

공개SW 솔루션 설치 & 활용 가이드

미들웨어 > WAS

uWSGI



제대로 배워보자

How to Use Open Source Software

---

Open Source Software Installation & Application Guide



오픈소스 소프트웨어 통합지원센터  
Open Source Software Support Center



# CONTENTS

1. 개요
2. 기능요약
3. 실행환경
4. 설치 및 실행
5. 기능소개
6. 활용예제
7. FAQ
8. 용어정리

# 1. 개요

uWSGI



<b>소개</b>	• WSGI란 Web Server Gateway Interface의 약자로, 웹서버와 웹어플리케이션이 어떤 방식으로 통신하는가에 관한 인터페이스를 의미, 이 WSGI 규격에 맞춘 코드를 실행해주는 프로그램이 uWSGI임. 이 외에 Gunicorn 같은 프로그램 또한 WSGI 규격 웹 애플리케이션을 배포할 때 쓰임		
<b>주요기능</b>	• 어플리케이션 인터페이스 • 미들웨어 인터페이스		
<b>대분류</b>	• 미들웨어	<b>소분류</b>	• WAS
<b>라이선스형태</b>	• GNU General Public License v2.0 with linking exception	<b>사전설치 솔루션</b>	• N/A
<b>운영제제</b>	• Windows, Linux, Unix	<b>버전</b>	• 2.0.15 (March 30, 2017)
<b>특징</b>	• 환경 변수를 적절히 변경하여 대상 URL에 따라 다른 애플리케이션 개체에 요청을 라우팅 • 동일한 프로세스에서 여러 애플리케이션 또는 프레임워크를 나란히 실행할 수 있도록 허용 • 네트워크를 통해 요청 및 응답을 전달하여 로드 밸런싱 및 원격 처리 • XSLT스타일을 적용하는 등의 콘텐츠 처리 수행		
<b>보안취약점</b>	• N/A		
<b>개발회사/커뮤니티</b>	• Unbit		
<b>공식 홈페이지</b>	• <a href="http://uwsgi-docs.readthedocs.io/en/latest/">http://uwsgi-docs.readthedocs.io/en/latest/</a>		

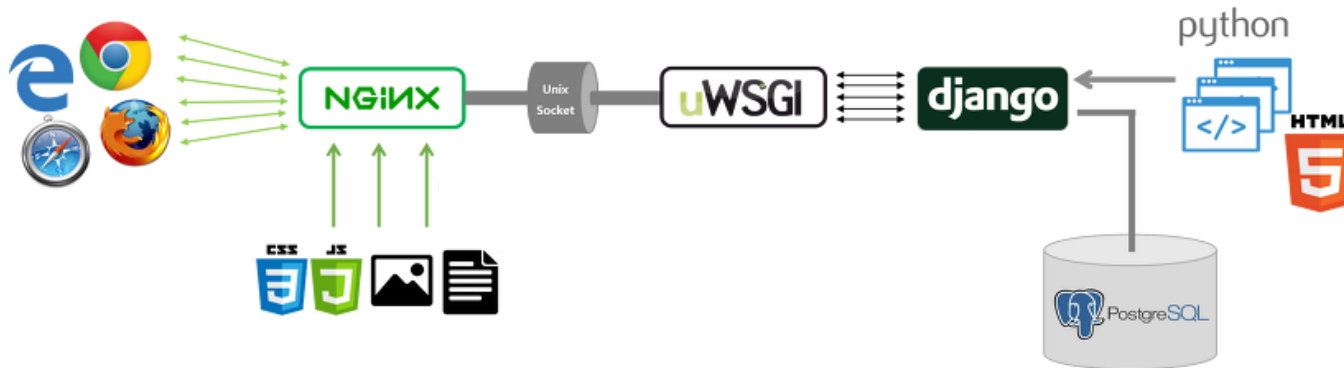


## 2. 기능요약

uWSGI



- 웹서버 게이트웨어 인터페이스(WSGI, Web Server Gateway Interface)는 웹서버와 웹 애플리케이션의 인터페이스를 위한 파이썬(Pyton) 프레임워크
  - ※ WSGI 종류: Bjoern, uWAGI, mod\_wsgi, Gunicorn
  - ※ Gunicorn: Unix용 Python WSGI HTTP Server 임(ruby unicorn 프로젝트에서 포크됨)
- uWSGI는 WSGI의 한 종류로 호스팅 서버를 구축할 수 있는 Full Stack 지원
- Django(파이썬 기반의 웹 프레임워크)와 호환성이 좋음
- uWSGI는 타 언어나 플랫폼을 다룰 수 있도록 Pluggable architecture 지원(예: C, C++지원)
- Plugin이 많을 수록 프로젝트의 규모가 커질 수 있음



< 서비스 아키텍처 도식도 > 출처: <http://technerd.tistory.com/55>



# 3. 실행환경

uWSGI



## 지원 가능한 플랫폼/시스템

- Linux 2.6/3.x
- FreeBSD
- NetBSD
- OpenBSD
- DragonFlyBSD
- Windows Cygwin
- Mac OSX
- Solaris >= 10
- NexentaOS
- SmartOS
- OpenSolaris
- OpenIndiana
- OmniOS
- Debian/kFreeBSD
- GNU/Hurd

## 지원 가능한 언어 및 플랫폼

기술	버전	
Python	0.9.1	Stable, 100% uWSGI API support
Lua	0.9.5	Stable, 60% uWSGI API support
Perl	0.9.5	Stable, 60% uWSGI API support
Ruby	0.9.7-dev	Stable, 80% uWSGI API support
Integrating uWSGI with Erlang	0.9.5	Stable, no uWSGI API support
Running CGI scripts on uWSGI	1.0-dev	Stable, no uWSGI API support
Running PHP scripts in uWSGI	1.0-dev	Stable from 1.1, 5% uWSGI API support
uWSGI Go support (1.4 only)	1.4-dev	15% uWSGI API support
JVM in the uWSGI server (updated to 1.9)	1.9-dev	Stable
The Mono ASP.NET plugin	0.9.7-dev	Stable
uWSGI V8 support	1.9.4	Early stage of development



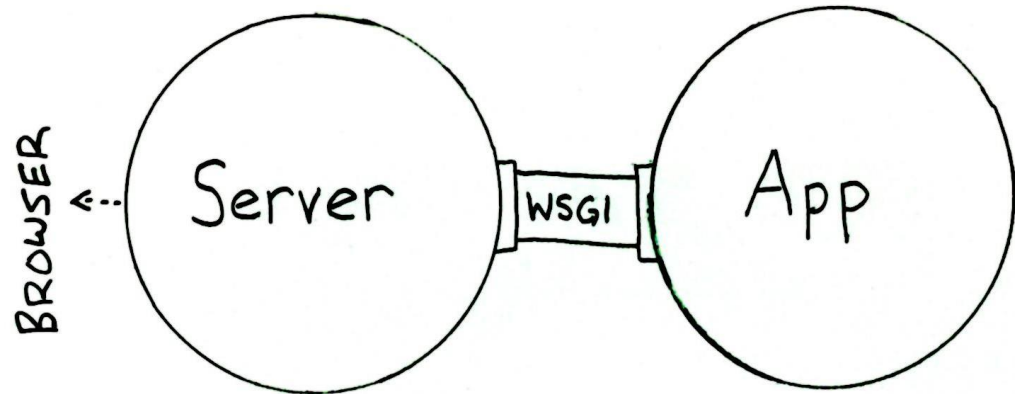
# 4. 설치 및 실행

uWSGI



세부 목차

1. uWSGI 설치
2. uWSGI 서버 실행
3. uWSGI 설정



\* 출처: <http://kernel-for-trace-based-systems.readthedocs.io>



# 4. 설치 및 실행

uWSGI



## 4.1 uWSGI 설치

여기서는 Ubuntu 기반에서 설치한다.

설치를 위해 Python과 C컴파일러(gcc, clang)가 필요하다.

Python은 linux에 기본으로 설치되어 있기 때문에 `pip install uwsgi` 로 설치 가능하다.

```
root@master: ~
root@master:~# pip install uwsgi
Collecting uwsgi
  Downloading uwsgi-2.0.15.tar.gz (795kB)
    100% |████████████████████████████████████████| 798kB 476kB/s
Building wheels for collected packages: uwsgi
  Running setup.py bdist_wheel for uwsgi ... done
  Stored in directory: /root/.cache/pip/wheels/26/d0/48/e7b0eed63b5d191e89d94e72196aafae93b2b6505a9feafdd9
Successfully built uwsgi
Installing collected packages: uwsgi
Successfully installed uwsgi-2.0.15
root@master:~#
```



# 4. 설치 및 실행

uWSGI



## 4.2 uWSGI 서버 실행

서버 실행은 uwsgi 명령어를 통해 실행 가능하지만 간단한 스크립트를 만들어 실행해 보자.

/etc/init 디렉토리에 있는 선언적 구성 파일을 기반으로 실행한다

### simple script (/etc/init/uwsgi.conf)

```
# simple uWSGI script
description "uwsgi tiny instance"
start on runlevel [2345]
stop on runlevel [06]

respawn

exec uwsgi --master --processes 4 --die-on-term --socket :3031 --wsgi-file /var/www/myapp.wsgi
```





# 4. 설치 및 실행

uWSGI



## 4.3 uWSGI 설정(1/2)

uWSGI는 여러가지 포맷(ini,xml, yaml, json)을 지원한다.

```
uwsgi --ini http://uwsgi.it/configs/myapp.ini # HTTP
```

```
uwsgi --xml - # standard input
```

```
uwsgi --yaml fd://0 # file descriptor
```

```
uwsgi --json 'exec://nc 192.168.11.2:33000' # arbitrary executable
```



# 4. 설치 및 실행

uWSGI



## 4.3 uWSGI 설정(2/2)

.INI 파일은 많은 응용 프로그램에서 사용되는 사실상 표준 구성 형식이다.

[section] key = value 쌍으로 구성된다.

```
[uwsgi]
socket = /tmp/uwsgi.sock
socket = 127.0.0.1:8000
workers = 3
master = true
```

기본적으로 uWSGI는 [uwsgi] 섹션을 사용하지만 filename : section 구문과 함께 INI 파일을 로드하는 동안 다른 섹션 이름을 지정할 수 있다.

```
uwsgi --ini myconf.ini:app1
```

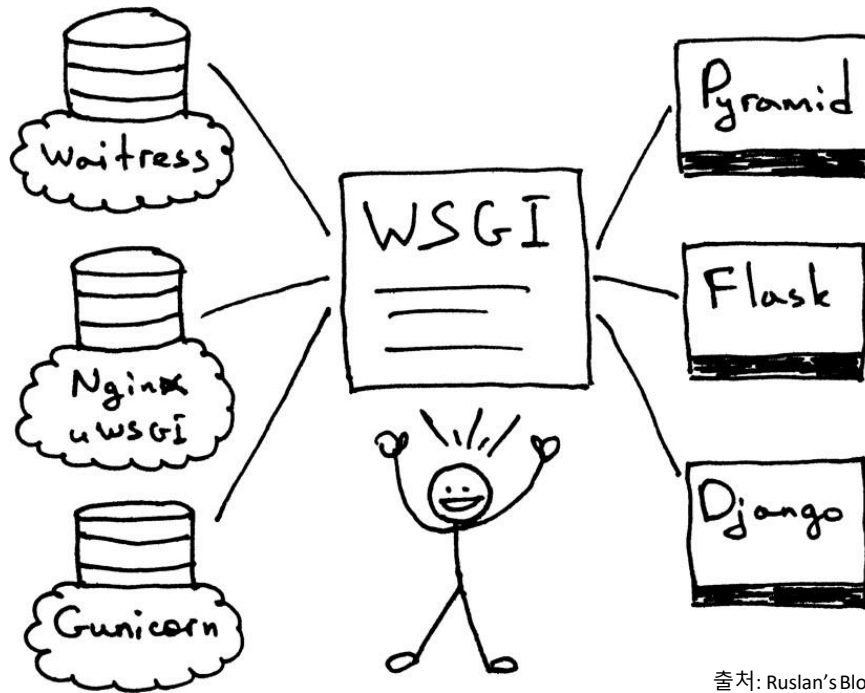


# 5. 기능소개



## 세부 목차

- 1. uWSGI 로깅
- 2. uWSGI 로그 포맷



출처: Ruslan's Blog



# 5. 기능소개

uWSGI



## 5.1 uWSGI 로깅

uWSGI에서 가장 기본적인 형태의 로깅은 요청, 오류 및 정보 메시지를 stdout / stderr 으로 생성한다. 로그 리다이렉션의 가장 기본적인 형태는 --logto / --logto2 / --daemonize 옵션으로, 로그를 파일로 정할 수 있다.

```
./uwsgi -s :3031 -w simple_app --daemonize /tmp/mylog.log  
./uwsgi -s :3031 -w simple_app --logto /tmp/mylog.log  
./uwsgi -s :3031 -w simple_app --uid 1001 --gid 1002 --logto2 /tmp/mylog.log
```

stdout / stderr 대신 파일에 기록하려면 --logto를 사용하거나 uWSGI, - daemonize를 동시에 데몬을 실행하면 된다.



# 5. 기능소개

uWSGI



## 5.2 uWSGI 로그 포맷

uWSGI에는 사용자 정의 요청 로그 라인을 작성하기 위한 `--logformat` 옵션이 있다.

- `%(uri)` -> `REQUEST_URI`
- `%(method)` -> `REQUEST_METHOD`
- `%(user)` -> `REMOTE_USER`
- `%(addr)` -> `REMOTE_ADDR`
- `%(host)` -> `HTTP_HOST`
- `%(proto)` -> `SERVER_PROTOCOL`
- `%(uagent)` -> `HTTP_USER_AGENT` (starting from 1.4.5)
- `%(referer)` -> `HTTP_REFERER` (starting from 1.4.5)

다음 로그 형식을 사용하면 코드 정의 `logvars`에 액세스 할 수 있다.

```
uwsgi --logformat 'worker id = %(worker_id) for request "%(method) %(uri) %(proto)" test = %(foo)
```



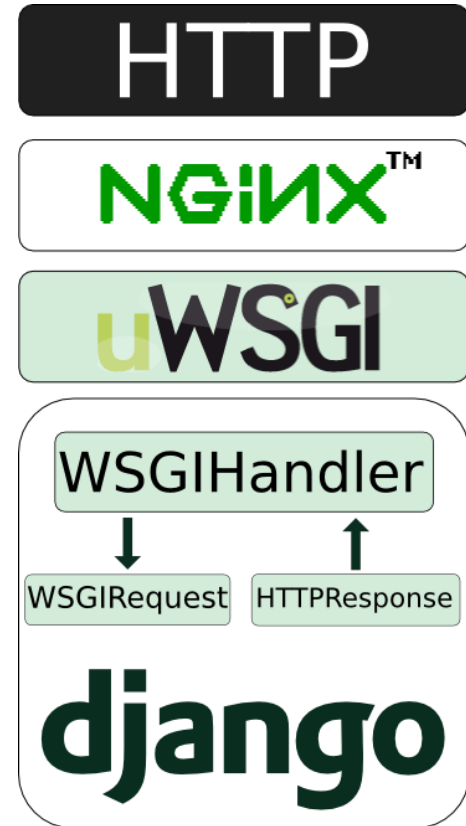
# 6. 활용예제

uWSGI



세부 목차

- 1. Python + Apache mod\_wsgi 연동
- 2. Apache mode\_wsgi + Django 연동



# 6. 활용예제

uWSGI



## 6.1 Python + Apache mod\_wsgi 연동(1/8)

### Apache mod\_wsgi란?

mod\_wsgi의 목적은 Python WSGI 인터페이스를 지원하는 Python 응용 프로그램을 호스팅 할 수 있는 간단한 ApacheModule을 구현하는 것이다. 이 모듈은 웹 호스팅 서비스에서 실행되는 고성능 운영 사이트뿐만 아니라 고성능 프로덕션 웹 사이트 호스팅에도 사용하기에 적합하다.

### 지원 가능한 웹프레임워크 및 어플리케이션

- CherryPy
- Django
- Pylons
- TurboGears
- Pyramid
- web.py
- Werkzeug
- Web2Py and Zope.





# 6. 활용예제



## 6.1 Python + Apache mod\_wsgi 연동(2/8)

### 아파치 설치(1/2)

```
# apt-get install apache2
```

```
root@master: /opt
root@master:/opt# apt-get install apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-headers-4.8.0-36 linux-headers-4.8.0-36-generic linux-headers-4.8.0-41
  linux-headers-4.8.0-41-generic linux-image-4.8.0-36-generic
  linux-image-4.8.0-41-generic linux-image-extra-4.8.0-36-generic
  linux-image-extra-4.8.0-41-generic
Use 'apt autoremove' to remove them.
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.1-0
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.1-0
0 upgraded, 9 newly installed, 0 to remove and 241 not upgraded.
Need to get 1,544 kB of archives.
After this operation, 6,369 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```





# 6. 활용예제



## 6.1 Python + Apache mod\_wsgi 연동(3/8)

### 아파치 설치(2/2)

```
# apt-get install apache2
```

```
root@master: /opt
Enabling module authn_core.
Enabling module auth_basic.
Enabling module access_compat.
Enabling module authn_file.
Enabling module authz_user.
Enabling module alias.
Enabling module dir.
Enabling module autoindex.
Enabling module env.
Enabling module mime.
Enabling module negotiation.
Enabling module setenvif.
Enabling module filter.
Enabling module deflate.
Enabling module status.
Enabling conf charset.
Enabling conf localized-error-pages.
Enabling conf other-vhosts-access-log.
Enabling conf security.
Enabling conf serve-cgi-bin.
Enabling site 000-default.
Processing triggers for libc-bin (2.23-0ubuntu7) ...
Processing triggers for systemd (229-4ubuntu16) ...
Processing triggers for ureadahead (0.100.0-19) ...
Processing triggers for ufw (0.35-0ubuntu2) ...
root@master:/opt#
```







# 6. 활용예제

uWSGI



## 6.1 Python + Apache mod\_wsgi 연동 (6/8)

### 아파치 wsgi 설정

```
$ a2enconf wsgi
```

```
$ systemctl restart apache2
```

```
root@master: ~  
root@master:~# a2enconf wsgi  
Enabling conf wsgi.  
To activate the new configuration, you need to run:  
  service apache2 reload  
root@master:~# systemctl restart apache2  
root@master:~#
```



# 6. 활용예제

uWSGI



## 6.1 Python + Apache mod\_wsgi 연동(7/8)

### test\_wsgi.py 생성

```
root@www:~# vi /var/www/html/test_wsgi.py
# create new
def application(environ,start_response):
    status = '200 OK' h
    tml = '<html>\n' \
        '<body>\n' \
        '<div style="width: 100%; font-size: 40px; font-weight: bold; text-align: center;">\n' \
        'mod_wsgi Test Page\n' \
        '</div>\n' \
        '</body>\n' \
        '</html>\n'
    response_header = [('Content-type','text/html')]
    start_response(status,response_header)
    return [tml]
```



# 6. 활용예제

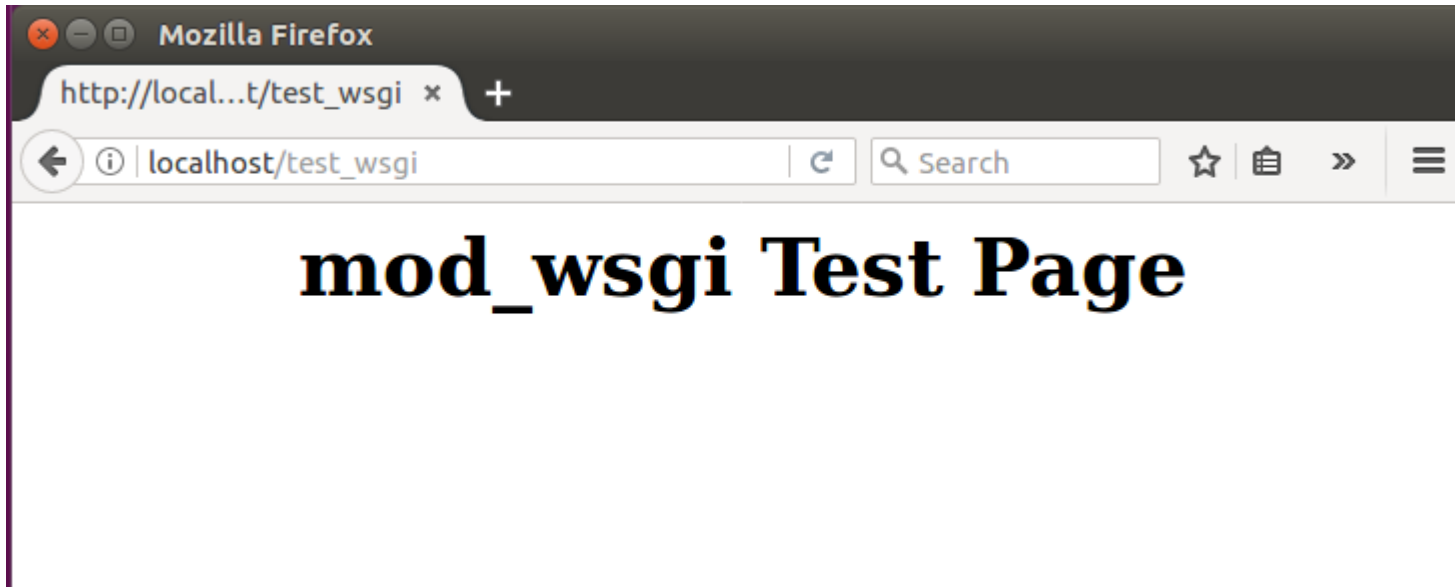
uWSGI



## 6.1 Python + Apache mod\_wsgi 연동 (8/8)

### test\_wsgi 페이지 확인

웹브라우저에서 [http://localhost/test\\_wsgi](http://localhost/test_wsgi) 접속하여 확인한다.



# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동(1/20)

### 사전 환경 구축

- virtualenv 설치
- django 설치
- Apache2 설치
- mod\_wsgi 설치



# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동(2/20)

### Django란?

Django는 안전하고 유지 관리 가능한 웹 사이트를 신속하게 개발할 수 있는 고차원적인 Python 기반 웹 프레임워크이다. 경험이 풍부한 개발자들에 의해 구축된 장고(Django)는 웹 개발의 번거로움을 해소할 수 있도록 하기 위해 고안되었다. 자유롭고 개방적이며 활발한 커뮤니티를 보유하고 있으며, 우수한 문서 및 유료 지원을 위한 다양한 옵션을 갖추고 있다.

# django





# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동(3/20)

### virtualenv 설치

가상 작업환경인 virtualenv를 설치한다. root

```
@www:~# apt-get -y install python-virtualenv
```

```
root@master: ~
root@master:~# apt-get -y install python-virtualenv
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-headers-4.8.0-36 linux-headers-4.8.0-36-generic linux-headers-4.8.0-41
  linux-headers-4.8.0-41-generic linux-headers-4.8.0-45
  linux-headers-4.8.0-45-generic linux-image-4.8.0-36-generic
  linux-image-4.8.0-41-generic linux-image-4.8.0-45-generic
  linux-image-extra-4.8.0-36-generic linux-image-extra-4.8.0-41-generic
  linux-image-extra-4.8.0-45-generic
Use 'apt autoremove' to remove them.
The following additional packages will be installed:
  python3-virtualenv virtualenv
The following NEW packages will be installed:
  python-virtualenv python3-virtualenv virtualenv
0 upgraded, 3 newly installed, 0 to remove and 222 not upgraded.
Need to get 94.1 kB of archives.
After this operation, 316 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 python-v
irtualenv all 15.0.1+ds-3ubuntu1 [46.6 kB]
Get:2 http://us.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 python3-
virtualenv all 15.0.1+ds-3ubuntu1 [43.2 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 virtuale
nv all 15.0.1+ds-3ubuntu1 [4,342 B]
Fetched 94.1 kB in 5s (18.3 kB/s)
Selecting previously unselected package python-virtualenv.
(Reading database ... 317199 files and directories currently installed.)
Preparing to unpack ../python-virtualenv_15.0.1+ds-3ubuntu1_all.deb ...
Unpacking python-virtualenv (15.0.1+ds-3ubuntu1) ...
Selecting previously unselected package python3-virtualenv.
Preparing to unpack ../python3-virtualenv_15.0.1+ds-3ubuntu1_all.deb ...
Unpacking python3-virtualenv (15.0.1+ds-3ubuntu1) ...
Selecting previously unselected package virtualenv.
Preparing to unpack ../virtualenv_15.0.1+ds-3ubuntu1_all.deb ...
Unpacking virtualenv (15.0.1+ds-3ubuntu1) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up python-virtualenv (15.0.1+ds-3ubuntu1) ...
Setting up python3-virtualenv (15.0.1+ds-3ubuntu1) ...
Setting up virtualenv (15.0.1+ds-3ubuntu1) ...
root@master:~#
```



# 6. 활용예제



## 6.2 Apache mode\_wsgi + Django 연동(4/20)

### virtualenv 설정

```
ubuntu@www:~$ virtualenv venv # virtualenv 생성
ubuntu@www:~$ cd ~/venv ubuntu@www:~/venv
$ source bin/activate (venv)ubuntu@www:~/venv$
pip install -U pip
(venv)ubuntu@www:~/venv$ pip install django #django 설치
```

```
root@master: ~/venv
root@master:~# virtualenv venv
New python executable in /root/venv/bin/python
Installing setuptools, pip, wheel...done.
root@master:~# cd ~/venv
root@master:~/venv# source bin/activate
(venv) root@master:~/venv# pip install -U pip
Requirement already up-to-date: pip in ./lib/python2.7/site-packages
(venv) root@master:~/venv# pip install django
Collecting django
  Downloading Django-1.11.4-py2.py3-none-any.whl (6.9MB)
    100% |████████████████████████████████████████| 7.0MB 165kB/s
Collecting pytz (from django)
  Downloading pytz-2017.2-py2.py3-none-any.whl (484kB)
    100% |████████████████████████████████████████| 491kB 1.7MB/s
Installing collected packages: pytz, django
Successfully installed django-1.11.4 pytz-2017.2
(venv) root@master:~/venv# django-admin --version
1.11.4
```



# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동(5/20)

### 테스트 프로젝트 생성

```
Ubuntu@www:~$ cd ~/venv Ubuntu@ww
w:~/venv$ source bin/activate # create "tes
tproject"
(venv)ubuntu@www:~/venv$ django-admin startproject testproject
(venv)ubuntu@www:~/venv$ cd testproject
```

```
root@master: ~/venv/testproject
root@master:~/venv# cd ~/venv
root@master:~/venv# source bin/activate
(venv) root@master:~/venv# django-admin startproject testproject
(venv) root@master:~/venv# cd testproject
(venv) root@master:~/venv/testproject#
```



# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동(6/20)

### 데이터베이스 설정(SQLite)(1/2)

```
# configure database (default is SQLite) (venv)ubuntu@www:~/venv/  
testproject$ python manage.py migrate
```

```
root@master: ~/venv/testproject  
(venv) root@master:~/venv/testproject# python manage.py migrate  
Operations to perform:  
  Apply all migrations: admin, auth, contenttypes, sessions  
Running migrations:  
  Applying contenttypes.0001_initial... OK  
  Applying auth.0001_initial... OK  
  Applying admin.0001_initial... OK  
  Applying admin.0002_logentry_remove_auto_add... OK  
  Applying contenttypes.0002_remove_content_type_name... OK  
  Applying auth.0002_alter_permission_name_max_length... OK  
  Applying auth.0003_alter_user_email_max_length... OK  
  Applying auth.0004_alter_user_username_opts... OK  
  Applying auth.0005_alter_user_last_login_null... OK  
  Applying auth.0006_require_contenttypes_0002... OK  
  Applying auth.0007_alter_validators_add_error_messages... OK  
  Applying auth.0008_alter_user_username_max_length... OK  
  Applying sessions.0001_initial... OK  
(venv) root@master:~/venv/testproject#
```



# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동(7/20)

### 데이터베이스 설정(SQLite)(2/2)

```
# create admin user  
(venv)ubuntu@www:~/venv/testproject$ python manage.py createsuperuser
```

```
root@master: ~/venv/testproject  
(venv) root@master:~/venv/testproject# python manage.py createsuperuser  
Username (leave blank to use 'root'):  
Email address: test@helloworld.co.kr  
Password:  
Password (again):  
Superuser created successfully.  
(venv) root@master:~/venv/testproject#
```





# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동(8/20)

### 서버 실행

```
# start server  
venv)ubuntu@www:~/venv/testproject$ python manage.py runserver 0.0.0.0:8000
```

```
root@master: ~/venv/testproject  
(venv) root@master:~/venv/testproject# python manage.py runserver 0.0.0.0:8000  
Performing system checks...  
  
System check identified no issues (0 silenced).  
August 17, 2017 - 08:40:30  
Django version 1.11.4, using settings 'testproject.settings'  
Starting development server at http://0.0.0.0:8000/  
Quit the server with CONTROL-C.
```



# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동(9/20)

### 서비스 확인

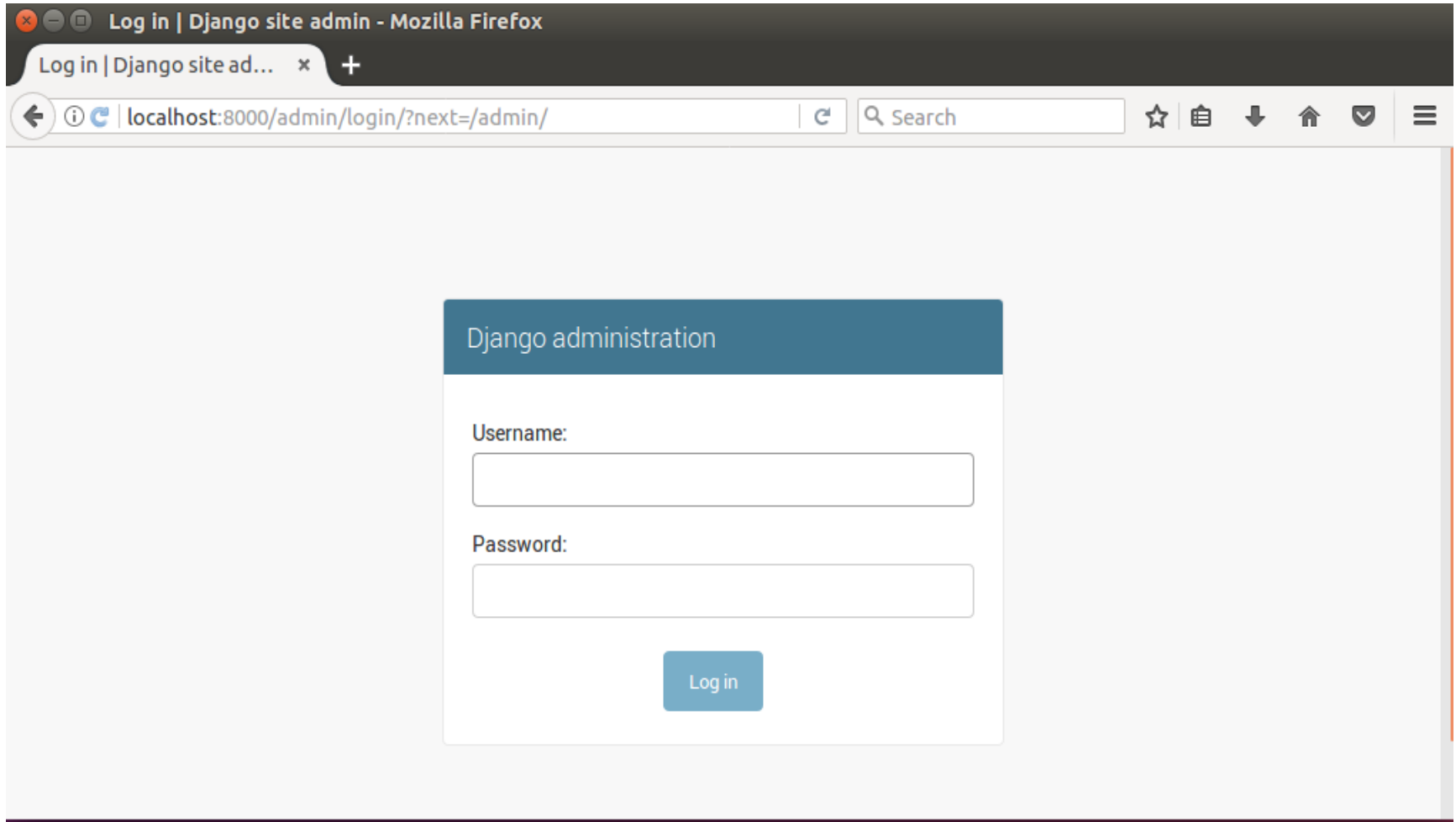


# 6. 활용예제



## 6.2 Apache mode\_wsgi + Django 연동 (10/20)

### 관리자 페이지 확인





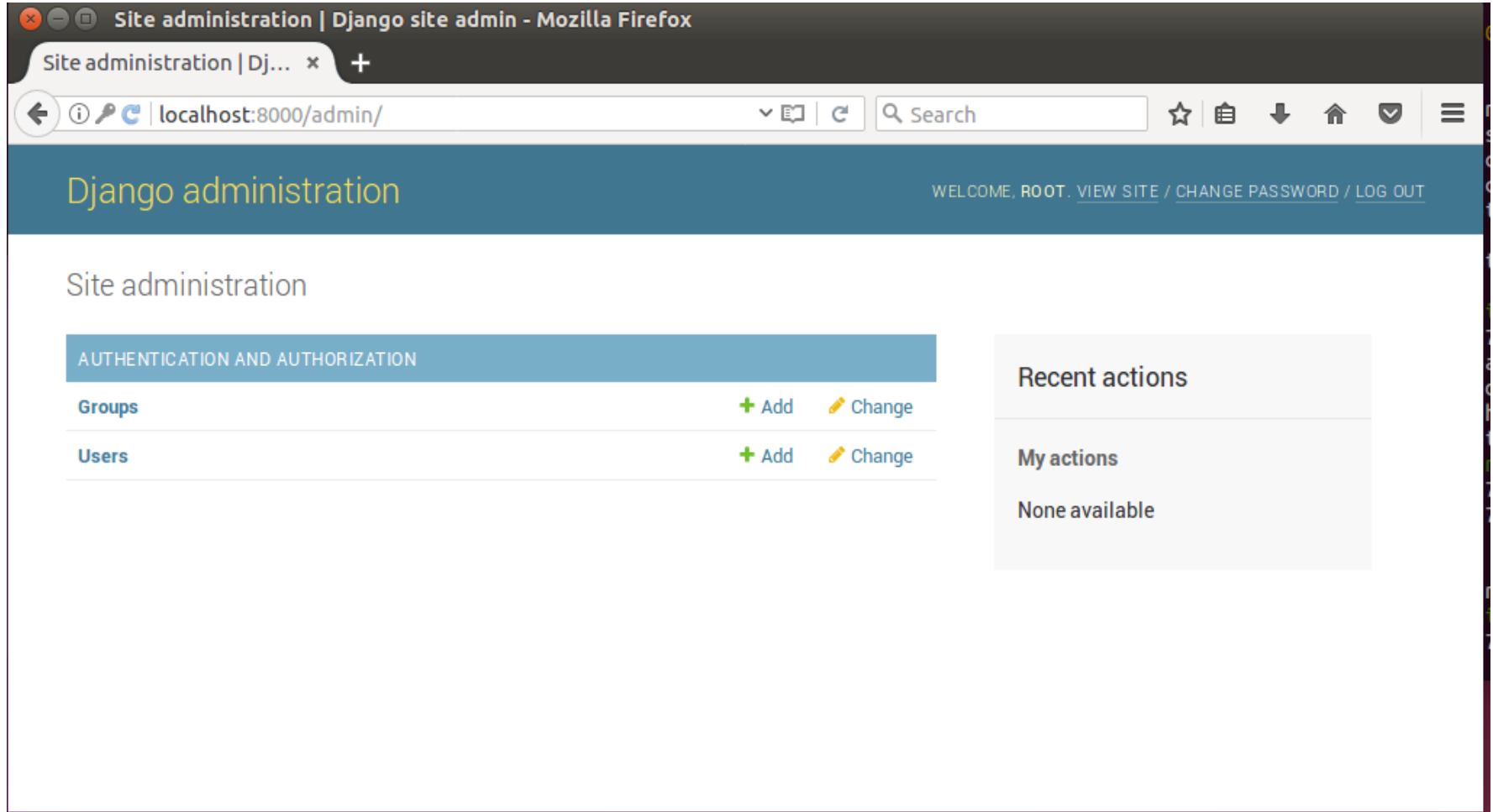
# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동(11/20)

### 관리자 페이지 확인



# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동(12/20)

### 테스트 프로젝트 생성

```
ubuntu@www:~$ cd ~/venv ubuntu@www
:~/venv$ source bin/activate (venv)ubuntu
@www:~/venv$ cd testproject
(venv)ubuntu@www:~/venv/testproject$ python manage.py startapp testapp
```

```
root@master: ~/venv/testproject
root@master:~# cd ~/venv/
root@master:~/venv# source bin/activate
(venv) root@master:~/venv# cd testproject/
(venv) root@master:~/venv/testproject# python manage.py startapp testapp
(venv) root@master:~/venv/testproject# vim testapp/views.py
(venv) root@master:~/venv/testproject#
```



# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동(13/20)

### views.py 작성

```
(venv)ubuntu@www:~/venv/testproject$ vim testapp/views.py
```

```
root@master: ~/venv/testproject
# -*- coding: utf-8 -*-
from __future__ import unicode_literals

from django.shortcuts import render

from django.http import HttpResponse
def main(request):
    html = '<html>\n' \
          '<body>\n' \
          '<div style="width: 100%; font-size: 40px; font-weight: bold; text-align: center;">\n' \
          'Django Test Page\n' \
          '</div>\n' \
          '</body>\n' \
          '</html>\n'
    return HttpResponse(html)

# Create your views here.
```



# 6. 활용예제



## 6.2 Apache mode\_wsgi + Django 연동(14/20)

### testproject/urls.py 수정

```
(venv)ubuntu@www:~/venv/testproject$ vim testproject/urls.py
```

```
root@master: ~/venv/testproject
"""testproject URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/1.11/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  url(r'^$', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  url(r'^$', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.conf.urls import url, include
    2. Add a URL to urlpatterns:  url(r'^blog/', include('blog.urls'))
"""
from django.conf.urls import include, url
from django.contrib import admin

urlpatterns = [
    url(r'^admin/', admin.site.urls),
    url(r'', include('testapp.urls')),
]
```



# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동 (15/20)

testapp/urls.py 생성

```
(venv)ubuntu@www:~/venv/testproject$ vim testapp/urls.py
```

```
root@master: ~/venv/testproject
from django.conf.urls import url
from . import views

urlpatterns = [
    url(r'^testapp/$', views.main, name='main'),
]
~
~
```



# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동 (16/20)

### settings.py 작성

```
(venv)ubuntu@www:~/venv/testproject$ vim testproject/settings.py
# add testapp like follows
INSTALLED_APPS = (
    'django.contrib.admin', 'django
    o.contrib.auth', 'django.contri
    b.contenttypes', 'django.contr
    ib.sessions', 'django.contrib.
    messages', 'django.contrib.sta
    ticfiles',
    'testapp', #추가
)
```

```
# Application definition
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'testapp'
]
```





# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동 (17/20)

### 서버 실행 후, 웹서비스 확인

브라우저를 통해 테스트 페이지가 확인되었다면 Django 연동 준비가 완료된 것이다.

```
(venv)ubuntu@www:~/venv/testproject$ python manage.py runserver 0.0.0.0:8000
```



# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동(18/20)

### Apache mod\_wsgi 설정

Django Test page까지 확인했으면 이제 Apache mod\_wsgi와 Django를 연동해본다.  
연동을 위해 django.conf 파일을 생성하여 설정한다.

```
root@www:~# vi /etc/apache2/conf-available/django.conf
```

```
root@master: ~
# create new
WSGIDaemonProcess testapp python-path=/home/ubuntu/venv/testproject:/home/ubuntu/venv/lib/python2.7/site-packages
WSGIProcessGroup testapp
WSGIScriptAlias /django /home/ubuntu/venv/testproject/testproject/wsgi.py

<Directory /home/ubuntu/venv/testproject>
    Require all granted
</Directory>
~
~
~
~
```





# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동(19/20)

### Apache 재구동

```
root@www:~# a2enconf wsgi root@ww  
w:~# systemctl restart apache2
```

```
root@master: ~  
root@master:~# a2enconf wsgi  
Enabling conf wsgi.  
To activate the new configuration, you need to run:  
  service apache2 reload  
root@master:~# systemctl restart apache2  
root@master:~#
```



# 6. 활용예제

uWSGI



## 6.2 Apache mode\_wsgi + Django 연동 (20/20)

브라우저에서 서비스 확인





**Q** 상업 서비스에 기술지원이 가능합니까?

&

**A** unbit.it 에서 메일 답변 가능합니다.

**Q** 클러스터 환경에서 사용할 수 있습니까?

&

**A** uWAGI 주요 기능 중 하나로 가능합니다. 여러 서버에 바인딩 된 여러 인스턴스와 로드 밸런싱을 사용하는 경우, webserver/proxy aerouter 부하를 분산시킬 수 있습니다.

**Q** 기존 서버를 uWSGI로 바꾸면 더 좋아집니까?

&

**A** 대체적으로 빠르지만 큰 병목현상이 있는 어플리케이션은 성능이 크게 나아지기 힘듭니다.





**Q** uWSGI 환경에서 성능 개선에 가장 중요한 옵션은 무엇입니까?

&

**A** 기본적으로 "almost-all-all"값으로 구성됩니다. 그 외에도 소켓 큐 사이즈나 쓰레딩, 멀티 인터럽트 튜닝 옵션을 고려 할 수 있습니다.

**Q** uWSGI에 적합한 웹 서버 핸들러는 무엇이 있습니까?

&

**A** Nginx, Apache, Lighttpd, Twisted, Tomcat 등이 있습니다.



# 8. 용어정리



용어	설명
<b>WSGI</b> (Web Server Gateway Interface)	웹서버와 웹 애플리케이션의 인터페이스를 위한 파이선 프레임워크
<b>장고(Django)</b>	파이썬으로 만들어진 무료 공개SW 웹 애플리케이션 프레임워크(web application framework)
<b>파이썬(Python)</b>	네덜란드 수학자 귀도 반 로섬에 의해 개발된 프로그램 언어로 문법이 쉽고 간결하고 무료로 사용할 수 있어 널리 쓰임
<b>엔진엑스(NginX)</b>	웹 서버(Web Server) 소프트웨어
<b>Mod_WSGI</b>	Graham Dumpleton이 개발된 Apache HTTP Server 모듈로 python web apps을 위한 WSGI 인터페이스지원
<b>Gunicorn</b>	Green Unicorn 약어로 Unix용 WSGI HTTP Server임. Ruby 프로젝트에서 포크됨
<b>Bjoern</b>	비동기 Cpython의 WSGI serve로 C언어로 작성되어 가볍고 속도가 빠름



# Open Source Software Installation & Application Guide



이 저작물은 크리에이티브 커먼즈 [저작자표시-비영리-동일조건 변경허락 2.0 대한민국 라이선스]에 따라 이용하실 수 있습니다.