

Chapter 09 소프트웨어 관리

목차

00 개요

01 우분투 패키지의 개요

02 우분투 패키지 설치

03 스냅 패키지 설치

04 파일 아카이브와 압축

05 소프트웨어 컴파일

학습목표



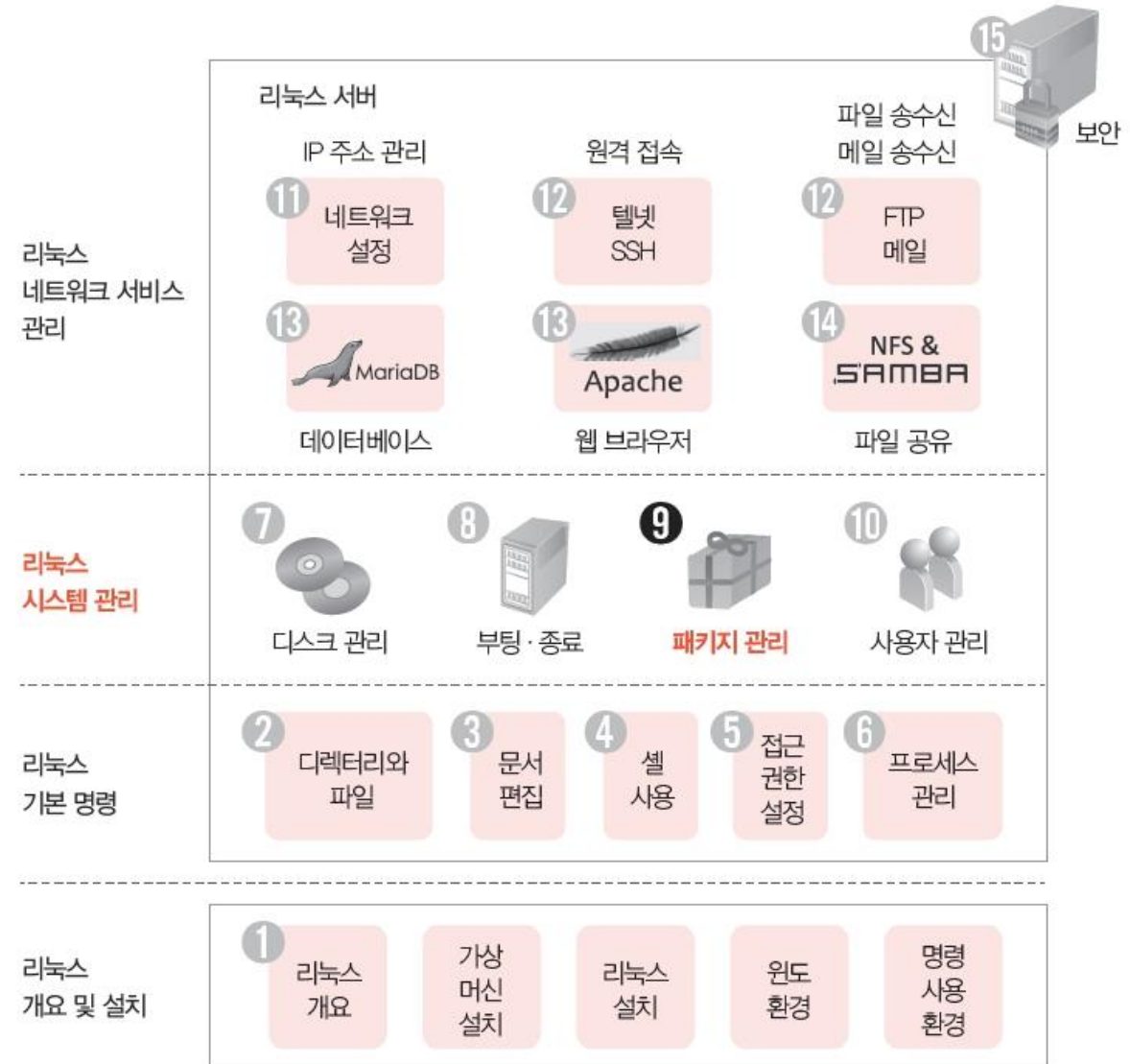
- 우분투 패키지의 특징을 이해한다.
- 우분투 패키지를 설치하거나 제거한다.
- 스냅 패키지를 설치하거나 제거한다.
- 파일을 묶어 아카이브 파일을 만들고 압축할 수 있다.
- 프로그램을 스스로 컴파일할 수 있다.
- makefile을 작성하고 make 명령으로 실행 파일을 만들 수 있다.

00 개요

00. 개요

■ 리눅스 학습 맵에서 9장의 위치

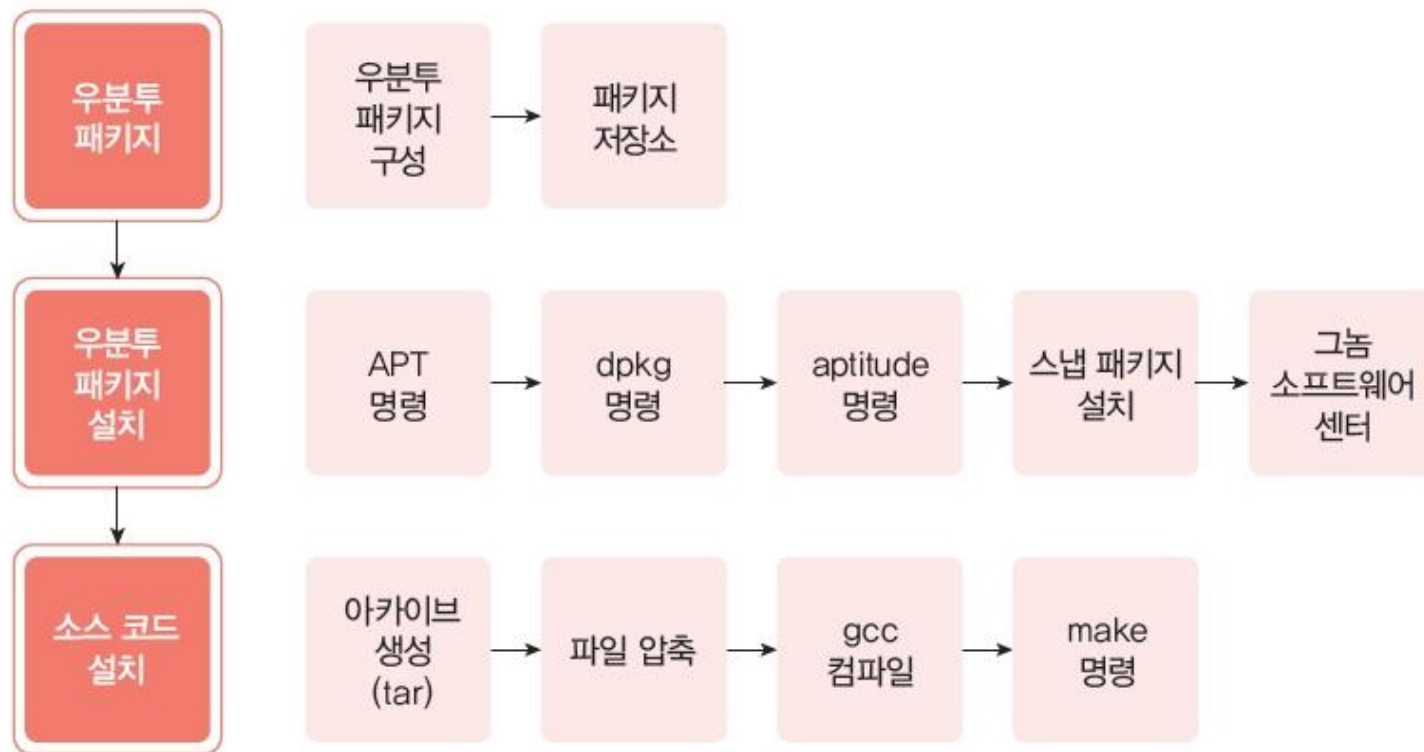
- 9장은 제3단계 중 세번째 항목으로 소프트웨어 패키지 관리를 다룬다.
- 우분투에서 패키지를 설치하고 삭제하는 여러가지 방법을 익히고, 아카이브 파일과 압축 파일을 만드는 방법을 익힌다.
- 우분투에서 C컴파일러는 사용하는 방법과 make 명령을 사용하는 방법일 익힌다.



00. 개요

■ 9장의 내용 구성

- 우분투 패키지 이해하기
- 우분투 패키지를 관리하는 명령 익히기
- 아카이브를 생성하고 파일을 압축하는 명령 익히기
- C컴파일러와 make 명령을 사용하여 실행 파일 만드는 방법 익히기



01 우분투 패키지의 개요

01. 우분투 패키지의 개요

- 우분투 리눅스는 데비안 계열의 패키지를 지원(deb 형식 패키지)
- 우분투 패키지의 특징
 - 바이너리 파일로 구성되어 있어 컴파일이 필요 없다.
 - 패키지의 파일이 관련 디렉터리에 바로 설치된다.
 - 패키지를 삭제할 때 관련된 파일을 일괄적으로 삭제할 수 있다.
 - 기존에 설치한 패키지를 삭제하지 않고 바로 업그레이드할 수 있다.
 - 패키지의 설치 상태를 검증할 수 있다.
 - 패키지에 대한 정보를 제공한다.
 - 해당 패키지가 의존하는 패키지가 무엇인지 알려준다. 따라서 의존성이 있는 패키지를 미리 설치할 수도 있고, apt-get 명령을 사용하면 의존성이 있는 패키지가 자동으로 설치된다.

01. 우분투 패키지의 개요

■ 우분투 패키지의 카테고리

- 우분투는 네 개의 카테고리로 나누어 소프트웨어를 제공
- main: 우분투에 의해 공식적으로 지원되며 자유롭게 배포할 수 있다.
- restricted: 우분투에 의해 지원되나 완전한 자유 라이선스 소프트웨어는 아니다.
- universe: 리눅스에서 사용할 수 있는 대부분의 소프트웨어로 자유 소프트웨어일 수도 있고 아닐 수도 있으며, 기술적 지원을 보장하지 않는다.
- multiverse: 자유 소프트웨어가 아닌 소프트웨어가 포함되어 있으며, 개인이 직접 라이선스를 확인해야 한다.

01. 우분투 패키지의 개요

■ 우분투 패키지의 이름 구성

패키지명_버전_데비안 리비전ubuntu우분투 리비전_아키텍처.deb

① ② ③ ④ ⑤ ⑥

그림 9-1 우분투 패키지의 이름 구성

- ① 패키지명: 패키지의 이름
- ② 버전: 오리지널 패키지의 버전
- ③ 데비안 리비전: 오리지널 패키지 버전을 데비안에 적용한 리비전 번호
- ④ ubuntu우분투 리비전: 오리지널 또는 데비안 패키지를 우분투에 적용한 리비전 번호
- ⑤ 아키텍처: 사용하는 시스템 아키텍처로 all은 시스템의 종류와 상관없이 사용할 수 있는 것이고, amd64는 64비트 운영체제임을 의미
- ⑥ 확장자: 확장자로 .deb를 사용

01. 우분투 패키지의 개요

■ '데비안 리비전ubuntu우분투 리비전' 세부 사례

- '데비안 리비전'이 0이면 이 패키지는 데비안 버전이 없거나 데비안 버전보다 새 버전임을 의미
- 'ubuntu우분투 리비전'은 데비안 패키지의 우분투 리비전 번호인데 다음과 같은 경우가 있음
 - ubuntu우분투 리비전 부분이 없으면 데비안 패키지를 변경 없이 사용
 - ubuntu우분투 리비전 부분이 있으면 데비안 패키지를 가져다가 우분투에서 기능 패치나 버그 수정 등 추가적인 작업을 했음을 의미
 - X_5.4.3-1: 패키지 X의 5.4.3 버전의 첫 번째 데비안 리비전이고 우분투에서는 데비안 리버전을 변경 없이 사용
 - X_5.4.3-2ubuntu1: 패키지 X의 5.4.3 버전의 두 번째 데비안 리비전이고 이를 바탕으로 한 첫 번째 우분투 리비전
 - X_5.4.3-0ubuntu2: 패키지 X의 5.4.3 버전의 데비안 리비전은 없고, 우분투는 두 번째 리비전

01. 우분투 패키지의 개요

■ 우분투 패키지 저장소

- 패키지 저장소에 대한 정보는 /etc/apt/sources.list 파일에 저장
- 이 파일을 수정하면 저장소를 추가하거나 삭제 가능

```
user1@myubuntu:~$ cat /etc/apt/sources.list
#deb cdrom:[Ubuntu 21.10 _Impish Indri_ - Release amd64 (20211012)]/ impish main
restricted

# See http://help.ubuntu.com/community/UpgradeNotes for how to upgrade to
# newer versions of the distribution.
deb http://kr.archive.ubuntu.com/ubuntu/ impish main restricted
```

- 패키지 유형: deb는 바이너리 패키지의 저장소를, deb-src는 패키지의 소스 저장소를 의미, 보통 한 저장소에 바이너리와 소스를 함께 저장
- 저장소 주소: http 프로토콜을 사용하는 URL 주소를 사용
- 우분투 버전 정보: 저장소에서 관리하는 패키지에 해당하는 우분투의 버전을 이름으로 표시
- 카테고리: 저장소가 가지고 있는 소프트웨어 카테고리(main, restricted 등)를 표시

02 우분투 패키지 설치

02. 우분투 패키지 설치

- APT 명령으로 패키지 관리하기: apt-get, apt-cache
- apt-cache 명령

apt-cache

- **기능** APT 캐시에 질의하여 여러 가지 정보를 검색한다.
- **형식** apt-cache [옵션] 서브 명령
- **옵션** -f: 검색 결과로 패키지에 대한 전체 기록을 출력한다.
-h: 간단한 도움말을 출력한다.
- **서브 명령** stats: 캐시에 대한 통계 정보를 출력한다.
dump: 현재 설치된 패키지를 업그레이드한다.
search 키워드: 캐시에서 키워드를 검색한다.
showpkg 패키지명: 패키지에 대한 의존성 정보와 역의존성 정보를 검색하여 출력한다.
show 패키지명: 패키지에 대한 간단한 정보를 출력한다.
pkgnames: 사용 가능한 모든 패키지의 이름을 출력한다.
- **사용 예** apt-cache stats
apt-cache show vsftpd
apt-cache search vsftpd

02. 우분투 패키지 설치

■ APT 캐시 통계 정보 보기: apt-cache stats 명령

- 전체 패키지 이름: 캐시에 있는 패키지 이름의 전체 개수
- 일반 패키지: 일반적으로 사용하는 패키지의 개수를 의미
- 순수 가상 패키지:
 - 패키지의 이름만 제공하며 별도의 패키지가 있는 것은 아님
- 단일 가상 패키지: 한 패키지가 특정한 가상 패키지의 기능을 제공
- 혼합 가상 패키지 : 특정한 가상 패키지를 제공하거나 가상 패키지의 이름을 패키지 이름으로 사용
- 빠짐: 의존성은 있지만 어떠한 패키지도 제공하지 않는 패키지
- 개별 버전 전체: 캐시에 있는 패키지 버전의 개수를 의미
- 전체 의존성 : 캐시에 있는 모든 패키지의 의존성 관계를 의미

```
user1@myubuntu:~$ apt-cache stats
전체 패키지 이름 : 106895 (2,993 k)
전체 패키지 구조: 102539 (4,512 k)
일반 패키지: 66883
순수 가상 패키지: 1538
단일 가상 패키지: 20508
혼합 가상 패키지: 2249
빠짐: 11361
개별 버전 전체: 70651 (6,217 k)
```

02. 우분투 패키지 설치

■ 사용 가능한 패키지 이름 보기: apt-cache pkgnames 명령

```
user1@myubuntu:~$ apt-cache pkgnames  
lib32stdc++-11-dev-s390x-cross  
bazel-bootstrap  
ruby-jmespath  
goverlay  
libgiza0  
librust-quickcheck-macros-dev  
r-cran-egg  
r-bioc-snpstats  
ruby-ami  
libghc-parseargs-dev
```


02. 우분투 패키지 설치

■ 패키지 이름 검색하기: apt-cache search 명령

```
user1@myubuntu:~$ apt-cache search vsftpd
resource-agents - Cluster Resource Agents
vsftpd - lightweight, efficient FTP server written for security
vsftpd-dbg - lightweight, efficient FTP server written for security (debug)
ccze - robust, modular log coloriser
ftpd - File Transfer Protocol (FTP) server
yasat - simple stupid audit tool
user1@myubuntu:~$
```

02. 우분투 패키지 설치

■ 패키지 정보 검색하기: apt-cache show 명령

- 버전, 패키지 크기, 카테고리, 체크섬 등 패키지에 관한 정보를 확인하려면 show 서브 명령을 사용

```
user1@myubuntu:~$ apt-cache show vsftpd
Package: vsftpd
Architecture: amd64
Version: 3.0.3-13build1
Priority: extra
Section: net
Origin: Ubuntu
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Original-Maintainer: Keng-Yu Lin <kengyu@debian.org>
Bugs: https://bugs.launchpad.net/ubuntu/+filebug
Installed-Size: 314
Provides: ftp-server
```

02. 우분투 패키지 설치

■ 패키지 의존성 검색하기: apt-cache showpkg 명령

```
user1@myubuntu:~$ apt-cache showpkg vsftpd
```

```
Package: vsftpd
```

```
Versions:
```

```
3.0.3-13build1 (/var/lib/apt/lists/kr.archive.ubuntu.com_ubuntu_dists_impish_main_
binary-amd64_Packages)
```

```
Dependencies:
```

```
3.0.3-13build1 - debconf (18 0.5) debconf-2.0 (0 (null)) libc6 (2 2.34) libcap2 (2
1:2.10) libpam0g (2 0.99.7.1) libssl1.1 (2 1.1.0) libwrap0 (2 7.6-4~) adduser (0
(null)) libpam-modules (0 (null)) lsb-base (2 3.0-6) netbase (0 (null)) procs (0
(null)) sysvinit-utils (2 2.96) ftp-server (0 (null)) logrotate (0 (null)) ssl-cert (0
(null)) ftp-server (0 (null)) ftp-server:i386 (0 (null)) ftp-server:i386 (0 (null))
```

```
Provides:
```

```
3.0.3-13build1 - ftp-server (= )
```

```
Reverse Provides:
```

```
user1@myubuntu:~$
```

02. 우분투 패키지 설치

■ apt-get 명령: 패키지 저장소를 업데이트하고 패키지를 설치하거나 제거

apt-get

- **기능** 패키지를 관리한다.
- **형식** apt-get [옵션] 서브 명령
- **옵션**
 - d: 패키지를 내려받기만 한다.
 - f: 의존성이 깨진 패키지를 수정하려고 시도한다.
 - h: 간단한 도움말을 출력한다.
- **서브 명령**
 - update: 패키지 저장소에서 새로운 패키지 정보를 가져온다.
 - upgrade: 현재 설치된 패키지를 업그레이드한다.
 - install 패키지명: 패키지를 설치한다.
 - remove 패키지명: 패키지를 삭제한다.
 - download 패키지명: 패키지를 현재 디렉터리로 내려받는다.
 - autoclean: 불완전하게 내려받았거나 오래된 패키지를 삭제한다.
 - clean: /var/cache/apt/archives에 캐시되어 있는 모든 패키지를 삭제하여 디스크 공간을 확보한다.
 - check: 의존성이 깨진 패키지를 확인한다.
- **사용 예** apt-get update apt-get install vsftpd apt-get clean

02. 우분투 패키지 설치

■ 패키지 정보 업데이트하기: apt-get update 명령

- update 서브 명령은 /etc/apt/sources.list에 명시한 저장소로부터 패키지 정보를 읽어와서 APT 캐시를 수정

```
user1@myubuntu:~$ sudo apt-get update
```

```
기존:1 http://kr.archive.ubuntu.com/ubuntu impish InRelease
```

```
기존:2 http://security.ubuntu.com/ubuntu impish-security InRelease
```

```
기존:3 http://kr.archive.ubuntu.com/ubuntu impish-updates InRelease
```

```
기존:4 http://kr.archive.ubuntu.com/ubuntu impish-backports InRelease
```

```
패키지 목록을 읽는 중입니다... 완료
```

```
(생략)
```

```
내려받기 1,914 k바이트, 소요시간 5초 (364 k바이트/초)
```

```
패키지 목록을 읽는 중입니다... 완료
```

02. 우분투 패키지 설치

■ 패키지 업그레이드하기: apt-get upgrade 명령

```
user1@myubuntu:~$ sudo apt-get upgrade
```

```
패키지 목록을 읽는 중입니다... 완료
```

```
의존성 트리를 만드는 중입니다... 완료
```

```
상태 정보를 읽는 중입니다... 완료
```

```
업그레이드를 계산하는 중입니다... 완료
```

```
다음 패키지를 업그레이드할 것입니다:
```

```
  alsa-ucm-conf bsdxtrautils bsduutils fdisk fonts-opensymbol gir1.2-mutter-8  
libasound2
```

```
  libasound2-data libatopology2 libayatana-indicator3-7 libblkid1 libfdisk1  
  libfprint-2-2 libmount1 libmutter-8-0 libnetplan0 libpulse-mainloop-glib0
```

```
(생략)
```

```
64개 업그레이드, 0개 새로 설치, 0개 제거 및 0개 업그레이드 안 함.
```

```
287 M바이트 아카이브를 받아야 합니다.
```

```
이 작업 후 1,457 k바이트의 디스크 공간을 더 사용하게 됩니다.
```

```
계속 하시겠습니까? [Y/n] n
```

```
중단.
```

업그레이드 시작하면 시간이 걸리므로 여기서는 n을 선택

02. 우분투 패키지 설치

■ 특정 패키지 설치 또는 업그레이드하기: apt-get install 명령

```
user1@myubuntu:~$ sudo apt-get install xterm
```

```
패키지 목록을 읽는 중입니다... 완료
```

```
의존성 트리를 만드는 중입니다... 완료
```

```
상태 정보를 읽는 중입니다... 완료
```

```
다음의 추가 패키지가 설치될 것입니다 :
```

```
libutempter0
```

```
제안하는 패키지:
```

```
xfonts-cyrillic
```

```
다음 새 패키지를 설치할 것입니다:
```

```
libutempter0 xterm
```

```
xterm (366-1ubuntu1) 설정하는 중입니다 ...
```

```
Processing triggers for desktop-file-utils (0.26-1ubuntu2) ...
```

```
Processing triggers for hicolor-icon-theme (0.17-2) ...
```

```
Processing triggers for gnome-menus (3.36.0-1ubuntu1) ...
```

02. 우분투 패키지 설치

- 여러 패키지를 한 번에 설치하려면 패키지 이름을 나열

```
sudo apt-get install xterm goaccess
```

- 패키지를 설치할 때 업그레이드를 하지 않으려면 --no-upgrade 옵션을 사용

```
sudo apt-get install xterm --no-upgrade
```

- 새로운 패키지를 설치하지 않고 업그레이드만 할 때는 --only-upgrade 옵션을 사용

```
sudo apt-get install netcat --only-upgrade
```


02. 우분투 패키지 설치

■ 패키지 삭제하기: apt-get remove 명령

```
user1@myubuntu:~$ sudo apt-get remove xterm
```

패키지 목록을 읽는 중입니다... 완료

의존성 트리를 만드는 중입니다... 완료

상태 정보를 읽는 중입니다... 완료

다음 패키지가 자동으로 설치되었지만 더 이상 필요하지 않습니다:

libutempter0

'sudo apt autoremove'를 이용하여 제거하십시오.

다음 패키지를 지울 것입니다:

xterm

계속 하시겠습니까? [Y/n]

(데이터베이스 읽는중 ...현재 183233개의 파일과 디렉터리가 설치되어 있습니다.)

xterm (366-1ubuntu1)를 제거합니다...

02. 우분투 패키지 설치

- 설정 파일을 포함하여 패키지를 삭제하려면 purge 서브 명령을 사용

```
sudo apt-get purge xterm
```

- remove 서브 명령과 --purge 옵션을 함께 사용 가능

```
sudo apt-get remove --purge xterm
```

02. 우분투 패키지 설치

■ 패키지 자동 정리 및 삭제하기: apt-get autoremove 명령

```
user1@myubuntu:~$ sudo apt-get autoremove
```

패키지 목록을 읽는 중입니다... 완료

의존성 트리를 만드는 중입니다... 완료

상태 정보를 읽는 중입니다... 완료

다음 패키지를 지울 것입니다:

libutempter0

0개 업그레이드, 0개 새로 설치, 1개 제거 및 64개 업그레이드 안 함.

이 작업 후 52.2 k바이트의 디스크 공간이 비워집니다.

계속 하시겠습니까? [Y/n]

(데이터베이스 읽는중 ...현재 183192개의 파일과 디렉터리가 설치되어 있습니다.)

libutempter0:amd64 (1.2.1-2)를 제거합니다...

Processing triggers for libc-bin (2.34-0ubuntu3) ...

02. 우분투 패키지 설치

- 디스크 공간 정리하기: apt-get clean 명령

```
sudo apt-get clean
```

- 패키지 내려받기: apt-get download 명령

```
user1@myubuntu:~$ sudo apt-get download xterm
받기:1 http://kr.archive.ubuntu.com/ubuntu impish/universe amd64 xterm amd64
366-1ubuntu1 [796 kB]
내려받기 796 k바이트, 소요시간 2초 (384 k바이트/초)
W: Download is performed unsandboxed as root as file '/home/user1/xterm_366-
1ubuntu1_amd64.deb' couldn't be accessed by user '_apt'. - pkgAcquire::Run (13: 허가
거부)
user1@myubuntu:~$ ls xt*
xterm_366-1ubuntu1_amd64.deb
```

02. 우분투 패키지 설치

■ 패키지의 소스 관련 서브 명령: apt-get source 명령

- 특정 패키지의 소스 코드를 내려받기만 하는 경우

```
sudo apt-get --download-only source 패키지명
```

- 특정 패키지의 소스 코드를 내려받고 압축을 푸는 경우

```
sudo apt-get source 패키지명
```

- 특정 패키지의 소스 코드를 내려받아 압축을 풀고 컴파일하는 경우

```
sudo apt-get --compile source 패키지명
```

455p. 따라해보기: APT 명령으로 패키지 설치하기

- ① python3과 관련된 패키지 중에서 업그레이드할 것이 있는지 확인

```
user1@myubuntu:~$ sudo apt-get upgrade | grep python3
python3-software-properties python3-uno python3-update-manager rfcill
^C
```

- ② 검색된 결과 중에서 python3-uno 패키지를 업그레이드

```
user1@myubuntu:~$ sudo apt-get install python3-uno
```

- ③ 설치한 패키지의 정보를 확인

```
user1@myubuntu:~$ apt-cache show python3-uno
Package: python3-uno
Architecture: amd64
Version: 1:7.2.3-0ubuntu0.21.10.1
```

02. 우분투 패키지 설치

■ <여기서 잠깐> apt 명령

- apt 명령은 패키지 관리 시스템을 위한 상위 레벨의 명령 인터페이스를 제공

apt 명령	apt-get, apt-cache 명령	기능
apt update	apt-get update	패키지 저장소에 새로운 패키지 정보를 가져온다.
apt upgrade	apt-get upgrade	패키지를 업그레이드한다.
apt install	apt-get install	패키지를 설치한다.
apt remove	apt-get remove	패키지를 삭제한다.
apt purge	apt-get purge	패키지와 설정 파일도 삭제한다.
apt autoremove	apt-get autoremove	필요 없는 패키지를 삭제한다.
apt search	apt-cache search	캐시에서 키워드로 패키지를 검색한다.
apt show	apt-cache show	패키지의 간단한 정보를 출력한다.
apt list		조건에 맞는 패키지 목록을 출력한다.

02. 우분투 패키지 설치

■ dpkg 명령으로 패키지 관리하기

dpkg

- **기능** 데비안의 패키지 관리 명령이다.
- **형식** dpkg [옵션] 파일명 또는 패키지명
- **옵션**
 - l: 설치된 패키지의 목록을 출력한다.
 - l 패키지명: 패키지의 설치 상태를 출력한다.
 - s 패키지명: 패키지의 상세 정보를 출력한다.
 - S 경로명: 경로명이 포함된 패키지를 검색한다.
 - L 패키지명: 패키지에서 설치된 파일의 목록을 출력한다.
 - c .deb 파일: 지정한 .deb 파일의 내용을 출력한다.
 - i .deb 파일: 해당 파일을 설치한다(sudo).
 - r 패키지명: 해당 패키지를 삭제한다(sudo).
 - P 패키지명: 해당 패키지와 설정 정보를 모두 삭제한다(sudo).
 - x .deb 파일 디렉터리: 해당 파일을 지정한 디렉터리에 풀어놓는다.
- **사용 예**
dpkg -l dpkg -s netcat dpkg -S /bin/ls
sudo dpkg -i xterm_361-1ubuntu3_amd64.deb

02. 우분투 패키지 설치

■ 패키지 목록 보기: `dpkg -l`

- 출력 결과에서 첫 글자는 상단의 희망 상태를 나타내고, 두 번째 글자는 상태를 의미
- 처음 두 글자 ii는 희망 상태가 '설치(I)', 상태도 '설치(I)'임을 의미

```
희망상태=알수없음(U)/설치(I)/지우기(R)/깨끗이(P)/고정(H)
| 상태=아님(N)/설치(I)/설정(C)/풀림(U)/절반설정(F)/일부설치(H)/트리거대기(W)/
| /      트리거밀림(T)
|/ 오류?=(없음)/다시설치필요(R) (상태, 오류가 대문자=불량)
||/ 이름                                     버전
Architec
ture 설명
+++=====
=====
=====
ii accountsservice                        0.6.55-0ubuntu14.1                amd64
    query and manipulate user account information
```

02. 우분투 패키지 설치

- -l 다음에 특정 패키지의 이름을 지정하면 해당 패키지에 대한 정보만 출력

```
user1@myubuntu:~$ dpkg -l zip
희망상태=알수없음(U)/설치(I)/지우기(R)/깨끗이(P)/고정(H)
! 상태=아님(N)/설치(I)/설정(C)/풀림(U)/절반설정(F)/일부설치(H)/트리거대기(W)/
! /      트리거밀림(T)
! / 오류?=(없음)/다시설치필요(R) (상태, 오류가 대문자=불량)
!|/ 이름                버전                Architecture 설명
+++-=====
=====
ii  zip                    3.0-12build1 amd64      Archiver for .zip files
```

02. 우분투 패키지 설치

■ 패키지 상세 정보 보기: `dpkg -s`

```
user1@myubuntu:~$ dpkg -s zip
```

```
Package: zip
```

```
Status: install ok installed
```

```
Priority: optional
```

```
Section: utils
```

```
Installed-Size: 531
```

```
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
```

```
Architecture: amd64
```

```
Multi-Arch: foreign
```

```
Version: 3.0-12build1
```

```
Depends: libbz2-1.0, libc6 (>= 2.34)
```

```
Recommends: unzip
```

```
Description: Archiver for .zip files
```

```
This is InfoZIP's zip program. It produces files that are fully  
compatible with the popular PKZIP program; however, the command line
```

02. 우분투 패키지 설치

■ 특정 파일이 포함된 패키지 검색하기: dpkg -S

```
user1@myubuntu:~$ dpkg -S /bin/ls  
coreutils: /bin/ls
```

■ 패키지가 설치한 파일 목록 검색하기: dpkg -L

```
user1@myubuntu:~$ dpkg -L zip  
/.  
/usr  
/usr/bin  
/usr/bin/zip  
/usr/bin/zipcloak  
/usr/bin/zipnote  
/usr/bin/zipsplit  
(생략)
```

02. 우분투 패키지 설치

■ 패키지의 파일 목록 검색하기: `dpkg -c`

- .deb 파일에 들어 있는 내용을 출력

```
user1@myubuntu:~$ dpkg -c xterm_366-1ubuntu1_amd64.deb
drwxr-xr-x root/root          0 2021-05-25 14:34 ./
drwxr-xr-x root/root          0 2021-05-25 14:34 ./etc/
drwxr-xr-x root/root          0 2021-05-25 14:34 ./etc/X11/
drwxr-xr-x root/root          0 2021-05-25 14:34 ./etc/X11/app-defaults/
-rw-r--r-- root/root      2400 2021-05-25 14:34 ./etc/X11/app-defaults/KOI8RXTerm
-rw-r--r-- root/root      6217 2021-05-25 14:34 ./etc/X11/app-defaults/KOI8RXTerm-
color
-rw-r--r-- root/root      3706 2021-05-25 14:34 ./etc/X11/app-defaults/UXTerm
```

02. 우분투 패키지 설치

■ 패키지 설치하기: dpkg -i

```
user1@myubuntu:~$ sudo dpkg -i xterm_366-1ubuntu1_amd64.deb
Selecting previously unselected package xterm.
(데이터베이스 읽는중 ...현재 183184개의 파일과 디렉터리가 설치되어 있습니다.)
Preparing to unpack xterm_366-1ubuntu1_amd64.deb ...
Unpacking xterm (366-1ubuntu1) ...
dpkg: 종속성 문제로 xterm의 구성이 차단되었습니다:
 xterm 패키지는 다음 패키지에 의존: libutempter0 (>= 1.1.5): 하지만:
  libutempter0 패키지는 설치하지 않았습니다.

dpkg: error processing package xterm (--install):
 의존성 문제 - 설정하지 않고 남겨둠
```

dpkg 명령은 의존성이 있는
패키지를 자동으로 설치해
주지 않음

```
처리하는데 오류가 발생했습니다:
xterm
```

02. 우분투 패키지 설치

- dpkg 명령은 의존성이 있는 패키지를 자동으로 설치하지 않으므로 사용자가 별도로 설치

```
user1@myubuntu:~$ apt-get download libutempter0
받기:1 http://kr.archive.ubuntu.com/ubuntu impish/main amd64 libutempter0 amd64
1.2.1-2 [8,860 B]
내려받기 8,860 바이트, 소요시간 1초 (8,661 바이트/초)
user1@myubuntu:~$ ls lib*
libutempter0_1.2.1-2_amd64.deb
```

의존성 있는 패키지
다운로드

```
user1@myubuntu:~$ sudo dpkg -i libutempter0_1.2.1-2_amd64.deb
Selecting previously unselected package libutempter0:amd64.
(데이터베이스 읽는중 ...현재 183225개의 파일과 디렉터리가 설치되어 있습니다.)
```

의존성 있는 패키지
먼저 설치

```
user1@myubuntu:~$ sudo dpkg -i xterm_366-1ubuntu1_amd64.deb
(데이터베이스 읽는중 ...현재 183233개의 파일과 디렉터리가 설치되어 있습니다.)
Preparing to unpack xterm_366-1ubuntu1_amd64.deb ...
Unpacking xterm (366-1ubuntu1) over (366-1ubuntu1) ...
xterm (366-1ubuntu1) 설정하는 중입니다 ...
```

xterm 다시 설치

02. 우분투 패키지 설치

■ 패키지 삭제하기: `dpkg -r`, `dpkg -P`

- `-r` : 설치된 패키지만 삭제, `-P` : 패키지와 설정 정보 모두 삭제

```
user1@myubuntu:~$ sudo dpkg -r xterm
```

```
(데이터베이스 읽는중 ...현재 183232개의 파일과 디렉터리가 설치되어 있습니다.)
```

```
xterm (366-1ubuntu1)를 제거합니다...
```

```
user1@myubuntu:~$ sudo dpkg -P libutempter0
```

```
(데이터베이스 읽는중 ...현재 183191개의 파일과 디렉터리가 설치되어 있습니다.)
```

```
libutempter0:amd64 (1.2.1-2)를 제거합니다...
```

```
Processing triggers for libc-bin (2.34-0ubuntu3) ...
```


02. 우분투 패키지 설치

■ deb 패키지 풀기: dpkg -x

- -x 옵션은 deb 패키지의 내용을 지정한 디렉터리에 풀어놓음

```
user1@myubuntu:~$ mkdir xterm
user1@myubuntu:~$ sudo dpkg -x xterm_366-1ubuntu1_amd64.deb xterm
user1@myubuntu:~$ ls -R xterm
xterm:
etc  usr

xterm/etc:
X11

xterm/etc/X11:
app-defaults

xterm/etc/X11/app-defaults:
KOI8RXTerm  KOI8RXTerm-color  UXTerm  UXTerm-color  XTerm  XTerm-color
```

465p. 따라해보기: dpkg 명령으로 패키지 설치하기

- ① **gnome-chess** 패키지가 설치되어 있는지 확인: `dpkg -l gnome-chess`
- ② 아직 설치되지 않은 패키지이므로 APT 명령을 사용하여 **gnome-chess** 패키지 다운로드: `apt-get download gnome-chess`
- ③ **gnome-chess** 패키지를 구성하는 파일의 내용을 확인:
: `dpkg -c gnome-chess_1%3a41.0-1_amd64.deb`
- ④ **gnome-chess** 패키지를 설치
: `sudo dpkg -i gnome-chess_1%3a41.0-1_amd64.deb`
- ⑤ 의존성이 있는 패키지가 있는데 이는 apt 명령으로 설치
: `sudo apt install hoichess`
- ⑥ 이제 다시 **gnome-chess** 패키지를 설치
- ⑦ 설치된 파일이 무엇인지 확인: `dpkg -L gnome-chess`

465p. 따라해보기: dpkg 명령으로 패키지 설치하기

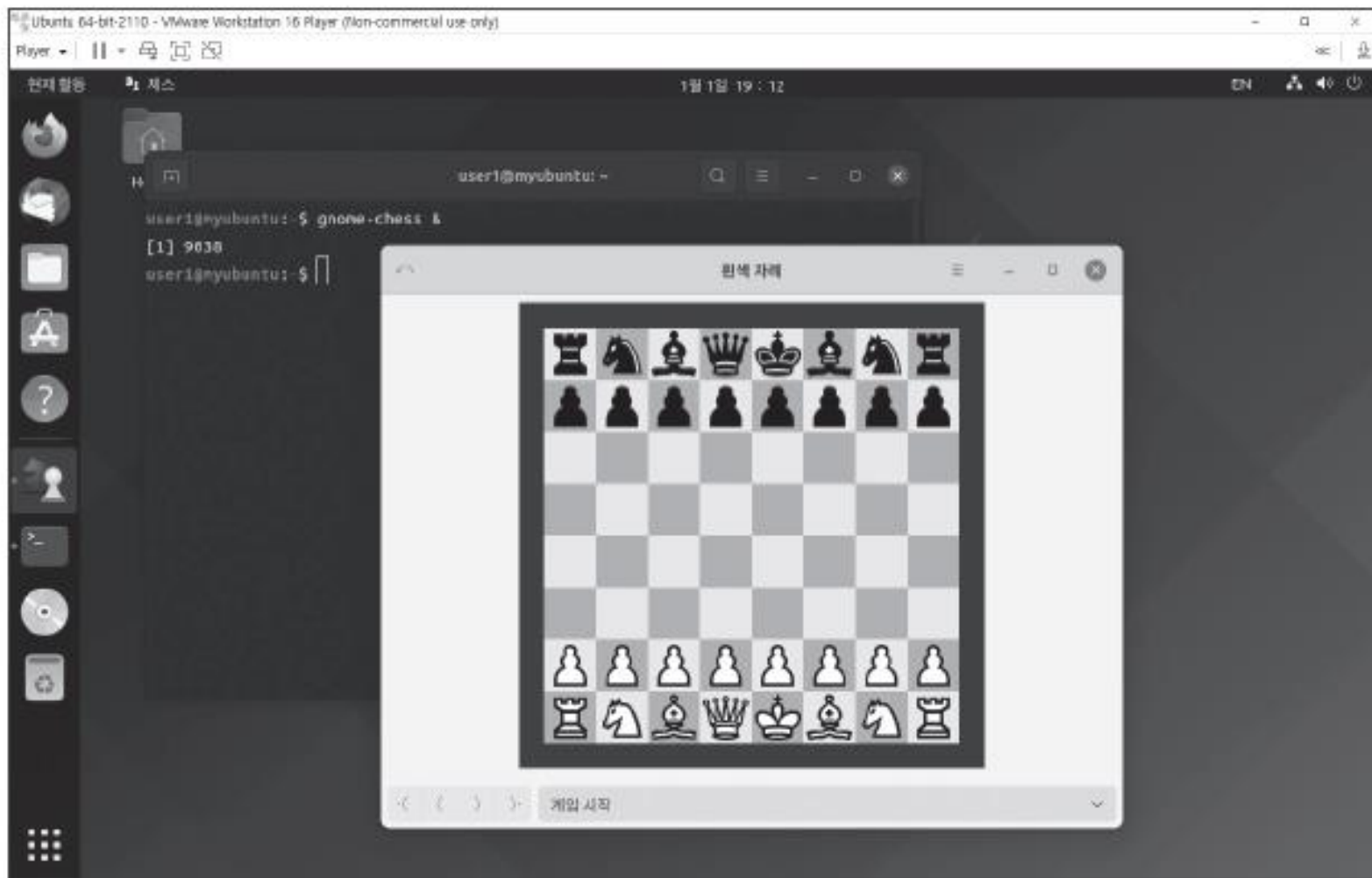


그림 9-2 gnome-chess 실행 화면

02. 우분투 패키지 설치

■ aptitude 명령으로 패키지 관리하기

aptitude

- **기능** 우분투에서 패키지를 관리한다.
- **형식** aptitude [서브 명령]
- **서브 명령** 단독 실행: curses 프로그램이 나타난다.
search 키워드: 키워드를 검색하여 일치하는 패키지 목록을 출력한다.
update: 패키지 저장소를 업데이트한다.
upgrade: 모든 패키지를 최신 버전으로 업그레이드한다.
show 패키지명: 패키지에 대한 자세한 정보를 보여준다.
download 패키지명: 패키지를 내려받는다.
clean: 패키지 캐시 디렉터리에서 모든 패키지 파일을 삭제한다.
install: 패키지를 설치한다.
remove: 패키지를 삭제한다.
purge: 패키지와 설정 파일을 모두 삭제한다.

02. 우분투 패키지 설치

■ aptitude 설치

```
user1@myubuntu:~$ sudo apt install aptitude
```

패키지 목록을 읽는 중입니다... 완료

의존성 트리를 만드는 중입니다... 완료

상태 정보를 읽는 중입니다... 완료

다음의 추가 패키지가 설치될 것입니다 :

aptitude-common libcwidget4 libxapian30

제안하는 패키지:

apt-xapian-index aptitude-doc-en | aptitude-doc debtags tasksel libcwidget-dev
xapian-tools

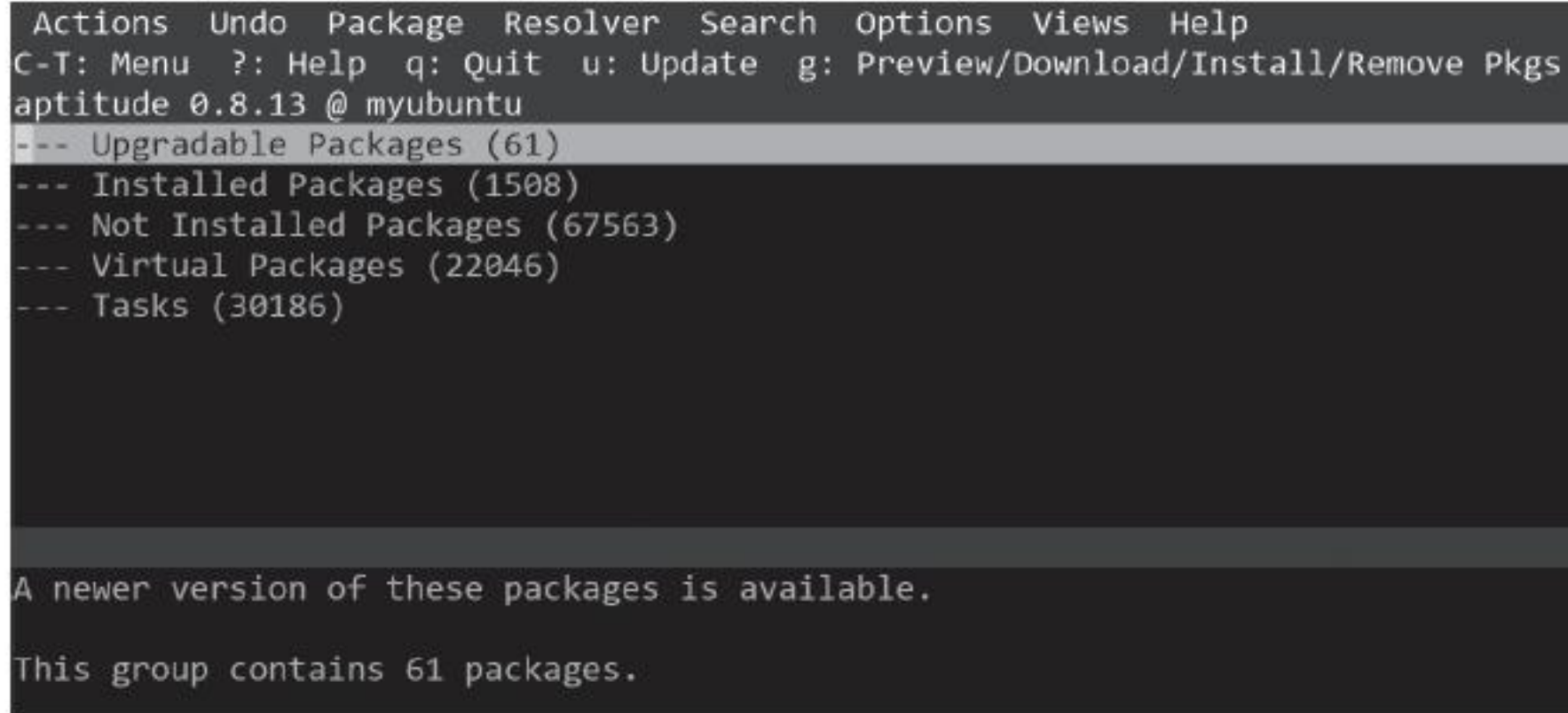
다음 새 패키지를 설치할 것입니다:

aptitude aptitude-common libcwidget4 libxapian30

0개 업그레이드, 4개 새로 설치, 0개 제거 및 61개 업그레이드 안 함.

02. 우분투 패키지 설치

- 옵션이나 서브 명령 없이 aptitude 명령 실행: 메뉴 화면 출력



```
Actions  Undo  Package  Resolver  Search  Options  Views  Help
C-T: Menu  ?: Help  q: Quit  u: Update  g: Preview/Download/Install/Remove Pkgs
aptitude 0.8.13 @ myubuntu
--- Upgradable Packages (61)
--- Installed Packages (1508)
--- Not Installed Packages (67563)
--- Virtual Packages (22046)
--- Tasks (30186)

A newer version of these packages is available.

This group contains 61 packages.
```

그림 9-4 aptitude 실행 화면

02. 우분투 패키지 설치

■ aptitude를 명령으로 사용하기

- 패키지 정보 업데이트하기: update 명령

```
user1@myubuntu:~$ sudo aptitude update
Hit http://kr.archive.ubuntu.com/ubuntu impish InRelease
Hit http://security.ubuntu.com/ubuntu impish-security InRelease
Hit http://kr.archive.ubuntu.com/ubuntu impish-updates InRelease
Hit http://kr.archive.ubuntu.com/ubuntu impish-backports InRelease
```

02. 우분투 패키지 설치

- 패키지 검색하기: search 명령

```
user1@myubuntu:~$ aptitude search gnome
p  amule-gnome-support - ed2k links handling support for GNOME web browsers
p  cairo-dock-gnome-integration-plug-in - GNOME integration plug-in for Cairo-dock
p  chrome-gnome-shell - GNOME Shell extensions integration for web browsers
p  clamtk-gnome - GNOME (Nautilus) MenuProvider extension for ClamTk
p  compiz-gnome - OpenGL window and compositing manager - GNOME window decorator
v  dh-sequence-gnome -
p  distccmon-gnome - GTK+ monitor for distcc a distributed client and server
p  geogebra-gnome - GNOME integration layer for GeoGebra
p  gir1.2-gnomeautoar-0.1 - GObject introspection data for GnomeAutoar
p  gir1.2-gnomeautoargtk-0.1 - GObject introspection data for GnomeAutoarGtk
(생략)
```


02. 우분투 패키지 설치

- 패키지의 상세 정보 확인하기: show 명령

```
user1@myubuntu:~$ aptitude show gnome-clocks
Package: gnome-clocks
Version: 40.0-2
State: not installed
Priority: 옵션
Section: universe/gnome
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Architecture: amd64
Uncompressed Size: 1,975 k
Depends: geoclue-2.0, dconf-gsettings-backend | gsettings-backend, libc6 (>=
        2.34), libgeoclue-2-0 (>= 2.4.0), libgeocode-glib0 (>= 1.0),
        libglib2.0-0 (>= 2.58), libgnome-desktop-3-19 (>= 3.17.92), libgsound0
        (>= 1.0.1), libgtk-3-0 (>= 3.21.4), libgweather-3-16 (>= 3.32.0),
        libhandy-1-0 (>= 1.0.0)
```

02. 우분투 패키지 설치

- 패키지 설치하기: install 명령

```
user1@myubuntu:~$ sudo aptitude install gnome-clocks
The following NEW packages will be installed:
  gnome-clocks
0 packages upgraded, 1 newly installed, 0 to remove and 61 not upgraded.
Need to get 293 kB of archives. After unpacking 1,975 kB will be used.
Get: 1 http://kr.archive.ubuntu.com/ubuntu impish/universe amd64 gnome-clocks amd64
40.0-2 [293 kB]
Fetched 293 kB in 2초 (156 kB/s)
Selecting previously unselected package gnome-clocks.
(데이터베이스 읽는중 ...현재 184147개의 파일과 디렉터리가 설치되어 있습니다.)
Preparing to unpack .../gnome-clocks_40.0-2_amd64.deb ...
Unpacking gnome-clocks (40.0-2) ...
gnome-clocks (40.0-2) 설정하는 중입니다 ...
```

02. 우분투 패키지 설치

■ 우분투 소프트웨어 센터

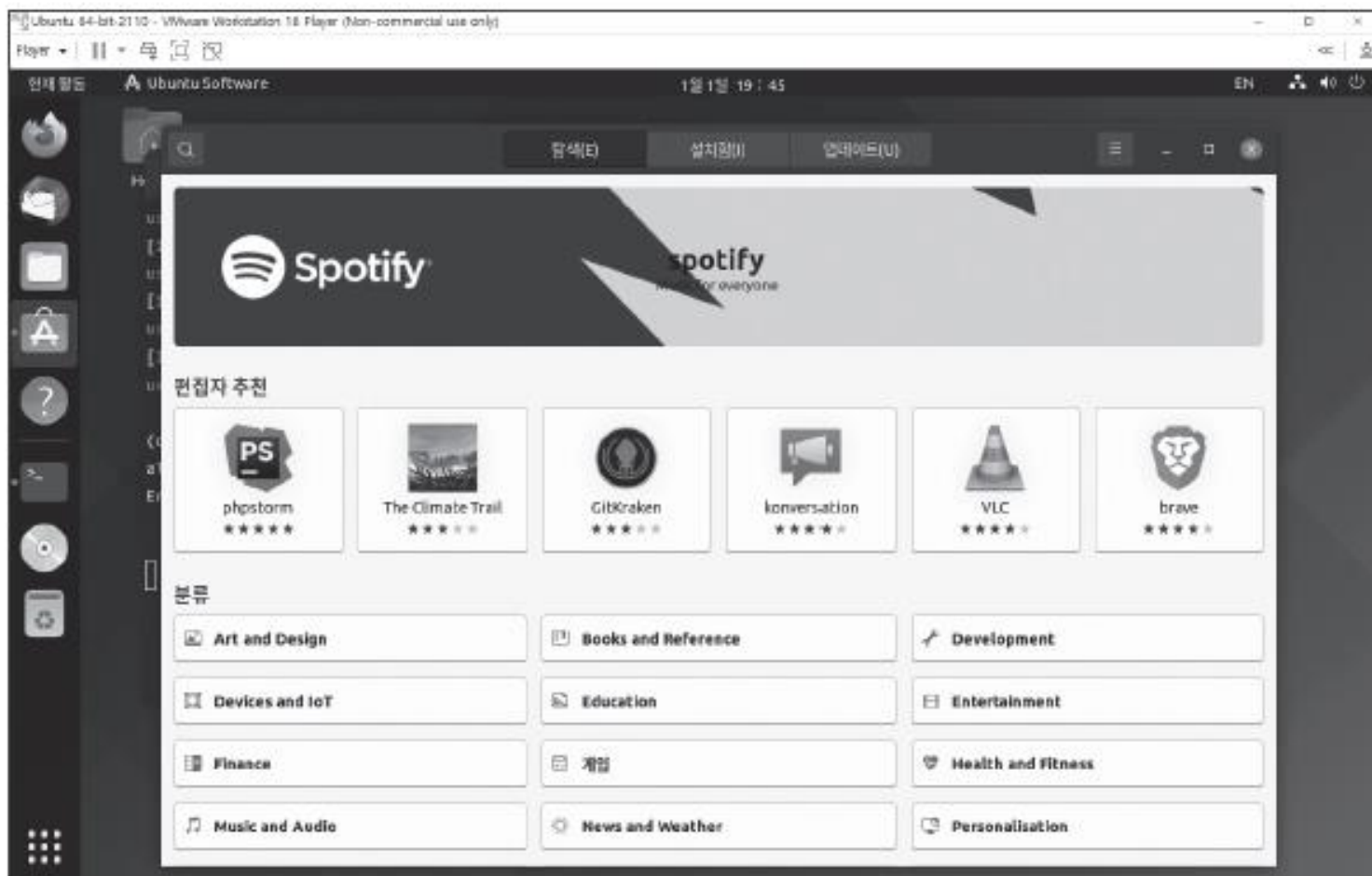


그림 9-8 우분투 소프트웨어 센터 실행 화면

03 스냅 패키지 설치

03. 스냅 패키지 설치

■ 스냅의 개념

- 샌드박스 형태의 패키지로 패키지를 만들 때 프로그램이 사용하는 모든 라이브러리를 패키지 안에 포함
- 샌드박스란 원래 외부에서 받은 파일을 그냥 실행하는 것이 아니라 보호된 영역에서 실행해 보는 것으로, 외부 파일이 내부 시스템에 악영향을 주는 것을 방지하는 보안 기술을 의미
- 스냅의 장점
 - 개발자가 다른 패키지나 라이브러리와 의존성을 신경 쓰지 않아도 된다.
 - 기존 시스템과 격리되어 실행하는 샌드박스 형식이므로 보안이 강화된다.
- 스냅의 단점
 - 패키지의 용량이 커짐

03. 스냅 패키지 설치

■ 스냅 사용하기

snap

- **기능** 스냅 패키지를 설치하고, 설정하고, 삭제한다.
- **형식** snap [옵션] 명령
- **옵션** -h: 도움말을 출력한다.
- **명령**
 - disable: 스냅 서비스와 실행 파일의 사용을 중지한다.
 - download 스냅명: 지정한 스냅 패키지를 내려받는다.
 - enable: 스냅 서비스와 실행 파일의 사용을 시작한다.
 - find 스냅명: 지정한 스냅을 검색한다.
 - info 스냅명: 지정한 스냅의 상세 정보를 출력한다.
 - install 스냅명: 지정한 스냅을 설치한다.
 - list: 설치한 스냅의 목록을 출력한다.
 - remove 스냅명: 지정한 스냅을 삭제한다.
- **사용 예**
 - snap list
 - snap install hello-world

03. 스냅 패키지 설치

- 스냅 목록 출력하기: list 명령

```
user1@myubuntu:~$ snap list
```

이름	버전	개정	추적	발행인	
노트					
bare	1.0	5	latest/stable	canonical	base
core	16-2.52.1	11993	latest/stable	canonical	core
core18	20211028	2253	latest/stable	canonical	base
core20	20211129	1270	latest/stable	canonical	base
firefox	95.0.2-1	777	latest/stable/...	mozilla	-
gitkraken	8.2.1	188	latest/stable	gitkraken	classic
gnome-3-34-1804	0+git.3556cb3	77	latest/stable	canonical	-
gnome-3-38-2004	0+git.cd626d1	87	latest/stable/...	canonical	-
gtk-common-themes	0.1-59-g7bca6ae	1519	latest/stable/...	canonical	-
snap-store	3.38.0-66-gbd5b8f7	558	latest/stable/...	canonical	-

03. 스냅 패키지 설치

- 스냅 찾기: find 명령

```
user1@myubuntu:~$ snap find hello-world
```

이름	버전	발행인	노트	요약
hello-world	6.4	canonical	-	The 'hello-world' of snaps
hello-world-lc	0.1	lolachangmwc2017	-	Just testing
hello-world-jbennett	0.1	jamiebennett	-	Just testing
hello-world-jbennett2	0.1	jamiebennett	-	Just testing

(생략)

03. 스냅 패키지 설치

- 스냅 설치하기: install 명령

```
user1@myubuntu:~$ sudo snap install hello-world  
core 14.62 MB / 83.73 MB [=====>-----] 17.47% 647.28 KB/s 1m49s
```

```
user1@myubuntu:~$ sudo snap install hello-world  
(생략)
```

스냅 "hello-world" (29) 보안 프로필 설정

스냅 "hello-world" (29) 보안 프로필 설정

스냅 "hello-world" (29) 보안 프로필 설정

스냅 "hello-world" (29) 보안 프로필 설정

hello-world6.4 from Canonical installed



```
user1@myubuntu:~$ hello-world  
Hello World!  
user1@myubuntu:~$
```

```
user1@myubuntu:~$ ls /snap
```

```
README  bin    core18  firefox  gnome-3-34-1804  gtk-common-themes  snap-store  
bare    core  core20  gitkraken  gnome-3-38-2004  hello-world
```

03. 스냅 패키지 설치

- 스냅의 상세 정보 확인하기: info 명령

```
user1@myubuntu:~$ snap info hello-world
name:      hello-world
summary:    The 'hello-world' of snaps
publisher: Canonical
store-url:  https://snapcraft.io/hello-world
contact:    snaps@canonical.com
license:    unset
description: |
  This is a simple hello world example.
commands:
  - hello-world.env
  - hello-world.evil
  - hello-world
  - hello-world.sh
snap-id:    buPKUD3TKqC0gLEjjHx5kSiCpIs5cMuQ
```

03. 스냅 패키지 설치

- 스냅 삭제하기: remove 명령

```
user1@myubuntu:~$ sudo snap remove hello-world
```

스냅 "hello-world" (29)에 대한 보안 프로필 제거

hello-world 삭제됨

```
user1@myubuntu:~$ ls /snap
```

README	bin	core18	firefox	gnome-3-34-1804	gtk-common-themes
bare	core	core20	gitkraken	gnome-3-38-2004	snap-store

465p. 따라해보기: 스냅 설치하고 삭제하기

- ① 설치할 스냅을 먼저 검색: game을 검색
- ② 검색된 스냅 중 game-2048 스냅을 설치, 설치가 끝나면 /snap 디렉터리를 확인
- ③ game-2048 스냅의 상세 정보를 확인
- ④ game-2048을 실행

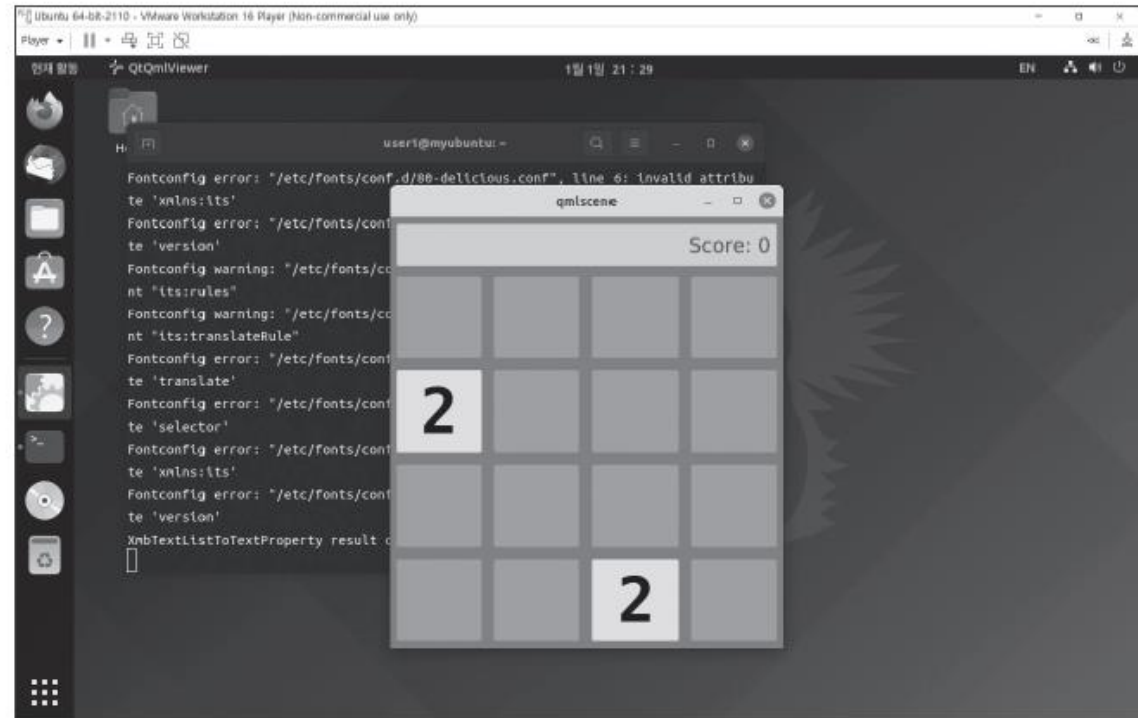


그림 9-12 game-2048 실행 화면

04 파일 아카이브와 압축

04. 파일 아카이브와 압축

■ 파일 아카이브

tar

- **기능** 파일과 디렉터리를 묶어 하나의 아카이브 파일을 생성한다.
- **형식** tar 기능[옵션] [아카이브 파일] [파일명]
- **기능**
 - c: 새로운 tar 파일을 생성한다.
 - t: tar 파일의 내용을 출력한다.
 - x: tar 파일에서 원본 파일을 추출한다.
 - r: 새로운 파일을 추가한다.
 - u: 수정된 파일을 업데이트한다.
- **옵션**
 - f: 아카이브 파일이나 테이프 장치를 지정한다. 파일명을 '-'으로 지정하면 tar 파일 대신 표준 입력에서 읽어들인다.
 - v: 처리하고 있는 파일의 정보를 출력한다.
 - h: 심볼릭 링크의 원본 파일을 포함한다.
 - p: 파일 복구 시 원래의 접근 권한을 유지한다.
 - j: bzip2로 압축하거나 해제한다.
 - z: gzip으로 압축하거나 해제한다.
- **사용 예**
 - tar cvf unix.tar Unix
 - tar xvf unix.tar

04. 파일 아카이브와 압축

■ 아카이브 생성하기: cvf 옵션

```
user1@myubuntu:~/linux_ex$ tar cvf ch2.tar ch2
ch2/
ch2/temp/
ch2/temp/text1
ch2/temp/hosts
ch2/temp/text2
ch2/temp/data1.cp
ch2/data1.ln
ch2/data
ch2/data1.sl
ch2/test
user1@myubuntu:~/linux_ex$ ls
ch2  ch2.tar  ch3  ch4  ch5  ch6
```

04. 파일 아카이브와 압축

■ 아카이브의 내용 확인하기: tvf 옵션

```
user1@myubuntu:~/linux_ex$ tar tvf ch2.tar
drwxrwxr-x user1/user1      0 2021-12-07 22:00 ch2/
drwxrwxr-x user1/user1      0 2021-12-07 22:00 ch2/temp/
-rw-r--r-- user1/user1     221 2021-12-07 20:20 ch2/temp/text1
-rw-r--r-- user1/user1     221 2021-12-07 20:21 ch2/temp/hosts
-rw-r--r-- user1/user1     221 2021-12-07 20:21 ch2/temp/text2
-rw-r--r-- user1/user1     223 2021-12-07 21:48 ch2/temp/data1.cp
-rw-r--r-- user1/user1     223 2021-12-07 21:36 ch2/data1.ln
-rw-r--r-- user1/user1    12813 2021-12-07 21:50 ch2/data
lrwxrwxrwx user1/user1      0 2021-12-07 21:38 ch2/data1.sl -> data1
-rw-rw-r-- user1/user1      0 2021-12-31 12:00 ch2/test
```


04. 파일 아카이브와 압축

■ 아카이브 풀기: xvf 옵션

```
user1@myubuntu:~/linux_ex$ cd ch9
user1@myubuntu:~/linux_ex/ch9$ tar xvf ch2.tar
ch2/
ch2/temp/
ch2/temp/text1
ch2/temp/hosts
ch2/temp/text2
ch2/temp/data1.cp
ch2/data1.ln
ch2/data
ch2/data1.sl
ch2/test
user1@myubuntu:~/linux_ex/ch9$ ls
ch2  ch2.tar
```

04. 파일 아카이브와 압축

■ 아카이브 업데이트하기: uvf 옵션

- 파일이 아카이브에 없는 파일이거나, 아카이브에 있는 파일이지만 수정된 파일일 때 아카이브의 마지막에 추가

```
user1@myubuntu:~/linux_ex/ch9$ touch ch2/data
user1@myubuntu:~/linux_ex/ch9$ tar uvf ch2.tar ch2
ch2/data
user1@myubuntu:~/linux_ex/ch9$ tar tvf ch2.tar
drwxrwxr-x user1/user1      0 2021-12-07 22:00 ch2/
```

수정된 파일 추가

```
-rw-r--r-- user1/user1 12813 2021-12-07 21:50 ch2/data
lrwxrwxrwx user1/user1    0 2021-12-07 21:38 ch2/data1.sl -> data1
-rw-rw-r-- user1/user1    0 2021-12-31 12:00 ch2/test
-rw-r--r-- user1/user1 12813 2022-01-02 10:26 ch2/data
```

04. 파일 아카이브와 압축

■ 아카이브에 파일 추가하기: rvf 옵션

- 지정한 파일을 무조건 아카이브의 마지막에 추가

```
user1@myubuntu:~/linux_ex/ch9$ cp /etc/services .
user1@myubuntu:~/linux_ex/ch9$ tar rvf ch2.tar services
services
user1@myubuntu:~/linux_ex/ch9$ tar tvf ch2.tar
drwxrwxr-x user1/user1      0 2021-12-07 22:00 ch2/
drwxrwxr-x user1/user1      0 2021-12-07 22:00 ch2/temp/
-rw-r--r-- user1/user1    221 2021-12-07 20:20 ch2/temp/text1
-rw-r--r-- user1/user1    221 2021-12-07 20:21 ch2/temp/hosts
-rw-r--r-- user1/user1    221 2021-12-07 20:21 ch2/temp/text2
-rw-r--r-- user1/user1    223 2021-12-07 21:48 ch2/temp/data1.cp
-rw-r--r-- user1/user1    223 2021-12-07 21:36 ch2/data1.ln
-rw-r--r-- user1/user1   12813 2021-12-07 21:50 ch2/data
lrwxrwxrwx user1/user1      0 2021-12-07 21:38 ch2/data1.sl -> data1
-rw-rw-r-- user1/user1      0 2021-12-31 12:00 ch2/test
-rw-r--r-- user1/user1   12813 2022-01-02 10:26 ch2/data
-rw-r--r-- user1/user1   12813 2022-01-02 10:27 services
```

04. 파일 아카이브와 압축

■ 아카이브 생성하고 압축하기

- 아카이브를 생성하면서 동시에 압축 가능: gzip으로 압축할 경우 z 옵션 사용

```
user1@myubuntu:~/linux_ex/ch9$ tar czvf ch2.tar.gz ch2
ch2/
ch2/temp/
ch2/temp/text1
ch2/temp/hosts
ch2/temp/text2
ch2/temp/data1.cp
ch2/data1.ln
ch2/data
ch2/data1.sl
ch2/test
user1@myubuntu:~/linux_ex/ch9$ ls
ch2  ch2.tar  ch2.tar.gz  services
```

04. 파일 아카이브와 압축

- bzip2로 압축할 경우 j 옵션을 사용

```
user1@myubuntu:~/linux_ex/ch9$ tar cvjf ch2.tar.bz2 ch2
ch2/
ch2/temp/
ch2/temp/text1
ch2/temp/hosts
ch2/temp/text2
ch2/temp/data1.cp
ch2/data1.ln
ch2/data
ch2/data1.sl
ch2/test
user1@myubuntu:~/linux_ex/ch9$ ls
ch2  ch2.tar  ch2.tar.bz2  ch2.tar.gz  services
```

04. 파일 아카이브와 압축

- 압축한 아카이브 파일의 내용은 tvf로 확인이 가능하며 xvf로 추출 가능

```
user1@myubuntu:~/linux_ex/ch9$ tar tvf ch2.tar.gz
drwxrwxr-x user1/user1      0 2021-12-07 22:00 ch2/
drwxrwxr-x user1/user1      0 2021-12-07 22:00 ch2/temp/
-rw-r--r-- user1/user1    221 2021-12-07 20:20 ch2/temp/text1
-rw-r--r-- user1/user1    221 2021-12-07 20:21 ch2/temp/hosts
-rw-r--r-- user1/user1    221 2021-12-07 20:21 ch2/temp/text2
-rw-r--r-- user1/user1    223 2021-12-07 21:48 ch2/temp/data1.cp
-rw-r--r-- user1/user1    223 2021-12-07 21:36 ch2/data1.ln
-rw-r--r-- user1/user1   12813 2022-01-02 10:26 ch2/data
lrwxrwxrwx user1/user1      0 2021-12-07 21:38 ch2/data1.sl -> data1
-rw-rw-r-- user1/user1      0 2021-12-31 12:00 ch2/test
```

04. 파일 아카이브와 압축

■ 파일 압축: gzip/gunzip: .gz 파일

gzip

- **기능** 파일을 압축한다.
- **형식** gzip [옵션] [파일명]
- **옵션**
 - d: 파일 압축을 해제한다.
 - l: 압축된 파일의 정보를 보여준다.
 - r: 하위 디렉터리를 탐색하여 압축한다.
 - t: 압축 파일을 검사한다.
 - v: 압축 정보를 화면에 출력한다.
 - 9: 최대한 압축한다.
- **사용 예**
 - gzip a.txt
 - gzip -v b.txt c.txt

- gzip에서 -l 옵션을 사용하여 압축 파일에 대한 정보 확인

```
user1@myubuntu:~/linux_ex/ch9$ gzip -l ch2.tar.gz
```

compressed	uncompressed	ratio	uncompressed_name
6182	51200	88.0%	ch2.tar

04. 파일 아카이브와 압축

- 압축 파일의 내용 보기: zcat 명령

zcat

- **기능** gzip으로 압축된 파일의 내용을 출력한다.
- **형식** zcat [파일명]
- **사용 예** zcat abc.gz

```
user1@myubuntu:~/linux_ex/ch9$ zcat ch2.tar.gz | more
ch2/0000775000175000017500000000000014153655341010423 5ustar  user1user1
ch2/temp/00007750001750000175000
000000000014153655341011370 5ustar  user1user1
ch2/temp/text10000644000175000017500000000033514153641624012356 0
ustar  user1user1127.0.0.1      localhost
127.0.1.1      ubuntu

# The following lines are desirable for IPv6 capable hosts
::1      ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```


04. 파일 아카이브와 압축

- 압축 풀기: gunzip 명령

gunzip

- **기능** gzip로 압축된 파일의 압축을 푼다.
- **형식** gunzip [파일명]
- **사용 예** gunzip abc.gz
gunzip abc

```
user1@myubuntu:~/linux_ex/ch9$ gunzip ch2.tar.gz
user1@myubuntu:~/linux_ex/ch9$ ls
ch2  ch2.tar  ch2.tar.bz2  services
```

04. 파일 아카이브와 압축

■ bzip2/bunzip2: .bz2 파일

bzip2

- **기능** 파일을 압축한다.
- **형식** bzip2 [옵션] [파일명]
- **옵션**
 - d: 파일 압축을 해제한다.
 - l: 압축된 파일의 정보를 보여준다.
 - t: 압축 파일을 검사한다.
 - v: 압축 정보를 화면에 출력한다.
 - best: 최대한 압축한다.
- **사용 예**
bzip2 abc.txt
bzip2 -v a.txt b.txt

```
user1@myubuntu:~/linux_ex/ch9$ rm ch2.tar.bz2
user1@myubuntu:~/linux_ex/ch9$ bzip2 ch2.tar
user1@myubuntu:~/linux_ex/ch9$ ls
ch2  ch2.tar.bz2  services
```

04. 파일 아카이브와 압축

- 압축 파일의 내용 보기: bzip2 명령

bzcat

- **기능** 압축된 파일의 내용을 출력한다.
- **형식** bzcat [파일명]
- **사용 예** bzcat abc.bz2
bzcat abc

- 압축 풀기: bunzip2 명령

bunzip2

- **기능** bzip2로 압축된 파일의 압축을 푼다.
- **형식** bunzip2 [파일명]
- **사용 예** bunzip2 abc.txt.bz2
bunzip2 abc.txt

```
user1@myubuntu:~/linux_ex/ch9$ bunzip2 ch2.tar.bz2
user1@myubuntu:~/linux_ex/ch9$ ls
ch2  ch2.tar  services
```

05 소프트웨어 컴파일

05. 소프트웨어 컴파일

■ C 컴파일러 설치: gcc

```
user1@myubuntu:~/linux_ex/ch9$ sudo apt install gcc
```

패키지 목록을 읽는 중입니다... 완료

의존성 트리를 만드는 중입니다... 완료

05. 소프트웨어 컴파일

■ 간단한 C 프로그램 컴파일 및 실행

```
user1@myubuntu:~/linux_ex/ch9$ vi hello.c

#include <stdio.h>

int main() {
    printf("Hello, World\n");
}

:wq
```



```
user1@myubuntu:~/linux_ex/ch9$ gcc hello.c
user1@myubuntu:~/linux_ex/ch9$ ls
a.out  ch2  ch2.tar  hello.c  services
```



```
user1@myubuntu:~/linux_ex/ch9$ a.out
a.out: 명령을 찾을 수 없습니다
```



```
user1@myubuntu:~/linux_ex/ch9$ ./a.out
Hello, World
```

05. 소프트웨어 컴파일

- 실행 파일명 변경하기: -o 옵션 사용

```
user1@myubuntu:~/linux_ex/ch9$ gcc -o hello hello.c
user1@myubuntu:~/linux_ex/ch9$ ./hello
Hello, World
```

05. 소프트웨어 컴파일

■ make 명령 사용하기

- make 명령은 makefile(또는 Makefile)에 설정된 정보를 읽어서 여러 소스 파일을 컴파일하고 링크하여 최종 실행 파일을 생성

■ make 명령 설치하기

```
user1@myubuntu:~/linux_ex/ch9$ sudo apt install make
패키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다... 완료
상태 정보를 읽는 중입니다... 완료
제안하는 패키지:
  make-doc
다음 새 패키지를 설치할 것입니다:
  make
```


05. 소프트웨어 컴파일

- 소스 파일 준비하기

```
user1@myubuntu:~/linux_ex/ch9$ vi one.c
#include <stdio.h>

extern int two();

int main() {
    printf("Go to Module Two--\n");
    two();
    printf("End of Module One.\n");
}

:wq
```

```
user1@myubuntu:~/linux_ex/ch9$ vi two.c
#include <stdio.h>

int two() {
    printf("In Module Two--\n");
    printf("--- This is a Module Two.\n");
    printf("End of Module Two.\n");
}

:wq
```

05. 소프트웨어 컴파일

- makefile 작성 및 실행

```
user1@myubuntu:~/linux_ex/ch9$ vi
TARGET=one
OBJECTS=one.o two.o

${TARGET} : ${OBJECTS}
    gcc -o ${TARGET} ${OBJECTS}

one.o : one.c
    gcc -c one.c
two.o : two.c
    gcc -c two.c

:wq
```

```
user1@myubuntu:~/linux_ex/ch9$ make
gcc -c one.c
gcc -c two.c
gcc -o one one.o two.o
user1@myubuntu:~/linux_ex/ch9$ ./one
Go to Module Two--
In Module Two--
--- This is a Module Two.
End of Module Two.
End of Module One.
```

Thank You !