즈베리파이에 데비안리눅스 설치기



오늘 라즈베리파이 수령후 간단하게 리눅스를 설치했습니다.

현재 라즈베리파이에 설치할 수 있는  OS는 리눅스입니다.

**1. 설치를 위한 준비물**

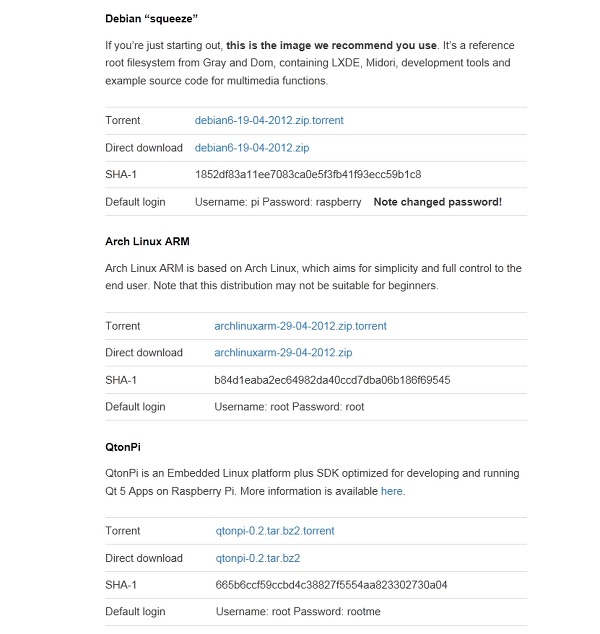


   1) 라즈베리파이 보드  
   2) SD Card (4기가 이상을 추천하네요. 저는 일단 설치만 해보기 위해 2기가 짜리를 이용했습니다.)  
   3) USB 타입의 마우스와 키보드 (저는 2.4GHz 무선키보드를 이용했습니다.)  
   4) HDMI 케이블 과 HDMI 를 지원하는 모니터

       (모니터에 HDMI 가 없을 경우 HDMI To DVI 젠더를 이용해도 됩니다.)  
   5) LAN Cable (공유기와 연결: 최대 100Mbps까지 지원합니다.  
   6) 리눅스 이미지 (SD Card에 설치해서 부팅을 하게 됩니다.)

**2. 라즈베리파이용 리눅스 이미지 다운로드**

   1) 라즈베리파이용 리눅스의 경우 <http://www.raspberrypi.org/downloads> 에서 다운로드 가능합니다.  
       세가지 종류의 리눅스 이미지가 있는데, 데비안 리눅스를 여기서는 데비안 리눅스를 추천합니다.

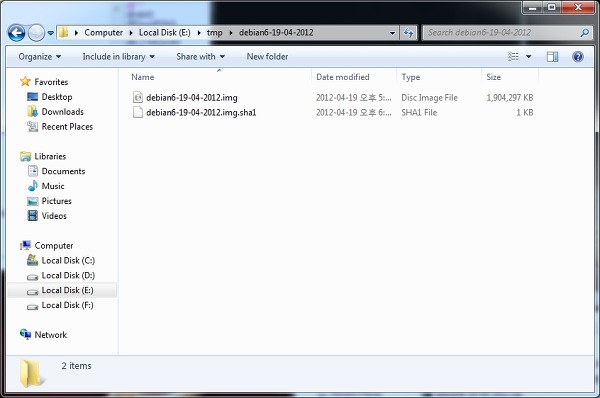
    ※ 다운로드 방법은 직접다운로드. 토렌트를 이용한 다운로드가 있습니다. 데비안의 경우 직접 다운로드보다,

        토렌트를 이용한 다운로드가 훨씬 빠릅니다.

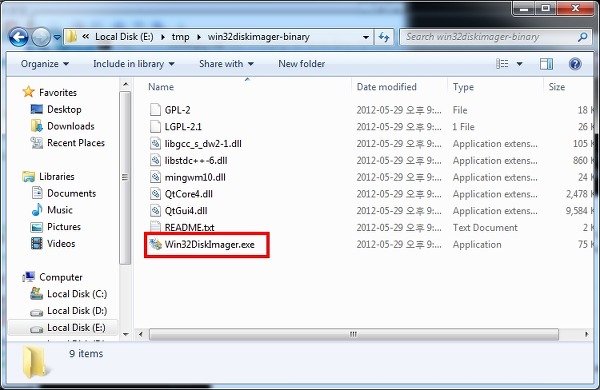
    2) 리눅스 이미지를 SD메모리에 쓰기 위해서는 Wine32 Disk Imager 라는 것이 필요합니다.  
        <http://www.softpedia.com/get/CD-DVD-Tools/Data-CD-DVD-Burning/Win32-Disk-Imager.shtml>

**2. SD Card에 리눅스 이미지 만들기**

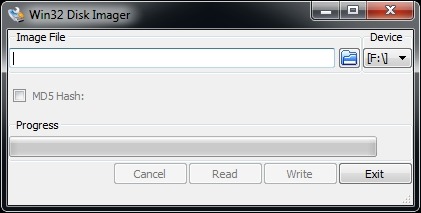
    1) 다운로드 받은 리눅스(여기서는 데이안입니다.)의 압축을 풉니다.



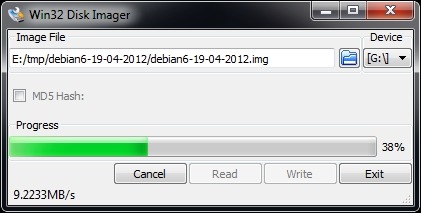
     2) Win32 Disk Imager 의 압축을 푼 후 Win32DiskImager.exe 를 실행합니다.



      3) Image File 부분에는 데비안리눅스 이미지를 Device 에는 SD Card 드라미버를 선택해 주면 됩니다.



     4) 모든 선택이 끝났으면 Write 버튼을 클릭하여 이미지를 SD Card에 기록합니다.



**3. SD Card를 이용해 부팅하기**

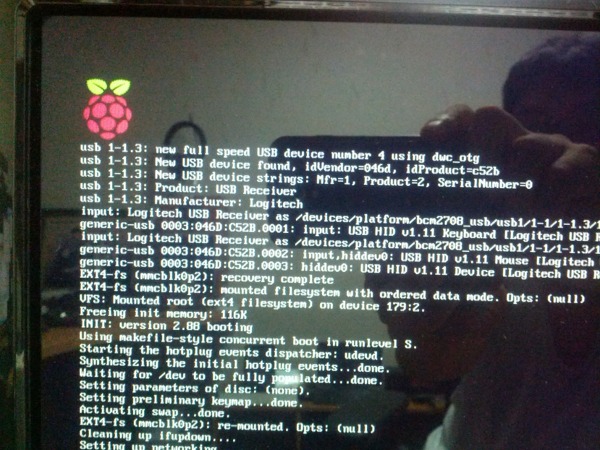
     1) 위의 과정을 통해 SD Card 에 리눅스 이미지를 기록했습니다. 이제 라즈베리파이에 연결해 부팅하면 됩니다.  
  
     2) 전원의 경우 Micro-USB 케이블과 PC 또는 외부전원 어댑터가 있는 USB허브를 이용할 수 있습니다.  
        저는 PC에 연결했습니다.



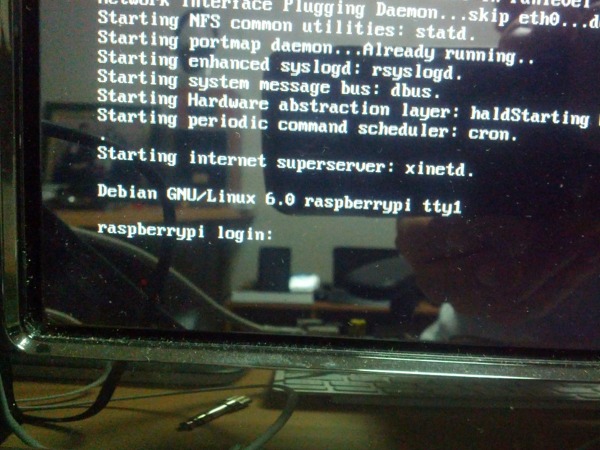
        - 위의 사진을 보면,  
           . 왼쪽상단 : LAN (10/100Mbps)  
           . 왼쪽하단 : USB 포트 \* 2  
                             USB 타입의 키보드와 마우스 연결이 가능합니다.

                             저는 무선키보드/마우스 동글을 연결했습니다.  
           . 상단가운데 : HDMI 포트 (모니터 연결 부분)  
           . 오른쪽상단: Micro-USB 전원어댑터 (PC 또는 USB 허브)  
           . 오른쪽하단: SD Card Adapter (OS)  
           . 하단첫번째: 오디오포트  
           . 하단두번째: 컴포지트 포트

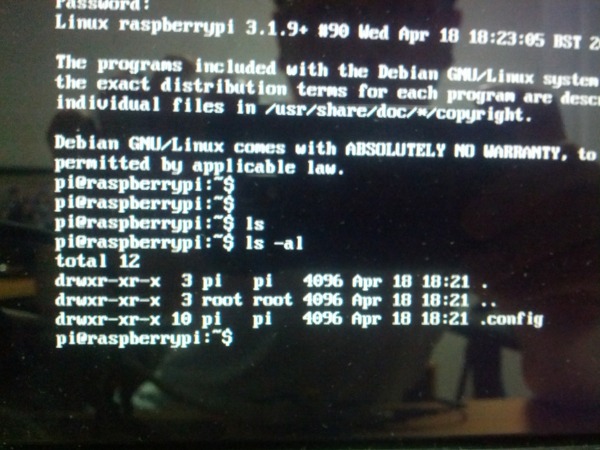
    3) 따로 전원 버튼이 없기 때문에 SD Card 어댑터에 SD Card를 장착한 후, Micro-USB 케이블을 연결하면 바로  
        리눅스로 부팅됩니다.



          - 일반 데탑용 리눅스와는 상단 로고외에는 별다은 차이점을 느낄 수 없습니다.



          - 부팅이 다 끝나고, 로그인 프롬프트가 나타납니다.



           - 로그인 한 후 의 화면입니다.

 이상으로 간단한 설치기를 써봤습니다.  
  
설렁한 로그인 화면으로 끝을 맺습니다. X-Window 등의 UI(UX)를 이용하는 방법등이 있을 것 같은데, 여건이 되면  
사용기를 하면 써보기로 하고, 허접한 설치기  마칩니다.

<http://m.clien.net/cs3/board?bo_table=use&bo_style=view&wr_id=379900>

## [라즈베리파이 N100mini 설정](http://pklazy.wordpress.com/2013/02/23/%eb%9d%bc%ec%a6%88%eb%b2%a0%eb%a6%ac%ed%8c%8c%ec%9d%b4-n100mini-%ec%84%a4%ec%a0%95/)

Posted on [2013/02/23](http://pklazy.wordpress.com/2013/02/23/%eb%9d%bc%ec%a6%88%eb%b2%a0%eb%a6%ac%ed%8c%8c%ec%9d%b4-n100mini-%ec%84%a4%ec%a0%95/) 작성자: [pklazy](http://pklazy.wordpress.com/author/pklazy/)

참고:

* <http://pingbin.com/2012/12/setup-wifi-raspberry-pi/>
* <http://cessnagi.tistory.com/20>

사용한 USB 무선랜은 ipTIME N100mini

(NCUBIC을 주문했지만 잘못 배송이 옴 -ㅅ-)

설치한 OS는 2013-02-09-wheezy-raspbian

lsusb  해보면 잡혀있는것을 알 수 있다. (RTL8188CUS 802.11n WLAN Adapter)

|  |  |
| --- | --- |
| 1  2  3  4  5 | pi@raspberrypi ~ $ lsusb  Bus 001 Device 002: ID 0424:9512 Standard Microsystems Corp.  Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub  Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp.  Bus 001 Device 004: ID 0bda:8176 Realtek Semiconductor Corp. RTL8188CUS 802.11n WLAN Adapter |

드라이버도 이미 설치되어 있는듯.

/etc/network/interfaces 는 따로 수정할 필요가 없고

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | auto lo  iface lo inet loopback    iface eth0 inet dhcp    allow-hotplug wlan0  iface wlan0 inet manual  wpa-roam /etc/wpa\_supplicant/wpa\_supplicant.conf    iface default inet dhcp |

sudo iwlist wlan0 scan 해보면 검색된 SSID 를 볼 수 있다.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | pi@raspberrypi ~ $ sudo iwlist wlan0 scan  wlan0 Scan completed :   Cell 01 - Address: XX:XX:XX:XX:XX:XX   ESSID:"SSID\_NAME"   Protocol:IEEE 802.11bgn   Mode:Master   Frequency:2.462 GHz (Channel 11)   Encryption key:on   Bit Rates:144 Mb/s   Extra:rsn\_ie=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx   IE: IEEE 802.11i/WPA2 Version 1   Group Cipher : CCMP   Pairwise Ciphers (1) : CCMP   Authentication Suites (1) : PSK   Quality=100/100 Signal level=91/100 |

wpa\_passphrase SSID\_NAME SSID\_PASSWORD

로 사용하고자 하는  SSID와 비밀번호를 암호화 한다.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | pi@raspberrypi ~ $ wpa\_passphrase SSID\_NAME SSID\_PASSWORD  network={   ssid="SSID\_NAME"   #psk="SSID\_PASSWORD"   psk=bea4ad3dc2e57e4f3db491da29231f1536151b221ee3ad76e5a4c25d739a267b  } |

이 내용을 /etc/wpa\_supplicant/wpa\_supplicant.conf 에 추가하고 몇가지 정보를 추가로 더 넣는다.

내 경우에는 AirPort Express의 개인용 WPA2를 사용하고 있어서 거기에 맞게 proto, pairwise, group를 수정했다.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | ctrl\_interface=DIR=/var/run/wpa\_supplicant GROUP=netdev  update\_config=1    network={   ssid="SSID\_NAME"   proto=WPA2   key\_mgmt=WPA-PSK   pairwise=CCMP   group=CCMP   psk=bea4ad3dc2e57e4f3db491da29231f1536151b221ee3ad76e5a4c25d739a267b  } |

수정 후 재부팅하면 무선랜에서 파란 불이 들어여면서 동작하는것을 볼수 있다.

공유기의 IP정보를 보면 raspberrypi로 할당된 IP를 확인할 수 있다.

[About these ads](http://en.wordpress.com/about-these-ads/)

이 글은 [Raspberry Pi](http://pklazy.wordpress.com/category/raspberry-pi/) 카테고리에 분류되었고 [라즈베리파이](http://pklazy.wordpress.com/tag/%eb%9d%bc%ec%a6%88%eb%b2%a0%eb%a6%ac%ed%8c%8c%ec%9d%b4/), [N100mini](http://pklazy.wordpress.com/tag/n100mini/), [Raspberry Pi](http://pklazy.wordpress.com/tag/raspberry-pi/), [USB 무선랜](http://pklazy.wordpress.com/tag/usb-%eb%ac%b4%ec%84%a0%eb%9e%9c/) 태그

<http://pklazy.wordpress.com/2013/02/23/%EB%9D%BC%EC%A6%88%EB%B2%A0%EB%A6%AC%ED%8C%8C%EC%9D%B4-n100mini-%EC%84%A4%EC%A0%95/>

# [**라즈베리파이 무선랜설정**](http://cessnagi.tistory.com/entry/%EB%9D%BC%EC%A6%88%EB%B2%A0%EB%A6%AC%ED%8C%8C%EC%9D%B4-%EB%AC%B4%EC%84%A0%EB%9E%9C%EC%84%A4%EC%A0%95)

sudo apt-get install wireless-tools

설치합니다

sudo nano /etc/network/interpaces

아래 내용을 추가

allow-hotplug wlan0

iface wlan0 inet dhcp

wpa-roam /etc/wpa\_supplicant/wpa\_supplicant.conf

iface default inet dhcp

sudo iwlist scan

ssid 정보 확인

비밀번호 암호화

wpa\_passphrase ESSID\_NAME  passwd

ESSID\_NAME에 ssid 넣고..

passwd 부분에 ssid에 해당하는 비밀번호 넣기

이 내용을 복사해서..

sudo nano /etc/wpa\_supplicant/wpa\_supplicant.conf

아래 내용에 추가.

나의경우는 WPA2-PSK / AES 셋팅

network={

ssid="공유기ssid"

key\_mgmt=WPA-PSK

proto=WPA2

pairwise=CCMP

group=CCMP

psk=170d0b9d1536fdd5884233b70b381abc3bd8216d3667a10987e7da4ff14da74b

}

무선랜 접속

sudo ifconfig wlan0 inet down

sudo ifconfig wlan0 inet up

sudo iwconfig wlan0 essid 공유기ssid channel 1

-----------------------------------------------------

Raspbmc의 경우..

sudo apt-get update

sudo apt-get upgrade

해주면 프로그램란에...

wireless manager가 생긴다

거기서 ssid 추가해주면 된다.

안녕하십니까.. 산딸기 마을에 이장 나무꾼 입니다.

오늘은 라즈베리파이 서버군 중 파일서버의 한 종류인 FTP Server 를 설치 및 환경설정을 해 보도록 하겠습니다.

<http://www.rasplay.org/?p=3051>

# **라즈베리파이 FTP Server 구축하기**

## ****0. 파일서버의 정의****

위키디피아 에서는 파일서버에 대한 정의를 아래와 같이 표현하고 있습니다.

컴퓨팅 환경에서 **파일 서버(file server)**는 워크스테이션이 접근할 수 있는 **컴퓨터 파일(문서, 사운드 파일, 사진, 영화, 그림, 데이터베이스 등)을 공유하고 있는 기억 공간의 위치를 제공하는 것이 주 목적을 가진 네트워크에 추가된 컴퓨터**를 말한다.”서버”(server)라는 용어는 클라이언트 서버 구조로 된 기기의 역할을 강조하며 여기서 클라이언트는 이 기억 공간을 사용하는 워크스테이션을 말한다.  
파일 서버는 일반적으로 어떠한 계산도 수행하지 않으며 클라이언트의 중간에 어떠한 프로그램도 실행하지 않는다. 데이터의 빠른 저장과 워크스테이션이 중대한 계산을 제공하는 데이터 수신이 파일 서버의 주된 설계 목적이다.

간단히 표현하자면 내부 또는 외부에서 파일서버가 구축되어 있는 네트워크 공간으로 접속을 하여 파일을 전달 받을 수 있는 컴퓨터 구성환경 이다.

그럼, 이제 본격적으로 사용 중인 R-Pi에 FTP Server 를 설치 해보도록 하겠습니다.

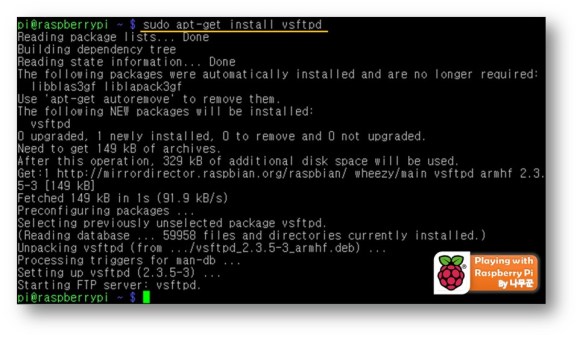
## ****1. 준비 사항.****

o Raspberry Pi 본체  
o Pre-install Raspbian SD 메모리카드

## ****2. vsftpd 설치하기****

apt-get install 설치실행 명령어로 FTP Server 프로그램 중 vsftpd 을 설치합니다.

sudo apt-get install vsftpd

[](http://i1.wp.com/www.rasplay.org/wp-content/uploads/vftpd_1.jpg)그림 2-1)

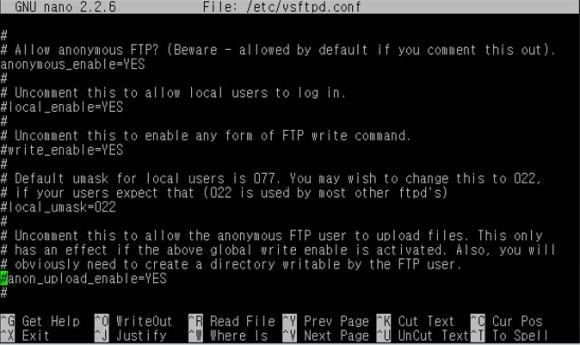
## ****3. vsftpd.conf 파일 설정하기****

2번에서 처럼 vsftpd 프로그램이 정상적으로 설치가 되었다면, 외부 또는 내부에서 FTP접속이 가능하도록 /etc/vsftpd.conf 파일을 열어 일부 내용을 수정을 합니다.

sudo nano /etc/vsftpd.conf

#### [vftpd_2](http://i0.wp.com/www.rasplay.org/wp-content/uploads/vftpd_2.jpg)

**그림 3-1)**

[](http://i1.wp.com/www.rasplay.org/wp-content/uploads/vftpd_3.jpg)

**그림 3-2)**

# anonymous 사용자의 접속 허용 여부, 즉 anonymous ftp (default = YES)

# 공개된 형태의 FTP 서버로 운영할 것이 아니라면 NO로 한다.

anonymous\_enable=NO

# 로컬 계정 사용자의 접속 허용 여부 (default = NO)

local\_enable=YES

# write 명령어 허용 여부 (defualt = NO)

write\_enable=YES

# 로컬 계정 사용자용 umask (default = 077)

local\_umask=022

# anonymous 사용자가 파일을 업로드 할 수 있는지 여부 (default = NO)

# anon\_upload\_enable=YES

# anonymous 사용자의 디렉토리 생성 허용 여부 (default = NO)

# anon\_mkdir\_write\_enable=YES

# 파일 전송 로그를 남길 것인지 여부 (default = YES)

xferlog\_enable=YES

# xferlog 표준 포맷으로 로그를 남길지 여부 (기본 설정파일은 YES)

# 아래에서 NO로 설정했을 때를 설명함

xferlog\_std\_format=YES

# 파일 전송 로그 파일명

xferlog\_file=/var/log/vsftpd.log

# FTP 서버 접속할 때 로긴 메시지 (default = vsFTPd 버전번호)

# 한글 사용 가능

# ftpd\_banner=Welcome to blah FTP service.

# 사용자의 홈디렉토리를 벗어나지 못하도록 제한하기 위한 설정 (default=NO)

# 제한이 필요할 경우 YES로 바꾼 후 제한할 사용자 ID를 chroot\_list\_file= 에 설정한 파일에

# 지정한다.

# chroot\_local\_user= 설정과 관련이 있으니 '3. 문제 해결'을 꼭 읽어보라.

#

# chroot\_list\_enable=YES

# chroot\_list\_file=/etc/vsftpd.chroot\_list

# ----------------------------------------------------------------

# 기본 설정 파일에는 없는 설정값으로 필요한 설정만 추가한다.

# ※ 중요한 설정은 굵은 글씨로 표시

# ----------------------------------------------------------------

# PAM 파일명을 지정 (설치할 때 /etc/pam.d/vsftpd명으로 복사함)

pam\_service\_name=vsftpd

# wtmp에 로그 남기기 (YES로 해야만 last 명령어로 접속 여부 확인 가능)

session\_support=YES

# 사용자가 자신의 home directory를 벗어나지 못하도록 설정

chroot\_local\_user=YES

# 새로운 디렉토리에 들어갔을 때 뿌려줄 환경 메시지를 저장한 파일명

# message\_file=.message

# xferlog 형식으로 log를 남기려면 (위에서 이미 YES로 했음)

# xferlog\_std\_format=NO

#

# - xferlog 표준 포맷은 로긴, 디렉토리 생성등의 로그를 남기지 않음

# 그러나 vsftpd 스타일 로그는 이를 포함한 보다 상세한 로그를 남김

# - vsftpd 스타일 로그 예

#

# Sun Jul 12 01:38:32 2003 [pid 31200] CONNECT: Client "127.0.0.1"

# Sun Jul 12 01:38:34 2003 [pid 31199] [truefeel] FAIL LOGIN: Client "127.0.0.1"

# Sun Jul 12 01:38:38 2003 [pid 31199] [truefeel] OK LOGIN: Client "127.0.0.1"

# Sun Jul 12 01:38:41 2003 [pid 31201] [truefeel] OK MKDIR: Client "127.0.0.1", "/mp3"

# Sun Jul 12 01:39:06 2003 [pid 31201] [truefeel] OK UPLOAD: Client "127.0.0.1", "/델리

# 스파이스 5집 - [04]키치죠지의 검은 고양이.mp3", 6855473 bytes, 3857.39Kbyte/sec

# 전송속도 제한 (0은 제한없음, 단위는 초당 bytes)

anon\_max\_rate=0

local\_max\_rate=0

trans\_chunk\_size=0

# 최대 접속 설정 (단 xinetd를 통하지 않고 standalone으로 동작할 때만 사용 가능)

# standalone을 위해서는 listen=YES 추가하고 별도로 vsftpd를 띄워야 함

#

# max\_clients=최대 접속자 수, max\_per\_ip=IP당 접속 수

# max\_clients=100

# max\_per\_ip=3

# Standalone 으로 운영할 때 listen=YES. 포트 변경을 원할 경우 listen\_port 설정

# 디폴트 포트는 21번 포트이다.

# listen=YES

# listen\_port=21

## ****[ 참고사이트]****

o [*https://security.appspot.com/vsftpd.html*](https://security.appspot.com/vsftpd.html)o [*http://jobdahan.net/server\_linux/651905*](http://jobdahan.net/server_linux/651905)

다음 글을 추천합니다.

# **라즈베리파이 GPIO 활용강좌 : 03 WIRINGPI2-PYTHON 설치하기**

by [나무꾼](http://www.rasplay.org/?author=1)



## ****0. WiringPi 이란 ?****

[](http://i1.wp.com/www.rasplay.org/wp-content/uploads/test.png)  
o WiringPi : 아두이노의 대부분의 기능을 라즈베리파이에 맞추어 재구성.  
o WiringPi2 : WiringPi2는 IO 확장장치를 관리하기 위한 일부 기능이 추가.

## ****1. WiringPi2-python 설치 준비****

라즈베리파이에 WiringPi2-Python 버전을 설치하기 위해선 두 가지 패키지(python-dev

python-setuptools)와 함께 WiringPi-Python이 사전 설치가 되어 있어야만 한다.

그럼 우선 Python 개발 패키지와 셋업 툴 그리고 WiringPi-Python을 먼저 설치해보도록 하자

pi@rasplay ~ $ sudo apt-get install -y python-dev python-setuptools

Reading package lists… Done

Building dependency tree

Reading state information… Done

The following extra packages will be installed:

libexpat1-dev libssl-dev libssl-doc python-pkg-resources python2.7-dev

Suggested packages:

python-distribute python-distribute-doc

The following NEW packages will be installed:

libexpat1-dev libssl-dev libssl-doc python-dev python-pkg-resources

python-setuptools python2.7-dev

0 upgraded, 7 newly installed, 0 to remove and 0 not upgraded.

Need to get 32.1 MB of archives.

………..[continue]……….

Setting up python2.7-dev (2.7.3-6) …

Setting up python-dev (2.7.3-4) …

Setting up python-pkg-resources (0.6.24-1) …

Setting up python-setuptools (0.6.24-1) …

## ****2. wiringPi 다운로드****

1번 항목을 통해 기본 패키지가 설치되었다 이제 WiringPi-python 설치를 위해  [WiringPi git](https://github.com/Gadgetoid)

[hub](https://github.com/Gadgetoid) 를 통해 Python 버젼을 다운로드 받자.

pi@rasplay ~ $ git clone git://git.drogon.net/wiringPi

|  |
| --- |
| Cloning into ‘wiringPi’… remote: Counting objects: 489, done. remote: Compressing objects: 100% (431/431), done. remote: Total 489 (delta 341), reused 95 (delta 58) Receiving objects: 100% (489/489), 209.46 KiB | 159 KiB/s, done. Resolving deltas: 100% (341/341), done. |

## ****3. WiringPi module 업데이트 하기****

라즈비안에서도 ‘update’ 명령어를 이용해 소스 패키지 업데이트를 통해 패키지 업데이트를

하듯이 WiringPi도 github 패키지소스 업데이트를 통한 모듈업데이트 기능을 이용할 수 있다.

pi@rasplay ~ $ cd wiringPi

pi@rasplay ~/wiringPi $ git pull origin

Already up-to-date.

## 4. WiringPi 설치하기

3번 항목을 통해 업데이트 된 모듈을 라즈베리파이에서 Wiring 이 구동 되도록 컴파일을 진행하자.

pi@rasplay ~/wiringPi $ ./build

|  |
| --- |
| wiringPi Build script ===================== WiringPi Library …… GPIO Utility [Compile] gpio.c [Compile] extensions.c [Link] [Install] All Done. NOTE: This is wiringPi v2, and if you need to use the lcd, Piface, Gertboard, MaxDetext, etc. routines then you must change your compile scripts to add -lwiringPiDev pi@rasplay ~/wiringPi $ |

## ****5. WiringPi-Python GPIO 테스트****

WiringPi-Python이 정상 설치가 이루어졌는지 gpio 명령어를 통해 간단한 확인 해 보도록 하자.

pi@rasplay ~/WiringPi-Python $ gpio -v

|  |
| --- |
| gpio version: 2.07 Copyright (c) 2012-2013 Gordon Henderson This is free software with ABSOLUTELY NO WARRANTY. For details type: gpio -warranty pi@rasplay ~/WiringPi-Python $ gpio readall +———-+-Rev2-+——+——–+——+——-+ | wiringPi | GPIO | Phys | Name   | Mode | Value | +———-+——+——+——–+——+——-+ |      0   |  17  |  11  | GPIO 0 | IN   | Low   | |      1   |  18  |  12  | GPIO 1 | IN   | Low   | |      2   |  27  |  13  | GPIO 2 | IN   | Low   | |      3   |  22  |  15  | GPIO 3 | IN   | Low   | |      4   |  23  |  16  | GPIO 4 | IN   | Low   | |      5   |  24  |  18  | GPIO 5 | IN   | Low   | |      6   |  25  |  22  | GPIO 6 | IN   | Low   | |      7   |   4  |   7  | GPIO 7  | IN   | Low   | |      8   |   2  |   3  | SDA     | IN   | High  | |      9   |   3  |   5  | SCL      | IN   | High  | |     10   |   8  |  24  | CE0     | IN   | Low   | |     11   |   7  |  26  | CE1     | IN   | Low   | |     12   |  10  |  19  | MOSI   | IN   | Low   | |     13   |   9  |  21  | MISO   | IN   | Low   | |     14   |  11  |  23  | SCLK   | IN   | Low   | |     15   |  14  |   8  | TxD     | ALT0 | Low   | |     16   |  15  |  10  | RxD    | ALT0 | Low   | |     17   |  28  |   3  | GPIO 8 | IN   | Low   | |     18   |  29  |   4  | GPIO 9 | IN   | Low   | |     19   |  30  |   5  | GPIO10 | IN   | Low   | |     20   |  31  |   6  | GPIO11 | IN   | Low   | +———-+——+——+——–+——+——-+ pi@rasplay ~/wiringPi $ |

## ****6. WiringPi2-Python 다운로드****

드디어 WiringPi2-Python설치에 끝자락으로 가고 있네요.

WiringPi2-Python 설치를 위해 github를 통해 source를 다운로드 받자.

pi@rasplay ~ $ cd ~

pi@rasplay ~ $ git clone https://github.com/Gadgetoid/WiringPi2-Python.git

|  |
| --- |
| Cloning into ‘WiringPi2-Python’… remote: Counting objects: 573, done. remote: Compressing objects: 100% (210/210), done. remote: Total 573 (delta 378), reused 550 (delta 355) Receiving objects: 100% (573/573), 285.82 KiB | 146 KiB/s, done. Resolving deltas: 100% (378/378), done. pi@rasplay ~ $ |

## ****7. WiringPi2-Python 설치하기****

pi@rasplay ~ $ cd WiringPi2-Python

pi@rasplay ~ ~/WiringPi2-Python $ sudo python setup.py install

|  |
| --- |
| running install Checking .pth file support in /usr/local/lib/python2.7/dist-packages/ /usr/bin/python -E -c pass TEST PASSED: /usr/local/lib/python2.7/dist-packages/ appears to support .pth files running bdist\_egg running egg\_info creating wiringpi2.egg-info writing wiringpi2.egg-info/PKG-INFO running install\_lib running build\_py creating build running build\_ext building ‘\_wiringpi2′ extension creating build/temp.linux-armv6l-2.7 creating build/temp.linux-armv6l-2.7/WiringPi gcc -pthread -fno-strict-aliasing -DNDEBUG -g -fwrapv -O2 -Wall -Wstrict-prototypes -fPIC -I/usr/include/python2.7 -c WiringPi/wiringPi/wiringPi.c -o build/temp.linux-armv6l-2.7/WiringPi/wiringPi/wiringPi.o …..[continue]….. Installed /usr/local/lib/python2.7/dist-packages/wiringpi2-1.0.10-py2.7-linux-armv6l.egg Processing dependencies for wiringpi2==1.0.10 Finished processing dependencies for wiringpi2==1.0.1.1 |

## [참고사이트]

*o ~~http://rasplay.tistory.com/14~~o*[*https://github.com/wiringPi*](https://github.com/wiringPi)*o*[*https://github.com/WiringPi/WiringPi2-Python*](https://github.com/WiringPi/WiringPi2-Python)

## Simple GPIO Control using Pi4J.

## <http://pi4j.com/example/control.html>

The following example demonstrates the simple control of a GPIO pin on the Raspberry Pi.

### Source Code

The source code for this example is included in the github repository:  
<https://github.com/Pi4J/pi4j/tree/master/pi4j-example/src/main/java/ControlGpioExample.java>

/\*

\* #%L

\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* ORGANIZATION : Pi4J

\* PROJECT : Pi4J :: Java Examples

\* FILENAME : ControlGpioExample.java

\*

\* This file is part of the Pi4J project. More information about

\* this project can be found here: http://www.pi4j.com/

\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* %%

\* Copyright (C) 2012 - 2013 Pi4J

\* %%

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* http://www.apache.org/licenses/LICENSE-2.0

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\* #L%

\*/

import com.pi4j.io.gpio.GpioController;

import com.pi4j.io.gpio.GpioFactory;

import com.pi4j.io.gpio.GpioPinDigitalOutput;

import com.pi4j.io.gpio.PinState;

import com.pi4j.io.gpio.RaspiPin;

/\*\*

\* This example code demonstrates how to perform simple state

\* control of a GPIO pin on the Raspberry Pi.

\*

\* @author Robert Savage

\*/

public class ControlGpioExample {

public static void main(String[] args) throws InterruptedException {

System.out.println("<--Pi4J--> GPIO Control Example ... started.");

// create gpio controller

final GpioController gpio = GpioFactory.getInstance();

// provision gpio pin #01 as an output pin and turn on

final GpioPinDigitalOutput pin = gpio.provisionDigitalOutputPin(RaspiPin.GPIO\_01, "MyLED", PinState.HIGH);

System.out.println("--> GPIO state should be: ON");

Thread.sleep(5000);

// turn off gpio pin #01

pin.low();

System.out.println("--> GPIO state should be: OFF");

Thread.sleep(5000);

// toggle the current state of gpio pin #01 (should turn on)

pin.toggle();

System.out.println("--> GPIO state should be: ON");

Thread.sleep(5000);

// toggle the current state of gpio pin #01 (should turn off)

pin.toggle();

System.out.println("--> GPIO state should be: OFF");

Thread.sleep(5000);

// turn on gpio pin #01 for 1 second and then off

System.out.println("--> GPIO state should be: ON for only 1 second");

pin.pulse(1000, true); // set second argument to 'true' use a blocking call

// stop all GPIO activity/threads by shutting down the GPIO controller

// (this method will forcefully shutdown all GPIO monitoring threads and scheduled tasks)

gpio.shutdown();

}

}

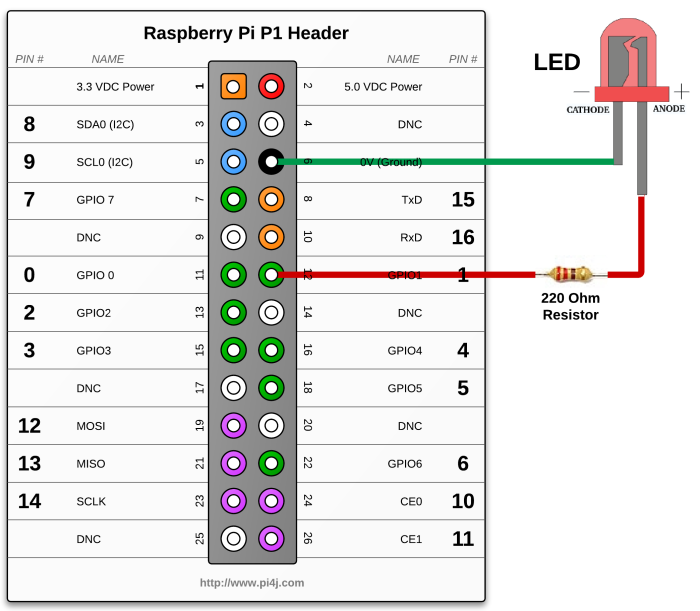
### JavaDoc

The following JavaDoc links are the primary interfaces used to control the Pi's GPIO state:

* [com.pi4j.io.gpio.GpioFactory](http://pi4j.com/apidocs/index.html?com/pi4j/io/gpio/GpioFactory.html)
* [com.pi4j.io.gpio.Gpio](http://pi4j.com/apidocs/index.html?com/pi4j/io/gpio/Gpio.html)
* [com.pi4j.io.gpio.GpioPin](http://pi4j.com/apidocs/index.html?com/pi4j/io/gpio/GpioPin.html)
* [com.pi4j.io.gpio.PinState](http://pi4j.com/apidocs/index.html?com/pi4j/io/gpio/PinState.html)

### Wiring Diagram

The following circuit can be used in conjunction with this sample code.



[(click here for hi-resolution image)](http://pi4j.com/images/gpio-control-example-large.png)

### Navigate

If you have not already downloaded and installed the Pi4J library on the RaspberryPi, then view this page for instructions on where to download and how to install Pi4J:   
[Download & Install Pi4J](http://pi4j.com/install.html)

First, locate the *ControlGpioExample.java* source file in the samples folder of the Pi4J installation on the RaspberryPi.   
You can use the following command on the Pi's console or SSH terminal to navigate to this path:   
cd /opt/pi4j/examples

### Compile

Next, use the following command to compile this example program:   
javac -classpath .:classes:/opt/pi4j/lib/'\*' -d . ControlGpioExample.java

### Execute

The following command will run this example program:   
sudo java -classpath .:classes:/opt/pi4j/lib/'\*' ControlGpioExample

### Output

You should see the attached LED perform as follows:

http://pi4j.com/images/pinon.png

Turn ON for 5 seconds

http://pi4j.com/images/pinoff.png

Turn OFF for 5 seconds

http://pi4j.com/images/pinon.png

Turn ON for 5 seconds

http://pi4j.com/images/pinoff.png

Turn OFF for 5 seconds

http://pi4j.com/images/pinon.png

Turn ON for 1 second

http://pi4j.com/images/pinoff.png

Turn OFF