

Git & GitHub

hojin chu



Day1

Agenda

- Introduction
- 버전 관리 시스템
- Git 설치, 셋팅
- 로컬 저장소 사용 위한 Git 기본
- 새로운 브랜치 생성, 이동
- 각 브랜치의 독립성 확인
- 실제 프로젝트에서 발생하는 상황들


Introduction

리눅스 토발즈 리눅스(Linux) 탄생



리눅스 토발즈

- 리눅스의 창시자
 - 1991년 취미로 개발
- Git의 창시자

A photograph of two men sitting at a wooden table in a dimly lit cafe or bar. They are playing a game of Connect Four on a yellow board. The man on the left is wearing a dark jacket and is looking towards the camera with a slight smile. The man on the right is wearing a white t-shirt and is looking down at the game. A glass of beer is on the table. In the background, there are large windows showing a street scene at night with lights and trees. The text '커뮤니케이션' is overlaid in the center of the image.

커뮤니케이션

짧은 소개

- 프로젝트 호스팅 사이트
 - GitHub
 - Git을 기반.
 - Git을 보다 쉽게 관리해줄 수 있는 웹 콘솔이 필요해 깃헙을 만들
 - 2008년 2월 서비스를 시작
- 깃(Git)
 - 리눅스 프로젝트에서 사용하던 Bitkeeper 가 불편해 만들
 - 불과 2주만에 :-)
 - 2005년 Git 탄생

Version control, Source control

- 데이터의 과거와 현재 상태 관리하는 것.
- 복수의 사람들이 사용하는 것임.

Git

- 동시 다발적인 브랜치 작업.
- Git 만들기 시작한 지 3일만 Git 자체의 버전 관리 시작함.
 - 2주 무렵 처음으로 복수의 브랜치 한번에 병합하기 시작하면서 완성됨.
- 대중성, 검증된 안정성!!

왜 Git?

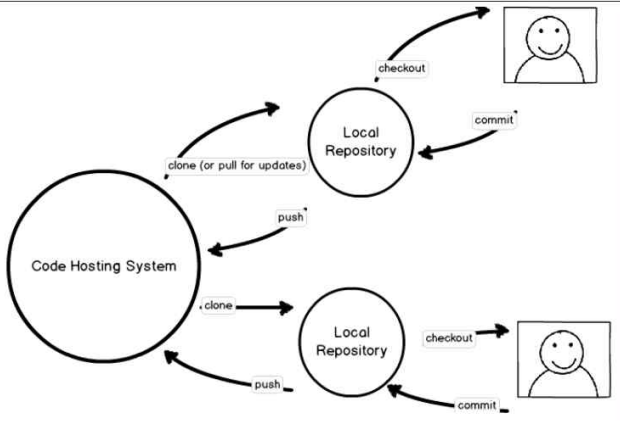
- 돈이 걸린, 프로로서 일하는 개발자, 디자이너와 협업!!

버전 관리 시스템

VCS

- Version Control System
- 프로젝트의 진행 상황을 저장.
- 협업시 서로 간에 상태를 똑같이 유지하는 기능까지 있음.

Git



분산 vcs

- 클라이언트 각자가 온전한 전체 저장소 사본을 로컬에 가짐.

Git 설치, 셋팅

git --distributed-is-the-new-centralized

Git is a **free and open source** distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is **easy to learn** and has a **tiny footprint with lightning fast performance**. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like **cheap local branching**, convenient **staging areas**, and **multiple workflows**.



Learn Git in your browser for free with **Try Git**.



About

The advantages of Git compared to other source control systems.



Documentation

Command reference pages, Pro Git book content, videos and other material.



Downloads

GUI clients and binary



Community

Get involved! Bug reporting.

Latest source Release

2.5.3

[Release Notes \(2015-09-17\)](#)

[Downloads for Linux](#)



git --distributed-even-if-your-world

About

Documentation

Blog

Downloads

GUI Clients

Logos

Community

Download for

It is easiest to install Git on Linux

Debian/Ubuntu

```
$ apt-get install git
```

Fedora

```
$ yum install git
```

Gentoo

```
$ emerge --ask --verbose
```

Arch Linux

```
$ pacman -S git
```

리눅스 Git 다운로드

- `sudo apt-get install git`

```
hojin@hojin: ~  
hojin@hojin:~$ apt-get install git
```

```
hojin@hojin: ~  
hojin@hojin:~$ apt-get install git  
E: 잠금 파일 /var/lib/dpkg/lock 파일을 열 수 없습니다 - open (13: 허가 거부)  
E: 관리 디렉터리를 (/var/lib/dpkg/) 잠글 수 없습니다. 루트 사용자가 맞습니까?  
hojin@hojin:~$ sudo apt-get install git
```

리눅스 Git 다운로드

```
hojin@hojin:~$ apt-get install git
E: 잠금 파일 /var/lib/dpkg/lock 파일을 열 수 없습니다 - open (13: 허가 거부)
E: 관리 디렉터리를 (/var/lib/dpkg/) 잠글 수 없습니다. 루트 사용자가 맞습니까?
hojin@hojin:~$ sudo apt-get install git
[sudo] password for hojin:
패키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다
상태 정보를 읽는 중입니다... 완료
다음 패키지가 자동으로 설치되었지만 더 이상 필요하지 않습니다:
  linux-headers-3.19.0-15 linux-headers-3.19.0-15-generic linux-image-3.19.0-15-generic linux-image-extra-3.19.0-15-generic
Use 'apt-get autoremove' to remove them.
다음 패키지를 더 설치할 것입니다:
  git-man liberror-perl
제안하는 패키지:
  git-daemon-run git-daemon-sysvinit git-doc git-el git-email git-gui gitk gitweb git-arch git-cvs git-mediawiki git-svn
다음 새 패키지를 설치할 것입니다:
  git git-man liberror-perl
0개 업그레이드, 3개 새로 설치, 0개 제거 및 7개 업그레이드 안 함.
3,112 k바이트 아카이브를 받아야 합니다.
이 작업 후 24.9 m바이트의 디스크 공간을 더 사용하게 됩니다.
계속 하시겠습니까? [Y/n] Y
```

리눅스 Git 다운로드

```
받기:1 http://kr.archive.ubuntu.com/ubuntu/ vivid/main liberror-perl all 0.17-1.1 [21.1 kB]
받기:2 http://kr.archive.ubuntu.com/ubuntu/ vivid/main git-man all 1:2.1.4-2.1 [700 kB]
받기:3 http://kr.archive.ubuntu.com/ubuntu/ vivid/main git i386 1:2.1.4-2.1 [2,391 kB]
내려받기 3,112 k바이트, 소요시간 0초 (4,170 k바이트/초)
Selecting previously unselected package liberror-perl.
(데이터베이스 읽는중 ...현재 327575개의 파일과 디렉터리가 설치되어 있습니다.)
Preparing to unpack .../liberror-perl_0.17-1.1_all.deb ...
Unpacking liberror-perl (0.17-1.1) ...
Selecting previously unselected package git-man.
Preparing to unpack .../git-man_1%3a2.1.4-2.1_all.deb ...
Unpacking git-man (1:2.1.4-2.1) ...
Selecting previously unselected package git.
Preparing to unpack .../git_1%3a2.1.4-2.1_i386.deb ...
Unpacking git (1:2.1.4-2.1) ...
Processing triggers for man-db (2.7.0.2-5) ...
liberror-perl (0.17-1.1) 설정하는 중입니다 ...
git-man (1:2.1.4-2.1) 설정하는 중입니다 ...
git (1:2.1.4-2.1) 설정하는 중입니다 ...
hojin@hojin:~$
```

git --local-branching-on-the-cheap

🔍 Search entire site...

Git is a [free and open source](#) distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is [easy to learn](#) and has a [tiny footprint with lightning fast performance](#). It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like [cheap local branching](#), convenient [staging areas](#), and [multiple workflows](#).



Learn Git in your browser for free with [Try Git](#).



About

The advantages of Git compared to other source control systems.



Documentation

Command reference pages, Pro Git book content, videos and other material.

Latest source Release

2.5.3

[Release Notes \(2015-09-17\)](#)

[Downloads for Windows](#)



git --distributed-is-the-new-centralized

About

Documentation

Blog

Downloads

GUI Clients

Logos

Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

Downloading Git



Your download is

You are downloading the
This is the most recent [ma](#)
09-18.

If your download has

Other Git for Wind

Git for Windows Setup
[32-bit Git for Window](#)

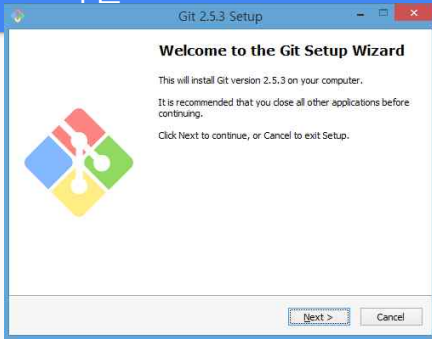
[64-bit Git for Window](#)

Git for Windows Portab
[32-bit Git for Window](#)

[64-bit Git for Window](#)

The current source code release
from [the source code](#).

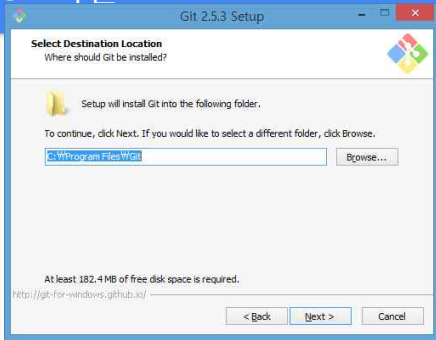
윈도우 Git 다운로드



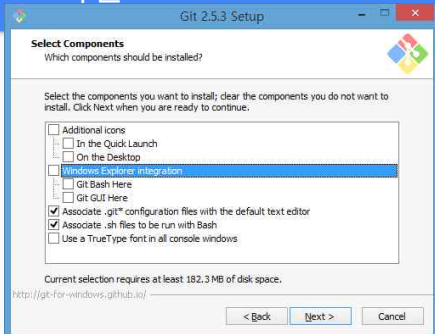
윈도우 Git 다운로드



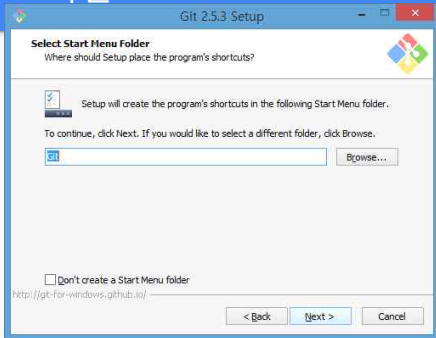
윈도우 Git 다운로드



윈도우 Git 다운로드

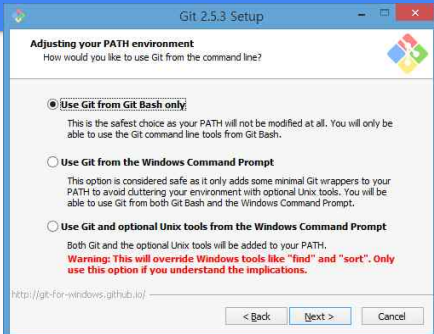


윈도우 Git 다운로드



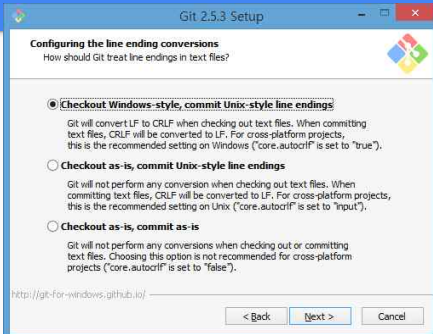
윈도우 Git 다운로드

- 환경변수 옵션
 - 환경변수 변경 안함.
 - Git은 오직 Git bash 커맨드 라인 도구에서만 실행됨.
 - 가장 안전함.



윈도우 Git 다운로드

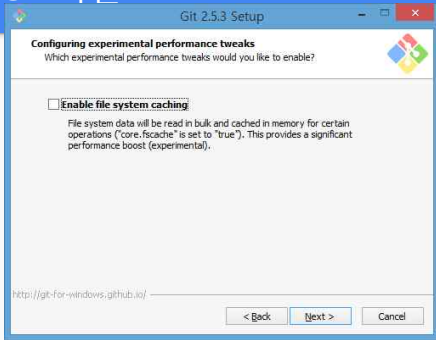
- Git에서 커밋시 작성하는 라인 끝을 어떻게 처리할지 결정.
 - 크로스 플랫폼 프로젝트 경우 윈도우의 권장 설정임.



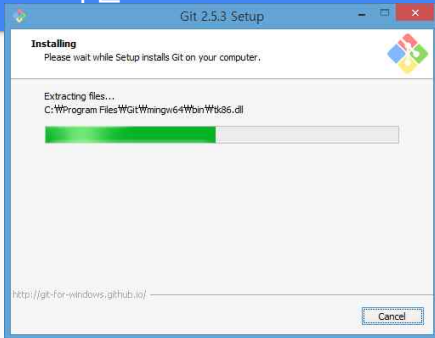
윈도우 Git 다운로드



윈도우 Git 다운로드



윈도우 Git 다운로드



윈도우 Git 다운로드



원도우 Git 사용자 이름, 이메일 설정

- 꼭 설정해 주어야 함.

아이름
이름
이메일

Chrome



Chrome

Daum



Daum ActiveX 매니저

Git



Git Bash 새로 설치됨



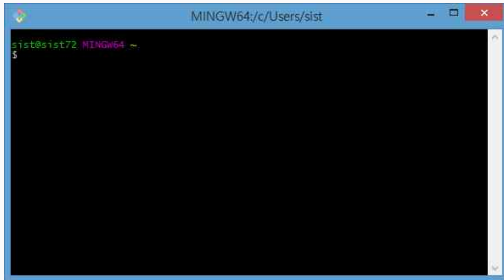
Git CMD 새로 설치됨



Git GUI 새로 설치됨

윈도우 Git 사용자 이름, 이메일 설정

- UNIX 명령어 실행 할 수 있는 Git Bash 실행됨.



우분투 Git 사용자 이름, 이메일 설정

- `git config --global user.name`
 - `git config --global user.email`
-
- 로컬 저장소 커밋에 이름,이메일 포함됨.

윈도우 Git 사용자 이름, 이메일 설정

- Git에서 커밋할 때마다 기록하는 사용자 이름, 메일주소 설정 하는 명령.

로컬 저장소 사용 위한 Git 기본

학습 방향

- 먼저 Git 혼자서 할 수 있는 것 배워야 함.
- 이후 협업 가능.

- 핵심은 Git 커맨드 라인 명령어.

가상 시나리오

- 로컬에 저장소 생성
- 저장소에 파일 생성, 추가
- 추가된 파일 수정
- 기본(마스터) 브랜치 외 별도 브랜치 생성.
- 브랜치 병합
- 충돌 해결
- 저장소 기록 보기

기본 명령어

- 저장소 생성
 - `git init`
 - 실행한 위치 Git 저장소로 초기화.

<실습>

- 원하는 위치에 디렉토리 하나 생성

```
hojin@hojin:/usr/local/Git_tutorial$ pwd  
/usr/local/Git_tutorial  
hojin@hojin:/usr/local/Git_tutorial$
```

<실습>

- 합계 8

```
hojin@hojin:/usr/local/Git_tutorial$ ls -al
drwxrwxr-x  2 hojin hojin 4096  9월 20 13:27 .
drwxr-xr-x 17 root  root  4096  9월 20 13:31 ..
hojin@hojin:/usr/local/Git_tutorial$ git init
Initialized empty Git repository in /usr/local/Git_tutorial/.git/
hojin@hojin:/usr/local/Git_tutorial$
```

<실습>

- 원하는 위치에 디렉토리 하나 생성

<실습>

- ```
sist@sist72 MINGW64 ~
$ ls -al git_tutorial/
total 12
drwxr-xr-x 1 sist 197121 0 9월 21 12:28 ./
drwxr-xr-x 1 sist 197121 0 9월 21 12:28 ../

sist@sist72 MINGW64 ~
$ |
```

## <실습>

- ```
sist@sist72 MINGW64 ~  
$ cd git_tutorial/  
  
sist@sist72 MINGW64 ~/git_tutorial  
$ ls  
  
sist@sist72 MINGW64 ~/git_tutorial  
$ git init  
Initialized empty Git repository in C:/Users/sist/git_tutorial/.git/  
  
sist@sist72 MINGW64 ~/git_tutorial (master)  
$ ls  
  
sist@sist72 MINGW64 ~/git_tutorial (master)  
$ ls -al  
total 16  
drwxr-xr-x 1 sist 197121 0 9월 21 12:31 ./  
drwxr-xr-x 1 sist 197121 0 9월 21 12:28 ../  
drwxr-xr-x 1 sist 197121 0 9월 21 12:31 .git/  
  
sist@sist72 MINGW64 ~/git_tutorial (master)  
$ |
```

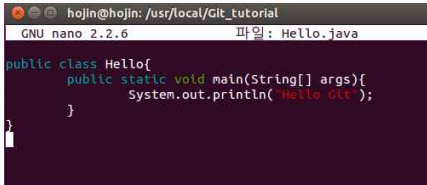
<실습>

- ```
hojin@hojin:/usr/local/Git_tutorial$ ls -al
합계 12
drwxrwxr-x 3 hojin hojin 4096 9월 20 13:36 .
drwxr-xr-x 17 root root 4096 9월 20 13:31 ..
drwxrwxr-x 7 hojin hojin 4096 9월 20 13:36 .git
hojin@hojin:/usr/local/Git_tutorial$ nano
```



## <실습>

- 



The screenshot shows a terminal window with a dark background. The title bar at the top reads 'hojin@hojin: /usr/local/Git\_tutorial'. Below the title bar, a status line indicates 'GNU nano 2.2.6' on the left and '파일: Hello.java' on the right. The main area of the terminal displays the following Java code:

```
public class Hello{
 public static void main(String[] args){
 System.out.println("Hello Git");
 }
}
```

A white cursor is visible at the end of the final closing brace '}' on the last line of the code.

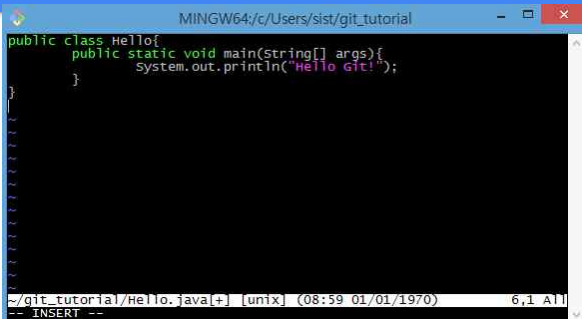
## <실습>

- ```
sist@sist72 MINGW64 ~/git_tutorial (master)  
$ vim Hello.java
```

<실습>

- vim
 - 리눅스, UNIX에서 사용 할 수 있는 텍스트 편집기.
 - 작성 시작
 - i 키 누름.
 - 일반 모드 ⇒ 입력 모드 전환됨.

<실습>



A screenshot of a MINGW64 terminal window. The title bar shows the path "MINGW64:/c/Users/sist/git_tutorial". The terminal displays a Java program with the following code:

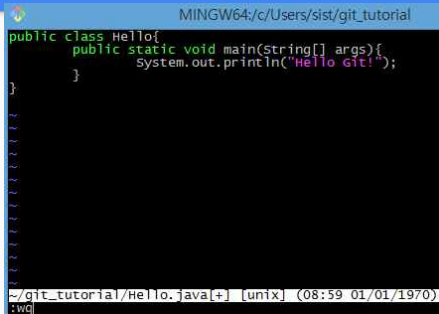
```
public class Hello{  
    public static void main(String[] args){  
        System.out.println("Hello Git!");  
    }  
}
```

The status bar at the bottom shows the file path "~/git_tutorial/Hello.java[+]", the editor mode "[unix]", the timestamp "(08:59 01/01/1970)", and the cursor position "6,1 All". The text "-- INSERT --" is visible at the bottom left of the status bar.

<실습>

- vim
 - 리눅스, UNIX에서 사용 할 수 있는 텍스트 편집기.
 - 작성 시작
 - i 키 누름.
 - 일반 모드 ⇒ 입력 모드 전환됨.
 - Esc 키 누름.
 - 입력 모드 ⇒ 일반 모드 전환됨.
 - : 키 누름.
 - 명령 모드 전환됨 (명령어 입력 가능)

<실습>



A screenshot of a MINGW64 terminal window. The title bar shows the MINGW64 logo and the path "MINGW64:/c/Users/sist/git_tutorial". The terminal displays a Java program with the following code:

```
public class Hello{  
    public static void main(String[] args){  
        system.out.println("Hello Git!");  
    }  
}
```

Below the code, there are several tilde (~) characters representing line continuation. At the bottom of the terminal, the command prompt shows the file path and editor:

```
~/git_tutorial/Hello.java[+] [unix] (08:59 01/01/1970)  
:wq|
```

<실습>

- vim
 - 리눅스, UNIX에서 사용 할 수 있는 텍스트 편집기.
 - 작성 시작
 - i 키 누름.
 - 일반 모드 ⇒ 입력 모드 전환됨.
 - Esc 키 누름.
 - 입력 모드 ⇒ 일반 모드 전환됨.
 - : 키 누름.
 - 명령 모드 전환됨 (명령어 입력 가능)
 - wq
 - 저장, 종료

<실습>

```
sist@sist72 MINGW64 ~/git_tutorial (master)
$ vim Hello.java

sist@sist72 MINGW64 ~/git_tutorial (master)
$ |
```


<실습>

```
sist@sist72 MINGW64 ~/git_tutorial (master)
$ cat Hello.java
public class Hello{
    public static void main(String[] args){
        System.out.println("Hello Git!");
    }
}

sist@sist72 MINGW64 ~/git_tutorial (master)
$ |
```

감사합니다

.