledger Nano S는 가장 저렴한 Ethereum 하드웨어 지갑 중 하나입니다(\$ 65). 여기서 Ether은 장치에 오프라인으로 저장됩니다. Ledger는 Ether을 쓰고 싶을 때마다 장치에 저장된 개인 키를 사용하여 서명합니다. ETH와 ETC를 모두 저장할 수 있습니다.

2. Trezor (Hardware Wallet)

Trezor는 Bitcoin을 위해 개발된 최초의 하드웨어 지갑이었습니다. Trezor는 MyEtherWallet 웹 인터페이스를 사용하여 Ethereum에도 사용할 수 있습니다. 또한 암호로 로그인 할 때만 활성화 할 수 있는 안전한 전자 칩에 Ether을 오프라인으로 저장합니다.

3. Atomic (Desktop)

Atomic Wallet은 Ethereum 및 ERC20 토큰을 위한 최상의 솔루션입니다. 지갑을 사용하면 ETH를 은행 카드로 저장, 교환 및 구입할 수 있습니다.

# 지갑(wallet)

---

질문) 가장 인기있는 암호화폐 지갑들은 어떤 것들이 있습니까?

## 4. Mist (Desktop Wallet)

미스트(Mist)는 공식 이더리움 지갑입니다. 미스트(Mist)를 설치하면 모든 이더리움 노드와 동기화 되기 때문에 시작하는데 시간이 걸립니다. 동기화가 완료 되면 보안 암호를 설정하라는 메시지가 표시됩니다. 이 비밀번호를 잊어 버리면 다른 방법으로 미스트에 접속할 수 없기 때문에 이 비밀번호를 반드시 기억해 두어야 합니다. 이 지갑 안에서 한 쌍의 공용키와 개인키에 액세스 하여 거래를 수행할 수 있습니다.

## 5. MetaMask (Desktop Wallet)

메타마스크(MetaMask)는 이더리움 네트워크에 접속하는 브라우저와 같습니다. 이 지갑을 사용하면 분산 된 이더리움 앱에 접근 할 수 있습니다. 직관적으로 설계되어 있어 테스트 네트워크와 메인 이더리움 네트워크를 신속하게 전환할 수 있습니다. 개인키는 암호화되어 있으며 시스템에 저장되어 있으므로 언제든지 내보낼 수 있습니다.

# 지갑(wallet)

---

질문) 암호화폐 지갑의 종류에는 어떤 것들이 있습니까?

## 6. MyEtherWallet (Web Wallet)

MyEtherWallet은 다른 전통적인 웹 지갑과 다르게 Ethereum의 개인 키를 제어 할 수 있습니다. 또한 스마트 계약을 작성하고 접근 할 수 있습니다. Trezor 또는 Ledger Nano S를 연결하여 MyEthereum의 브라우저 환경에서 자금을 이용할 수도 있습니다.

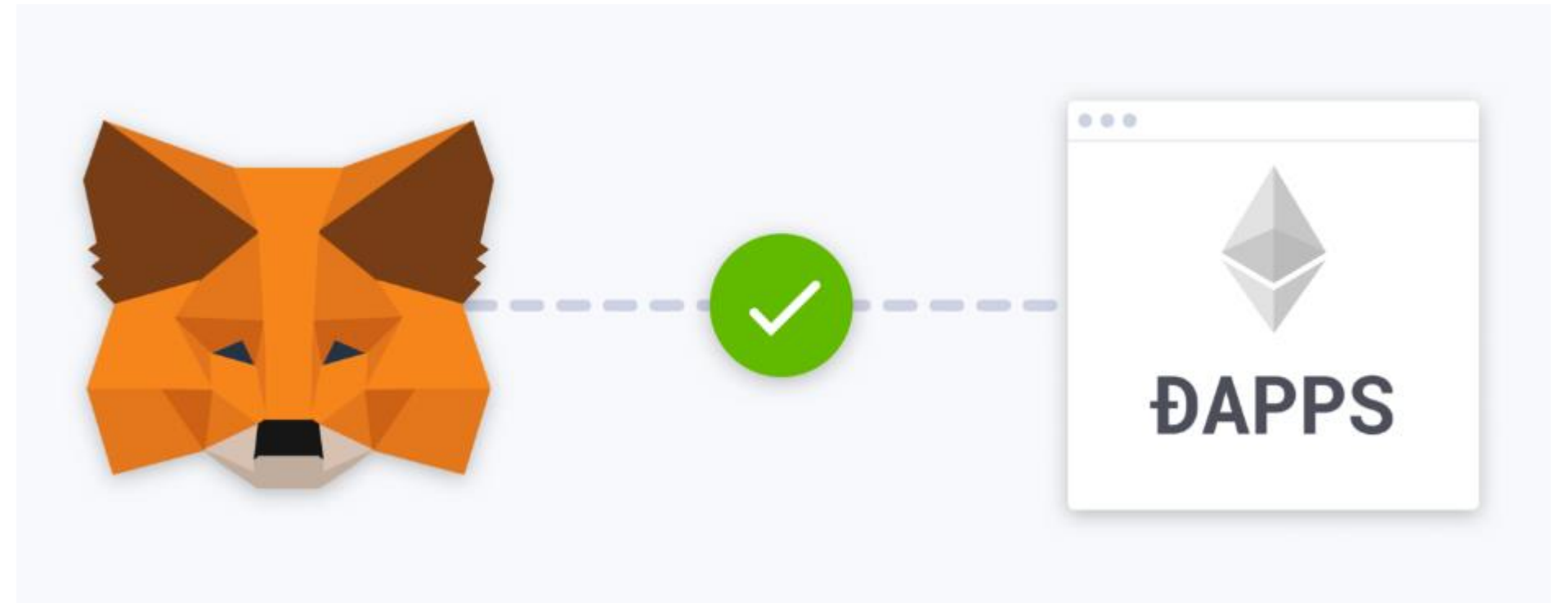
## 7. Coinbase (Web Wallet)

Coinbase는 가장 인기있는 Bitcoin 웹 지갑 중 하나입니다. 현재는 Ethereum 지원도 포함 되었습니다. 단점은 개인 키가 Coinbase의 호스팅 서버에 저장되어 있기 때문에 개인 키를 제어 할 수 없다는 것입니다.

# 메타마스크

질문) MetaMask의 기능은 어떤 것들이 있습니까?

- 계정 생성
- 이더 송금
- 스마트 컨트랙트 배포
- 스마트 컨트랙트 함수 호출
- 옵션으로 UI 제공



메타 마스크는 이더리움 개인 지갑을 편리하고 안전하게 관리할 수 있는 구글 크롬 확장 프로그램이며, 이더리움 블록체인 기반의 DApps로 연동되는 개인지갑 시스템입니다.

메타마스크의 기반 기술은 ethereumjs-wallet이며 사용자에게 비밀번호를 입력 받아 해당 정보로 하여금 private key를 안전하게 저장하고 보관하며 각 계좌들을 쉽게 사용할 수 기능들을 제공합니다.

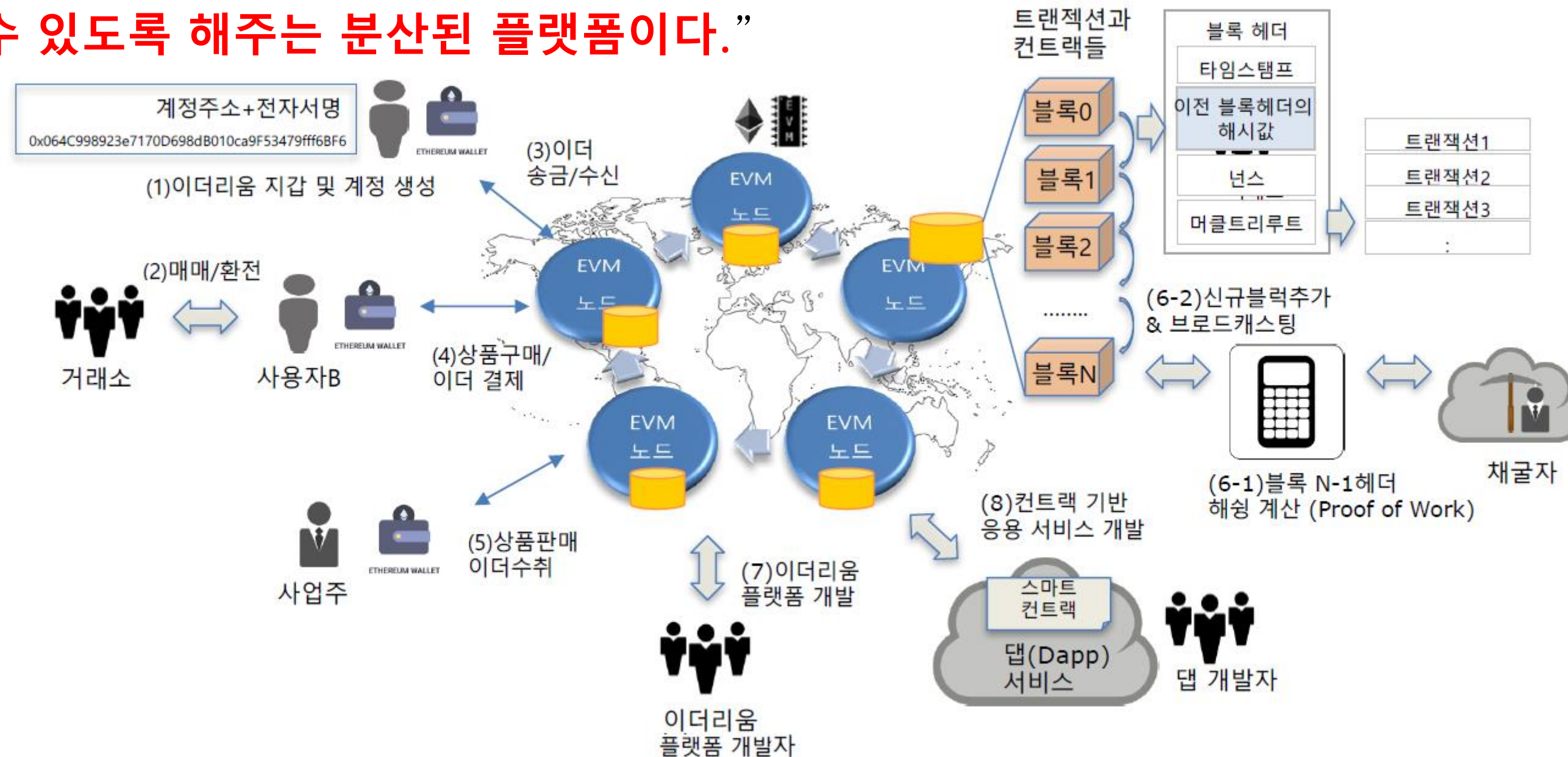
메타 마스크는 미스트(Mist)처럼 geth 프로그램을 따로 설치하지 않아도 되며, 스마트 컨트랙트를 발행할 수 있는 DApp입니다.



# 이더리움 플랫폼 작동 과정

## 이더리움 플랫폼의 작동 과정 - I

: 이더리움 플랫폼은 비트코인 처럼 블록체인 기술 기반하에 이더(Ether)와 같은 다양한 암호화폐를 생성하고 운용할 수 있도록 해준다. 또한 이더리움은 네트워크 상에서 서로 신뢰할 수 없는 대상 간에 서로 합의한 계약을 준수하도록 강제하는 스마트 컨트랙트를 지원함으로써 비트코인 같은 다른 암호화폐 시스템과는 달리 분산 플랫폼을 목표로 한다. **“이더리움은 정확히 프로그래밍 한대로 동작하는 스마트 컨트랙트를 실행할 수 있도록 해주는 분산된 플랫폼이다.”**





# 이더리움 플랫폼 작동 과정

---

## 이더리움 플랫폼의 작동 과정 - II

### 1. 이더리움의 지갑 설치와 사용

- 일반 사용자는 암호화폐 이더를 사용하기 위해서 메타마스크(Mist)나 이더리움 월렛(Ethereum Wallet)을 설치한다(①) 어카운트(Account)를 생성된 후에 해당 암호는 절대 변경될 수 없다. 메인 어카운트를 이더베이스(Etherbase)라고 한다. 이더베이스는 마이닝 등의 작업 대가를 이더로 지급 받을 때 기본 어카운트로 사용된다. 복수개의 어카운트를 생성할 수 있다.
- 사용자 입장에서 쉽게 이더를 획득하는 방법은 빗썸이나 업비트처럼 이더를 거래하는 거래소에서 이더를 구매하거나(②), 다른 사용자에게 이더를 송금 받는 것이다(③) 또한 이더리움 채굴자가 되어 마이닝 작업을 통해 이더를 확보할 수도 있지만 많은 장비 투자가 필요하기 때문에 적합한 방법은 아니다. 사업주도 미리 이더리움 월렛과 사용자 어카운트를 생성해 두어야 결제 대가로 이더를 전송 받을 수 있다. 상품이나 서비스를 판매 후 이더로 결제하고 이를 사용자로 부터 획득할 수 있다.(④,⑤)

# 이더리움 플랫폼 작동 과정

---

## 이더리움 플랫폼의 작동 과정 - III

### 2. 모든 거래 기록의 공유 및 블록체인 구성

- 이더 금액의 이동은 특정 시점에 특정 사용자의 어카운트 상태를 다른 상태로 전이 시키는 것이다 이런 일련의 상태변화를 일으키는 모든 활동(②,③,④,⑤)을 트랜잭션이라고 부른다. 이 트랜잭션들이 모여 하나의 블록이 만들어 지고, 이 블록이 시간순으로 연결되면서 일련의 블록체인이 된다.
- 기존의 은행에서 사용하는 중앙집중 원장에서는 중앙의 은행이 각 트랜잭션의 이상유무를 확인하고 보장해 준다. 그러나 블록체인은 중앙의 인증기관 없이 네트워크로 연결된 컴퓨터 간에 공유되기 때문에 정보도달에 시차가 발생할 수 있고, 정보 지연이나 미도달 사태가 발생할 수 있다. 이러한 문제를 해결하기 위해서는 해당 정보가 정확하고 문제가 없는지를 확인하기 위한 방법이 필요하다.

# 이더리움 플랫폼 작동 과정

---

## 이더리움 플랫폼의 작동 과정 - III

### 2. 모든 거래 기록의 공유 및 블록체인 구성

- 이 방법이 합의(동의) 알고리즘이다. 합의 알고리즘은 “작업증명(PoW, Proof of Work)” 알고리즘이다. PoW 알고리즘을 수행하는 사람은 채굴자(마이너)이다. 채굴자는 각 트랜잭션이 모여 있는 블록에 정의된 난이도보다 적은 수의 해쉬값(hash value)을 찾는 컴퓨터 해쉬연산을 한 후, 해당 값을 찾으면 네트워크 상에 연결되어 있는 모든 참여자 노드에게 전파하여 알린다.(⑥-1)
- 여러 채굴자 중 가장 빠르게 값을 찾은 채굴자가 블록당 3이더(Ether)와 블록 내에 포함되어 있는 트랜잭션의 처리비용을 함께 획득한다.(⑥-2) PoW를 통한 채굴에 성공하기 위해서는 막대한 장비 투자와 전기사용, 대형 채굴업자에 의한 영향도 집중 등 문제점이 많다. 이를 해결하기 위해 “지분증명(PoS, Proof of Stake)”으로 합의 방식을 변경 중에 있다.

# 이더리움 플랫폼 작동 과정

---

## 이더리움 플랫폼의 작동 과정 - IV

### 3. 다양한 응용 앱 개발

- 개발자는 오픈 소스 방식으로 개발,운영되는 이더리움 개발 커뮤니티에 참여할 수 있다.(⑦) 소스 공유 및 컨트롤 서비스인 깃허브(<https://github.com/ethereum>)에 접속 후 이더리움 소스를 다운로드하여 해당 소스코드의 오류를 수정하거나 신규기능을 추가할 수 있다. 또한, 이더리움의 스마트 컨트랙트를 이용하여 다양한 응용 서비스를 개발할 수도 있다.(⑧) 스마트 컨트랙트를 이용해서 개발한 일련의 서비스를 탈 중앙화 앱인 DApp(Decentralized App, 뎡)이라고 한다.

# 이더리움 플랫폼 작동 과정

---

## 이더리움 플랫폼의 작동 과정 - IV

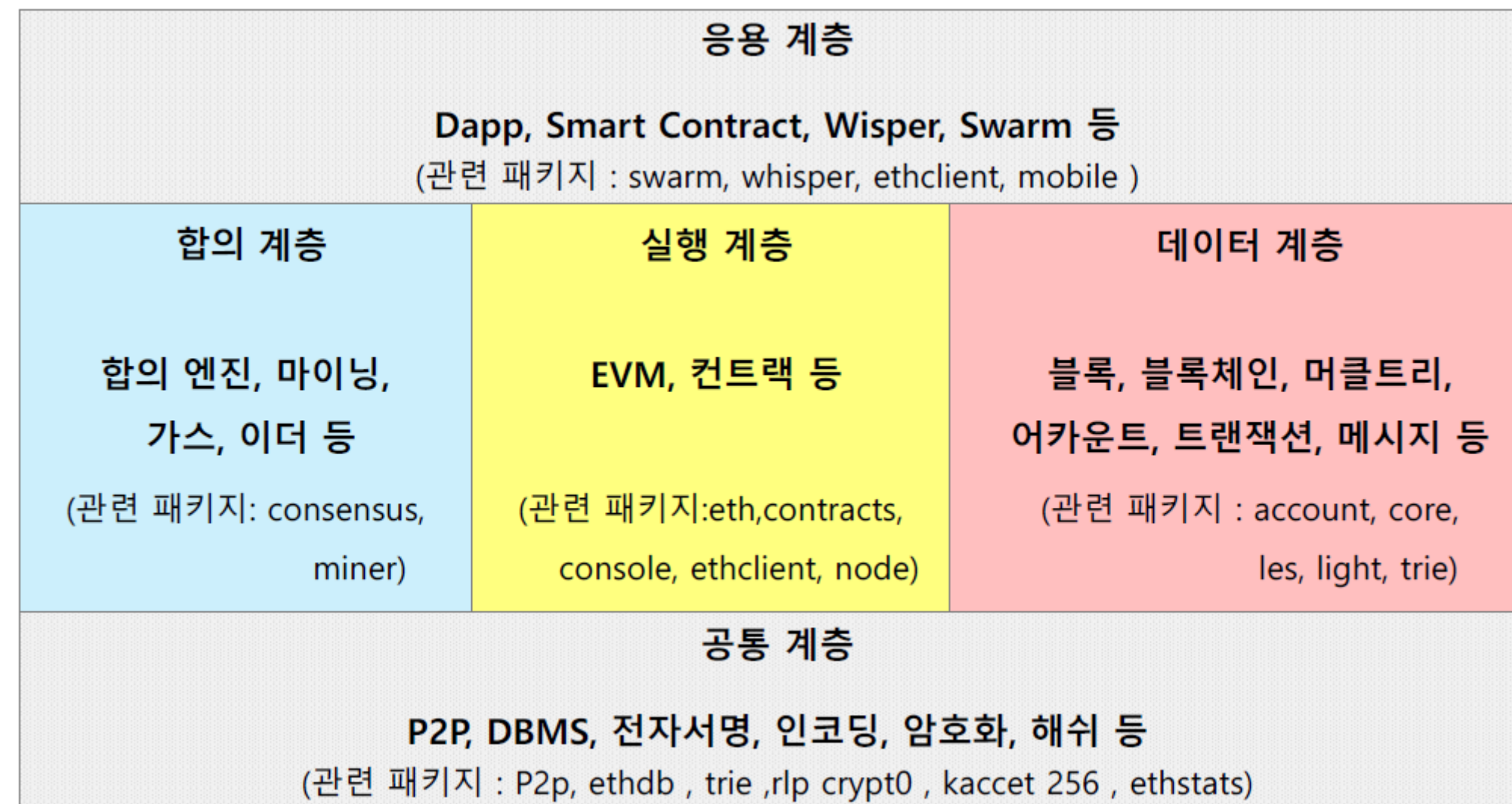
### 3. 다양한 응용 앱 개발

- DApp은 기존의 자바스크립트나 HTML, CSS 등을 사용하여 스마트 컨트랙트를 조작함으로써 다양한 서비스를 개발할 수 있다. 솔리디티로 개발된 컨트랙트 프로그램을 바이트 코드로 컴파일한 후에 블록체인에 배포하고 저장한다. 배포된 스마트 컨트랙트의 바이트 코드는 일련의 검증 과정을 통해 이상이 없을 경우 이더리움 가상 머신(EVM, Ethereum Virtual Machine)에서 실행 가능한 코드(Op코드)로 변환되어 실행된다. 사용자가 스마트 컨트랙트를 이용하기 위해서는 가스(GAS)라는 내부 운용 토큰을 사용 대가로 지급해야 한다. DApp은 플랫폼으로서 이더리움의 목표를 실현시켜 주는 가장 중요한 개념이자 비트코인과 같은 다른 암호화폐와 이더리움을 차별화 하는 중요한 요소이다.

# 이더리움 플랫폼 참조 모델

## 이더리움 플랫폼 참조 모델 - I

- 이더리움 플랫폼은 P2P 네트워크를 기반으로 서로 신뢰할 수 없는 대상들이 모여 일련의 트랜잭션의 유효성을 합의한 후, 전체 데이터를 분산된 원장을 통해 전체가 공유한다. 이더리움은 단순 암호화폐의 트랜잭션 전송 뿐만 아니라 스마트 컨트랙트라는 프로그램을 작동시키기 위한 내부 매커니즘을 지원한다. 5개의 데이터 계층, 합의 계층, 실행 계층, 응용계층, 공통계층으로 모델링할 수 있다.



# 이더리움 플랫폼 참조 모델

---

## 이더리움 플랫폼 참조 모델 - II

- 데이터 계층은 이더리움에서 다루는 각종 데이터 구조를 정의하고 관련 데이터를 관리한다. 주요 데이터 구조로는 Account와 트랜잭션, 메시지와 리시트(receipt), 데이터의 집합인 블록과 블록이 연결된 블록체인이 있다. 데이터 모델은 하부에 ethdb 패키지를 통해 구글이 만든 빠르고 가벼운 키/값(key/value) 데이터베이스인 레벨 DB(LevelDB)에 바이너리 형태로 저장된다.
- 합의계층은 Account에 의해 생성된 트랜잭션과 트랜잭션들과 관련된 데이터들이 모여 있는 블록의 유효성을 검증하는 합의 엔진과 이 과정을 수행하는 채굴과 채굴의 난이도, 채굴자들에게 지급할 인센티브인 가스(GAS), Ether등의 처리를 담당한다.
- 실행계층은 이더리움 블록체인에서 구동 가능한 스마트 컨트랙트와 스마트 컨트랙트를 수행시켜 줄 EVM(Ethereum Virtual Machine)의 처리를 담당한다.
- 공통계층은 이더리움에서 공통적으로 사용하는 기능들을 제공한다. 노드 간의 연결과 동기화를 위한 P2P 네트워크 프로토콜을 비롯하여 암호해쉬, 전자서명, 각종 인코딩, 공통 저장소 등 모든 계층에서 공통적으로 이용할 기능들을 담당한다.



# 3. Geth(Go Ethereum)

# 이더리움 네트워크 유형



← 테스트 순서

# 이더리움 메인 넷

---

- MainNet ( <https://etherscan.io/> )
- ✓ 실제 거래가 이루어지는 이더리움 메인 네트워크
- ✓ 메인넷은 블록체인 네트워크 시스템 운영을 통해 디지털 화폐 생성뿐 아니라 다른 디앱(Dapp)을 탄생하게 하는 기반을 제공해 독자적인 생태계를 구성하는 것을 의미한다.
- ✓ 자체 프로토콜인 메인넷을 보유하고 있다면 '코인'(이더리움, 이오스, 퀀텀, 리플, 네오 등), 다른 플랫폼에서 파생되어 만들어진 것을 '토큰'(트론, 비체인 등)이라고 합니다.
- ✓ 메인넷이 만들어지는 과정은 다음과 같습니다. 우선, 이더리움과 같은 기존 코인을 기반으로 토큰을 제작한 뒤 암호화폐공개(ICO)를 합니다. 또 테스트넷을 운영하면서 독자 플랫폼으로 자리 잡을 수 있는지 테스트를 하는 것입니다. 이 기간은 짧게는 수개월, 길게는 몇 년이 걸립니다. 지갑 생성과 거래소 연결 문제 등 안정화 작업이 쉽지 않기 때문입니다.
- ✓ 메인넷을 갖는 것은 매우 어렵습니다. 그만큼 기술력을 인정받을 수 있습니다. 글로벌 대형 거래소들은 우선 상장 기준 중 하나로 메인넷 구축 여부를 꼽기도 합니다.

# 이더리움 테스트 넷

---

- TestNet

이더리움의 퍼블릭 테스트 네트워크. 현재 아래와 같은 3개의 퍼블릭 테스트넷을 제공한다.

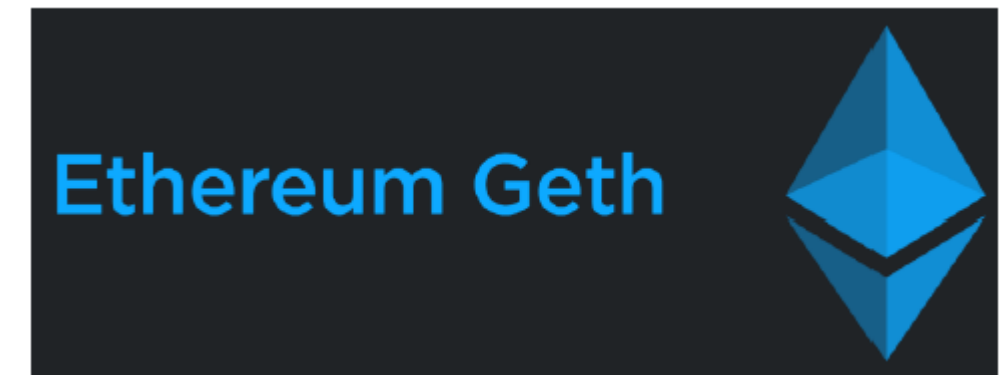
- 1) Ropsten (Proof of Work) : 프로덕션 환경이 실제 메인넷이랑 가장 비슷하여 (PoW 이기 때문에) 현재 가장 많이 쓰여지고 있는 테스트 네트워크. Geth 와 Parity 클라이언드 둘다 지원. 이전에 스팸 어택을 당한 이력이 있음. 이더 마이닝이 가능하며, 테스트 Ether를 요청할 수 있다.
- 2) Kovan (Proof of Authority) : Parity 팀이 Ropsten의 문제를 해결하기 위해 개발한 테스트 네트워크. 스팸어택으로 부터 안전하며, Parity 클라이언트만 지원. Ether 마이닝 불가능하며, 테스트 Ether를 요청할 수 있다.
- 3) Rinkeby (Proof of Authority) : Rinkeby 역시 스팸 어택을 방지하고자 만들어 졌으며, 이더리움 팀이 개발한 테스트넷. Geth 클라이언트만 지원. 이더 마이닝 불가능하며, 테스트 Ether를 요청 할 수 있다.

# 이더리움 프라이빗 넷

---

## Geth 와 Ganache

- Geth는 이더리움 재단(Ethereum Foundation)이 제공하는 공식 클라이언트 소프트웨어로써, Go언어로 개발되었다.  
Geth는 블록체인의 복사본을 최신 상태로 유지하기 위해 다른 노드와 통신한다. 또한 블록을 채굴하고, 블록체인에 트랜잭션을 추가하고, 검증하며, 실행도 할 수 있다.
- Ganache는 메모리 내 블록체인을 사용하는 방법으로서 트랜잭션의 실행 시간을 단축해 준다.



# 이더리움 실습

---

Geth 란?

## 1. Geth(go-ethereum)

: Geth는 Go언어로 된 이더리움 노드 소프트웨어 코드입니다.

Geth 소프트웨어: 이더리움 프로토콜 블록체인 기술을 이용한 golang 구현체로서, 전 세계 공유 컴퓨팅 플랫폼을 구현한다.

**go-ethereum : 커맨드 라인 인터페이스 형태의 GO 언어로 만들어진 full 이더리움 노드이다.**

The go-ethereum client is commonly referred to as geth, which is the the command line interface for running a full ethereum node implemented in Go. By installing and running geth, you can take part in the ethereum frontier live network and:

# 이더리움 실습

---

## Geth로 할 수 있는 일

- mine real ether ( 이더를 마이닝 할 수 있다. )
- transfer funds between addresses ( 주소 간 자금이체 )
- create contracts and send transactions ( 계좌를 생성하거나 트랜잭션을 보낸다. )
- explore block history ( 블록 기록 탐색 )
- and much more ( 기타 등등 )



# 이더리움 실습

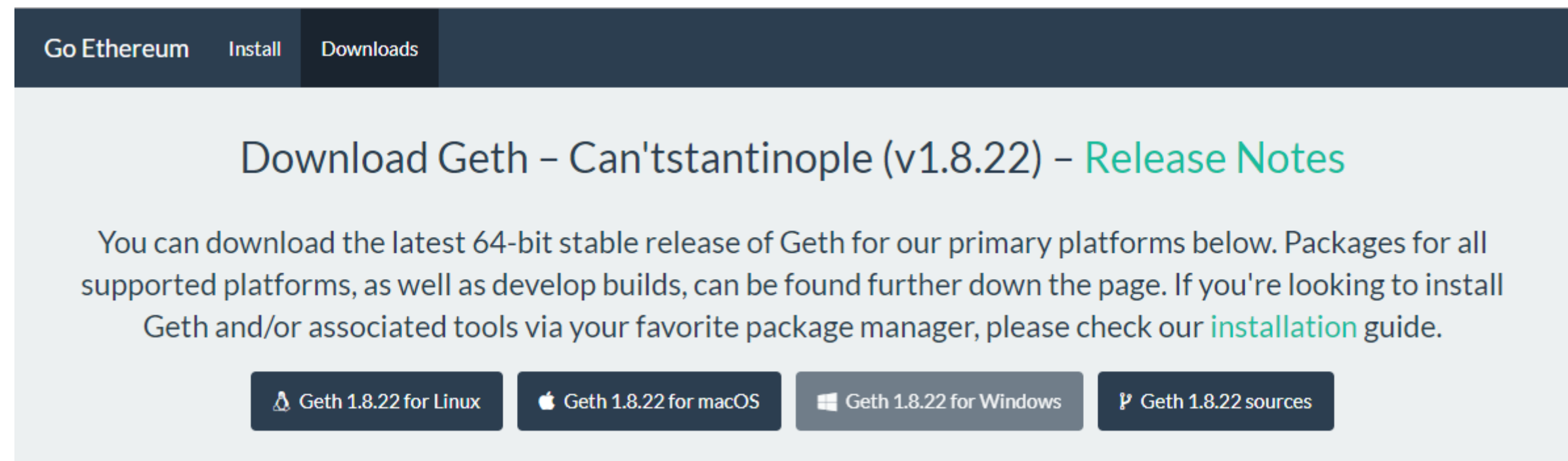
---

## Geth의 설치 부터 Mining 까지

### Step1. Geth 설치 및 실행

#### 1. Geth 프로그램 다운로드

<https://ethereum.github.io/go-ethereum/downloads/>



# 이더리움 실습

---

## Step1. Geth 설치 및 실행

### 1-1. Geth 설치하기

Windows : 인스톨러의 가이드에 따라 설치하세요.

Mac : 원하는 디렉토리에 설치하세요.

### 1-2. Geth 디렉토리를 path에 추가한다.

: C:\Program Files\Geth 디렉토리를 추가한다.

### 1-3. Geth 버전 확인

geth version 명령어를 통해서 geth의 버전을 확인할 수 있다.

> `geth version`

# 이더리움 실습

## Step2. Genesis 블록 생성

제네시스 블록(Genesis block)이란?

: **블록체인 네트워크가 시작되었음을 상징하는 첫번째 블록이다.** 첫 번째 블록이 있어야 다음 블록을 계속 연결할 수 있다는 점에서 의미를 갖는다. 이에 창세기라는 뜻을 가진 “제네시스”로 명명하였다. 비트코인의 경우 2009년 1월 사토시 나카모토에 의해 비트코인 제네시스 블록이 생성되었다. 한편 블록이 생성된 순서는 높이(height)로 표현하는데, 제네시스 블록의 높이는 0이다.

- Geth genesis.json 파일의 내용은 아래와 같음

```
{
  "config": {
    "chainId": 0,
    "homesteadBlock": 0,
    "eip155Block": 0,
    "eip158Block": 0
  },
  "alloc"      : {},
  "coinbase"   : "0x0000000000000000000000000000000000000000",
  "difficulty" : "0x20000",
  "extraData"  : "",
  "gasLimit"   : "0x2fefd8",
  "nonce"      : "0x0000000000000042",
  "mixhash"    : "0x0000000000000000000000000000000000000000000000000000000000000000",
  "parentHash" : "0x0000000000000000000000000000000000000000000000000000000000000000",
  "timestamp"  : "0x00"
}
```

# 이더리움 실습

---

Step2. Genesis 블록 생성

puppeth 모듈이란?

puppeth라는 go-ethereum에서 제공하는 모듈이 있습니다. 2017년 11월 데브콘3에서 발표된 것인데, 커맨드라인을 통해서 손쉽게 프라이빗 이더리움 네트워크를 구축할 수 있도록 해줍니다.

```
c:\blockchain>puppeth
```

```
Please specify a network name to administer (no spaces, hyphens or capital letters please)
```

```
>mynetwork
```

```
What would you like to do? (default = stats)
```

```
1. Show network stats
```

```
2. Configure new genesis
```

```
3. Track new remote server
```

```
4. Deploy network components
```

```
> 2
```

# 이더리움 실습

---

Step2. Genesis 블록 생성

which consensus engine to use? (default = clique)

1. Ethash - proof-of-work

2. Clique - proof-of-authority

> 1

which accounts should be pre-funded? (advisable at least one)

> 0x

should the precompile-addresses (0x1 .. 0xff) be pre-funded with 1 wei? (advisable yes)

> yes

specify your chain/network ID if you want an explicit one (default = random)

> 4386

[32mINFO [0m[01-29|12:10:18.986] Configured new genesis block

# 이더리움 실습

---

## Step2. Genesis 블록 생성

what would you like to do? (default = stats)

1. Show network stats
2. Manage existing genesis
3. Track new remote server
4. Deploy network components

> 2

1. Modify existing fork rules
2. Export genesis configurations
3. Remove genesis configuration

> 2

which folder to save the genesis specs into? (default = current)

will create mynetwork.json, mynetwork-aleth.json, mynetwork-harmony.json, mynetwork-parity.json

> ctrl+C로 exit한다.

# 이더리움 실습

---

## Step2. Genesis 블록 생성

vs(visual studio) Code에서 Genesis Block 파일 열기

```
c:\blockchain>code mynetwork.json
```

이더리움 노드에서 Genesis 블록의 내용을 읽고 초기화 하는 것이다.

config : 체인의 파라미터들을 정의하는데 사용이 된다.

chainId : 만들 때 정해진 id이고,

etash : 체인의 합의 알고리즘이 작업증명이라는 것을 말한다.

timestamp : 이더리움 가상 머신(EVM)에서 블록 생성의 난이도를 조절하는데 사용된다.

연속되는 두개의 블록의 timestamp가 작으면 난이도는 올라가고, 크면 난이도는 내려간다.

또한 timestamp가 블록들이 올바른 순서대로 진행되고 있는지를 확인한다.

gasLimit: 가스한도 인데 블록내의 트랜잭션이 소비할 수 있는 최대 gas 값을 의미한다.

각 블록마다 처리할 수 있는 트랜잭션의 개수를 제한을 시켜서 블록의 사이즈를 조절하는 것이다.

difficulty : 블록의 유효성을 검사할 때 사용되는 난이도를 말한다. 채굴자가 블록을 채굴하기 위해 퍼즐을 풀면서 연산을 몇 번이나 해야 하는지 이 difficulty의 값과 직접적인 연관 관계가 있다.

difficulty가 높으면 블록을 채굴하는 시간도 길어진다.



# 이더리움 실습

---

## Step2. Genesis 블록생성

`alloc` : 어떤 지갑 주소에 자금을 미리 할당하는 내용이다. 필드의 내용이 길다.

`number` : 블록 넘버를 뜻한다. 제니시스 블록 이므로 0이다.

`gasUsed` : 이 블록내에서 여러 개의 트랜잭션을 처리하는데 사용된 모든 `gas`의 합계

`parentHash` : 부모 블록의 `hash` 값을 가지고 있다. 제니시스 블록 이므로 0이다

# 이더리움 실습

---

## Step2. Genesis 블록 생성

### 2-1. genesis 블록을 사용해서 Node 초기화 완성하기

```
c:\blockchain>geth --datadir . init mynetwork.json
```

geth 커맨드를 사용해서 현재 디렉토리에 private node 데이터를 저장한다는 뜻입니다.

init은 private node를 초기화 하기 위한 genesis 블록의 파일 이름을 지정할 때 사용하는 옵션이다.

즉, genesis 블록을 사용해서 node 초기화를 완성시킨 것이다.

c:\blockchain 디렉토리 아래에 geth와 keystore 디렉토리가 생성 되었습니다.

geth 폴더에는 모든 체인에 대한 정보가 있고, keystore 폴더에는 생성할 계정에 대한 정보가 저장되어 있다.

# 이더리움 실습

---

## Step3. Account 생성 및 관리

### 3-1. Account 생성 (3개의 계정을 만든다. Keystore 디렉토리에 계정정보가 저장된다.)

```
c:\blockchain>geth --datadir . account new
```

Your new account is locked with a password. Please give a password. Do not forget this password.

Passphrase:

Repeat passphrase:

Address: {8ec77d693fd7cde71bec86853f4ca1de153e59b7}

### 3-2. Account 목록 보기

```
c:\blockchain>geth --datadir . account list
```

```
INFO [02-08|15:23:03.322] Maximum peer count                      ETH=25 LES=0 total=25
Account #0: {87b945c0a3c90beb97483c142dd64f45d20da050} keystore://C:\blockchain\keystore\UTC--2019-01-23T10-54-19.401449000Z--87b945c0a3c90beb97483c142dd64f45d20da050
Account #1: {c8283d5ec83c97e767a4e1b68eb7cce1bc03afbb} keystore://C:\blockchain\keystore\UTC--2019-01-23T10-54-47.113947100Z--c8283d5ec83c97e767a4e1b68eb7cce1bc03afbb
Account #2: {5af09c175d804cf08ceebb4c100daf72411e0aa0} keystore://C:\blockchain\keystore\UTC--2019-01-23T10-55-15.124467400Z--5af09c175d804cf08ceebb4c100daf72411e0aa0
```

# 이더리움 실습

---

## Step4. Geth 노드 실행

### 4-1. password.sec 파일 작성

```
c:\blockchain>code password.sec
```

패스워드를 입력한다.

### 4-2. Geth Node 시작 파일 작성 (※ 반드시 한 줄로 작성해야 한다)

```
c:\blockchain>code nodestart.cmd
```

```
geth --networkid 4386 --mine --minerthreads 2 --datadir "./" --nodiscover --rpc --rpcport  
"8545" --rpccorsdomain "*" --nat "any" --rpcapi eth,web3,personal,net --unlock 0 --  
password ./password.sec
```

### 4-3. nodestart.cmd 파일 실행

```
c:\blockchain>nodestart.cmd
```

# 이더리움 실습

---

## Step4. Geth 노드 실행

### 4-4. Geth 노드 시작에 필요한 옵션 설명

`--networkid` : 네트워크 식별자

0 : Olympic, Ethereum public pre-release testnet

1: Frontier, Homestead, Metropolis, the Ethereum public main network

1: Classic, the (un)forked public Ethereum Classic main network, chain ID 61

1: Expanse, an alternative Ethereum implementation, chain ID 2

2: Morden, the public Ethereum testnet, now Ethereum Class testnet

3: Ropsten, the public cross-client Ethereum testnet

4: Rinkeby, the public Geth PoA testnet

[Other]: Could indicate that your connected to a local development test network

`--mine` : 채굴을 활성화 한다.

`--minerthreads 2` : 채굴에 사용할 CPU 스레드 수를 지정한다. 기본값은 1이다.

`--nodiscover` : 생성자의 노드를 다른 노드에서 검색할 수 없게 하는 옵션

(탐색 프로토콜 : 다른 노드가 우리 체인에 연결하려는 것을 못하게 하는 것입니다.)

# 이더리움 실습

---

## Step4. Geth 노드 실행

### 4-4. Geth 노드 시작에 필요한 옵션 설명

- datadir : 체인 파일을 어디에 저장할 것인지를 지정함 ( 현재 디렉토리에 저장함 )
- rpc : HTTP-RPC 서버를 활성화 하고, 별도의 콘솔을 연결할 때 필요한 옵션임 메타 마스크에서 geth로 실행한 노드에 연결할 수 있다.
- rpcport : 어떤 포트에 접속해야 하는지 명시하는 파라미터
- rpccorsdomain : 아무 도메인에서나 우리 rpc도메인에 접속할 수 있도록 해주는 파라미터
- nat : 네트워킹에 사용됨
- rpcapi : rpc 엔드포인트에서 어떠한 API를 커맨드로 사용하기 위한 파라미터 (eth,web3,personal,net)
- unlock 0 : 채굴이 시작됐을 때 모든 채굴 보상금이 디폴트로 첫번째 계정으로 들어가게 된다. 보상을 받기 위해서는 첫번째 계정을 unlock 시켜야 한다.
- password : 비밀번호를 담고 있는 파일(password.sec)을 지정한다.

Geth 마이닝을 위한 명령 라인 옵션

(<https://github.com/ethereum/go-ethereum/wiki/Command-Line-Options>)

# 이더리움 실습

---

## Step4. Geth 노드 실행

### 4-5. nodestart.cmd 실행 후의 결과

```
c:\blockchain>nodestart.cmd
```

: C:\Users\사용자계정\AppData\Ethash\ 아래에 보면 DAG 파일이 생성되어 있다.

노드가 채굴을 하기 위해서는 먼저 DAG 파일이 생성되어 있어야 한다. Node를 처음 시작할 때만 DAG 파일을 생성하고, 이미 생성되어 있으면 바로 채굴을 시작한다.

: DAG(Directed Acyclic Graph) 일방향적 비순환 그래프란?

DAG알고리즘은 일방향적 비순환 그래프 구조로 인하여 뛰어난 트랜잭션 처리속도, 확장성, 저렴한 수수료의 장점을 가진다.

etash 채굴 알고리즘에서 필요로 하는 데이터 구조이다.

: epoch=0 일 때 100% 까지 진행되고, epoch=1 일 때 100% 까지 완료되면 DAG 파일 생성이 끝난다.

: DAG 파일 생성이 끝나면, Commit new mining work 메시지가 출력되면서 채굴이 시작된다.

# 이더리움 실습

---

## Step5. Geth 콘솔

5-1. 실행 중인 노드에 연결해서 javascript console을 여는 명령

```
c:\blockchain> geth attach ipc:\\.pipe\geth.ipc
```

5-2. Ether API 사용

코인베이스 계정(첫번째 계정) 주소가 출력됨

```
>eth.coinbase
```

현재 이 노드에 등록된 모든 계정들의 주소가 출력됨

```
>eth.accounts
```

두번째 계정의 잔액 (wei 단위로 표시됨)

```
>eth.getBalance(eth.accounts[1])
```

코인베이스 계정의 잔액 (wei 단위로 표시됨)

```
>eth.getBalance(eth.coinbase)
```



# 이더리움 실습

---

Step5. Geth 콘솔

5-2. Ether API 사용

현재 채굴된 블록의 번호

```
>eth.blockNumber
```

현재 해시레이트(연산 처리능력을 측정하는 단위로 해시 속도를 의미함. 해시레이트가 높아져 연산량이 많아질 경우, 더 빠른 채굴이 이루어지기 때문에 채굴 난이도가 높아진다.)

```
>eth.hashrate
```

현재 채굴 중 인지 여부

```
>eth.mining
```

GAS Price 정보 확인

```
>eth.gasPrice
```

현재 진행을 기다리고 있는 트랜잭션

```
>eth.pendingTransactions
```

# 이더리움 실습

---

Step5. Geth 콘솔

5-3. Admin API 사용

실행중인 Geth 노드가 현재 모든 데이터베이스를 저장하는 데 사용하는 절대 경로

```
>admin.datadir
```

nodeInfo는 실행중인 Geth 노드에 대한 정보

```
>admin.nodeInfo
```

admin, eth, personal, web3, miner API들을 Javascript 콘솔에서 관리할 수 있고, 더 자세한 사항은 <https://github.com/ethereum/go-ethereum/wiki/Management-APIs> 에서 확인 가능합니다.

# 이더리움 실습

---

## Step5. Geth 콘솔

### 5-4. web3 API 사용

기본적인 화폐의 단위는 wei이다. Ether의 단위로 환산하기 위해서는 web3 객체를 사용해야 하고 fromWei 함수를 사용하면 된다.

코인베이스 계정의 잔액 : wei 단위를 ether로 변환시켜 출력

```
>web3.fromWei(eth.getBalance(eth.coinbase), "ether")
```

### 5-5. 채굴 멈추기 / 다시 시작하기

채굴 멈추기

```
>miner.stop()
```

채굴 다시 시작하기( 2개의 스레드에서 )

```
>miner.start(2)
```

# 이더리움 실습

---

## Step5. Geth 콘솔

### 5-6. 계좌 간에 송금

송금을 하기 위해 200초 동안 두번째 계정을 unlock 시킨다.

```
>personal.unlockAccount(eth.accounts[1], "bimilbunho", 200)
```

코인베이스 계정에서 두번째 계정으로 20 ether를 송금한다. Log에서 tx commit 되었다는 것을 확인한다.

```
>var tx = {from:eth.coinbase,to:eth.accounts[1],value:web3.toWei(20,"ether")}
```

```
>personal.sendTransaction(tx, "bimilbunho")
```

트랜잭션 id로 트랜잭션 정보를 확인한다.

```
>eth.getTransaction(txid)
```

두번째 계정의 잔액 확인한다.

```
>eth.getBalance(eth.accounts[1])
```

```
>web3.fromWei(eth.getBalance(eth.accounts[1]), "ether")
```

# 이더리움 실습

Ethereum 대시 보드 와 Ethereum 네트워크 상태를 모니터링 하는 방법

1. DashBoard(<https://ethstats.net/>)

2. eth-netstats(<https://github.com/cubedro/eth-netstats>)

: 이더리움 네트워크 상태를 추적하기 위한 비주얼 인터페이스입니다. WebSocket을 사용하여 실행중인 노드로 부터 통계를 수신 받아서 Angular 인터페이스를 통해 출력합니다.

3. eth-net-intelligence-api(<https://github.com/cubedro/eth-net-intelligence-api>)

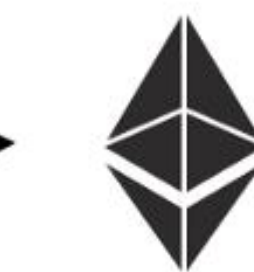
: Ethereum과 함께 실행되고 네트워크 상태를 추적하고 JSON-RPC를 통해 정보를 가져오고 WebSocket를 통해 eth-netstats에 연결하여 정보를 제공하는 백엔드 서비스입니다.



eth-netstats



eth-net-intelligence-api.



Ethereum network

# 이더리움 실습

---

모니터링 도구들 사용하기 위한 작업 순서

1. Execute ethereum private/public node
2. Download and Install Tools
3. Configure the Node Monitoring App
4. Start the Node App
5. Start the Frontend

## 2.1 eth-netsates 설치

```
C:\eth-netstats> npm install
```

```
C:\eth-netstats> npm install -g grunt-cli
```

```
C:\eth-netstats> grunt
```

## 2.2 eth-net-intelligence-api 설치

```
C:\eth-net-intelligence-api> npm install
```

```
C:\eth-net-intelligence-api> npm install -g pm2
```

# 이더리움 실습

---

## 3. Configure the Node Monitoring App

eth-net-intelligence-api 디렉토리의 app.json 수정

```
"INSTANCE_NAME"      : "Geth/node1/v1.8.21-stable-9dc5d1a9/windows-amd64/go1.11.4",  
"WS_SERVER"          : "http://localhost:3000",  
"WS_SECRET"          : "mysecret",
```

## 4. Start The Node App

### 4.1 Execute the backend tool ( eth-net-intelligence-api)

```
C:\eth-net-intelligence-api> pm2 start app.json
```

### 4.2 Execute the front-end tool (eth-netstats)

```
C:\eth-netstats> set WS_SECRET=mysecret&& npm start
```

## 5. Start the Frontend

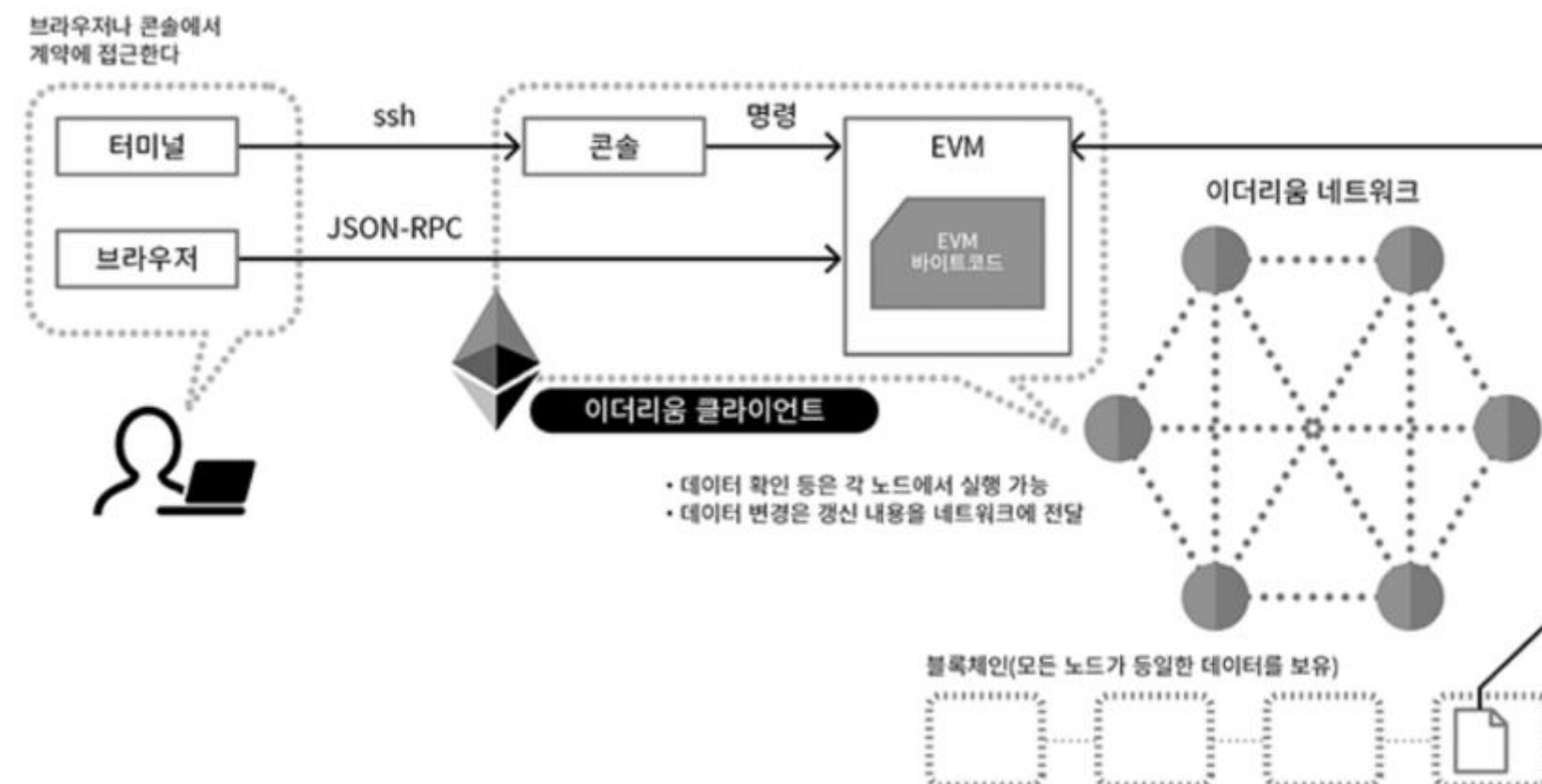
<http://localhost:3000>

# 스마트 컨트랙트 배포

## 스마트 컨트랙트를 이더리움에 배포하는 방법

이더리움에 두가지 방식으로 접근이 가능합니다. 터미널과 DApp방식, 또는 브라우저로 접근이 가능합니다.

- ① 터미널은 Geth를 통한 콘솔로 EVM의 스마트 컨트랙트를 올릴 수 있습니다.
- ② 브라우저 방식은 보통 Remix(<https://remix.ethereum.org/>)에서 Solidity라는 언어로 스마트컨트랙트를 작성하여 올립니다. Remix브라우저는 일련의 명령어(web3)들이 내장되어 있어 쉽게 바이트 코드로 컴파일을 하고 Deploy를 할 수 있습니다.





# 스마트 컨트랙트 배포

---

## 스마트 컨트랙트 작성 및 배포 순서

### 1. 스마트 컨트랙트 코딩

: 구현하고자 하는 내용을 Solidity 언어로 코딩합니다.

### 2. 구현한 소스 코드를 컴파일

: 컴파일 결과 EVM 바이트 코드가 생성됩니다.

### 3. 스마트 컨트랙트 배포

: 스마트 컨트랙트를 배포하는 것은 컴파일된 EVM 코드를 하나의 트랜잭션 처럼 블록에 추가시켜 블록체인에 등록시키는 작업입니다.

: 소스 컴파일 -> EVM 바이트 코드 -> 구체적인 작업은 ABI(Application Binary Interface) 취득 -> ABI로부터 컨트랙트 객체 생성 -> 트랜잭션 생성하여 블록에 추가

: 채굴자가 해당 블록을 채굴하게 되면 블록체인에 포함됨

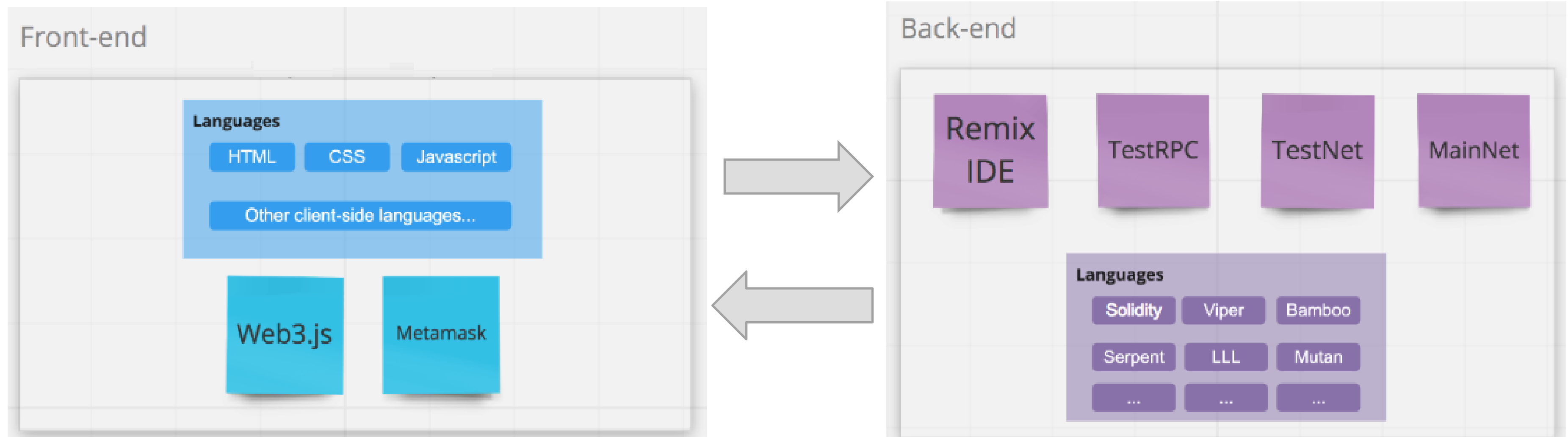
## 4. DApp 개발 사전지식

## 질문) Solidity란 무엇입니까?

- 솔리디티(Solidity)는 계약 지향 프로그래밍 언어로 다양한 블록체인 플랫폼의 스마트 컨트랙트 (Smart Contract) 작성 및 구현에 사용된다.
- 2014년 8월에 Gavin Wood 가 처음으로 제안했으며, 제안 이후 이더리움 프로젝트의 Christian Reitwiessner 가 이끄는 솔리디티팀에 의해 개발 되었다.
- Solidity is statically typed, supports inheritance, libraries and complex user-defined types among other features.
- Ethereum Virtual Machine(EVM)을 목표로 설계된 4가지 언어 중 하나이다.(Serpent, LLL, Viper (실험용)) 현재는 솔리디티가 이더리움의 주요 언어이다.
- 정적타입(statically-typed)의 언어이며, ECMAScript 문법을 기반으로 하였으며 EVM상에서 작동하는 스마트컨트랙트를 개발하기 위해 설계되었고 EVM에서 작동 가능한 바이트코드로 컴파일 된다. 개발자는 솔리디티를 통해서 스스로 실행되는 비즈니스 로직을 스마트컨트랙트에 담아서 Application을 구현할 수 있습니다.

# DApp 개발환경

- DApp을 구현하기 위한 개발 환경은 크게 두 가지로 나눌 수 있습니다. 스마트 컨트랙트를 작성하고 배포하는 Back-end Side와, 사용자 인터페이스를 구현할 Front-end Side입니다.



# DApp 개발환경

---

## Back-End

- Remix IDE

Solidity 코드를 작성하고 컴파일 및 배포까지 쉽게 자동으로 할 수 있도록 돕는 온라인 컴파일러 툴. 디버깅 툴로도 많이 쓰임. 원래는 로컬 컴퓨터에 직접 솔리디티 컴파일러를 설치해야 하는데, Remix 웹브라우저를 사용하면 그런 과정 없이 쉽고 빠르게 컴파일 및 배포가 가능하다.

- TestRPC(현재는 Ganache로 업데이트 되었음)

개발 단계에서 실제 이더 없이 시뮬레이션 테스트 환경을 구성할 수 있게 해주는 툴. 원래 이더리움 메인넷에서 테스트를 하려면 실제 이더가 필요하지만, TestRPC 를 사용하면 마이닝 없이 가상환경에서 리소스, 트랜잭션에 대한 제한 없이 이더리움을 만들어 내서 테스트를 할 수 있다.

# DApp 개발환경

---

## Back-End

- TestNet

이더리움의 퍼블릭 테스트 네트워크. 현재 아래와 같은 3개의 퍼블릭 테스트넷을 제공한다.

1) Ropsten (Proof of Work) : 프로덕션 환경이 실제 메인넷이랑 가장 비슷하여 (PoW 이기 때문에) 현재 가장 많이 쓰여지고 있는 테스트 네트워크. Geth 와 Parity 클라이언트 둘다 지원. 이전에 스팸 어택을 당한 이력이 있음. 이더 마이닝이 가능하며, 테스트 이더를 요청할 수 있다.

2) Kovan (Proof of Authority) : Parity 팀이 랍스텐의 문제를 해결하기 위해 개발한 테스트 네트워크. 스팸 어택으로 부터 안전하며, Parity 클라이언트만 지원. 이더 마이닝 불가능하며, 테스트 이더 요청 할 수 있다.

3) Rinkeby (Proof of Authority) : Rinkeby 역시 스팸 어택을 방지하고자 만들어 졌으며, 이더리움 팀이 개발한 테스트넷. Geth 클라이언트만 지원. 이더 마이닝 불가능하며, 테스트 이더 요청 할 수 있다.

- MainNet

실제 거래가 이루어지는 이더리움 메인 네트워크

# DApp 개발환경

---

## Front-End

- HTML/CSS/Javascript

웹 프론트엔드를 작성에 사용하는 언어들. 블록체인 기반 서비스(DApp)의 클라이언트를 웹 브라우저 기반으로 만들 때 필요하다.

- Web3.js

이더리움 자바스크립트 API.

- Metamask

이더리움 풀노드를 운영하지 않고도 웹 브라우저에서 DApp을 사용할 수 있게 해주는 크롬 및 파이어폭스 확장 프로그램. 즉, 일부 필수적인 블록 헤더 데이터만 외부의 풀노드로부터 받아와 검증하는 라이트 클라이언트 중 하나이며 메타마스크를 사용하면 여러 사이트의 계정(Account)을 관리하고, 블록체인 트랜잭션에 서명할 수 있음.

## 질문) Remix란 무엇입니까?

- 솔리디티 프로그래밍을 위해서는 로컬에 솔리디티 컴파일러를 설치하거나, Remix와 같은 개발 도구를 사용하는 방법이 있습니다. Remix(<https://remix.ethereum.org>)는 브라우저를 통해 항상 최신 버전을 사용할 수 있기 때문에 편리하다.
- Remix는 Solidity 개발을 위한 브라우저 기반의 IDE입니다. 즉, 브라우저상에서 테스트 환경을 제공하고 작성된 Smart Contract를 실행,디버깅 할 수 있게 도와줍니다. Smart Contract의 실행은 블록체인 위에서 이루어집니다. 실제 이더리움 체인이나 테스트넷에서 실행할 수도 있고 로컬 가상머신 위에서 할 수도 있습니다.
- Remix는 3개의 영역으로 구성되어 있습니다. 가장 왼쪽이 소스파일 브라우저, 가운데가 소스코드 에디터, 오른쪽이 컨트랙트의 컴파일 및 배포,디버깅,분석 등 다양한 관련 옵션을 처리하는 영역입니다.
- Contract 조정 영역 중 Run 탭에는 Environment 메뉴가 있는데 3개의 실행 환경 모드를 제공합니다.  
: Javascript VM, Injected Web3, Web3 Provider



# Remix

## 질문) Remix는 어떻게 사용하나요?

- Environment

- 1) Web3 Provider : 외부 도구가 필요합니다. Geth가 로컬에서 작동되고 있어야 하며 이더를 할당하거나 직접 마이닝 해야 합니다.
- 2) Injected Provider : 외부 도구가 필요합니다. Mist 브라우저나 MetaMask를 통해 테스트넷에 연결하여 사용합니다.
- 3) JavaScript VM : 셋 중 가장 간단하고 편리하며, 외부도구가 필요 없으며 Remix의 메모리상에서 모든 작업이 이루어집니다.

- Account

JavaScript VM을 선택하고 Account를 보면 5개의 계정이 생성되어 있고, 각 계정에는 100ETH의 가상화폐가 지급되어 있습니다.

- Gas Limit

Gas limit는 작업이 발생시킬 가스의 양을 지정하는 것입니다. Gas limit를 작게 설정하면 실행 시 out of gas 오류를 만날 수 있음.

The screenshot shows the 'Run' tab in the Remix IDE. It features a menu bar with 'Compile', 'Run', 'Analysis', 'Testing', 'Debugger', and 'Settings'. Below the menu, there are two main sections. The top section is for 'Environment' and 'Account'. The 'Environment' dropdown is set to 'JavaScript VM'. The 'Account' dropdown is also set to 'JavaScript VM'. Below these, there is a 'Gas limit' field set to '3000000' and a 'Value' field set to '0' with a unit dropdown set to 'wei'. The bottom section is for 'Account' and 'Gas limit'. The 'Account' dropdown is set to '0xca3...a733c (100 ether)'. Below it, there is a 'Gas limit' field set to '3000000'.

Environment	Account	Gas limit	Value
JavaScript VM	0xca3...a733c (100 ether)	3000000	0

# 스마트 컨트랙트 작성

질문) 스마트 컨트랙트는 예제는 어떻게 작성하나요?

- Remix docs에 있는 예제를 통해 알아보도록 하겠습니다.

## testContract.sol (수정 전)

```
pragma solidity ^0.4.16;
contract testContract {
    uint value;
    function testContract(uint _p) {
        value = _p;
    }
    function setP(uint _n) payable {
        value = _n;
    }
    function setNP(uint _n) {
        value = _n;
    }
    function get () constant returns (uint) {
        return value;
    }
}
```

## testContract.sol (수정 후)

```
pragma solidity ^0.5.1;
contract testContract {
    uint value;
    constructor(uint _p) public {
        value = _p;
    }
    function setP(uint _n) payable public {
        value = _n;
    }
    function setNP(uint _n) public {
        value = _n;
    }
    function get () public view returns (uint) {
        return value;
    }
}
```

# 스마트 컨트랙트 작성

- Deploy

Run 탭에서 값을 넣지 않고 Deploy 합니다.

성공적으로 실행이 되면 Run 탭의 내용이 조금 바뀌게 됩니다.

내용을 살펴보면 Contract를 생성하면서 생긴 트랜잭션과 실행 비용이 소모되어 현재 선택된 계정의 이더가 100 ETH에서 조금 줄어 들어 있습니다.

또한 Contract가 성공적으로 Deploy 되어 하단에 작성한 함수를 실행시킬 수 있는 섹션이 생겼습니다.

The screenshot displays the Remix IDE interface during the deployment of a smart contract. At the top, the 'Environment' is set to 'JavaScript VM'. The 'Account' is '0xca3...a733c (99.99999999999998665)'. The 'Gas limit' is '3000000' and the 'Value' is '0 wei'. Below this, the 'testContract' is selected, and the 'Deploy' button is highlighted. The 'Load contract from Address' button is also visible. A message indicates '0 pending transactions'. At the bottom, the 'testContract at 0x692...77b3a (memory)' section is expanded, showing the 'setNP' and 'setP' functions, both with 'uint256 \_n' as the argument, and the 'get' function.

Environment	JavaScript VM
Account	0xca3...a733c (99.99999999999998665)
Gas limit	3000000
Value	0 wei

testContract

Deploy uint256 \_p

Load contract from Address At Address

0 pending transactions

testContract at 0x692...77b3a (memory)

Function	Argument
setNP	uint256 _n
setP	uint256 _n
get	

# 스마트 컨트랙트 작성

- Deploy 후 Log

로그창을 보면 testContract가 잠시 pending 되었다가 실행에 성공한 것을 볼 수 있습니다.

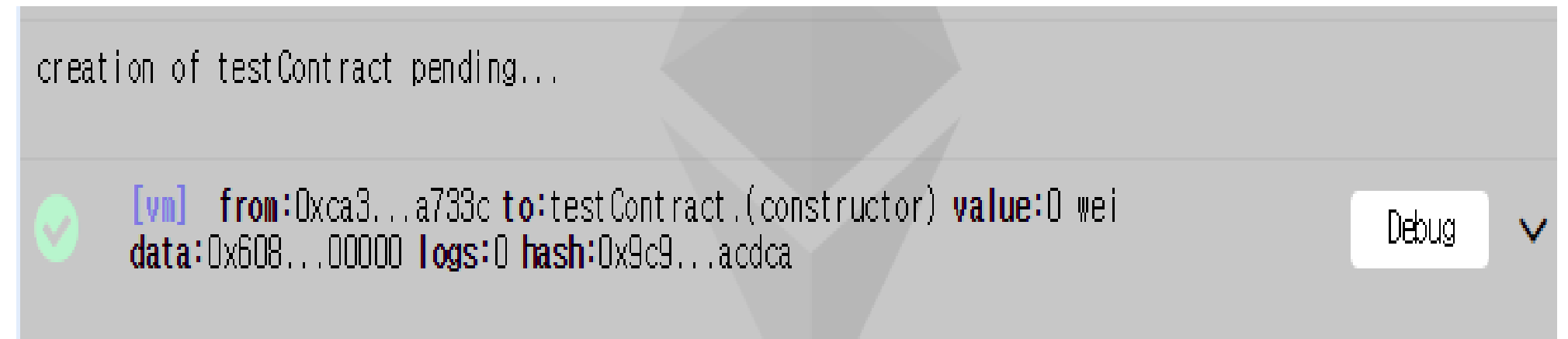
실제 이더리움 체인 위에서 라면 시간이 다소 걸리겠지만 여기에서는 JavaScript VM 위에서 동작하므로 거의 지연 없이 실행됩니다.

- 함수 실행

값을 넣지 않고 실행 했는데, testContract의 value 값이 0인 것을 확인할 수 있습니다.

setNP와 setP에 값을 넣고 빨간색 영역을 누르면 이더를 소비하면서 해당 함수를 실행합니다.

get 버튼을 누르면 value 값을 가져와 사용자에게 보여줍니다. 하지만 이 동작은 이더를 소비하지 않습니다.

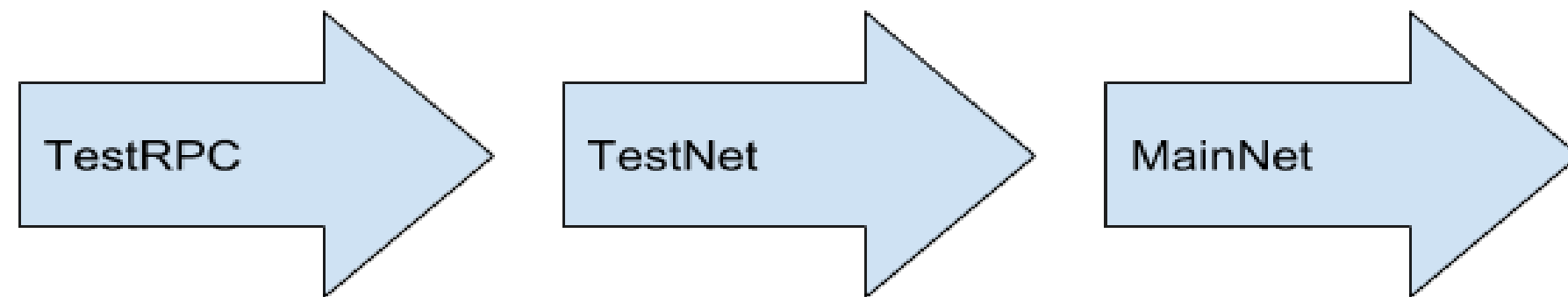


# DApp 배포

---

## 배포 환경

- "이더리움 블록체인에 배포한 컨트랙트는 지울 수 없다. 즉 한 번 배포하면 이더리움이 유지되는 한, 계속 존재한다." 즉, "분산 애플리케이션"이기 때문에, 내 컴퓨터에서 지웠다고 다른 사람들 컴퓨터에서도 지워지는 것은 아니기 때문입니다.
- 이런 특징 때문에 반드시 아래와 같은 절차를 거쳐서 이더리움 메인넷에 컨트랙트를 배포해야 합니다.



# DApp 배포

---

## 배포 환경

### 1. TestRPC(Ganache)와 같은 가상 환경에서 솔리디티 컨트랙트 테스트

: 가상 환경은 실제 블록체인이 아니며, 채굴 없이도 마음대로 이더를 발급하고 컨트랙트를 배포한 후 지우는 기능이 가능함

### 2. 테스트넷에 배포

: 실제 블록체인이어서 채굴을 통해 이더를 생성해야 하지만, 채굴 난이도가 매우 낮음. 다양한 종류의 테스트넷이 있으며, 물론 테스트넷에서 받은 이더는 실제 이더리움 메인넷에서는 사용할 수 없음

### 3. 이더리움 메인넷에 배포

: 컨트랙트 생성 등 모든 트랜잭션이 발생할 때마다 가스를 소모해야 함. 즉, 실제 이더를 지불해야 함.

# 이더리움 로컬 개발환경

## • 1.Node.js 설치하기

Node.js 를 현재 기준 LTS 버전인 v10 버전을 설치하세요.  
윈도우의 경우에 [노드 공식 홈페이지](#) 다운로드 페이지 에서 설치를 하면 됩니다.

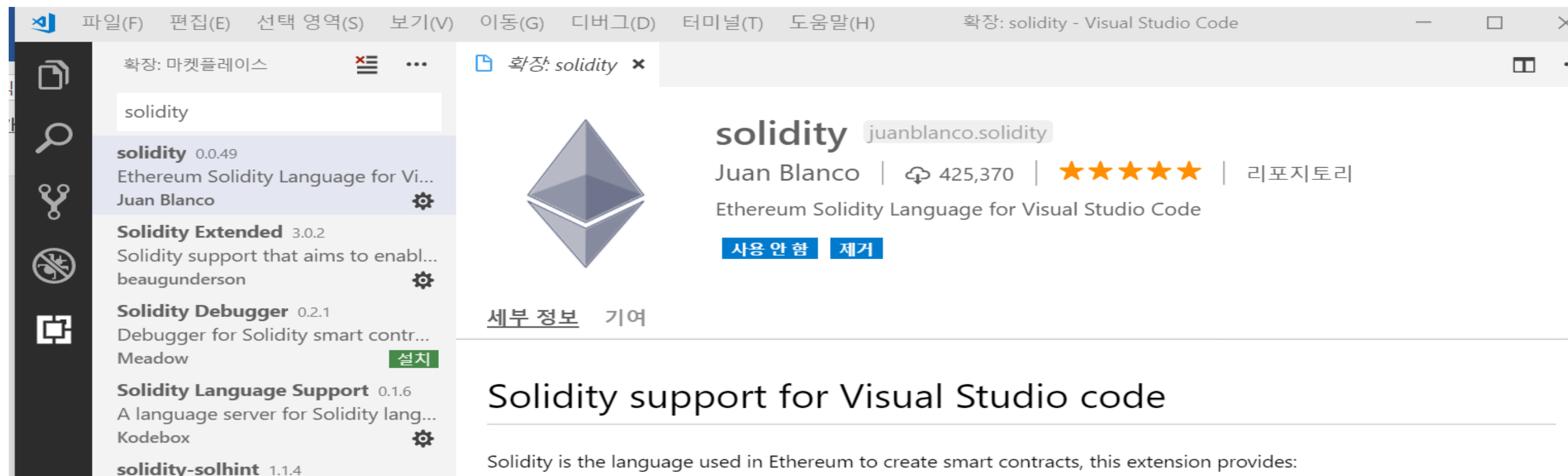
nodejs 8이상, npm 5 이상 이면 가능합니다.

## • 2.Visual Studio Code 설치 및 Plug In 설치하기

VS Code 설치는 [Visual Studio Code](#) 에서 하실 수 있습니다.

VS Code의 확장 프로그램(플러그인) 설치

: Solidity 언어 확장팩 설치

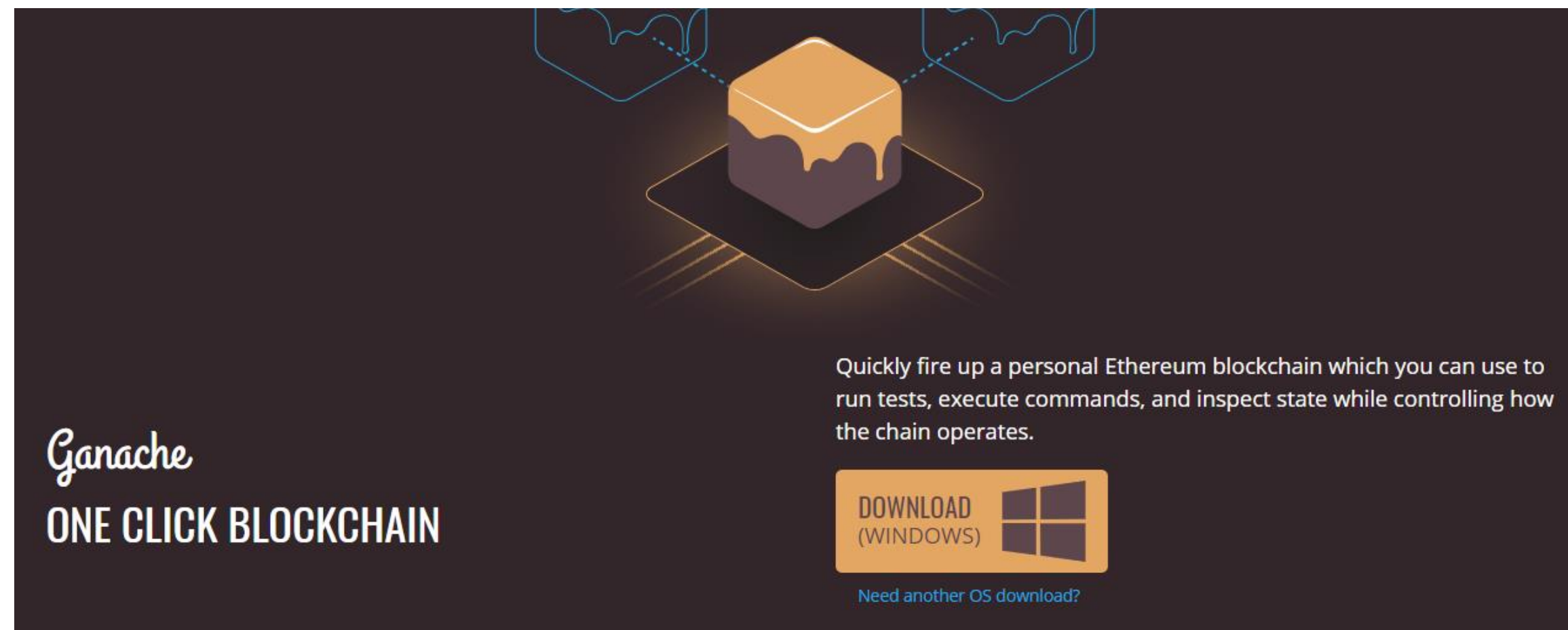


# 이더리움 로컬 개발환경

## 3. Ganache란? <https://truffleframework.com/ganache>

Ganache는 이더리움의 로컬 테스트 네트워크를 구축하는 소프트웨어로 원래의 이름은 TestRPC다. 개발을 위해 geth 또는 패리티 같은 클라이언트를 사용하면 각 트랜잭션을 실행하는 데 15 초씩 걸리기 때문에 개발 속도가 느려질 수 있습니다.

이 문제를 해결하기 위해 일반적으로 개발 목적으로 가나슈(ganache)라는 메모리 내 블록체인을 사용할 수 있습니다. 즉, Ganache는 이더리움 개발을 위한 개인 블록체인으로 계약을 배포하고, 앱을 개발해 실행 테스트를 할 수 있는 개인 블록체인이다. (프라이빗 블록체인은 아님)






# 이더리움 로컬 개발환경

## 3-1. Ganache 설치하기 <https://truffleframework.com/ganache>

Ganache는 윈도우 버전 뿐만 아니라 맥과 리눅스 버전도 있으니 자신의 운영체제에 맞는 것을 다운로드 받으면 된다.

다운로드 받은 파일(Ganache-1.x.x.appx)을 윈도우10에서 실행시키면 간단한 설치 동의와 함께 다음과 같은 화면을 확인할 수 있다.

 Ganache

ACCOUNTS

BLOCKS

TRANSACTIONS

LOGS

UPDATE AVAILABLE

SEARCH FOR BLOCK NUMBERS OR TX HASHES

CURRENT BLOCK  
0

GAS PRICE  
20000000000

GAS LIMIT  
6721975



NETWORK ID  
5777

RPC SERVER  
HTTP://127.0.0.1:8545

MINING STATUS  
AUTOMINING

MNEMONIC ?  
shaft clap gun expire course crouch magnet furnace grant shop used vacant

HD PATH  
m/44'/60'/0'/0/account\_index

ADDRESS 0x20BB5789f444e47a88c366f0bfE41EcB3c75BD4C	BALANCE 100.00 ETH	TX COUNT 0	INDEX 0	
ADDRESS 0x2DcCa9B61E50D79A90a813fcD6a42c3A3Ac52e6f	BALANCE 100.00 ETH	TX COUNT 0	INDEX 1	

# 이더리움 로컬 개발환경

4.Truffle이란? <http://truffleframework.com/>

SpringMVC, Ruby on Rails, Django 등과 같은 웹 애플리케이션 개발을 위한 프레임워크처럼, Truffle은 DApps 개발에 가장 널리 사용되는 프레임워크 중 하나로서 블록체인에서 스마트 계약을 컴파일하고 배포하는 복잡성을 많이 추상화 해줍니다. 즉, **Truffle은 이더리움을 위한 개발 테스트 프레임워크로서, 스마트컨트랙트 개발과 컴파일, 블록체인에 컨트랙트를 배포하는 스크립트 마이그레이션, 스마트컨트랙트 테스트 등을 지원한다**

## TRUFFLE

SMART CONTRACTS MADE SWEETER

A world class development environment, testing framework and asset pipeline for blockchains using the Ethereum Virtual Machine (EVM), aiming to make life as a developer easier.

LEARN MORE

GITHUB REPO

DOCS

# 이더리움 로컬 개발환경

---

4-1.Truffle 설치하기 <http://truffleframework.com/>

```
> npm install -g truffle@4.1.15  
> truffle version
```

## \* Truffle의 역할

- Built-in smart contract compilation, linking, deployment and binary management.
- Automated contract testing for rapid development.
- Scriptable, extensible deployment & migrations framework.
- Network management for deploying to any number of public & private networks.
- Package management with EthPM & NPM, using the ERC190 standard.
- Interactive console for direct contract communication.
- Configurable build pipeline with support for tight integration.
- External script runner that executes scripts within a Truffle environment.

# 이더리움 로컬 개발환경

---

5.web3.js란? <https://github.com/ethereum/web3.js>

web3.js는 Ethereum JavaScript API로서, 블록체인과 상호작용 할 수 있도록, 트랜잭션을 만들고 스마트 컨트랙트를 호출하는 것을 지원한다.

이 API는 이더리움 클라이언트와 커뮤니케이션하는 것을 추상화 하고, 개발자가 어플리케이션의 콘텐츠에 보다 초점을 맞추게 한다. 이를 위해, web3 인스턴스를 브라우저에 임베드 시켜야 한다.

**web3.js is a collection of libraries which allow you to interact with a local or remote ethereum node, using a HTTP or IPC connection.**

즉, web3.js는 이더리움 JSON RPC를 구현한 자바스크립트 API라고 할 수 있다.





판교 | 경기도 성남시 분당구 삼평동 대왕판교로 670길 유스페이스2 B동 8층 T. 070-5039-5805  
가산 | 서울시 금천구 가산동 371-47 이노플렉스 1차 2층 T. 070-5039-5815  
웹사이트 | <http://edu.kosta.or.kr> 팩스 | 070-7614-3450