



제품소프트웨어 패키징 (Git과 GitHub를 활용한 소스코드 관리)

Git clone



학습내용

- Git clone
- Tortoise Git



학습목표

- Git clone 명령을 통해 저장소를 복사하는 방법을 설명할 수 있다.
- Tortoise Git을 활용하여 GUI환경에서 Git을 활용할 수 있다.



Git clone

1. Https를 활용한 clone

1) Git clone

- 원격 저장소에 있는 소스코드를 내 컴퓨터로 복사해 오는 역할
 - 변경한 내용을 반영하기 위한 것
 - 소스코드 파일 내용을 변경한 후 원격 저장소에 최종적으로 수정 사항을 반영하는 단계
- Git clone 명령어 사용방법
 - git clone을 입력 한 후, 한 칸을 띄워 원격 저장소의 주소를 입력하면 해당 저장소에서 소스코드를 내 컴퓨터로 복사
 - 해당 서버에 인증을 위한 Git 설정 필요
- Git clone 명령 수행
 - URL에 있는 저장소를 통째로 모두 내 컴퓨터의 작업 공간 복제
 - file, git, SSH, http, https 지원

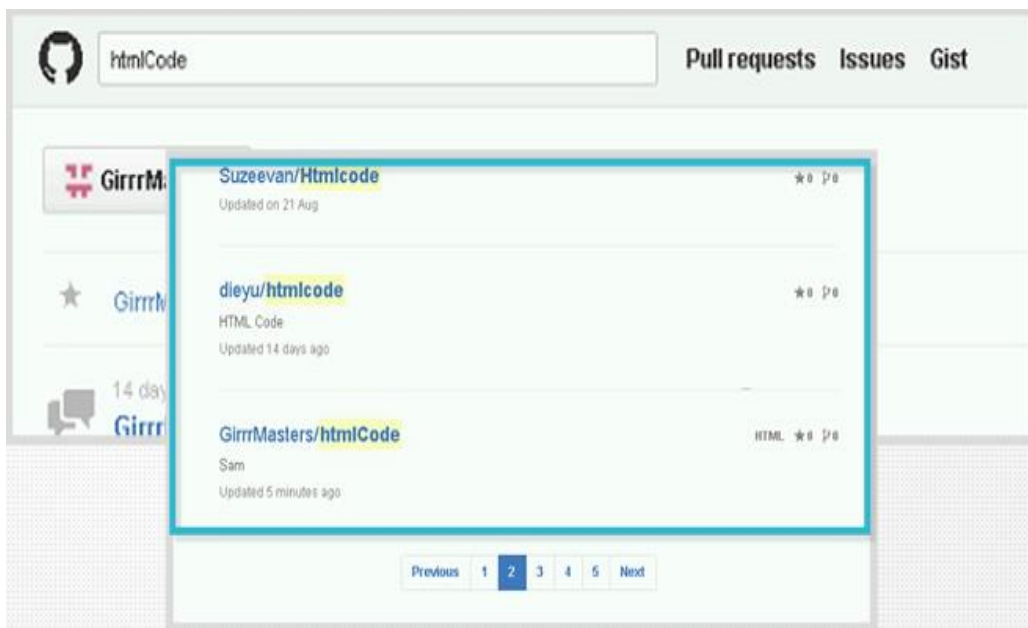


Git clone

1. Https를 활용한 clone

2) Git clone 준비

- htmlCode 저장소 clone
 - GirrrMasters 계정의 htmlCode 저장소 개설
- htmlCode 저장소 검색



- htmlCode : 원하는 키워드 입력

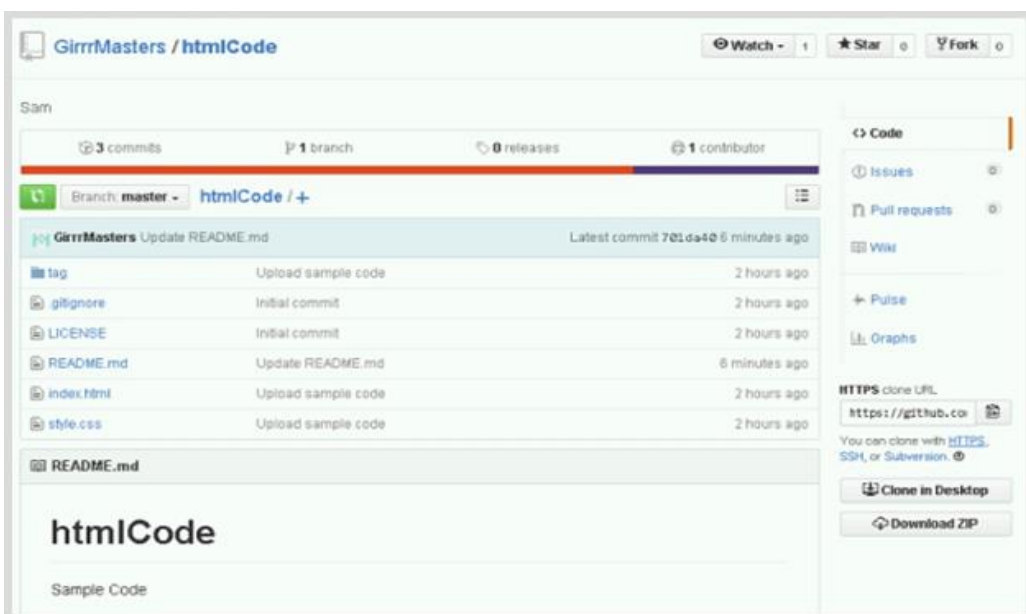


Git clone

1. Https를 활용한 clone

2) Git clone 준비

- GirrrMasters/html Code로 이동



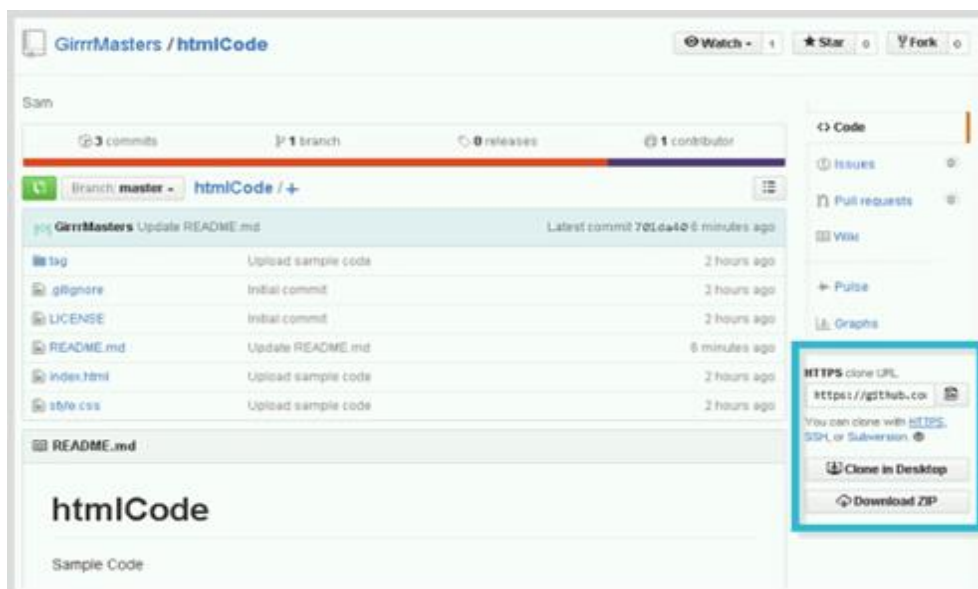


Git clone

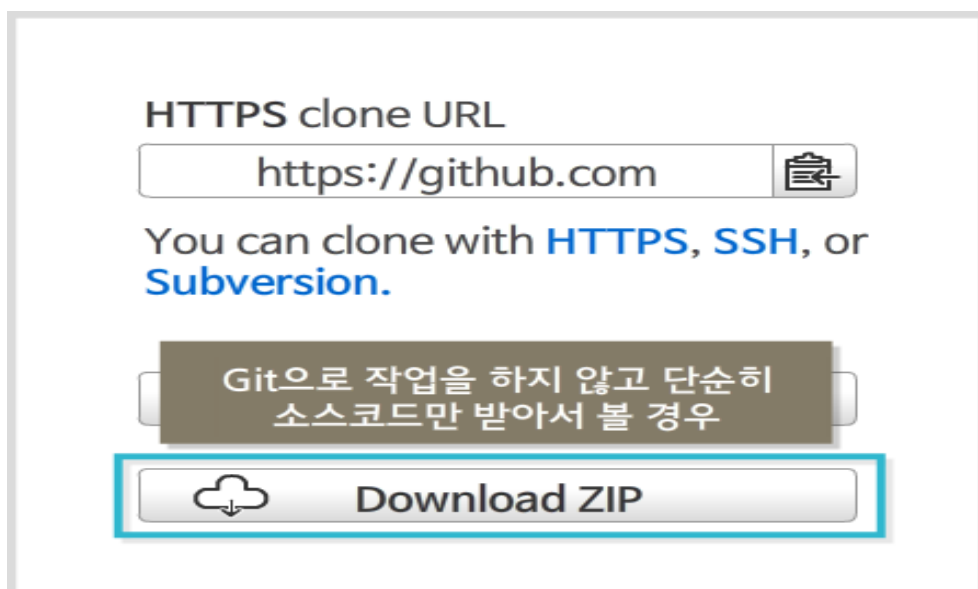
1. Https를 활용한 clone

2) Git clone 준비

- 하단 HTTPS / SSH clone URL 복사



- Git를 활용한 프로젝트는 사용자들의 접근이 용이하도록 Git clone 주소를 보기 편하게 제공



- Git으로 작업하지 않고 단순히 소스코드만 받을 경우에는 ZIP 형태로 압축하여 제공



Git clone

1. Https를 활용한 clone

3) Git bash에서 clone

- Git bash 실행
 - Git bash에서 Git clone으로 저장소 가져오기
- 사용자 지정 폴더로 이동

```
user@userpc MINGW64 ~
$ cd c:\\gitFolder
$ git clone https://github.com/GirrrMasters/htmlCode.git
Cloning into 'htmlCode'...
remote: Counting objects: 16, done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 16 (delta 5), reused 7 (delta 3), pack-reused 0
Unpacking objects: 100% (16/16), done.
Checking connectivity... done.

user@userpc MINGW64 /c/gitFolder
$ ls
developer.github.com/  girrrDev/  htmlCode/

user@userpc MINGW64 /c/gitFolder
$ cd htmlCode/

user@userpc MINGW64 /c/gitFolder/htmlCode (master)
$ ll
total 21
-rw-r--r-- 1 user None 2093 11월 11 17:57 index.html
-rw-r--r-- 1 user None 11560 11월 11 17:57 LICENSE
-rw-r--r-- 1 user None 25 11월 11 17:57 README.md
-rw-r--r-- 1 user None 1577 11월 11 17:57 style.css
drwxr-xr-x 1 user None 0 11월 11 17:57 tag/
```

```
user@userpc MINGW64 ~
$ cd c:\\gitFolder
$ git clone https://github.com/GirrrMasters/htmlCode.git
Cloning into 'htmlCode'...
remote: Counting objects: 16, done.
remote: Compressing objects: 100% (11/11), done.
$ git clone https://github.com/Girrrmaters/htmlCode.git

user@userpc MINGW64 /c/gitFolder
$ ls
developer
$ cd html
$ ll
total 21
-rw-r--r-- 1 user None 2093 11월 11 17:57 index.html
-rw-r--r-- 1 user None 11560 11월 11 17:57 LICENSE
-rw-r--r-- 1 user None 25 11월 11 17:57 README.md
-rw-r--r-- 1 user None 1577 11월 11 17:57 style.css
drwxr-xr-x 1 user None 0 11월 11 17:57 tag/
```

htmlCode 저장소의 https URL과
Git clone 명령을 이용하여
저장소를 Local 저장소 이동



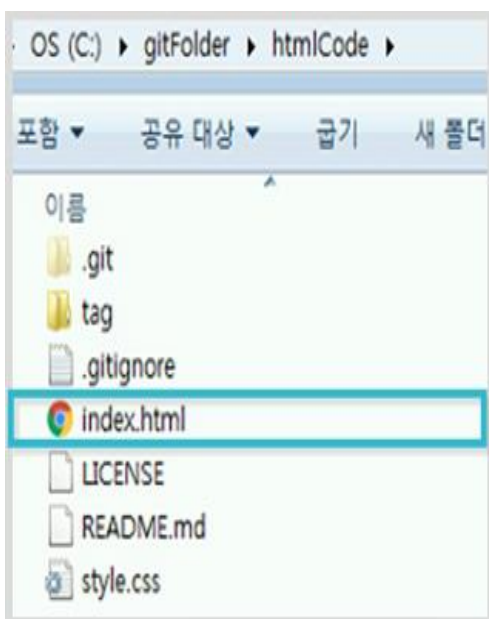
Git clone

1. Https를 활용한 clone

3) Git bash에서 clone

- 원본 저장소와 clone 저장소 비교
 - htmlCode 저장소와 GitHub의 구성이 같다면 clone이 제대로 이뤄진 것

GirrrMasters Update README.md		Latest commit 701da40 29 minutes ago
tag	Upload sample code	2 hours ago
.gitignore	Initial commit	3 hours ago
LICENSE	Initial commit	3 hours ago
README.md	Update README.md	29 minutes ago
index.html	Upload sample code	2 hours ago
style.css	Upload sample code	2 hours ago



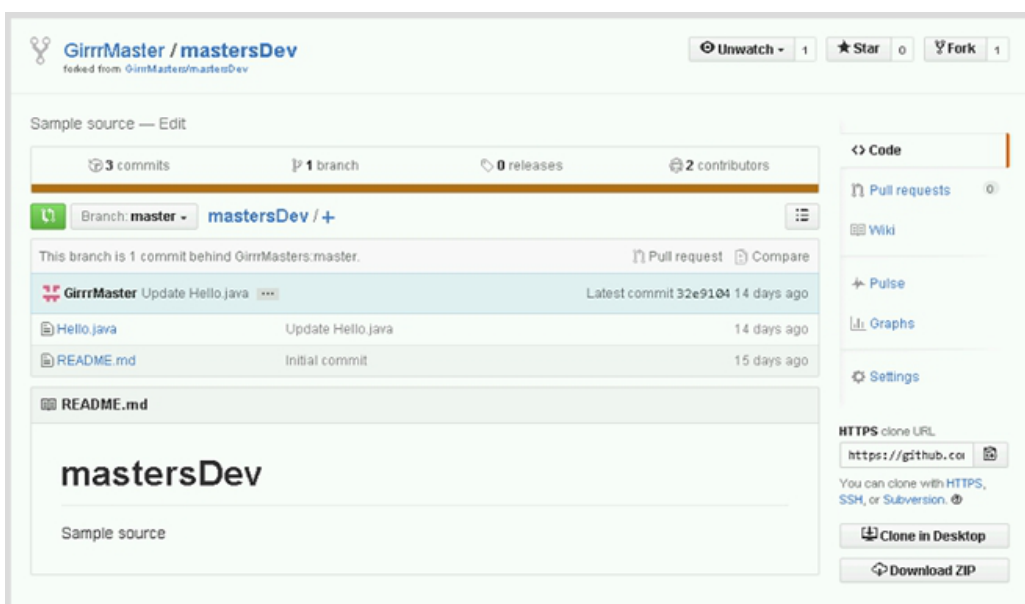


Git clone

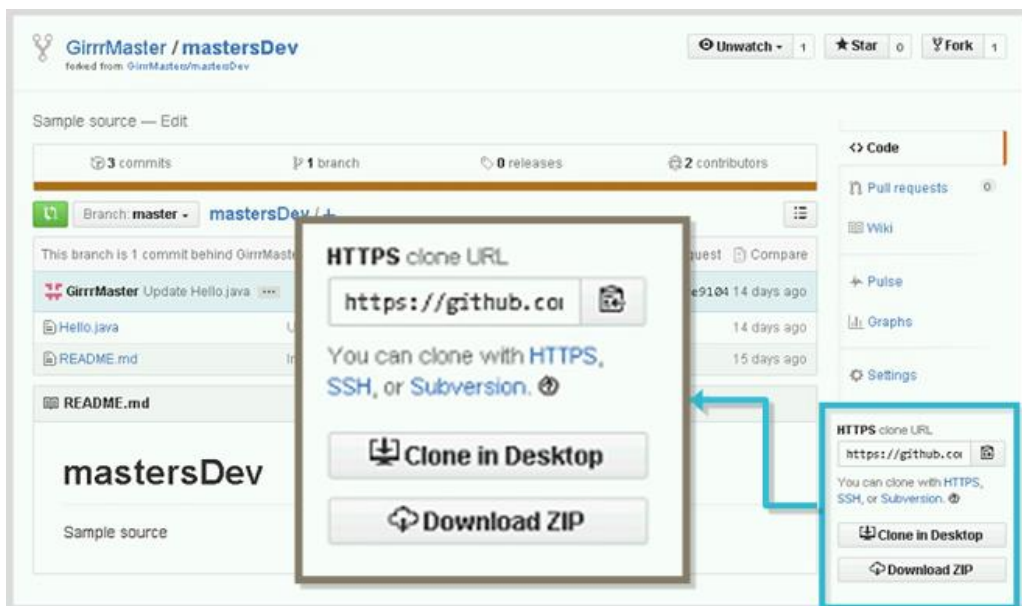
1. Https를 활용한 clone

4) Git bash에서 clone (Local 저장소 폴더 지정)

- mastersDev 저장소 선택
 - Local 저장소를 사용자가 직접 특정 위치에 생성



- HTTPS clone URL





Git clone

1. Https를 활용한 clone

4) Git bash에서 clone (Local 저장소 폴더 지정)

- Pwd로 Git bash 위치 확인

```
user@userpc ~  
$ pwd  
/c/users/  
user@userpc MINGW64 ~  
$ git clone https://github.com/GirrrMaster/mastersDev.git c:\\gitFolder\\masters  
Cloning into 'c:\\gitFolder\\masters'...  
remote: Counting objects: 9, done.  
remote: Total 9 (delta 0), reused 0 (delta 0), pack-reused 9  
Unpacking objects: 100% (9/9), done.  
Checking connectivity... done.  
user@userpc MINGW64 ~  
$ |
```

```
user@userpc MINGW64 ~  
$ pwd  
/c/Users/user  
user@userpc MINGW64 ~  
$ git clone https://github.com/GirrrMaster/mastersDev.git c:\\gitFolder\\masters  
Cloning into 'c:\\gitFolder\\masters'...  
remote: Counting objects: 9, done.  
remote: Total 9 (delta 0), reused 0 (delta 0), pack-reused 9  
Unpacking objects: 100% (9/9), done.  
Checking connectivity... done.  
user@userpc MINGW64 ~  
$ |
```

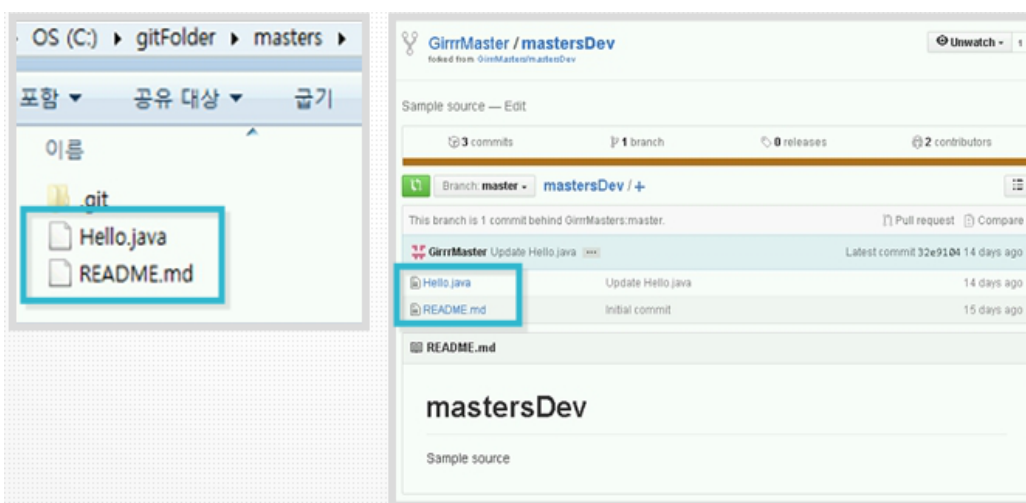


Git clone

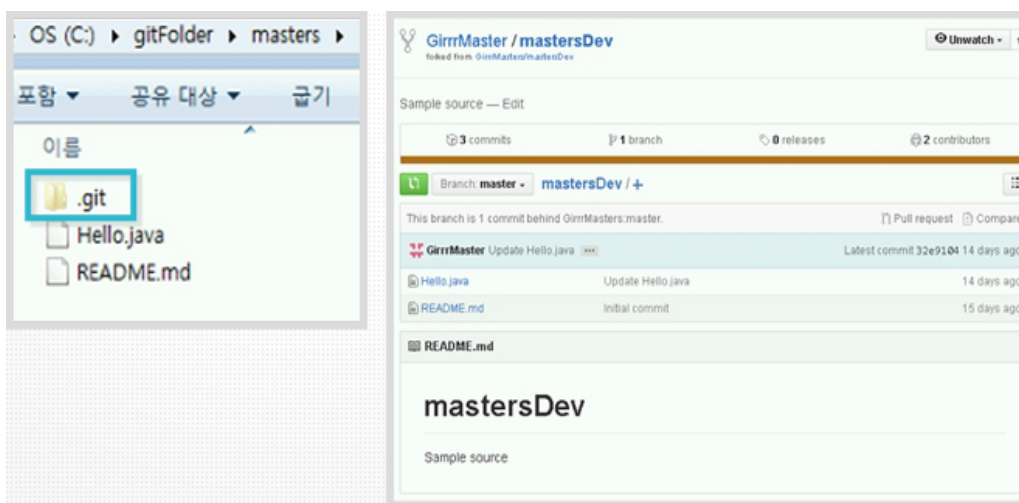
1. Https를 활용한 clone

4) Git bash에서 clone (Local 저장소 폴더 지정)

- 폴더 생성
 - 사용자가 입력한 경로에 폴더가 존재하지 않아도 명령어 실행 시 폴더를 생성하고 clone을 수행



- clone한 내용 확인
 - .git 폴더는 git 관리를 위해 다양한 정보를 저장하는 역할
 - .git 폴더가 삭제되면 일반 파일 목록으로만 인식





Git clone

2. SSH를 활용한 clone

1) SSH clone 방식의 특징

- 사용자 컴퓨터의 SSH키를 미리 생성하여 GitHub에 등록하고 해당 SSH키로 사용자 인증
 - Https를 활용한 clone은 매번 사용자 계정 및 비밀번호를 입력
- 인증키를 원격 저장소 관리자에게 전달하여 등록
- SSH 방식은 http 방식보다 좀더 보안에 강화된 형태

2) SSH clone

- SSH 폴더로 이동
- 폴더 내용 조회 및 모든 파일 삭제
 - c:\user\사용자 계정명\.ssh 폴더에 저장

```
kris8553@kr-083jslee MINGW64 ~  
$ cd ~/.ssh
```

```
kris8553@kr-083jslee MINGW64 ~/.ssh  
$ ls  
id_rsa  id_rsa.pub  known_hosts
```

```
kris8553@kr-083jslee MINGW64 ~/.ssh  
$ rm *
```

- 기존 다른 시스템에서 사용하는 SSH키가 있다면 주의



Git clone

2. SSH를 활용한 clone

2) SSH clone

- SSH키 생성
 - SSH-keygen 명령어를 이용하여 private / public 키 생성

```
krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ssh-keygen -t rsa -b 4096 -C "girrr.master@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/krjs8553/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/krjs8553/.ssh/id_rsa.
Your public key has been saved in /c/Users/krjs8553/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:rx0Tt5rh5Guf+fU8vwZb4hFt0ioXCdtVdyGjTbnxk6U girrr.master@gmail.com
The key's randomart image is:
+----[RSA 4096]-----+
|
|   +..=|
|   +OO=|
|   ...+.+|
|   =EO|
|   S.  .*=|
|   +..  *|
|   o+... *o|
|   +o+.o =Oo|
|   .B=+.. +O*|
+----[SHA256]-----+
```

- -t 옵션 : 암호화 타입 입력

```
krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ssh-keygen -t rsa -b 4096 -C "girrr.master@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/krjs8553/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/krjs8553/.ssh/id_rsa.
Your public key has been saved in /c/Users/krjs8553/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:rx0Tt5rh5Guf+fU8vwZb4hFt0ioXCdtVdyGjTbnxk6U girrr.master@gmail.com
The key's randomart image is:
+----[RSA 4096]-----+
|
|   +..=|
|   +OO=|
|   ...+.+|
|   =EO|
|   S.  .*=|
|   +..  *|
|   o+... *o|
|   +o+.o =Oo|
|   .B=+.. +O*|
+----[SHA256]-----+
```




Git clone

2. SSH를 활용한 clone

2) SSH clone

- SSH키 생성
 - -b 옵션 : 생성할 키의 비트수 지정

```
krjs8553@kr-083jslee MINGW64 ~/ssh
$ ssh-keygen -t rsa -b 4096 -C "girrr.master@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/krjs8553/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/krjs8553/.ssh/id_rsa.
Your public key has been saved in /c/Users/krjs8553/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:rx0Tt5rh5Guf+fU8vWzb4hFt0ioxCdtVdyGjTbnxk6U girrr.master@gmail.com
The key's randomart image is:
+---[RSA 4096]-----+
|
|      +..=
|      +oo =
|      ...+.+
|      =EO
|      S.  * =
|      +.. *
|      o+... * o
|      +o+.o = Oo
|      .B=+.. +o*
+-----[SHA256]-----+
```

- 각 암호화 타입마다 필요한 비트수가 다름 : rsa 타입은 최소 786 비트, default 2048 비트로 설정
- -C : 주석 입력

```
krjs8553@kr-083jslee MINGW64 ~/ssh
$ ssh-keygen -t rsa -b 4096 -C "girrr.master@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/krjs8553/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/krjs8553/.ssh/id_rsa.
Your public key has been saved in /c/Users/krjs8553/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:rx0Tt5rh5Guf+fU8vWzb4hFt0ioxCdtVdyGjTbnxk6U girrr.master@gmail.com
The key's randomart image is:
+---[RSA 4096]-----+
|
|      +..=
|      +oo =
|      ...+.+
|      =EO
|      S.  * =
|      +.. *
|      o+... * o
|      +o+.o = Oo
|      .B=+.. +o*
+-----[SHA256]-----+
```



Git clone

2. SSH를 활용한 clone

2) SSH clone

- SSH키 생성
 - \$ ls : 생성된 키 확인

```
krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ls -a
./ ../ id_rsa id_rsa.pub
```

\$ ls : 생성된 키 확인

- id_rsa : private 키

```
krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ls -a
./ ../ id_rsa id_rsa.pub
```

id_rsa : private 키

- id_rsa.pub : public 키

```
krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ls -a
./ ../ id_rsa id_rsa.pub
```

id_rsa.pub : public 키



Git clone

2. SSH를 활용한 clone

2) SSH clone

- SSH agent에 생성된 키 등록
 - \$SSH-add SSH키 위치 명령어를 사용

```
krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ls -a
./ ../ id_rsa id_rsa.pub

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ssh-add ~/.ssh/id_rsa → $SSH-add SSH키 위치 명령어
Could not open a connection

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ eval $(ssh-agent)
Agent pid 15944

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ssh-add -l
The agent has no identities.

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ssh-add ~/.ssh/id_rsa
Identity added: /c/Users/krjs8553/.ssh/id_rsa (/c/Users/krjs8553/.ssh/id_rsa)
```

- 오류 메시지 발생 : eval \$(SSH-agent) 명령어로 설정

```
krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ls -a
./ ../ id_rsa id_rsa.pub

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ssh-add ~/.ssh/id_rsa
Could not open a connection to your authentication agent.

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ eval $(ssh-agent) → eval $(SSH-agent)
Agent pid 15944

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ssh-add -l
The agent has no identities.

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ssh-add ~/.ssh/id_rsa
Identity added: /c/Users/krjs8553/.ssh/id_rsa (/c/Users/krjs8553/.ssh/id_rsa)
```



Git clone

2. SSH를 활용한 clone

2) SSH clone

- SSH agent에 생성된 키 등록
 - passphrase 키 설정

```
krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ls -a
./ ../ id_rsa id_rsa.pub

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ssh-add ~/.ssh/id_rsa
Could not open a connection to your authentication agent.

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ eval $(ssh-agent)
Agent pid 15944

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ssh-add -l
The agent has no identities.

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ssh-add ~/.ssh/id_rsa
Identity added: /c/Users/krjs8553/.ssh/id_rsa (/c/Users/krjs8553/.ssh/id_rsa)
```

passphrase 키를 설정 시 SSH-add-l 명령어를 이용하여 비밀번호 묻는 과정 생략

- 다시 SSH-add 명령어로 키를 등록

```
krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ls -a
./ ../ id_rsa id_rsa.pub

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ssh-add ~/.ssh/id_rsa
Could not open a connection to your authentication agent.

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ eval $(ssh-agent)
Agent pid 15944

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ssh-add -l
The agent has no identities.

krjs8553@kr-083jslee MINGW64 ~/.ssh
$ ssh-add ~/.ssh/id_rsa
Identity added: /c/Users/krjs8553/.ssh/id_rsa (/c/Users/krjs8553/.ssh/id_rsa)
```

SSH-add 명령어로 키 등록



Git clone

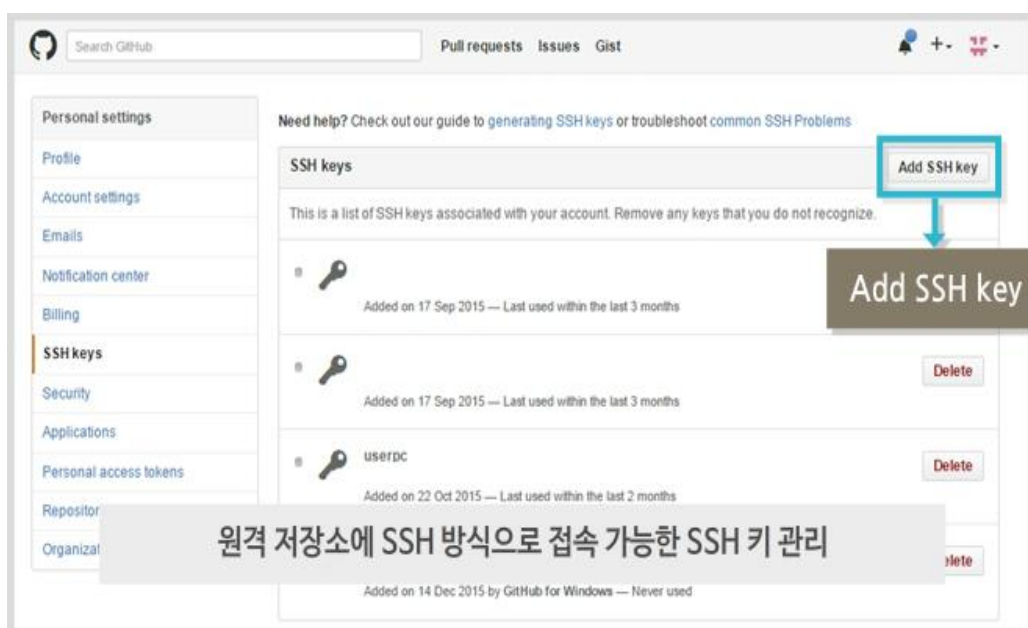
2. SSH를 활용한 clone

2) SSH clone

- SSH keys로 이동하여 생성한 key 등록
 - 간단하게 notepad를 이용하여 id_rsa.pub 파일을 열어 내용 복사



- 복사한 키는 GitHub 페이지의 옵션 → SSH keys 메뉴에서 add SSH key를 이용하여 등록





Git clone

2. SSH를 활용한 clone

2) SSH clone

- SSH keys로 이동하여 생성한 key 등록

Add an SSH key

Title

SSH-pubkey-girrrMaster

Key

ssh-rsa

XqqobYgAVSvYN1oStt5tXl48vHXW== girrr.master@gmail.com

Add key

Add key

- known_hosts 파일 생성
 - GitHub계정에 자신의 컴퓨터 SSH키로 접속 위치 알림

```
krjs85530kr-083jslee MINGW64 ~/.ssh
$ ssh -T git@github.com
Warning: Permanently added the RSA host key for IP address '192.30.252.129' to the list of known hosts.
Hi GirrrMaster! You've successfully authenticated, but GitHub does not provide shell access.
```



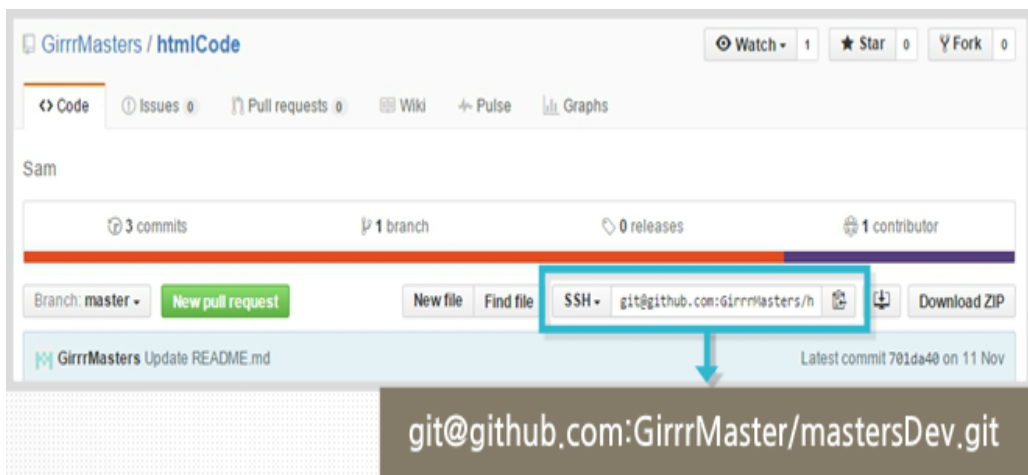


Git clone

2. SSH를 활용한 clone

2) SSH clone

- SSH 주소 복사



- Git bash 창에서 저장소 clone
 - ID, 비밀번호를 묻지 않고 clone 완료

```
krjs8553@kr-083jslee MINGW64 /c/gitFolder
$ git clone git@github.com:GirrrMaster/mastersDev.git
Cloning into 'mastersDev'...
remote: Counting objects: 14, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 14 (delta 0), reused 0 (delta 0), pack-reused 11
Receiving objects: 100% (14/14), done.
Resolving deltas: 100% (1/1), done.
Checking connectivity... done.
```




! 핵심정리



Git clone

1. Git clone

- 원격 저장소에 있는 소스코드를 내 로컬 저장소로 복사
- 원격 저장소의 프로젝트를 내 로컬 저장소에 새롭게 설정
- 프로토콜은 file, git, SSH, http(s) 가능

2. Https를 활용한 clone

- 매번 사용자 계정 및 비밀번호를 입력하여 인증

3. SSH를 활용한 clone

- 사용자 컴퓨터의 SSH키를 미리 생성하여 GitHub에 등록하고 해당 SSH키로 사용자 인증
- 인증키를 원격 저장소 관리자에게 전달하여 등록
- SSH 방식은 http 방식보다 좀더 보안에 강화된 형태



! 핵심정리



Tortoise Git

1. Tortoise Git에서 Git Clone

- Clone을 위한 폴더 생성
- 원격 저장소의 Clone 주소 받아오기
- Clone 수행
- Clone 결과 확인

2. 다른 프로젝트를 Clone해 보기

- NASA 프로젝트 다운로드
- GitHub URL 복사
- 폴더 생성 후 Clone 주소 받아오기
- README.md 파일로 Clone 결과 확인
- 특정 Branch 다운로드 방법