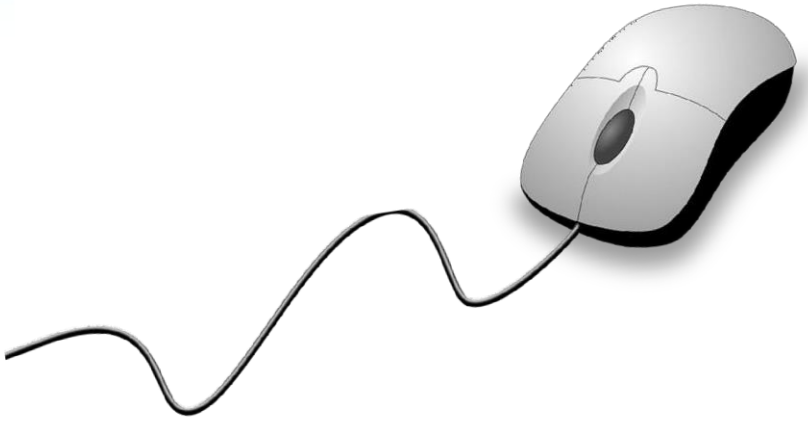
공개SW 솔루션 설치 & 활용 가이드

응용SW > 콘텐츠 배포



TRUFFLE

제대로 배워보자

How to Use Open Source Software

Open Source Software Installation & Application Guide

OPENUP

오픈소스 소프트웨어 통합지원센터

Open Source Software Support Center

CONTENTS

1. 개요

소개	• 이더리움 Dapp을 작성하고 사용하려면 스마트 컨트랙트를 작성 후 컴파일하고, 네트워크에 배포하고, 배포된 스마트 컨트랙트를 연결해야 하는 과정을 거치게 된다. 이러한 복잡한 작업들을 쉽게 처리할 수 있도록 스마트 컨트랙트 컴파일, 배포 및 테스트 등의 기능을 제공한다.		
주요기능	• 컨트렉트 컴파일 • 컨트렉트 배포 • 컨트렉트 디버그 • 컨트렉트 테스트		
대분류	• 응용SW	**소분류**	• 콘텐츠 배포
라이선스형태	• The MIT License (MIT)	**사전설치 솔루션**	• Node.js
		버전	• 5.0.36(2019년 9월 기준)
특징	•컴파일 된 컨트랙트를 이더리움 네트워크에 배포(depoly) 및 관리, SmartContract 개발 중 디버깅 등을 할 수 있도록 지원하는 개발 환경 도구이다. • 테스트 케이스를 작성해서 빠르고 손쉽게 테스트를 진행해 볼 수 있다.		
개발회사/커뮤니티	• Truffle Suite / https://gitter.im/ConsenSys/truffle		
공식 홈페이지	• https://www.trufflesuite.com/docs/truffle/overview		

2. 기능요약

TRUFFLE

truffle 주요 기능

컨트랙트 컴파일	스마트 컨트랙트 컴파일을 진행할 수 있다.
컨트랙트 배포	Solidity로 작성된 스마트 컨트랙트를 블록체인 네트워크 상에 손쉽게 배포 할 수 있다.
컨트랙트 테스트	컨트랙트를 테스트 환경에 배포 후 일일이 테스트 하려면 많은 시간이 소요된다. Truffle을 사용하면, 다양한 테스트케이스를 빠르고 쉽게 실행할 수 있다.

3. 실행환경

- ✓ standard JSON RPC API를 지원하는 Ethereum Client가 있어야 함.
- NodeJs v8.9.4 이상
- Windows, Linux 또는 Max OS X

4. 설치 및 실행

세부 목차

4. 설치 및 실행

4.1 Truffle 설치

- 사전에 npm, node(v8.9.4 이상)가 설치되어 있어야 함.

npm -g install truffle

4. 설치 및 실행

4.2 Truffle Init

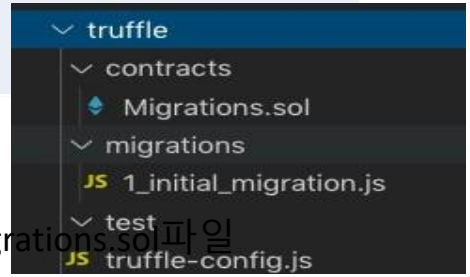
truffle init

```
[redacted] $ truffle init

This directory is non-empty...
? Proceed anyway? Yes
✔ Preparing to download
✔ Downloading
✔ Cleaning up temporary files
✔ Setting up box

Unbox successful. Sweet!

Commands:

  Compile:        truffle compile
  Migrate:        truffle migrate
  Test contracts: truffle test
```

4. 설치 및 실행

4.3 Truffle init 후 폴더구조

파일 /폴더명	비고
Contracts	Solidity로 개발된 스마트 컨트랙트 소스 폴더
Contracts/Migrations.sol	배포를 도와주는 Solidity 파일
migrations	배포를 위한 스크립트 파일 폴더
migrations/1_initial_migration.js	Migrations.sol을 배포하는 스크립트
test	개발된 스마트 컨트랙트를 테스트 하기 위한 테스트 케이스 파일 폴더
truffle-confing.js	Truffle 설정 파일



- Migrations/1_*.js : 배포시 접두사로 붙여진 숫자에 따라 순차적으로 실행한다. -1_initial_migration.js 파일은 스마트 계약의 마이그레이션을 관찰하기 위해 Migrations.sol파일을 배포하며, 나중에 변경되지 않은 컨트랙트를 이중 마이그레이션하지 않도록 보장한다.(1_initial_migration.js, Migrations.sol 파일은 수정, 삭제 하지 않기를 권한다.)

5. 기능소개

세부 목차

1. 스마트 컨트랙트 컴파일
2. 스마트 컨트랙트 배포
3. 스마트 컨트랙트 테스트
4. Truffle 명령어

5. 기능소개

1. 스마트 컨트랙트 컴파일(1/2)

- contracts 폴더에 Solidity로 스마트 컨트랙트를 작성한다.
- Solidity 소스코드를 컴파일하면 byte code 형태로 변환된다.
- 스마트 컨트랙트를 컴파일 하기 위해선, contract 폴더 안에서 다음 명령어를 실행한다.

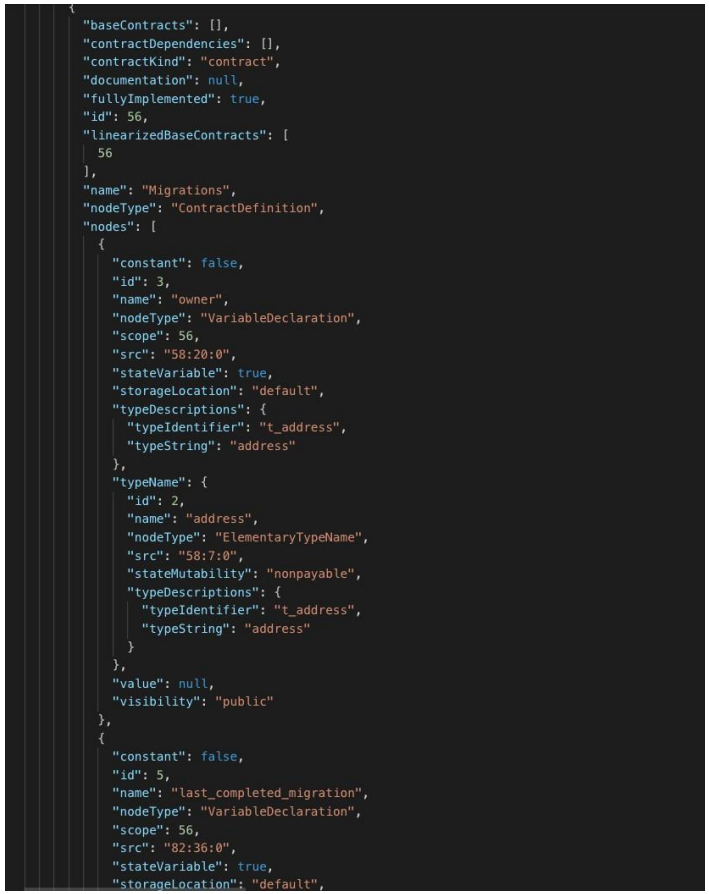
•contracts/ 폴더 내에 모든 파일이 컴파일이 진행된다. #

truffle compile

```
Compiling your contracts...
===========================
> Compiling ./contracts/Migrations.sol
> Artifacts written to /Users/gimgijeong/Desktop/truffle/test2/build/contracts
> Compiled successfully using:
   - solc: 0.5.8+commit.23d335f2.Emscripten.clang
```

5. 기능소개

5.1 스마트 컨트랙트 컴파일(2/2)



Migrations.json

- Compile이 완료 되면 build /contracts폴더 내에 json파일이 생성된 것을 확인할 수 있다.
- json파일에는 컨트랙트의 ABI정보, bytecode 등 배포할 때 필요한 정보가 들어있다.

5. 기능소개

5.2 스마트 컨트랙트 배포(1/2)

TRUFFLE

```
const Migrations = artifacts.require("Migrations");

module.exports = function(deployer) {
  deployer.deploy(Migrations);
};
```

- artifacts.require()는 nodedml require과 비슷한 메소드이다.
- artifacts.require() 로 배포를 원하는 컨트랙트의 정보를 획득 한 후 deployer.deploy()를 사용해서 배포를 한다.(해당 파일은 Migrations 컨트랙트를 배포하는 파일)

5. 기능소개

5.2 스마트 컨트랙트 배포(2/2)

- 작성된 스마트 컨트랙트를 블록체인으로 배포한다.

- # truffle migrate

- 배포가 완료되면 transaction 주소 및 block 에 대한 정보가 출력된다.

- 이미 배포를 하였고 배포를 재설정 하려면 다음 명령어 입력한다.

  # truffle deploy --reset 또는

  # truffle migrate –compile-all -reset

```
Compiling your contracts...
===========================
> Compiling ./contracts/Migrations.sol
> Artifacts written to /Users/gimgijeong/Desktop/truffle/test3/build/contracts
> Compiled successfully using:
   - solc: 0.5.8+commit.23d335f2.Emscripten.clang

gimgijecBookPro:contracts gimgijeong$ cd ..
gimgijecBookPro:test3 gimgijeong$ ls
build                   contracts               migrations              test                    truffle-config.js
gimgijecBookPro:test3 gimgijeong$ truffle migrate

Compiling your contracts...
===========================
> Everything is up to date, there is nothing to compile.



Starting migrations...
======================
> Network name:    'development'
> Network id:      5777
> Block gas limit: 0x6691b7


1_initial_migration.js
======================

   Deploying 'Migrations'
   ----------------------
   > transaction hash:    0x5b5a5dd5c8136d84a38a2f05bdfdf4ce06e5689ebad5b6ca9b135cadef139ee7
   > Blocks: 0            Seconds: 0
   > contract address:    0x75ea2b1e0Ec3B42fE6De4a1950f374fbd25e646E
   > block number:        13
   > block timestamp:     1568962807
   > account:             0x6FC742B52CB9eF1f917978575C79b79B6bcCF2cd
   > balance:             99.96334578
   > gas used:            261393
   > gas price:           20 gwei
   > value sent:          0 ETH
   > total cost:          0.00522786 ETH


   > Saving migration to chain.
   > Saving artifacts
   -------------------------------------
   > Total cost:          0.00522786 ETH


Summary
=======
> Total deployments:   1
> Final cost:          0.00522786 ETH
```

5. 기능소개

5.3 스마트 컨트랙트 테스트

작성된 test 케이스가 작성된 파일을 실행시켜 작성한 스마트 컨트랙트가 잘 작성되었는지 test 한다. (테스트는 .js 또는 .sol 파일로 작성한다.)

truffle test

```
Contract: SampleToken
  ✓ has a name
  ✓ has a symbol
  ✓ has 18 decimals
  ✓ has 1e+24 total supply
  ✓ assigns the initial total supply to the creator (86ms)
  ✓ transfer token to the investor (139ms)
  ✓ transfer token to the investor (93ms)
  ✓ should reject transfer token(more than has) to the investor (51ms)


8 passing (2s)
```

Cannot find module 'web3' error 발생시, web3모듈을 설치한다.

npm install web3

5. 기능소개

5.4 Truffle 명령어(1/3)


```
Usage: truffle <command> [options]

Commands:
  build     Execute build pipeline (if configuration present)
  compile   Compile contract source files
  config    Set user-level configuration options
  console   Run a console with contract abstractions and commands available
  create    Helper to create new contracts, migrations and tests
  debug     Interactively debug any transaction on the blockchain (experimental)
  deploy    (alias for migrate)
  develop   Open a console with a local development blockchain
  exec      Execute a JS module within this Truffle environment
  help      List all commands or provide information about a specific command
  init      Initialize new and empty Ethereum project
  install   Install a package from the Ethereum Package Registry
  migrate   Run migrations to deploy contracts
  networks  Show addresses for deployed contracts on each network
  obtain    Fetch and cache a specified compiler
  opcode    Print the compiled opcodes for a given contract
  publish   Publish a package to the Ethereum Package Registry
  run       Run a third-party command
  test      Run JavaScript and Solidity tests
  unbox     Download a Truffle Box, a pre-built Truffle project
  version   Show version number and exit
  watch     Watch filesystem for changes and rebuild the project automatically
```

5. 기능소개

5.4 Truffle 명령어(2/3)

명령어	비고	명령어	비고
Build	컨트랙트 빌드	debug	블록체인상에 배포된 트랜잭션 디버깅
compile	컨트랙트 컴파일	deploy	컨트랙트 배포
config	user-level 옵션 설정	develop	개발모드로 접속 (ganache-cli 실행)
console	truffle 콘솔 접속	exec	Truffle 환경에서 JS모듈을 실행
create	새로운 컨트랙트, 마이그레이션, 테스를 생성	help	모든 명령 목록이나 특정 명령에 대한 정보를 표시

5. 기능소개

5.4 Truffle 명령어(3/3)

명령어	비고	명령어	비고
init	Truffle 프로젝트 초기화	publish	Ethereum Package Registry에 패키지 publish
install	Ethereum Package Registry에서 패키지를 설치	run	Third-party plugin command 실행
migrate	컨트랙트 배포	test	Solidity로작성된 컨트랙트 테스트
networks	각 네트워크에 배포된 컨트랙트의 주소 표시	unbox	사전 구축 된 Truffle 프로젝트 인 Truffle Box를 다운로드
opcode	컨트랙트에 대해 컴파일된 opcode를 표시		

6. 활용예제

세부 목차

6. 활용예제

1. 예제 설명

- Ganache를 이용하여 가상의 이더리움 블록체인 환경을 구성하고, Truffle프레임워크에서 컴파일, 블록체인(Ganache)으로 배포, 테스트 등을 진행한다.

- Truffle box의 pet-shop tutorial을 이용하여 간단한 스마트 컨트랙트 컴파일, 배포 및 테스트를 진행한다.
- Truffle box 란, 다른 사람들이 프로젝트 템플릿을 제작해 놓은 것이다.
- Truffle box 내 pet-shop 예제를 활용한다.

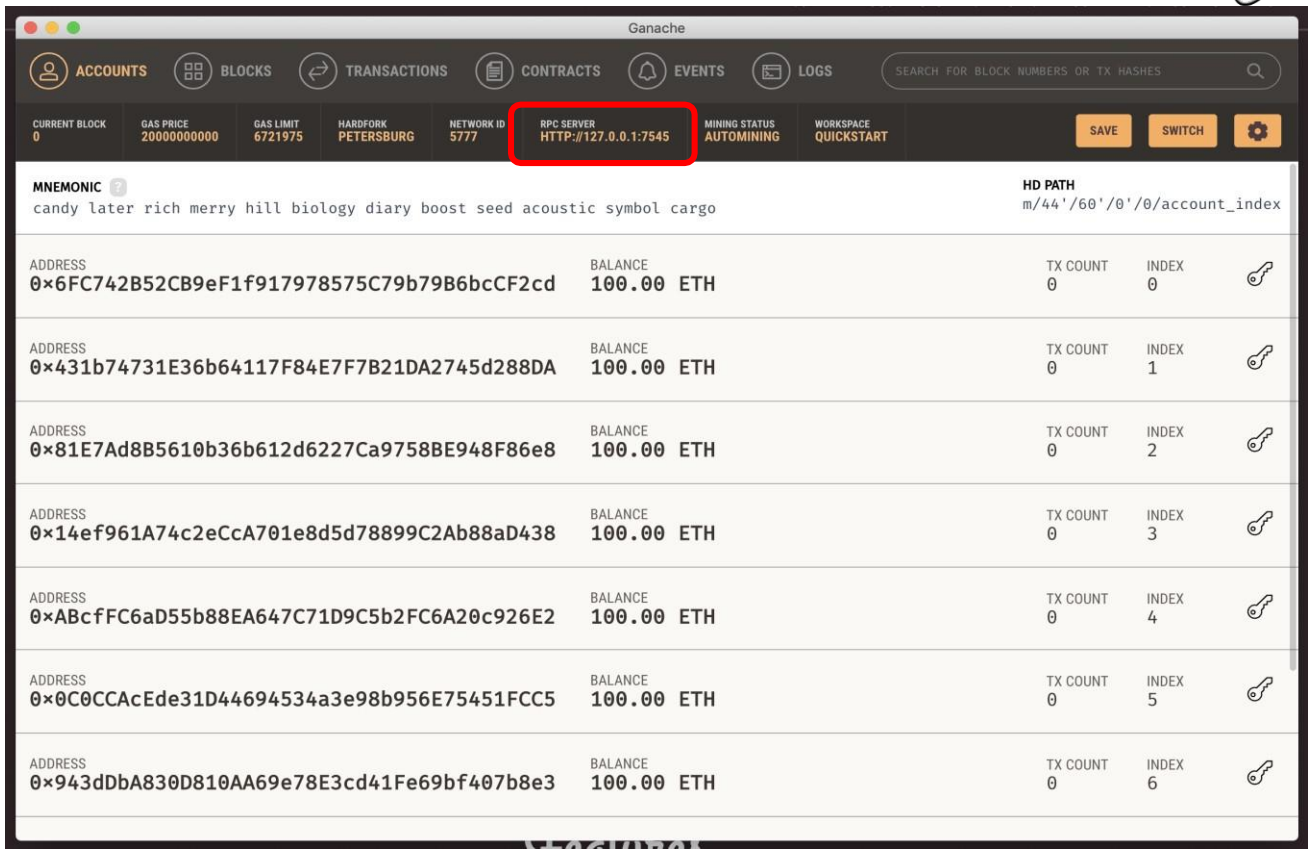
6. 활용예제

2. Ganache 설치 및 실행하기(1/2)

- Ganache: “가나슈” 라고 읽는다. 가상의 이더리움 네트워크를 생성해 스마트 컨트랙트를 실행할 수 있도록 해주는 프로그램이다. Public Blockchain이 아니라, 혼자서만 실행해서 테스트 해볼수 있는 Personal Ethereum Blockchain이다.
- Ganache공식 사이트(https://www.trufflesuite.com/ganache)에서 다운로드 받는다.

6. 활용예제

6.2 Ganache 설치 및 실행하기(2/2)



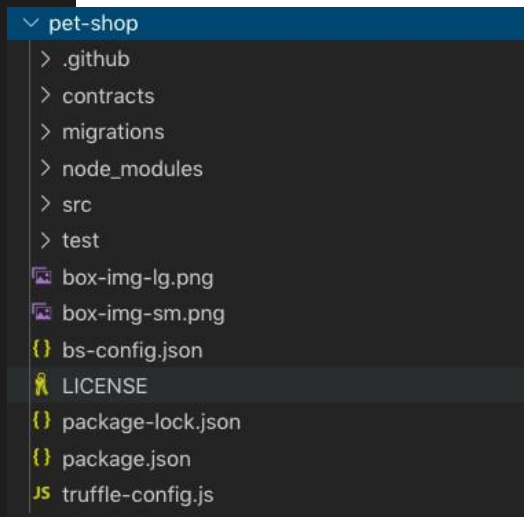
- 설치를 완료하면 다음과 같은 화면이 나타나며, 가상의 이더리움 네트워크가 생성되었다.
- Default로 10개의 계정에 100eth가 잔액이 존재한다.
- 해당 네트워크는 http://127.0.0.1:7545 이다.

6. 활용예제

3. truffle unbox

- 디렉토리 생성 후 Pet-shop tutorial를 unbox 합니다.
- # truffle unbox pet-shop

```
✓ Preparing to download
✓ Downloading
✓ Cleaning up temporary files
✓ Setting up boxnpm WARN pet-shop@1.0.0 No description
npm WARN pet-shop@1.0.0 No repository field.

✓ Setting up box

Unbox successful. Sweet!

Commands:

  Compile:        truffle compile
  Migrate:        truffle migrate
  Test contracts: truffle test
  Run dev server: npm run dev
```



6. 활용예제

4. 스마트 컨트랙트 작성하기

TRUFFLE

- contracts 폴더 내에 Adoption.sol 파일을 추가 후 컨트랙트를 작성합니다.

```
pragma solidity ^0.5.0;

contract Adoption {

  address[16] public adopters;

  // Adopting a pet
  function adopt(uint petId) public returns (uint) {
    require(petId >= 0 && petId <= 15);

    adopters[petId] = msg.sender;

    return petId;
  }

  // Retrieving the adopters
  function getAdopters() public view returns (address[16] memory) {
    return adopters;
  }
}
```

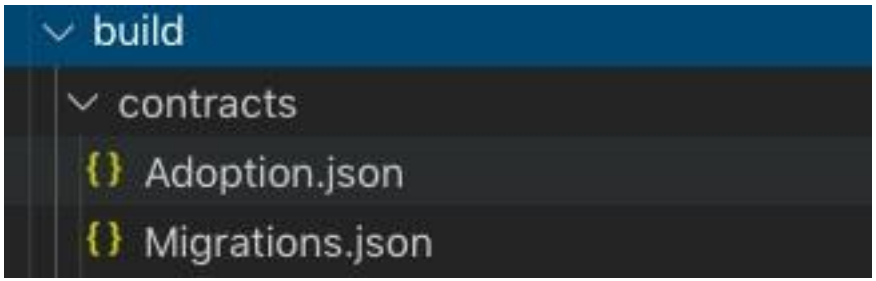
- Function adopt: 애완동물 입양을 요청할 수 있도록 허용
- Function getAdopters: 입양한 pet 소유자 주소 반환

6. 활용예제

5. 스마트 컨트랙트 컴파일하기

- 작성된 컨트랙트를 EVM(Ethereum Virtual Machian)에서 실행 하기 위해 bytecode로 컴파일을 진행한다.
- # truffle compile

```
Compiling your contracts...
===========================
> Compiling ./contracts/Adoption.sol
> Compiling ./contracts/Migrations.sol
> Artifacts written to /Users/gimgijeong/Desktop/truffle/pet-shop/build/contracts
> Compiled successfully using:
   - solc: 0.5.8+commit.23d335f2.Emscripten.clang
```



- 컴파일이 완료되면 build/contracts 내에 json파일 이 생성된다.

6. 활용예제

6.6 truffle 환경설정하기

```
module.exports = {
  networks: {
    development: {
     host: "127.0.0.1",     // Localhost (default: none)
     port: 7545,            // Standard Ethereum port (default: none)
     network_id: "*",       // Any network (default: none)
    },
  },
```

- truffle-config.js 파일 내에 배포할 때 사용할 네트워크를 알맞게 설정한다.
- Networks: 현재 Dapp의 네트워크 환경 정의
  - development: 개발용
  - Ropsten, private 등 등
- network_id: 이더리움 네트워크 고유 아이디(사설(Private) 네트워크를 사용하므로 * 로 설정)

6. 활용예제

7. 스마트 컨트랙트 배포하기(1/3)

- Migratios 디렉토리 내에 2_deploy_contracts.js 파일을 생성한다.

```
var Adoption = artifacts.require("Adoption");

module.exports = function(deployer) {
  deployer.deploy(Adoption);
};
```

6. 활용예제

7. 스마트 컨트랙트 배포하기(2/3)

TRUFFLE

•작성한 스마트 컨트랙트를 블록체인으로 migrate한다.

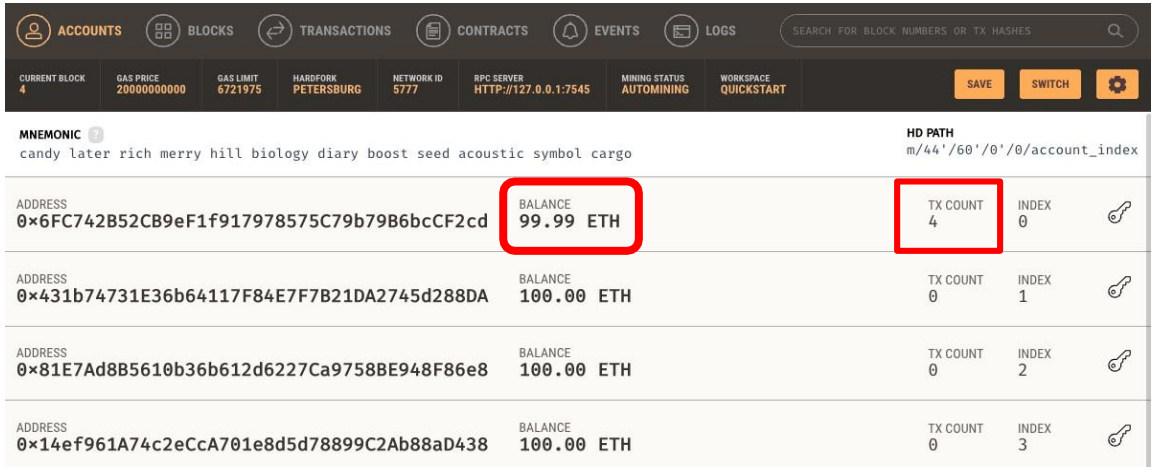
- # truffle migrate

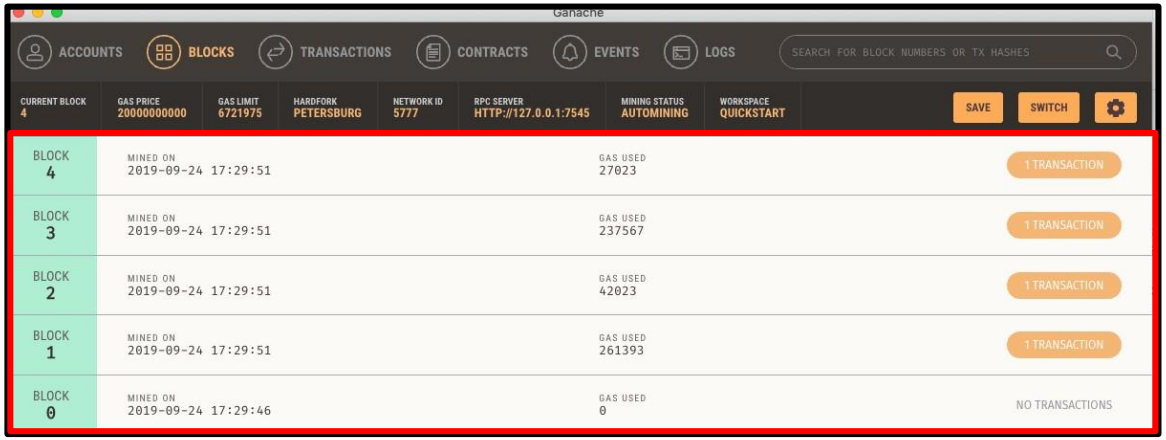
```
Compiling your contracts...
===========================
> Everything is up to date, there is nothing to compile.


Starting migrations...
======================
> Network name:    'development'
> Network id:      5777
> Block gas limit: 0x6691b7


1_initial_migration.js
======================

   Replacing 'Migrations'
   ----------------------
   > transaction hash:    0x997b57278c38ba9ca22a227fcb39df895e28838d278a2b178bfdc56db8cf58fb
   > Blocks: 0            Seconds: 0
   > contract address:    0xdfD7c8Ff2313016a2A48aa8fE768007e757EB4Ce
   > block number:        1
   > block timestamp:     1569313791
   > account:             0x6FC742B52CB9eF1f917978575C79b79B6bcCF2cd
   > balance:             99.99477214
   > gas used:            261393
   > gas price:           20 gwei
   > value sent:          0 ETH
   > total cost:          0.00522786 ETH


   > Saving migration to chain.
   > Saving artifacts
   -------------------------------------
   > Total cost:          0.00522786 ETH


2_deploy_contracts.js
=====================

   Replacing 'Adoption'
   --------------------
   > transaction hash:    0xa30325fcdb0bca4d298f2d80cab72ae5849dfb66716b33462137afa27d075749
   > Blocks: 0            Seconds: 0
   > contract address:    0x9F320CD9bfB5ad999FD472098B31191A4bCfB97d
   > block number:        3
   > block timestamp:     1569313791
   > account:             0x6FC742B52CB9eF1f917978575C79b79B6bcCF2cd
   > balance:             99.98918034
   > gas used:            237567
   > gas price:           20 gwei
   > value sent:          0 ETH
   > total cost:          0.00475134 ETH


   > Saving migration to chain.
   > Saving artifacts
   -------------------------------------
   > Total cost:          0.00475134 ETH


Summary
=======
> Total deployments:   2
> Final cost:          0.0099792 ETH
```

6. 활용예제

7. 스마트 컨트랙트 배포하기(3/3)

- 첫번째 계정에트랜잭션이 4개 증가하고, 트랜잭션 비용으로 eth가 감소했음을 확인 할 수 있다.
- 블록이 4개 추가 되었다.





6. 활용예제

8. 스마트 컨트랙트 테스트하기(1/3)

- test 폴더 내에 TestAdoption.sol 파일을 추가 후 테스트케이스를 작성한다.

```
pragma solidity ^0.5.0;

import "truffle/Assert.sol";
import "truffle/DeployedAddresses.sol";
import "../contracts/Adoption.sol";

contract TestAdoption {
  Adoption adoption = Adoption(DeployedAddresses.Adoption());

  uint expectedPetId = 8;

  address expectedAdopter = address(this);

  function testUserCanAdoptPet() public {
    uint returnedId = adoption.adopt(expectedPetId);

    Assert.equal(returnedId, expectedPetId, "Adoption of the expected pet should match what is returned.");
  }

  function testGetAdopterAddressByPetId() public {
    address adopter = adoption.adopters(expectedPetId);

    Assert.equal(adopter, expectedAdopter, "Owner of the expected pet should be this contract");
  }


  function testGetAdopterAddressByPetIdInArray() public {
    // Store adopters in memory rather than contract's storage
    address[16] memory adopters = adoption.getAdopters();

    Assert.equal(adopters[expectedPetId], expectedAdopter, "Owner of the expected pet should be this contract");
  }
}
```

8. 스마트 컨트랙트 테스트하기(2/3)

- Assert.sol: 테스트에 사용할 다양한 assertion을 제공한다.
  같은지, 같지 않은지, 테스트를 통과했는지, 실패했는지, 반환하기 위한 공백과 같은 것들을 확인한다. (전역truffle파일)
- DeployedAddresses.sol: depoly된 컨트랙트의 주소를 가져온다. (전역truffle파일)
- Adoption.sol: 테스트하려는 컨트랙트이다.

- TestUserCanAdoptPet(): adopt()함수 테스트
- TestGetAdopterAddressByPetId(): 입양한 Pet 소유자 검색 테스트
- TestGetAdopterAddressByPetIdInArray: 모든 Pet의 소유자 검색 테스트

8. 스마트 컨트랙트 테스트하기(3/3)

- 테스트를 진행한다.
- # truffle test
- 모든 테스트를 통과하면 다음과 같은 콘솔이 출력된다.

```
Using network 'development'.


Compiling your contracts...
===========================
> Compiling ./test/TestAdoption.sol



  TestAdoption
    ✓ testUserCanAdoptPet (59ms)
    ✓ testGetAdopterAddressByPetId (53ms)
    ✓ testGetAdopterAddressByPetIdInArray (72ms)


  3 passing (6s)
```

7. FAQ

Q **truffle deploy와 truffle migrate은 차이점이 무엇인가요?**

A truffle deploy는 truffle migrate의 별칭입니다.

두 명령어 모두 배포하는 역할을 하고 있습니다.

Q truffle-config.js 파일 내에 development네트워크 외에 다른 네트워크를 정의 할 경우 어떤 네트워크에 배포가 되나요?

A --networks 옵션을 사용하여 배포하려는 네트워크를 정의 하지 않은 경우 development 네트워크에 배포가 됩니다.

Q **Truffle을 windows에도 사용할 수 있나요?**

&

A 네. 사용할 수 있습니다.

설치시 오류가 발생한다면, 환경을 올바르게 구성해야 합니다.

자세한 내용은 링크참조 에서 확인 할 수 있습니다.

또한, Windows에서 명령 프롬프트를 사용하는 경우

truffle 실행 파일과 충돌 할 수 있습니다.

Windows PowerShell 또는 **Git BASH** 를 사용하는 것이 좋습니다.

8. 용어정리

용어	설 명
Solidity	Ethereum 블록체인 플랫폼 위에서 Smart Contract를 정의하기 위해 개발된 프로그래밍 언어이다. 정적타입 언어이며, 상속과 라이브러리, 사용자 정의 타입을 지원하는 등의 특징을 가지고 있다.
DAap	블록체인 위에서 돌아가는 어플리케이션이다. 오픈소스 기반이며 자율적으로 운영된다.
Ganache	가상의 이더리움 네트워크를 생성해 스마트 컨트랙트를 실행할 수 있도록 해주는 프로그램이다. Public blockchain이 아니라, 혼자서만 실행해서 테스트 해볼 수 있는 personal Ethereum blockchain 이다.
Web3js	EVM에서 작동하는 스마트 컨트랙트와 Dapp Client이 서로 통신할 수 있도록 하는 Javascript라이브러리이다

Open Source Software Installation & Application Guide