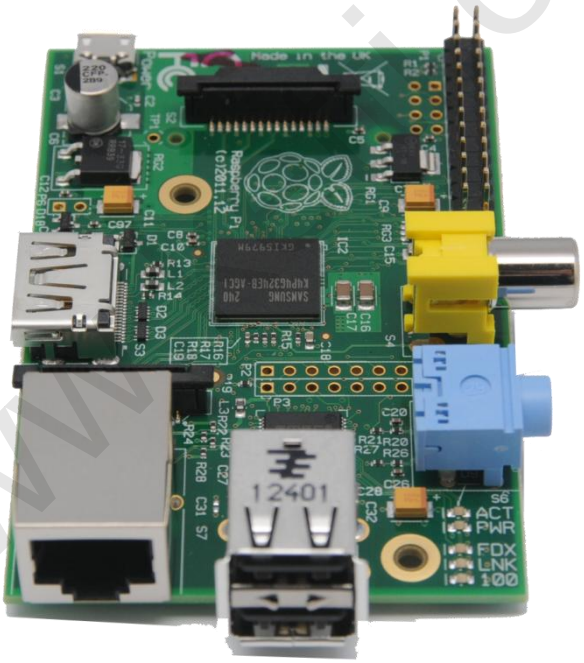
라즈베리파이를 활용한 오픈 소스 R&D 플랫폼

(1,2장)



www.ketti.or.kr

한국전자통신교육원

본 문서는 자유롭게 배포 가능합니다.
다만 상업적 목적 및 문서 수정 배포시
한국전자통신교육원(kks@ketti.or.kr)로 문의 부탁드립니다.

목차

목차 1

Chapter 1. 라즈베리파이 소개 5

1. 프로그래밍의 재미 5

2. 라즈베리파이의 역사 5

3. 라즈베리파이란? 6

4. 라즈베리파이로 무엇을 할 수 있나? 7

Chapter 2. 라즈베리파이와의 첫 만남 9

1. ARM 대 x86 9

2. 리눅스 대 윈도우 10

3. 라즈베리파이에 포팅 된 운영체제 10

4. 주변기기 연결 11

5. SD카드에 운영체제 이미지 써넣기 14

6. USB저장 장치 연결 15

7. 네트웍 연결 15

8. 전원 연결 16

Chapter 3. 리눅스 시스템 관리 17

1. 리눅스 개요 17

2. 리눅스 기초 17

3. 데비안(Debian) 소개 18

4. 외부 저장 장치 사용 21

5. 사용자 계정 만들기 22

6. 파일 시스템 구성 23

www.ketti.or.kr

7. 소프트웨어 설치 및 제거 25

Chapter 4. 문제 해결 29

1. 키보드 및 마우스 진단 29

2. 전원 진단 29

3. 디스플레이 진단 31

4. 부팅 진단 31

5. 네트웍 진단 32

6. 비상용 커널 34

Chapter 5. 네트웍 설정 36

1. 유선 네트워킹 36

2. 무선 네트워킹 39

Chapter 6. 파티션 관리 49

1. 새 파티션 작성 49

2. 파티션 크기 변경 52

3. 수작업으로 파티션 크기 변경 53

4. 더 큰 SD카드로 옮기기 57

Chapter 7. 하드웨어 구성 설정 60

1. 하드웨어 설정 – config.txt 60

2. 디스플레이 설정 변경 61

3. 부팅 옵션들 64

4. 라즈베리파이 오버 클럭킹 64

5. 메모리 파티셔닝 – start.elf 67

6. 소프트웨어 설정들 – cmdline.txt 67

Chapter 8. 홈씨어터로 활용 69

www.ketti.or.kr

1. Rasbmc를 이용한 전용 HTPC 구축 69

2. 인터넷 미디어 스트리밍 70

3. 로컬 네트웍 미디어 스트리밍 72

4. Rasbmc 설정 변경 74

Chapter 9. 업무용으로 활용 76

1. 클라우드 기반 앱 사용 76

2. 오픈오피스 사용 77

3. Gimp를 이용한 이미지 편집 78

Chapter 10. 웹 서버로 활용 80

1. LAMP 스택 설치 80

2. WordPress 설치 83

Chapter 11. 스크래치로 프로그래밍 88

1. 예제 1: Hello World 88

2. 예제 2: 애니메이션 및 사운드 92

3. 예제 3: 간단한 게임 94

4. 로보틱스 및 센서들 99

5. 더 읽어보기 100

Chapter 12. 파이썬 프로그래밍 101

1. 파이썬 소개 101

2. 예제 1: Hello World 101

3. 파이썬 프로그램 실행 가능하게 만들기 105

4. 예제 2: 주석, 입력, 변수 및 반복문 106

5. 예제 3: pygame을 이용하여 게임 만들기 109

6. 파이썬과 네트워킹 118

www.ketti.or.kr

7. 더 읽어보기 123

Chapter 13. 하드웨어 해킹 125

1. 전자 부품 125

2. 저항 값 읽기 127

3. GPIO 포트 128

4. 파이썬에서 GPIO 사용하기 131

5. GPIO 출력: LED 밝히기 133

6. GPIO 입력: 버튼 읽기 137

7. 브레드보드 벗어나기 140

8. 납땜에 대한 간단한 가이드 143

참조 A: Python Recipes 148

1. Raspberry Snake (Chapter 11, Example 3) 148

2. IRC User List (Chapter 11, Example 4) 151

3. GPIO Input and Output (Chapter 12) 154

참조 B: HDMI 디스플레이 모드 목록 155

참조 C: 참고 문헌 162

1. 참고 문헌 162

2. 참고 사이트 162

www.ketti.or.kr

Chapter 1. 라즈베리파이 소개

작년에 불꽃놀이 파티에서 만난 누군가가 나에게 "요즘 아이들은 다들 엄마 배 속에서부터 컴퓨터를 배워 가지고 나온 것 같아요", "난 당신이 왜 이런걸 만들고 있는지 이유를 모르겠어요", "우리 아이들은 다 나 보다 PC를 잘 다뤄요!" 같은 말들을 하는 것을 들은 적이 있다. 내가 그에게 자녀들이 프로그래밍도 할 줄 아는지를 물었더니 그는 "왜 우리 아이들이 프로그래밍을 할 줄 알아야 하죠? 아이들이 원하는 건 이미 컴퓨터에 다 있지 않나요?"라고 답을 하는 것이었다.

물론 그렇기는 하다. 하지만 컴퓨터가 스스로를 프로그래밍 할 수 있는 것은 아니지 않은가? 우리는 기술을 계속 발전시켜나갈 수 많은 엔지니어들이 필요하다. 우리는 은퇴할 때가 되어 업계를 떠나가는 선배 엔지니어들의 뒤를 이어, 그들이 하던 일을 물려받을 젊은이들이 필요하다. 뿐만 아니라 프로그래밍 학습을 통해 길러지는, 창조적인 생각과 복잡한 작업들을 구조화할 수 있는 '능력'은 컴퓨터 엔지니어뿐만이 아니라 컴퓨터와는 관계가 없는 역사가나, 디자이너, 변호사, 화학자를 비롯한 어느 누구에게나 큰 도움이 될 수 있는 것이다.

1. 프로그래밍의 재미

- 범위가 방대하다.
- 보상이 있다.
- 창조적 재미가 있다.

2. 라즈베리파이의 역사

2006년	. 아트멜의 ATmega644 마이크로 컨트롤러를 기반으로 초기 개념이 만들어짐 . 재단 이사인 에반 업톰은 교사, 대학생, 아이들에게 컴퓨터에 대한 영감을 주고자 노력하는 컴퓨터 광들을 모아 재단을 설립
2011년 8월	. 15번째 알파 보드 제작 . 기능적으로는 계획했던 B 모델과 같았지만 커다란 물리적 디버그 헤더를 장착 . 데비안 LXDE 데스크톱과 1080p환경에서 퀘이크3를 구동 . HDMI포트를 통해 1080p MPEG-4 동영상 재생
2011년 10월	. 라즈베리파이에 포팅된 RISC OS의 개발 버전 공개 시연
2011년 12월	. 25개의 B모델 베타버전 조립 시험 . 리눅스 구동 . 1080P 영화 트레일러와 더불어 Rightware 사무라이 OpenGL ES를 실행
2012년 첫 주	. 10개의 라즈베리파이 보드가 이베이 경매에 등록 . 10개의 라즈베리파이 보드는 16,000파운드, 일련번호 1번인 라즈베리파이 보드는

www.ketti.or.kr

	3,500 파운드에 판매 . 익명의 구매자는 영국 셔퍼크 컴퓨터 역사 센터에 기부
2012년 2월 29일	. UTC 06:00 시에 판매 시작 . 모델A의 램이 128MB에서 256MB로 변경 공지 . 비공식 확인 2백만건의 선주문 폭주
2012년 4월 16일	. 최초 구매자들에게 라즈베리파이 배송
2012년 5월 22일	. 2만개 이상의 라즈베리파이 선적

3. 라즈베리파이란?

라즈베리파이(Raspberry Pi)는 영국의 라즈베리파이 재단이 기초 컴퓨터 과학 교육 증진 및 취미를 위해 만든 ARM 기반의 싱글 보드 컴퓨터

그래픽 성능이 뛰어나면서도 가격은 저렴 (모델A는 $25, 모델B는 $35, 부가세 별도)

SD카드에 설치된 데비안 리눅스를 실행

- 표준 이미지 다운로드: http://www.raspberrypi.org/downloads
- 다양한 도구들과 예제를 포함
- GPIO(General-purpose IO 커넥터를 이용하여 간단한 인터페이스 가능)

기능 및 사양

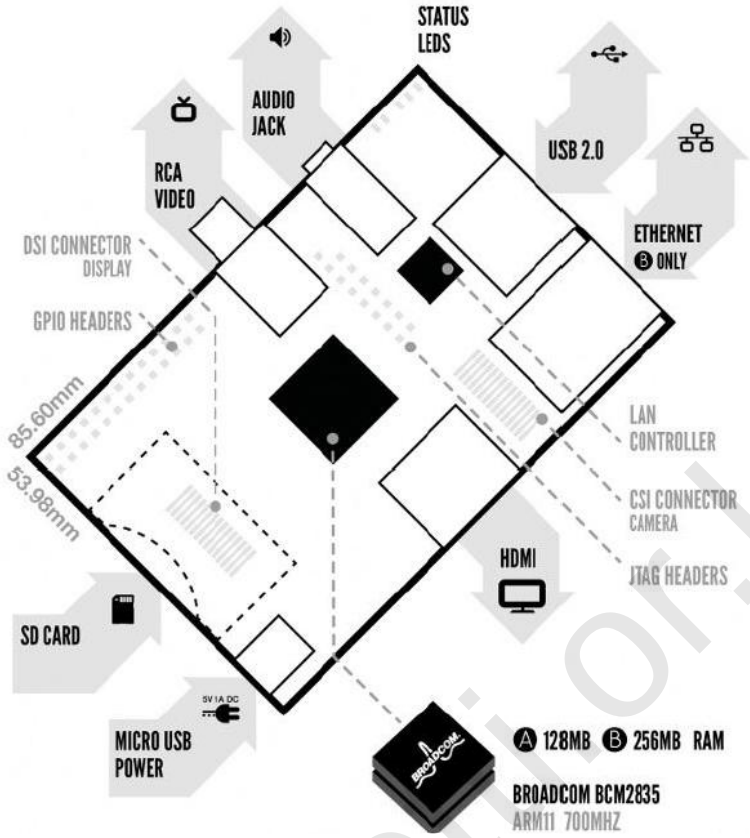
<표 1-1. 라즈베리파이 사양>

기능	사양
CPU	700MHz ARM1176-JZFS
GPU	Broadcom VideoCore IV
Memory	256MB LPDDR2-800
Video	HDMI, composite
Audio	HDMI, stereo analog
USB	2 x USB2.0 (model B)
Storage	SD card
Networking	10/100 Ethernet
Power	5V micro USB

라즈베리파이 하드웨어 구성

<그림 1-1. 라즈베리파이 하드웨어 구성>

www.ketti.or.kr



케이블 및 액세서리

<그림 1-2. 케이블 및 액세서리>

4. 라즈베리파이로 무엇을 할 수 있나?

물리적 컴퓨팅

- 소프트웨어로 센서, 모터, 광원 등의 하드웨어를 직접 제어
- 어린이를 대상으로 하는 다양한 물리적 컴퓨팅 프로젝트가 진행 중 - 로봇들, 쿼드콥터, 부모감지 문 센서, ...
- 일반 PC는 GPIO 포트 결여

미디어센터로 활용

게임개발 플랫폼

- 강력한 멀티미디어 및 3D 그래픽 성능 –적합
- 스크래치를 이용한 손쉬운 게임 개발

라즈베리의 매력은 매우 작은 범용 컴퓨터라는 점

www.ketti.or.kr

Chapter 2. 라즈베리파이와의 첫 만남

라즈베리파이는 신용카드 크기에 상당한 컴퓨팅 파워를 담은 미니 컴퓨터 보드이다. 이 보드는 놀라운 일들을 할 수 있는 능력을 가지고 있지만 본격적으로 발을 담그기 전에 미리 알아야 할 것들이 있다.

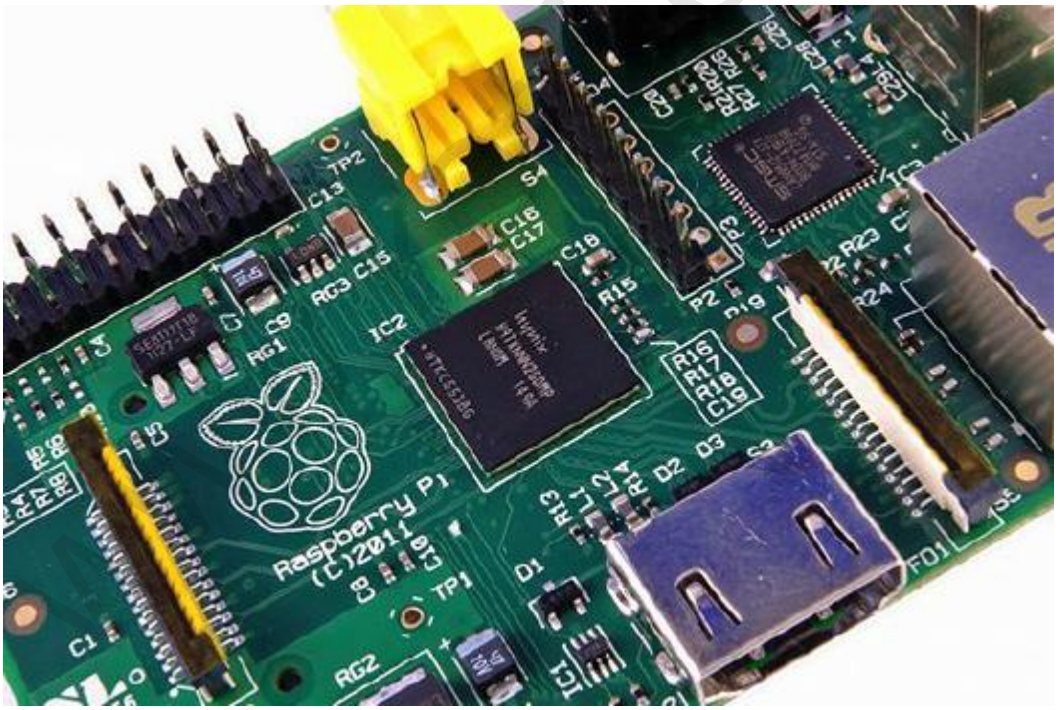
1. ARM 대 x86

라즈베리파이는 브로드컴 BCM2835 시스템온칩(SoC) 멀티미디어 프로세서 탑재

그래픽 프로세싱, 오디오, 통신 유닛 등이 포함

보드 중앙의 5612MB 메모리 칩 밑에 위치

<그림 2-1. BCM2835 SoC, 메모리 칩 및에 위치>



PC에 주로 사용되는 x86계열 프로세서와는 다른 ARM 명령어 셋을 사용

- 주로 휴대기기에 많이 사용
- 축소 명령 집합 컴퓨터 (RISC; Reduced Instruction Set Computer) <> 복잡 명령 집합 컴퓨터(CISC; Complex Instruction Set Computer)

- 기존의 PC 소프트웨어와는 호환되지 않음

저전력 특성

- 보드에 탑재된 micro-USB 포트를 통해 5V-1A의 USB전력공급으로 동작 가능

ARM11 코어

- ARMv6 명령어 셋
- ARM Cortex 패밀리에 사용되는 ARMv7 명령어 셋과도 호환되지 않으나 쉽게 이전이 가능

2. 리눅스 대 윈도우

오픈 소스 라이선스인 리눅스를 포팅

윈도우는 소스코드가 비공개인 반면 리눅스는 모든 소스 및 변경사항 공개

라즈베리파이에 포팅된 운영체제는 모두 오픈 소스

3. 라즈베리파이에 포팅 된 운영체제

Raspbian "wheezy"

- 입문 시 추천
- 리눅스 배포판 중 하나인 데비안을 최적화
- LXDE, Midori, 개발도구들 및 멀티미디어 기능 예제 소스들을 포함
- 이미지: http://downloads.raspberrypi.org/images/raspbian/2012-10-28-wheezy-raspbian/2012-10-28-wheezy-raspbian.zip

Soft-float Debian "wheezy"

- Raspbian "wheezy"와 동일하나 속도가 약간 느리긴 하지만 부동 소수점 연산을 소프트웨어로 처리하는 ABI를 사용하는 버전. 하드웨어 부동 소수점 처리를 지원하지 않는 오라클 JVM을 위한 버전
- 이미지: http://downloads.raspberrypi.org/images/debian/7/2012-08-08-wheezy-

www.ketti.or.kr

armel/2012-08-08-wheezy-armel.zip

Arch Linux ARM

- Arch Linux를 기반으로 ARM에 최적화
- 최종 사용자에게 단순성과 완전한 제어권을 주는 것을 목표로 개발
- 초보자에게는 권장하지 않음
- 최신 버전은 하드웨어 부동 소수점 연산을 지원
- 10초 내외에 부팅 완료 후 커맨드 프롬프트 표시
- 이미지: http://downloads.raspberrypi.org/images/archlinuxarm/archlinux-hf-2012-09-18/archlinux-hf-2012-09-18.zip

RISC OS

- 영국 캠브리지에서 디자인된 운영체제
- 1987년에 처음 공개됨
- ARM 마이크로프로세서를 개발한 팀에서 시작
- 캐슬 테크놀로지(Castle Technology)가 소유권을 가지고 있으며 RISC OS Open (https://www.riscosopen.org/content/)이 소스코드를 유지관리. 라즈베리파이 사용자에게는 무료 버전 제공
- 이미지 링크: http://downloads.raspberrypi.org/images/riscos/riscos-2012-11-01-RC6/riscos-2012-11-01-RC6.zip

4. 주변기기 연결

디스플레이 연결

- 세가지 비디오 출력 지원: composite, HDMI, DSI 비디오
- 컴포지트 비디오
  구형 디스플레이를 위해 지원

www.ketti.or.kr

<그림 2-2. 컴포지트 비디오 출력을 위한 노란색 RCA 포노 커넥터>

- HDMI 비디오

  High Definition Multimedia Interface

  Full HD 1920x1080 해상도 지원

  HDMI 포트가 없는 모니터는 HDMI-to-DVI(Digital Video Interconnect) 케이블 또는 DVI-D 이용

<그림 2-3. 은색 HDMI 커넥터, 고해상도 비디오 출력>

www.ketti.or.kr

- DSI 비디오

  인쇄회로기판의 윗면 SD카드 슬롯 위에 있는 플라스틱 레이어로 보호되어 있는 작은 리본 커넥터를 이용

  Digital Serial Interface

  태블릿 또는 스마트폰 등의 평면 디스플레이에 주로 사용

  주로 엔지니어들을 위한 커넥터

오디오 연결

- HDMI 포트 이용 시 비디오 신호와 오디오 신호 동시 출력
- 3.5mm 오디오 잭을 통해 아날로그 오디오 출력 제공

키보드 및 마우스

- 라즈베리파이 모델 B는 2개의 USB 제공, 모델 A는 1 개 제공 (USB 허브 필요)

<그림 2-4. 모델 B의 두개의 USB 포트>

- 전원 공급을 별도로 받지 않고 USB를 통해 받으려면 700mA이상을 공급할 수 있는 전원 공급형 USB허브를 이용

www.kotti.or.kr

5. SD카드에 운영체제 이미지 써넣기

운영체제 이미지가 미리 쓰여진 SD카드를 구매한 경우 건너뜀

2GB SD카드 준비

SD카드리더기를 PC에 연결하고 SD카드를 리더기에 삽입

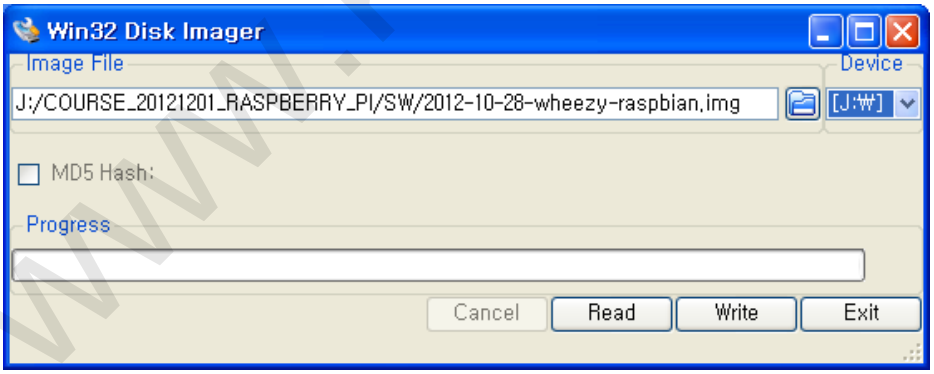
운영체제 이미지 다운로드

- 다운로드 링크: http://downloads.raspberrypi.org/images/raspbian/2012-10-28-wheezy-raspbian/2012-10-28-wheezy-raspbian.zip
- 압축을 풀면 2012-10-28-wheezy-raspbian.img 파일이 나온다. 파일 크기는 대략 1.8GB

이미지쓰기 유틸리티 Win32DiskImager 다운로드

- 다운로드 링크: https://launchpad.net/win32-image-writer/+download
- 압축을 풀고 Win32DiskImager.exe를 실행. 운영체제 이미지 파일의 위치를 지정하고 SD 카드 드라이브를 지정

<그림 2-5. Win32DiskImager 실행 화면>



Write 버튼을 클릭

경고: SD카드가 Device 항목에 나타나지 않는 경우 다음 방법 사용

이미지 쓰기 유틸리티 flashnull 을 사용하는 방법

- 다운로드 링크: http://shounen.ru/soft/flashnul/flashnul-1rc1.zip

www.kett.or.kr

- 윈도우의 커맨드 창을 연다
  장치 목록을 조회
  C:₩Program Files₩flashnull-1rc1> flashnull –p
  Available Physical Devices항목에서 SD카드의 장치 번호를 확인
  운영체제 이미지를 쓴다
  C:₩Program Files₩flashnull-1rc1> flashnull 1 –L e:₩jung10₩downloads₩2012-10-28-wheezy-raspbian.img

6. USB저장 장치 연결

부팅에 사용된 SD카드가 2GB인 경우 저장 용량이 부족

UMS(USB Mass Storage)를 저장장치로 마운트하여 사용 가능

마운트하여 사용하는 방법은 리눅스 시스템 관리 장에서 다룸

7. 네트워 연결

모델 B는 10/100 이더넷 포트 제공

모델A는 이더넷 포트 없음. USB 이더넷 어댑터 이용

유선 이더넷 연결

- 허브에 연결
- PC와 1:1 연결
  - 인터넷 연결을 위해서는 PC네트웍 설정에서 브리지 설정 필요
- 라즈베리파이는 auto-MDI를 지원하므로 스트레이트 또는 크로스 랜 케이블 모두 사용 가능
- 라즈베리파이는 기본적으로 DHCP 클라이언트 설정

무선 이더넷 연결

- USB 무선랜 어댑터를 이용하면 무선랜 접속 가능
- 라즈베리파이와 호환여부 확인 필요

www.ketti.or.kr

참고: http://elinux.org/RPi_VerifiedPeripherals - USB_WiFi_Adapters

- 802.11a 타입 어댑터는 호환, 802.11g는 지원 안됨
- 지원 주파수 확인 필요. 802.11a 규격은 다수의 주파수를 지원.
  2.4GHz만 지원하는 어댑터로 5GHz를 지원하는 AP에 접속은 불가
- 암호화 방식 확인 필요. 대부분 모든 타입의 암호화 방식을 지원하나 구형의 어댑터는 최신 AP가 요구하는 암호화 방식을 지원하지 않을 수도 있음. 가장 보편적이고 오래된 방식은 WEP, 좀 더 나중에 나온 방식은 WPA, WPA2
- 별도의 설정 필요 없이 USB포트에 꽂으면 됨. 자세한 설정 방식은 "네트웍 설정" 장에서 다룸

8. 전원 연결

라즈베리파이는 보드 좌측의 대부분의 스마트폰과 태블릿에 사용되고 있는 것과 같은 마이크로 USB 커넥터를 통해 전원 공급

대부분의 스마트폰 용 전원 어댑터가 호환되나 최소 700mA이상 공급 가능해야 함

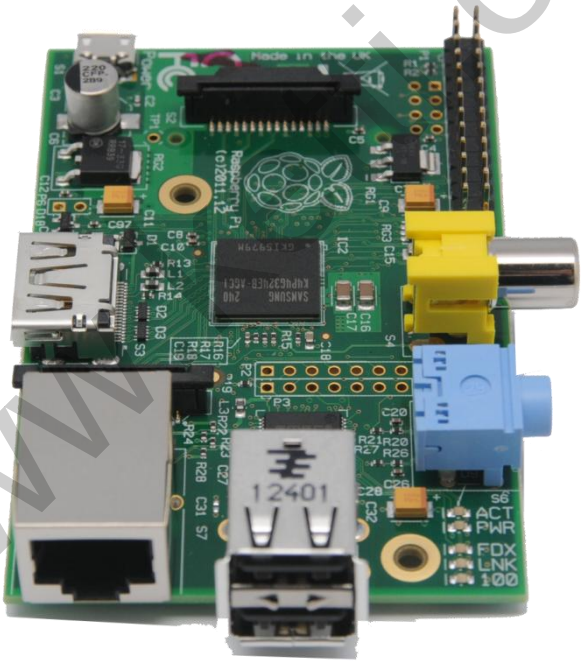
PC의 USB 포트는 공급할 수 있는 전력이 충분치 않으므로 전원 공급형 USB허브 사용 권장.

본 문서는 자유롭게 배포 가능합니다.
다만 상업적 목적 및 문서 수정 배포시
한국전자통신교육원(kks@ketti.or.kr)로 문의 부탁드립니다.

www.ketti.or.kr

라즈베리파이를 활용한 오픈 소스 R&D 플랫폼

(3,4,5장)



한국전자통신교육원

본 문서는 자유롭게 배포 가능합니다.
다만 상업적 목적 및 문서 수정 배포시
한국전자통신교육원(kks@ketti.or.kr)로 문의 부탁드립니다.

www.ketti.or.kr

목차

목차 1

Chapter 1. 라즈베리파이 소개 6

1. 프로그래밍의 재미 6

2. 라즈베리파이의 역사 6

3. 라즈베리파이란? 7

4. 라즈베리파이로 무엇을 할 수 있나? 8

Chapter 2. 라즈베리파이와의 첫 만남 10

1. ARM 대 x86 10

2. 리눅스 대 윈도우 11

3. 라즈베리파이에 포팅 된 운영체제 11

4. 주변기기 연결 12

5. SD카드에 운영체제 이미지 써넣기 15

6. USB저장 장치 연결 16

7. 네트워 연결 16

8. 전원 연결 17

Chapter 3. 리눅스 시스템 관리 18

1. 리눅스 개요 18

2. 리눅스 기초 18

3. 데비안(Debian) 소개 19

4. 외부 저장 장치 사용 22

5. 사용자 계정 만들기 23

www.ketti.or.kr

6. 파일 시스템 구성 24

7. 소프트웨어 설치 및 제거 26

Chapter 4. 문제 해결 30

1. 키보드 및 마우스 진단 30

2. 전원 진단 30

3. 디스플레이 진단 32

4. 부팅 진단 32

5. 네트웍 진단 33

6. 비상용 커널 35

Chapter 5. 네트웍 설정 37

1. 유선 네트워킹 37

2. 무선 네트워킹 40

Chapter 6. 파티션 관리 50

1. 새 파티션 작성 50

2. 파티션 크기 변경 53

3. 수작업으로 파티션 크기 변경 54

4. 더 큰 SD카드로 옮기기 58

Chapter 7. 하드웨어 구성 설정 61

1. 하드웨어 설정 – config.txt 61

2. 디스플레이 설정 변경 62

3. 부팅 옵션들 65

4. 라즈베리파이 오버 클럭킹 65

www.ketti.or.kr

5. 메모리 파티셔닝 – start.elf 68

6. 소프트웨어 설정들 – cmdline.txt 68

Chapter 8. 홈씨어터로 활용 70

1. Rasbmc를 이용한 전용 HTPC 구축 70

2. 인터넷 미디어 스트리밍 71

3. 로컬 네트웍 미디어 스트리밍 73

4. Rasbmc 설정 변경 75

Chapter 9. 업무용으로 활용 77

1. 클라우드 기반 앱 사용 77

2. 오픈오피스 사용 78

3. Gimp를 이용한 이미지 편집 79

Chapter 10. 웹 서버로 활용 81

1. LAMP 스택 설치 81

2. WordPress 설치 84

Chapter 11. 스크래치로 프로그래밍 89

1. 예제 1: Hello World 89

2. 예제 2: 애니메이션 및 사운드 93

3. 예제 3: 간단한 게임 95

4. 로보틱스 및 센서들 100

5. 더 읽어보기 101

Chapter 12. 파이썬 프로그래밍 102

1. 파이썬 소개 102

www.ketti.or.kr

2. 예제 1: Hello World 102

3. 파이썬 프로그램 실행 가능하게 만들기 106

4. 예제 2: 주석, 입력, 변수 및 반복문 107

5. 예제 3: pygame을 이용하여 게임 만들기 110

6. 파이썬과 네트워킹 119

7. 더 읽어보기 124

Chapter 13. 하드웨어 해킹 126

1. 전자 부품 126

2. 저항 값 읽기 128

3. GPIO 포트 129

4. 파이썬에서 GPIO 사용하기 132

5. GPIO 출력: LED 밝히기 134

6. GPIO 입력: 버튼 읽기 138

7. 브레드보드 벗어나기 141

8. 납땜에 대한 간단한 가이드 144

참조 A: Python Recipes 149

1. Raspberry Snake (Chapter 11, Example 3) 149

2. IRC User List (Chapter 11, Example 4) 152

3. GPIO Input and Output (Chapter 12) 155

참조 B: HDMI 디스플레이 모드 목록 156

참조 C: 참고 문헌 163

1. 참고 문헌 163

www.ketti.or.kr

2. 참고 사이트.....163

www.ketti.or.kr

Chapter 3. 리눅스 시스템 관리

근래의 대부분의 리눅스 배포판들은 대부분의 공통적인 작업을 쉽게 할 수 있도록 도와주는 그래픽 사용자 인터페이스(GUI; Graphic User Interface)를 포함하고 있어 매우 사용자 친화적이다. 그렇지만 윈도우나 OS-X와는 사뭇 다른 점이 많다. 라즈베리파이를 최대한 활용하기 위해서는 운영체제에 대한 간단한 사전 지식이 필요하다.

1. 리눅스 개요

커널

- 리눅스의 핵심 역할을 하는 중심 부분

GNU Linux

- 오리지널 버전의 리눅스는 커널에 GNU 그룹의 여러 가지 도구들을 더하여 만들어짐
- 기본적이지만 강력, 다중 사용자 기능을 처음부터 제공

리눅스에서는 대부분의 작업을 제한된 기능의 사용자 계정에서 작업

- 기능은 제한되지만
- 실수로 인한 사고를 예방
- 바이러스 등으로부터 보안상 더 안전

2. 리눅스 기초

어떤 목적을 달성하는 방법은 GUI(Graphic User Interface)를 이용하는 방법과 명령어(Command)를 사용하는 방법이 있다.

리눅스의 가장 기초적인 명령어들

- ls <디렉터리 또는 파일명>
  대상의 리스트를 표시. 윈도우의 dir 명령에 해당
- cd <디렉터리>
  대상 디렉터리로 이동

www.kett.or.kr

- mv <대상파일> <새파일>
  <대상파일>을 <새파일>로 이름을 변경하거나 <새파일>이 이미 존재하는 디렉터리명이면 그 밑으로 이동. 윈도우의 ren 명령에 해당

- rm <대상파일>
  <대상파일>을 삭제. 윈도우의 del 명령에 해다

- mkdir <디렉터리명>
  디렉터리 작성. 윈도우의 md 명령어에 해당

3. 데비안(Debian) 소개

데비안은 가장 오래된 배포판 중 하나

가벼운 특성

라즈베리파이 버전은 데스크톱 버전의 일부만 포함

라즈베리파이 버전 데비안에 포함된 기능

- 웹브라우징 도구
- 파이썬(Python) 프로그래밍 도구 및 수 많은 예제 포함
- GUI – LXDE(Lightweight X11 Desktop Environment)

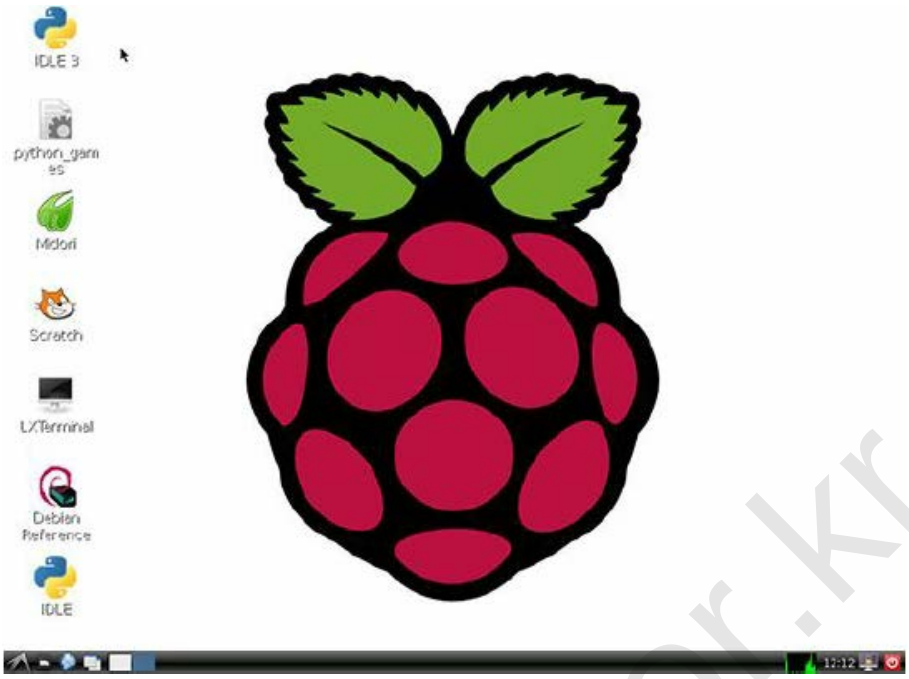
추가 소프트웨어는 패키지 배포 관리도구인 apt를 이용해 빠르게 설치 가능

경고: 라즈베리파이 부팅 후 GUI를 기동하기 위해서는 startx 명령어 이용

화면 좌하단의 메뉴 아이콘을 마우스 주 버튼으로 클릭하면 카테고리별로 분류된 소프트웨어들을 실행 가능

그림 2-1 - LXDE 데스크톱 처음 화면

www.ketti.or.kr



카테고리별 소프트웨어

액세서리

- Debian Reference 참조 가이드
- File Manager PCManFM 파일 매니저
- Image View GPicView 이미지 뷰어
- Leafpad 간단한 텍스트 편집기
- LXTerminal LXDE 터미널 패키지, 명령어 실행 창
- Root Terminal 실행 창 LXTerminal과 비슷하나 슈퍼유저인 root로 로그인 된 명령어
- Xarchiver 압축파일 작성, 추출 도구

교육

- Scratch 어린이들을 위한 그래픽 기반 프로그래밍 언어.
- Squeak 스크래치가 실행되는 플랫폼. 직접 실행할 필요는 거의 없음

인터넷

www.ketti.or.kr

- Midori 웹 브라우저. 매우 가볍다.
- Midori Private Browsing 미도리 웹 브라우저를 프라이빗 모드(사이트 방문 기록이 남지 않는 모드)로 기동할 때 사용
- NetSurfWeb Brower 또 다른 웹 브라우저. 미도리와 보완적으로 사용

프로그래밍

- IDLE 파이썬을 위한 통합개발환경(IDE). "파이썬 소개" 장에서 더 자세히 다룸
- IDLE 3 기본적인 파이썬 2.7 버전이 아닌 파이썬 3 버전에 맞춰 구성된 IDLE을 구동한다. 대 부분 두 버전간에 호환이 되지만 몇 가지 기능은 파이썬 3 버전이 필요하다.
- Scratch 교육 카테고리의 Scratch 항목과 동일하다.
- Squeak 교육 카테고리의 Squeak 항목과 동일하다.

사운드 및 비디오

- Music Player LXMusic은 XMMS2 음악 플레이어 소프트웨어의 간단하고 가벼운 인터페이스
- 시스템 도구
- Task Manager 메모리 여유 확인, 프로세서의 작업 부하, 프로그램 강제 종료 등을 할 수 있는 도구

사용자화(Preferences)

- Customise Look and Feel 창의 스타일과 색상 등을 포함하여 GUI의 외관을 조절
- Desktop Session Settings 사용자가 로그인 했을 때 어떤 프로그램을 자동으로 실행하고 어떤 윈도우 매니저(창 틀, 제목 표시줄 등을 관리하는 소프트웨어)를 사용할지를 지정
- Keyboard and Mouse 키보드 타입 설정, 마우스 감도 조절 등의 입력장치 설정
- Monitor Settings 라즈베리파이에 연결된 모니터 또는 TV 등의 해상도 변경, 좀 더 세부적인 설정은 설정 파일을 수정해야 된다 (라즈베리 설정 장에서 다룸)

www.kett.or.kr

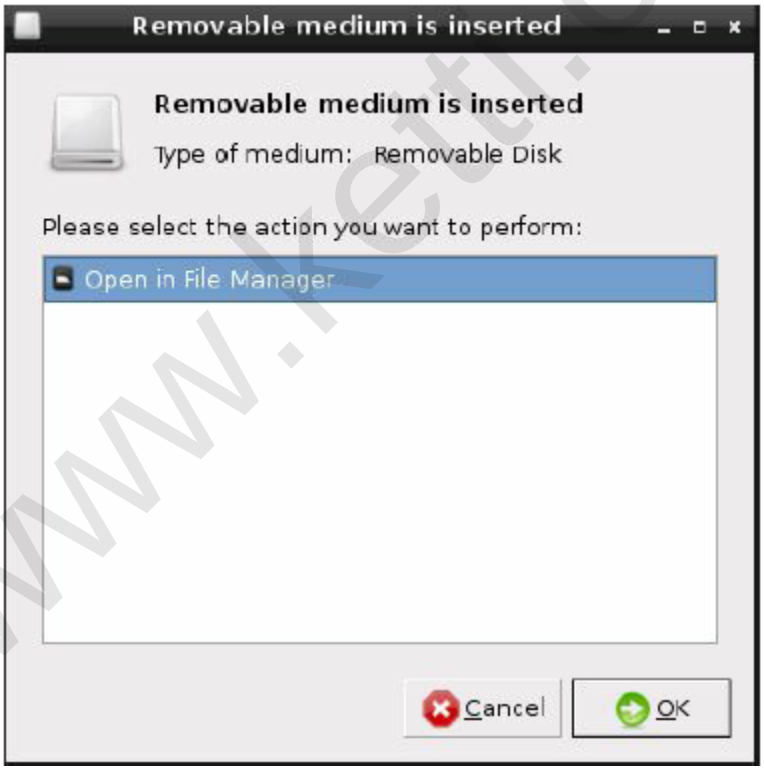
- Openbox Configuration Manger LXDE GUI는 OpenBox라고 하는 데스크톱 환경 설정 관리자를 이용하여 새로운 테마를 적용하거나 사용자 인터페이스의 특정 속성을 변경 할 수 있다.
- Preffered Applications 파일 타입에 따라 어떤 응용프로그램을 연결하여 실행할지를 변경.

4. 외부 저장 장치 사용

SD 카드의 최대 저장 공간은 64 GB (3 TB를 지원하는 데스크톱 하드드라이브에 비하면 너무 작은 용량)

LXDE상에서는 외부 USB 저장장치를 연결하면 자동으로 마운트 됨

그림 2-2 - LXDE는 자동으로 USB 저장장치를 마운트



콘솔 상태에서는 다음과 같이 마운트

부팅 시 데스크톱(GUI)으로 들어가지 않고 콘솔 모드로 남도록 하기 위해서는

$ sudo raspi-config 실행

www.ketl.or.kr

boot_behaviour Start desktop on boot? 를 선택한 후

Should we boot straight to desktop? 에서 No 를 선택

빠져나온 후 리부팅 한다.

USB 저장장치를 라즈베리파이의 USB 포트에 연결

라즈베리파이에 연결된 장치 목록 조회

$ sudo fdisk –l

사이즈로 어떤 장치인지를 판단한다. 장치명은 /dev/sdXN 형태로 나타나는데 X는 드라이브를 의미하고, N은 파티션 번호를 의미한다. USB포트에 연결된 유일한 저장장치인 경우 첫 번째 파티션은 /dev/sda1, 두 번째 파티션은 /dev/sda2 가 될 것이다.

마운트 포인트를 만들어 준다.

$ sudo mkdir /media/usbdisk

모든 사용자에게 사용 권한을 부여한다.

$ sudo chgrp –R users /media/usbdisk && sudo chmod –R g+w /media/usbdisk

USB 저장장치를 마운트 한다.

$ sudo mount /dev/sdXN /media/usbdisk(첫 번째 디스크 첫 번재 파티션은 sda1)

5. 사용자 계정 만들기

라즈베리파이는 기본적으로 일반 사용자계정 pi(패스워드는 raspberry)와 슈퍼 유저 계정 root가 있다. pi 계정을 그대로 사용해도 되지만 보안상의 이유로 또는 새로운 사용자를 위해 새로운 계정을 만들 필요가 있다. 새로운 계정을 만들 때는 다음과 같이 한다.

기존의 pi 계정으로 로그인 한다.

pi계정이 속해 있는 그룹을 확인

$ groups

pi adm dialout cdrom sudo audio video plugdev games users netdev input

www.ketti.or.kr

새로운 사용자 계정을 생성

$ sudo useradd –m –G adm,dialout,cdrom,sudo,audio,video,plugdev,games,users,netdev,input username (주의: 그룹명 콤마 사이에 공백 없음)

username에 해당하는 새로운 계정이 만들어 진다.

-m 옵션은 계정에 해당하는 홈 디렉터리 작성

-G 옵션은 새로운 계정이 속할 그룹

새로운 계정에 패스워드를 지정

$ sudo passwd username

패스워드를 2회 입력한다

새로 만들어진 계정이 속한 그룹을 확인

$ groups username

6. 파일 시스템 구성

논리적 구성

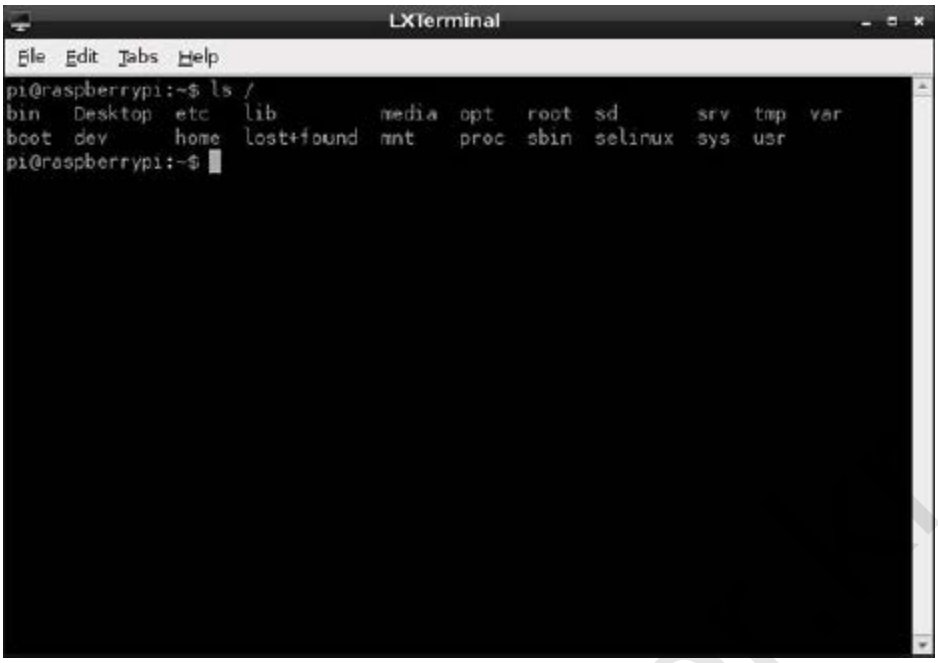
리눅스에는 윈도우와 같은 드라이브명이 없다.

모든 파일과 디렉터리는 루트 파일 시스템(/) 밑에 계층적으로 만들어진다.

루트 디렉터리를 ls / 명령으로 조회해보면 다음 그림과 같다.

그림 2-3 - 라즈베리파이의 루트 파일 시스템

www.ketti.or.kr



기본 데비안 배포판의 디렉터리 구성은 다음과 같다

- boot – 라즈베리파이가 기동될 때 필요한 리눅스 커널 및 여타 패키지를 보관
- bin – GUI를 실행하는 파일들과 같은 운영체제 관련 바이너리 파일들을 보관
- dev - SD카드에 실존하는 것이 아닌 가상 디렉터리, 저장장치, 사운드 카드, HDMI 등의 시스템에 연결된 모든 장치들을 이 곳을 통해 접근 할 수 있다.
- etc – 사용자 계정과 암호화된 패스워드를 비롯한 여러 가지 설정파일들을 보관
- home – 각 계정의 홈 디렉터리들
- lib – 수 많은 애플리케이션들에 의해 공유되는 라이브러리들을 보관
- lost+found – 파일이 깨졌을 때 파일 시스템 체크에 의해 발견된 깨진 조각들을 보관하기 위한 특별한 디렉터리
- media – USB 메모리 스틱, 외장형 CD 드라이브 등의 분리 가능한 저장장치를 위한 특별한 디렉터리
- mnt – 외부 저장 장치 등을 수동으로 마운트하는데 사용되는 디렉터리
- opt – 운영체제의 일부가 아닌 소프트웨어를 보관. 사용자가 새로운 소프트웨어를 설치하면 대부분 이곳에 설치

- proc – 리눅스에서 실행되고 있는 프로그램 즉, 프로세스에 대한 정보를 제공하는 가상 디렉터리
- seliunx – 미국 국립 보안국(US National Security Agency)에서 개발한 보안 강화 리눅스 관련 파일들을 보관
- sbin – 슈퍼유저에 의해 사용되는 시스템 관리 유틸리티 바이너리 파일들을 보관
- sys – 특별한 운영체제 파일들이 있는 디렉터리
- tmp – 임시 파일들이 자동으로 보관되는 곳
- usr – 사용자가 접근할 수 있는 프로그램들을 보관하기 위한 디렉터리
- var – 프로그램들이 값이 변하는 자료나 변수들을 저장하기 위한 가상 디렉터리

물리적 구성

기본 데비안 배포판은 SD카드 상에 두 개의 파티션으로 존재

첫 번째 파티션은 마이크로소프트의 분리가능 장치들을 위한 포맷인 VFAT 포맷의 75MB 크기. 라즈베리파이를 설정하는데 필요한 파일들과 리눅스 자체를 로드하기 위한 파일들을 가지고 있으며 부팅 후에는 /boot로 마운트 된다.

두 번째 파티션은 리눅스의 고속 접근 및 데이터 안정성을 위한 기본 포맷인 EXT4 포맷으로 훨씬 크기가 크다. 나머지 모든 프로그램, 사용자가 설치한 프로그램, 사용자 데이터가 모두 이 곳에 저장된다. SD 카드의 대부분을 차지한다.

7. 소프트웨어 설치 및 제거

소프트웨어를 설치하고자 할 때는 강력한 패키지 매니저인 apt 도구를 사용

UI 기반의 시냅틱 패키지 매니저도 있지만 메모리 부족으로 실행이 어려울 수 있어 커맨드 기반의 apt 도구 사용을 권장

패키지 매니저는 소프트웨어 설치뿐만 아니라, 제거, 업데이트 등 전반을 관리

주의: 새로운 소프트웨어를 설치하고나 업그레이드할 때는 apt의 캐시가 최신 상황을 반영하고 있는지를 확실히 하기 위해 sudo apt-get update를 해주는 것이 좋다.

www.keti.or.kr

소프트웨어 찾기

apt를 통해 설치 가능한 소프트웨어의 목록인 캐시를 가지고 있는 있는 인터넷상의 서버를 리포지토리라고 한다.

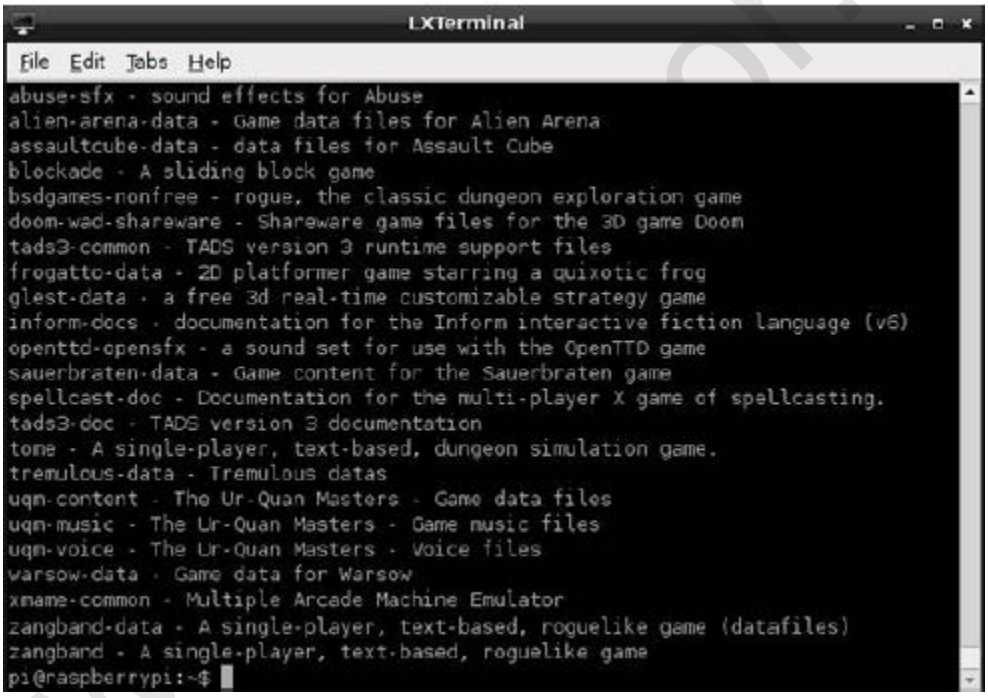
캐시 목록을 관리하기 위해서는 apt-cache를 이용

제목 또는 설명에 game이라는 단어를 가진 소프트웨어를 검색

```
$ apt-cache search game
```

결과가 너무 많을 수 있으므로 가능한 세밀하게 하는 것이 좋다.

그림 2-4 - apt-cache "game" 검색 결과의 마지막 부분



결과가 너무 많을 때는 less 도구를 써서 페이지 단위로 멈춰서 볼 수 있다.

```
$ apt-cache search game | less
```

커서키로 상하 스크롤, q로 끝냄

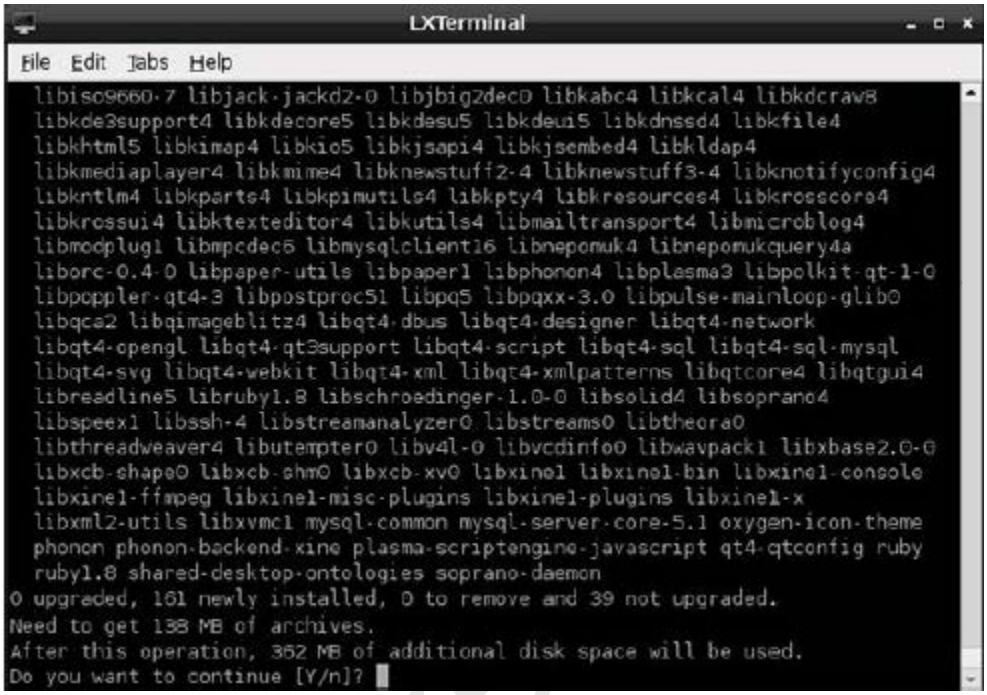
소프트웨어 설치

소프트웨어를 설치할 때는 sudo를 써야 되며 apt-get 명령 사용, 예를 들어 thrust 게임을 설치하려면 다음과 같이 입력

```
$ sudo apt-get install thrust
```

apt 패키지 매니저는 의존성을 자동으로 체크하여 필요한 패키지를 설치

그림 2-5 - apt가 OpenOffice 패키지가 의존하는 패키지 목록을 표시



소프트웨어 설치 제거

apt-get remove 명령어를 사용하여 제거하면 더 이상 필요치 않은 의존관계 패키지까지 자동으로 제공

thrust 패키지를 제거하려면 다음과 같이 한다

```
$ sudo apt-get remove thrust
```

remove 명령은 설정 파일은 남겨두는 반면 purge 명령어를 사용하면 완전히 제거한다.

```
$ sudo apt-get purge thrust
```

소프트웨어 업그레이드

apt-get upgrade를 사용하면 설치된 소프트웨어들의 최신 버그 수정과 보안 패치를 적용할 수 있다. 업그레이드를 할 때는 캐시를 반드시 업데이트한 후 실행하는 것이 좋다.

```
$ sudo apt-get update
```

```
$ sudo apt-get upgrade
```

개별 패키지만 업그레이드하고자 할 때는 해당 패키지를 다시 install 하면 된다. 이미 설치된 패키지에 대해 install이 실행되면 패키지 매니저는 업그레이드 의사로 판단한다.

```
$ sudo apt-get install thrust
```

www.ketti.or.kr

Chapter 4. 문제 해결

때때로 뜻하는 대로 작업이 매끄럽게 진행되지 않을 때가 있다. 장비들이 복잡해질수록 문제 또한 더 복잡하게 발생한다. 라즈베리파이는 참으로 복잡한 장비이다.

대부분의 일반적인 문제들은 곧바로 진단하고 고칠 수 있다. 가장 보편적으로 발생하는 이상 증상들을 살펴보고 어떻게 고칠 수 있는지를 살펴본다.

1. 키보드 및 마우스 진단

가장 흔하게 발생하는 문제는 글자가 중복 입력되는 현상

예) startx 입력 시 stttttttttttttttttttarxxxxxxxxxxxxxxxxxxx처럼 입력됨

주 원인은 전력 부족, 키보드 뒷면의 소비전력을 확인해 본다. LED를 사용하는 키보드는 그렇지 않은 일반 키보드 보다 전력 소모가 많다.

라즈베리파이의 소비 전력은 500mA(2.5W, Model A), 700mA(3.5W, Model B)

라즈베리파이의 USB 포트에 연결된 plyfuse 라는 부품이 USB에 연결된 주변기기로 150mA이상의 전력이 소모되면 전류를 차단하므로 주변기기가 이상 작동하거나 아예 작동하지 않게 된다.

주변기기를 전력소모가 낮은 기기로 교체하거나 전원을 사용하는 USB 허브를 통해서 연결

호환성 문제 – USB 포트에 무선랜 어댑터 같은 다른 장비를 추가했을 때 키보드가 이상 동작을 일으키고, 빼면 정상으로 돌아오는 경우 간혹 라즈베리파이와의 호환성 문제일 수 있다. 다른 장비로 교체해야 된다. 알려진 비호환 장비 목록은 다음 링크를 참조한다: http://elinux.org/RPi_VerifiedPeripherals#Problem_USB_Keyboards

마우스나 트랙볼도 키보드와 마찬가지이다. 다음 호환성 목록을 참조한다: http://elinux.org/RPi_VerifiedPeripherals#Working_USB_Mouse_Devices

2. 전원 진단

라즈베리파이에서 발생하는 많은 수의 문제들이 전원 공급과 관련이 있다. 모델 A는 5V 500mA의 전력이 필요하고 모델 B는 5V 700mA의 전력을 필요로 한다. 모든 USB 전원어댑터들이 정격 규격은 그렇게 표기되어 있을지라도 이 요건을 온전히 충족시켜 주는 것은 아니다.

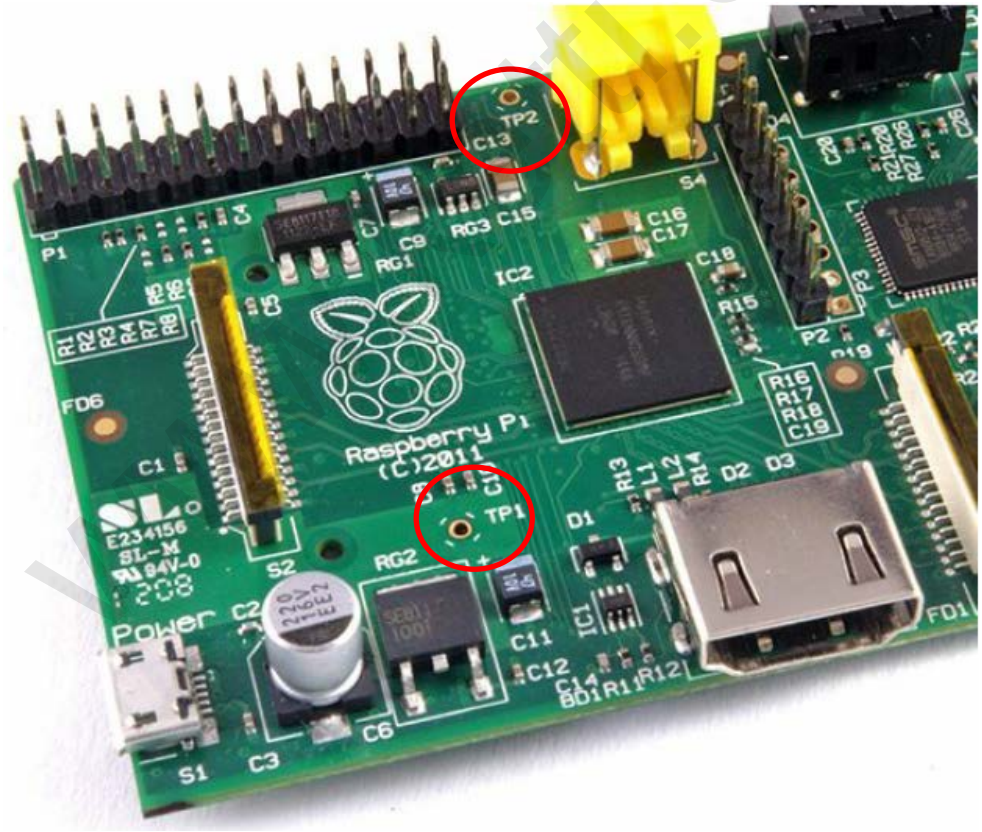
www.ketti.or.kr

경고: 공식 USB 표준에 의하면 USB에 연결된 기기는 500mA이상을 소비하지 않아야 하고 그 이상의 전력은 협상절차를 거쳐야 얻을 수 있도록 되어있다. 라즈베리파이는 전력 전력 협상 절차 동작을 하지 않으므로 데스크톱 PC의 USB 포트에 직접 연결하면 동작하지 않을 가능성이 높다.

정상적으로 동작하던 라즈베리파이가 USB 포트에 주변기기를 연결했을 때, 또는 비디오 플레이 같은 프로세서 집중적인 동작을 했을 때 이상 증세를 보이면 전압강하 문제일 가능성이 높다. 라즈베리파이는 전압 테스트를 위한 두 지점을 제공하므로 상대적으로 쉽게 테스트를 해볼 수 있다. DC 전압 측정기 또는 멀티 미터를 이용하여 측정한다.

경고: 테스트 지점 이외의 곳을 건드리지 않도록 주의한다. 라즈베리파이의 내부는 3.3V 로 동작하므로 5V의 전압이 닿으면 회로가 망가지기 쉽다. 특히 헤더 핀을 건드리지 않도록 주의한다. TP1에는 적색 프루브(+), TP2에는 흑색(GND)을 대어 측정한다.

그림 2-1 - 전압 테스트 지점 TP1과 TP2



측정값은 4.8V ~ 5V 가 되어야 한다. 그 이하면 전원 어댑터를 교체한다. 1A 용량의 어댑터를 권장. 저가 모델은 표기된 정격에 미치지 못하는 전력을 제공하는 수가 많다.

3. 디스플레이 진단

라즈베리파이는 대부분의 HDMI, DVI, 컴포지트 비디오 디스플레이와 동작하도록 만들어졌지만 간혹 기대와는 다르게 동작하기도 한다.

우선 연결된 장치의 타입을 확인한다. 특히 RCA 커넥터를 이용해 TV에 연결했을 때 특히 중요하다. 국가마다 TV를 위한 표준이 다르기 때문에 흑백으로 표시되는 경우 대부분 표준이 다르기 때문이다. 설정을 조정하는 방법은 "라즈베리파이 설정" 장에서 다룬다.

HDMI 출력 사용 시 보통 자동으로 디스플레이를 감지한다. HDMI-DVI 어댑터를 이용하여 컴퓨터 모니터를 연결한 경우 눈 같은 점들이 생기거나, 화면의 일부가 보이지 않거나, 아예 화면이 표시되지 않을 수 있다. 이 경우 해상도와 refresh rate를 조절해야 된다. "라즈베리파이 설정"장에서 수동으로 조정하는 방법을 다룬다.

화면이 너무 크거나 작게 나오는 경우 화면이 모니터 가장 자리를 벗어나거나 가장 자리가 많이 남게 되는 경우는 오버스캔이 설정되어 있기 때문이다. 오버스캔은 화면이 모니터크기를 벗어나 표시되는 것을 방지하기 위한 설정으로 값을 조정하거나 꺼주면 된다. 그 방법은 "라즈베리 파일 설정"장에서 다룬다.

4. 부팅 진단

라즈베리파이 부팅 실패의 원인은 대부분 SD카드와 관련이 있다. 데스크톱이나 랩톱 컴퓨터와 달리 라즈베리파이는 모든 것을 전적으로 SD 카드에 의존한다. SD카드를 읽지 못하면 아무런 동작도 하지 않는다.

전원을 연결했을 때 라즈베리파이의 전원 등은 켜지는데 아무 일도 일어나지 않고 OK 등이 꺼진 상태이면 SD카드에 문제가 있는 것이다. SD카드를 PC에 연결해서 파티션과 파일이 제대로 있는지를 확인한다.

SD카드가 PC에서는 정상적으로 보인다면 호환성 문제이다. 특히 클래스 10 급의 고속 카드일수록 흔하다. 다음의 호환성 목록을 참조한다:
http://elinux.org/RPi_VerifiedPeripherals#Problem_SD_Cards

www.keti.or.kr

5. 네트워 진단

네트웍 문제를 진단하는데 있어 가장 유용한 도구는 ifconfig 명령이다. 무선랜 설정은 "네트웍 설정"장을 참조한다.

ifconfig 명령을 사용하면 라즈베리파이의 모든 이더넷 포트를 볼 수 있는데 2개의 포트가 있다. 하나는 보드의 우측에 있는 물리적인 포트이고, 다른 하나는 라즈베리파이 내의 프로그램들간에 통신을 할 수 있도록 해주는 가상 루프백 인터페이스이다.

그림 2-2 - ifconfig 실행 결과

ifconfig 출력 결과의 각 부분은 다음과 같다.

- Link encap – 네트웍에 의해 사용되는 캡슐화 타입, 물리적 네트웍 포트는 ethernet, 가상 루프백 어댑터는 Local Loopback 타입이다.
- Hwaddr – 네트웍 인터페이스의 미디어 접근 제어(MAC; Media Access Control) 주소. 16진수로 표시되며 네트웍상의 각 장치마다 고유한 번호를 갖는다. 각 라즈베리파이도 각각 고유의 MAC 주소를 가지며 공장에서 설정된다.
- inet addr – 네트웍 인터페이스의 인터넷 프로토콜(IP; Internet Protocol) 주소. 이 주소를 이용하여 네트웍상에서 운용되는 웹 서버 또는 파일 서버 등의 위치를 찾을 수 있다.

- Bcast – 라즈베리파이가 연결되어 있는 네트웍을 위한 브로드캐스트 주소. 이 주소로 보내지는 트래픽은 같은 네트웍 상의 모든 장치가 수신한다.
- Mask – 라즈베리파이가 연결되어 있는 네트웍의 크기를 결정. 대부분의 가정에서는 255.255.255.0이다.
- MTU – 최대 전송 단위 크기(Maximum transmission unit size). 시스템이 전송을 위해 데이터를 여러 개의 패킷으로 자를 때 단일 패킷의 최대 크기를 지정
- RX – 수신된 네트웍 트래픽에 대한 보고. 오류 및 누락된 패킷 수를 포함. 오류 숫자가 표시되면 네트웍에 뭔가 문제가 있다는 뜻.
- TX – 송신된 네트웍 트래픽에 대한 보고.
- collisions – 네트웍 상의 시스템들이 동시에 전송 시 발생하는 충돌 횟수. 소량의 충돌은 상관없지만 많은 수의 충돌이 발생하면 네트웍에 뭔가 문제가 있다는 뜻
- txqueuelen – 송신 대기열의 길이. 대개 1000으로 설정되며 바꿀 일은 거의 없다.
- RX bytes, Tx bytes – 네트웍 인터페이스를 통과한 트래픽 양

네트웍에 문제가 발생하면 우선 네트웍 인터페이스를 비활성화시켰다가 다시 활성화 시킨다. 가장 쉬운 방법은 ifup과 ifdown을 사용하는 것이다.

네트웍이 활성화되어 있는데 ifconfig를 쳤는데 inet addr 섹션에 아무것도 나타나지 않으면 다음의 명령을 사용한다.

```
$ sudo ifdown eth0
```

네트웍 케이블 등을 점검하고 잠시 후

```
$ sudo ifup eth0
```

ping 명령어를 사용하여 네트웍 상태를 점검. 다른 컴퓨터에 데이터를 보내고 응답을 확인한다.

그림 2-3 - ping 명령 실행 결과 성공

www.ketti.or.kr

6. 비상용 커널

리눅스 커널은 /boot 디렉터리에 kernel.img 단일 파일. 전원이 들어오면 이 파일을 찾아서 로드한다.

비상용 커널은 /boot 디렉터리에 kernel_emergency.img 파일

비상용 커널은 표준 커널과 동일. 표준 커널에 변경이 가해지더라도 비상용 커널은 원래의 상태를 유지. 표준 커널에 문제가 생겼을 때 비상용 커널을 로드 하도록 할 수 있다.

첫 번째 방법은 PC에서 원래의 kernel.img파일을 kernel.img.bak으로 이름을 변경하고, kernel_emergency.img 파일을 kernel.img로 변경하여 라즈베리파이를 부팅한다. 다시 표준 커널로 되돌아가 가려면 kernel.img를 kernel_emergency.img로 변경하고, kernel.img.bak을 kernel.img로 변경하면 된다.

두 번째 방법은 /boot 디렉터리의 cmdline.txt 파일의 맨 끝에 다음 커맨드를 추가한다.

kernel=kernel_emergency.img

라즈베리파이가 로드할 커널 명칭을 지정해주는 것이다. 되돌아가는 방법은 추가된 행을 다시 지우면 된다. cmdline.txt 에 대해서는 "라즈베리파이 설정" 장에서 더 다룬다.

www.kotti.

www.ketti.or.kr

Chapter 5. 네트웍 설정

대부분의 사용자들에게 있어서 라즈베리파이의 네트웍을 설정하는 것은 모델 A의 경우 이더넷 포트에 네트웍 케이블을 연결하는 것 또는 모델 B의 경우 USB이더넷 어댑터를 연결하는 정도면 된다. 하지만 어떤 경우에는 수작업으로 설정이 필요한 경우도 있다.

1. 유선 네트워킹

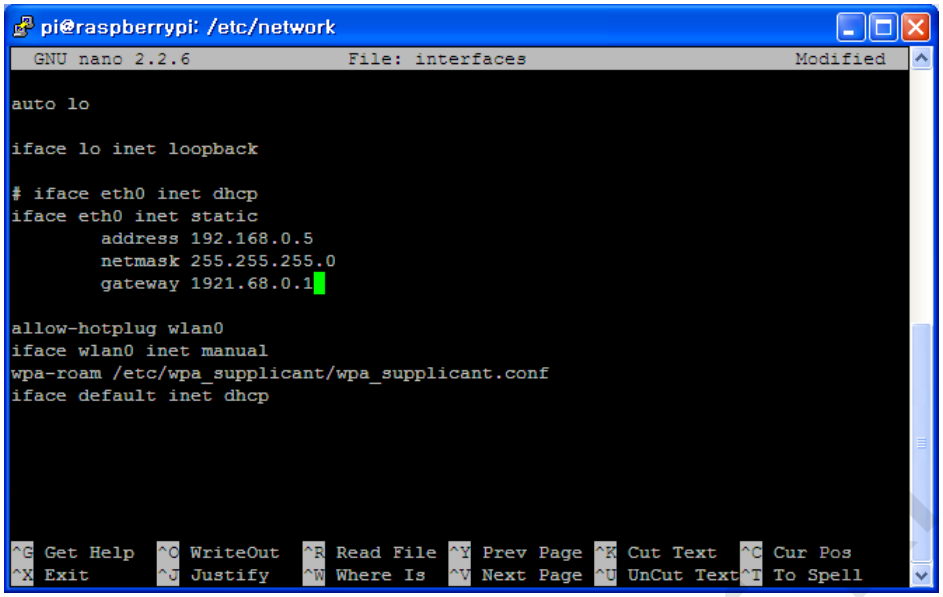
대부분의 가정이나 사무실에서는 DHCP 서버가 자동으로 설정 값을 부여하므로 수동 설정이 필요 없다.

수동(고정 IP) 설정은 /etc/network/interfaces 파일 편집

```
$ cd /etc/network

$ sudo cp interfaces interfaces.org

$ sudo nano interfaces

  iface eth0 inet static                  ← dhcp를 static으로 수정

  [Tab] address xxx.xxx.xxx.xxx           ← 192.168.0.XY – 2 ~ 254

  [Tab] netmask xxx.xxx.xxx.xxx           ← 255.255.255.0

  [Tab] gateway xxx.xxx.xxx.xxx           ← 192.168.0.1
```

그림 2-1 - 나노 에디터로 /etc/network/interfaces 파일 편집

www.ketti.or.kr



네트웍 서비스를 재시작 한다.

$ sudo service networking restart

IP주소가 의도대로 설정되었는지 확인

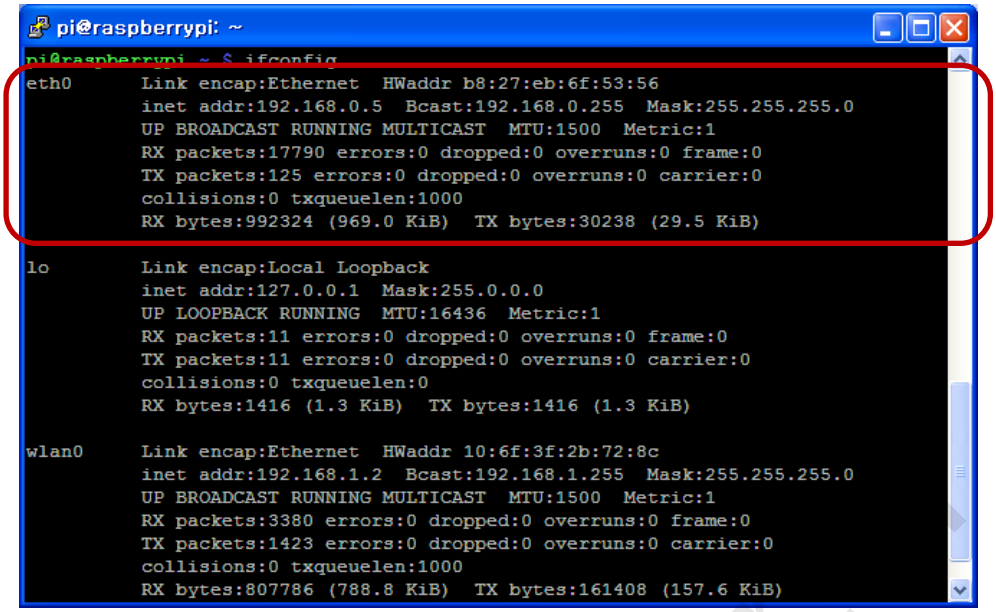
$ ifconfig

잘 안될 경우 시스템을 재부팅해 본다.

$ sudo shutdown –r now

그림 2-2- 고정 IP 주소 설정 확인

www.ketti.or.kr



DNS 서버 목록 등록

DNS서버는 질의 받은 도메인명에 해당하는 호스트의 IP주소를 알려주는 서버이다. DHCP 클라이언트인 경우 /etc/resolv.conf파일에 DNS서버 목록이 자동으로 채워진다. 수동으로 IP을 설정할 경우 직접 써넣어야 된다.

$ cd /etc

$ sudo cp resolv.conf resolv.conf.org

$ sudo nano resolv.conf

domain ketti.or.kr ← 내 네트웍의 도메인명

search ketti.or.kr ← 호스트명만으로 찾을 때 검색할 도메인명

nameserver 192.168.0.1 ← 게이트웨이(공유기)의 dns 서버를 지정

nameserver 8.8.8.8 ← 구글의 공개된 DNS 서버

nameserver 8.8.4.4

저장 후 빠져 나온다.

$ sudo service networking restart

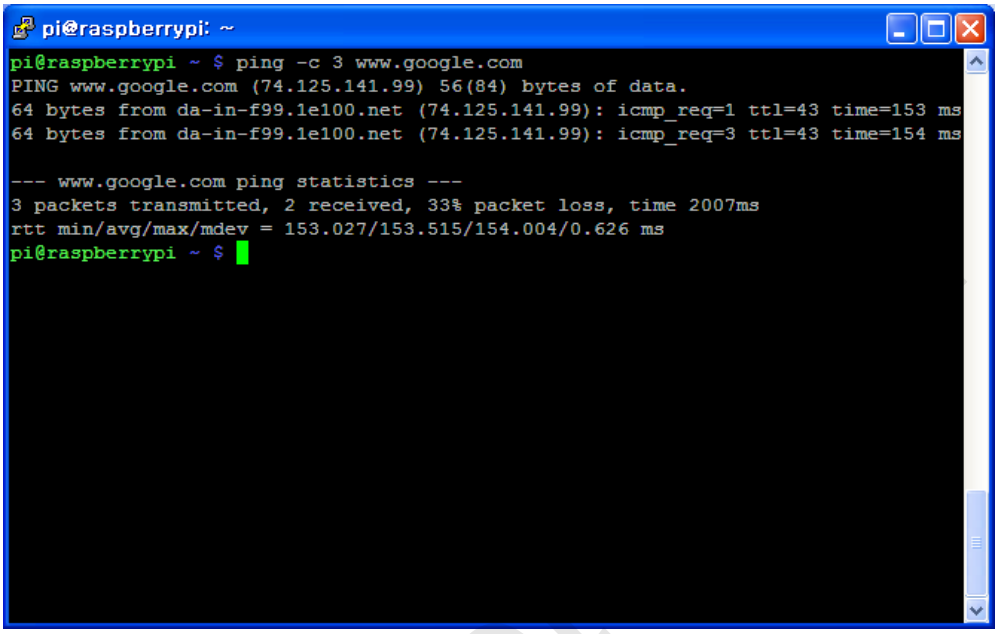
$ ping www.google.com

www.ketti.or.kr

도메인명으로 연결이 가능함을 확인할 수 있다.

$ ping –c 3 www.google.com

그림 2-3 - 도메인명으로 연결 확인



2. 무선 네트워킹

무선 네트웍 어댑터는 기본으로 장착되어 있지 않지만 USB WiFi 어댑터를 연결하여 사용할 수 있다.

경고: USB WiFi 어댑터는 전력을 많이 소모한다. 라즈베리파이의 USB 포트에 직접 연결하기 보다는 유전원 USB 허브를 통해 연결할 것을 권한다.

우선 사용할 무선 액세스 포인트의 설정 값들을 알아야 한다.

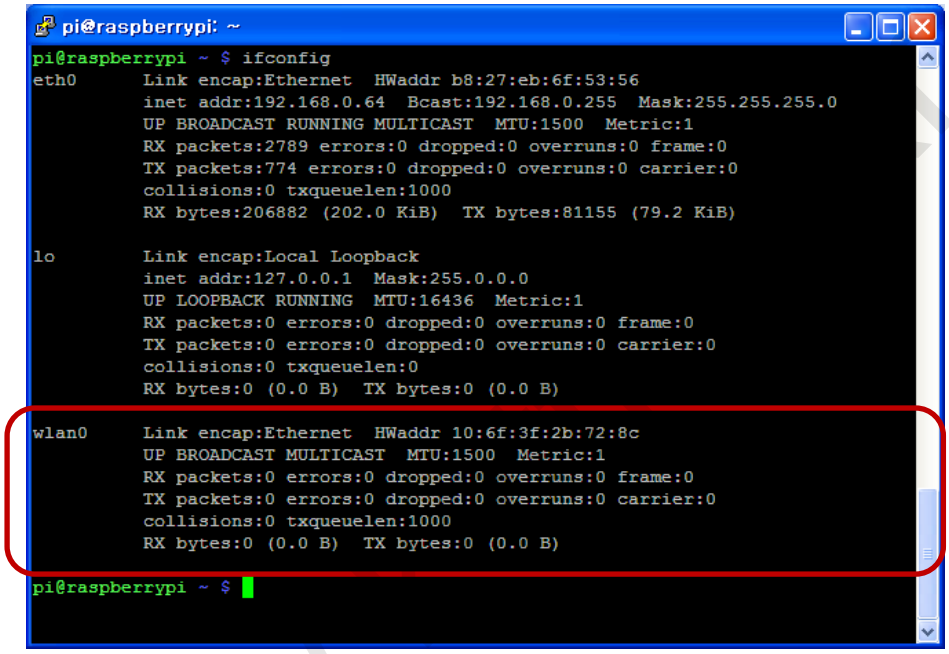
- 네트웍이름: SSID(Service Set Identifier)
- 암호화 종류
- 패스워드
- 무선네트웍의 종류: USB 어댑터가 802.11a만 지원하고 AP가 802.11g만 지원하면 접속 불가

USB 무선랜 어댑터를 시스템이 인식하기 위해서는 펌웨어가 필요하다. 가장 일반적인 무선랜 펌

웨어는 대부분의 배포판에 기본적으로 포함되어 있어 별도로 설치할 필요가 없지만 배포판 크기를 최소화하기 위해 베포판에 기본으로 포함되지 못한 펌웨어들도 있다. 그럴 경우 수동으로 펌웨어를 설치해 주어야 한다.

USB 무선랜 어댑터를 USB 포트에 연결한다. 정상적으로 인식이 되는 경우 ifconfig명령을 이용해 wlan0 인터페이스를 확인할 수 있다.

그림 2-4 - 버팔로 무선랜 어댑터가 인식된 것을 확인



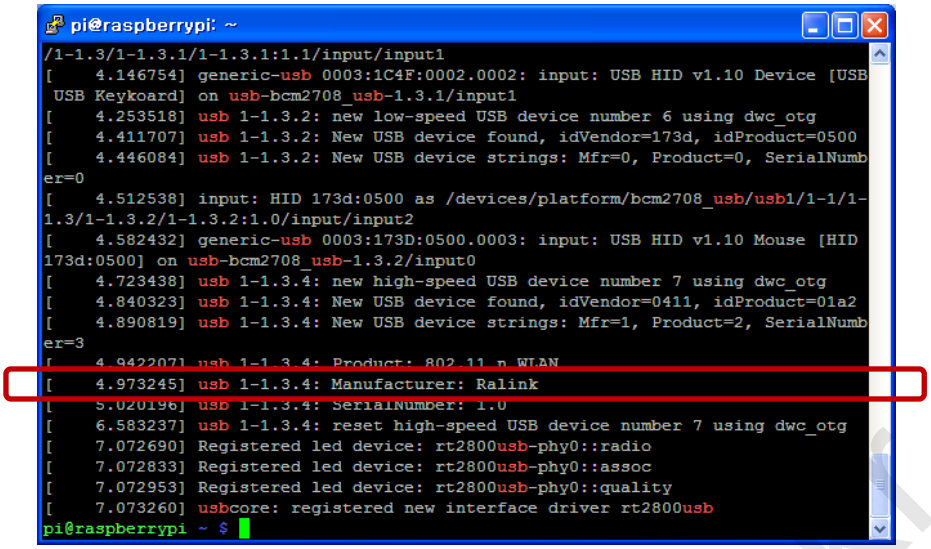
인식이 되지 않는 경우에는 링버퍼에 출력되는 오류 메시지를 확인한다.

```
$ dmesg | grep ^usb          ← usb로 시작되는 행을 찾는다.
```

또는 그냥 usb가 포함된 행을 찾는다.

```
$ dmesg | grep usb
```

그림 2-5 - 무선랜 어댑터가 장착된 USB를 커널 링버퍼에서 검색



Manufacturer를 찾아 어댑터에 장착되어 있는 칩셋 제조사를 확인한다.

경고: 결과를 찾을 수 없으면 firmware, wlan 또는 wireless를 검색해 본다. 그래도 찾지 못하면 lsusb 명령으로 모든 USB 장치를 찾아본다.

apt-cache 명령으로 펌웨어 파일을 찾는다.

```
$ apt-cache search ralink
```

apt-cache 명령으로 찾을 수 없으면 다음 목록을 이용하여 유추해 본다.

- atmel-firmware - Atmel AT76C50X chipset
- firmware-atheros - Atheros chipsets
- firmware-brcm80211 - Broadcom chipsets
- firmware-intelwimax - Intel WiMAX chipsets
- firmware-ipw2x00 - Intel Pro Wireless adapters (2100, 2200, 2915 포함)
- firmware-iwlwifi – 기타 Intel wireless adapters (3945, 4965, 5000 시리즈 포함)
- firmware-ralink - Ralink chipsets
- firmware-realtek - Realtek chipsets
- zd1211-firmware - ZyDAS 1211 chipset

www.ketti.or.kr

유선랜으로 인터넷에 연결되어 있으면 다음 커맨드로 펌웨어를 설치

$ sudo apt-get install firmwarepackage

네트웍 연결이 없는 경우 다른 컴퓨터에서

구글 검색창에 firmaware-realtek Debian wheezy 입력 검색

확장자가 .deb인 펌웨어 설치 파일을 받아 SD카드의 /home/pi에 복사

라즈베리파이에 SD카드를 넣고 부팅 후

$ cd ~

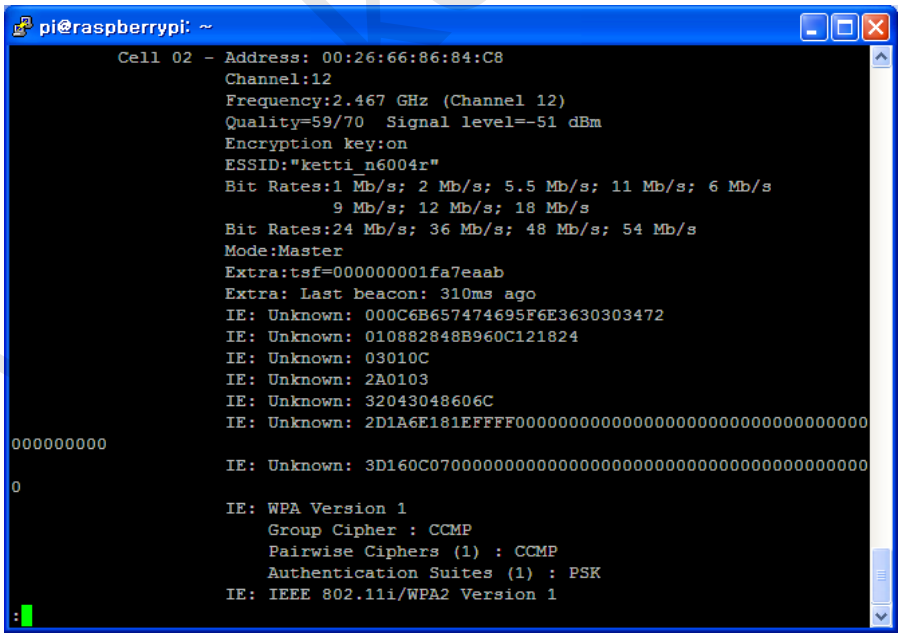
$ sudo apt-get install firmware-ralink_0.35_all.deb

펌웨어 설치 후 USB 무선랜 카드를 분리했다가 다시 연결한다.

주변의 무선랜 액세스 포인트를 검색한다.

$ sudo iwlist scan | less ← 스크린 단위로 멈춘다.

그림 2-6 - iwlist로 무선 네트웍을 검색



이 시점에서 네트웍 인터페이스가 다운되었다거나 하는 등의 오류를 접하게 되면 올바른 펌웨어를 설치했는지 점검해보고 USB 무선랜 어댑터가 유전원 USB 허브에 연결되어 있는지를 확인한

다.

iwconfig명령으로 무선랜 네트웍 인터페이스 상태를 조회하거나 설정 가능하다.

iwconfig 명령 출력에 보이는 항목들의 의미는 다음과 같다.

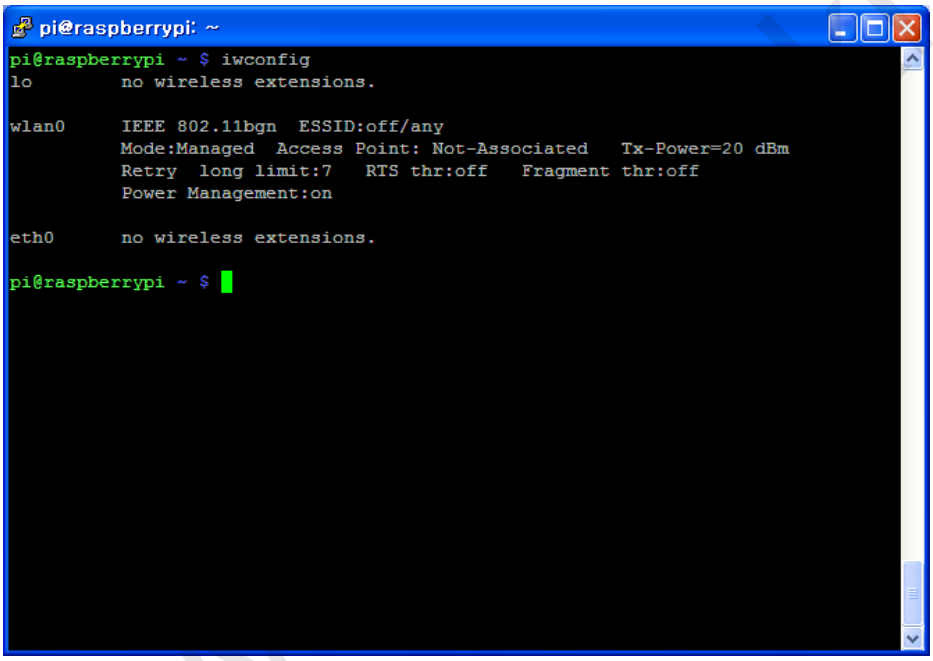
- Interface Name - 각 네트웍 장치의 이름. 무선 장치가 아니면 상세 정보가 표시되지 않는다. 라즈베리파이에서 무선 연결의 기본 명칭은 wlan0 이다.
- Standard - 무선랜 표준인 IEEE 802.11표준은 다양한 변형이 존재하며 뒤에 붙는 알파벳으로 구별된다. 이 곳에는 장작된 USB 무선랩 어댑터가 지원하는 표준의 목록이 표시된다. 예시된 어댑터가 지원하는 표준은 IEEE 802.11bgn으로 3가지 표준을 지원한다.
- ESSID – 어댑터가 연결된 네트웍의 SSID. 연결되지 않은 경우에는 off/any로 표시된다.
- Mode – 어댑터가 동작중인 모드. 다음 목록 중의 하나.
  - Managed – 표준 무선 네트웍, 클라이언트들이 무선 액세스 포인트에 접속. 대부분의 사무실과 가정의 환경이 여기에 해당.
  - Ad-Hoc - 장치 대 장치 무선 네트웍, 무선 액세스 포인트가 없음.
  - Monitor – 어댑터가 식별이 가능하든 하지 않든 모든 트래픽에 귀를 기울이고 있는 특별한 모드. 주로 무선 네트웍 트래픽을 수집하는 네트웍 문제 해결에 사용된다.
  - Repeater – 어댑터를 다른 네트웍 클라이언트에 트래픽을 전달하는 신호 증폭기로 동작시킴
  - Secondary – Repeater 모드의 하위셋. 어댑터를 백업 리피터로 동작시킨다.
- Access Point – 무선랜 어댑터가 연결되어 있는 액세스 포인트의 주소. 무선랜 어댑터가 연결되어 있는 않은 경우 Not-Associated로 표시된다.
- Tx-Power – 무선랜 어댑터를 송신 강도. 숫자가 클수록 신호 강도가 세다.
- Retry – 무선랜 어댑터의 재전송 횟수. 절충형 네트웍 프로토콜에 사용된다. 설정 값 변경할 필요가 거의 없으며 일부 어댑터는 아예 변경이 불가.
- RTS – 혼잡한 네트웍에서 충돌을 방지하기 위한 RTS(Ready To Send)/CTS(Clear To Send) 핸드쉐이킹을 위한 어댑터의 현재 설정값. 이 값은 보통 액세스 포인트 연결 시 액세스

www.ketti.or.kr

포인트에 의해 설정된다.

- Fragment – 혼잡한 네트웍에서 패킷을 다수의 조각으로 나눌 때의 최대 조각 수. 이 값은 보통 액세스 포인트 연결 시 액세스 포인트에 의해 설정된다.
- Power Management – 어댑터가 놀고 있을 때 소비 전력 절감을 위한 어댑터의 전력 관리 기능의 현재 상태. 배터리로 구동되는 랩톱에서 주로 활성화되며 라즈베리파이에는 거의 무효하다.

그림 2-7 - 무선 네트웍에 연결되지 않은 상태에서의 iwconfig 출력



라즈베리파이를 무선 네트웍에 연결하려면 /etc/network/interfaces파일에 무선랜 설정 추가 필요

$ sudo nano /etc/network/interfaces

유선랜 설정 밑에 다음 내용을 추가

auto wlan0

allow-hotplug wlan0

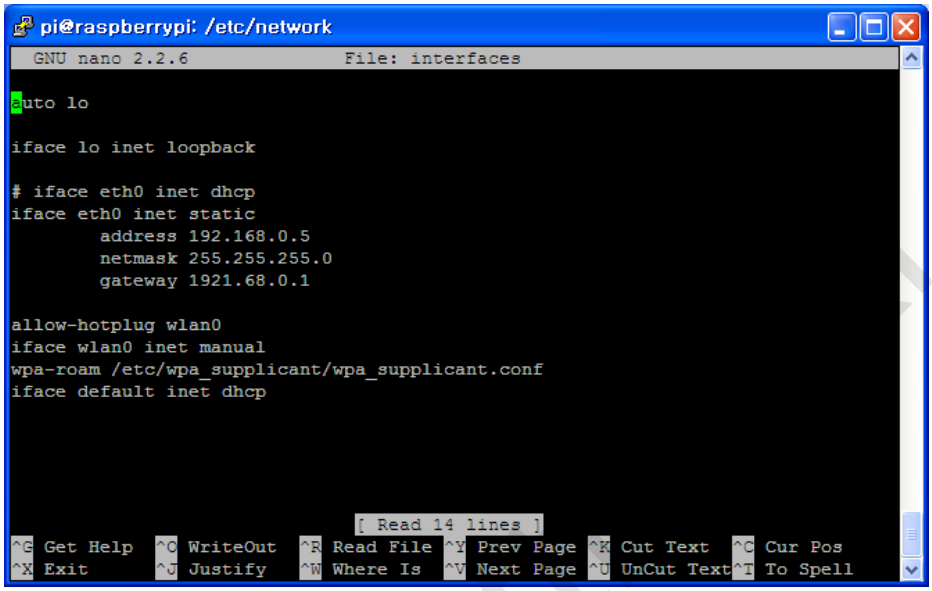
iface wlan0 inet manual

wpa-roam /etc/wpa_supplicant/wap_supplicant.conf

iface default inet dhcp

저장 후 빠져 나온다.

그림 2-8 - 무선 네트워 액세스를 위한 interfaces 파일 수정



wpa-roam으로 시작하는 행에서 참조하고 있는 /etc/wpa_supplicant/wpa_supplicant.conf파일은 WPA(Wireless Protected Access) 암호화를 사용하는 네트웍에 쉽게 접속할 수 있도록 고안된 wpasupplicant에 의해 이용된다.

이름과 달리 wpasupplicant 유틸리티는 WPA는 물론 WPA2, AES(Advanced Encryption Standard), TKIP(Temporal Key Integrity Protocol), 비교적 오래된 암호화 방식인 WEP(Wired Equivalent Privacy) 등 거의 모든 무선 네트웍에 사용할 수 있다.

wpa_supplicant.conf파일을 열어 맨 밑에 다음 내용을 추가한다.

$ cd /etc/wpa_supplicant

$ sudo nano wpa_supplicant.conf

network={

[Tab] ssid="Your_SSID"

암호화를 사용하지 않는 경우

[Tab] key_mgmt=NONE

}

WEP 암호화를 사용하는 경우

[Tab] key_mgmt=NONE

[Tab] wep_key0="Your_WEP_Key"

}

경고: WEP 암호화는 그다지 안전하지 못하다. 손쉽게 입수할 수 있는 소프트웨어를 써서 몇 분이면 암호를 깰 수 있다. WPA 또는 WPA2 암호화를 사용할 것을 권한다.

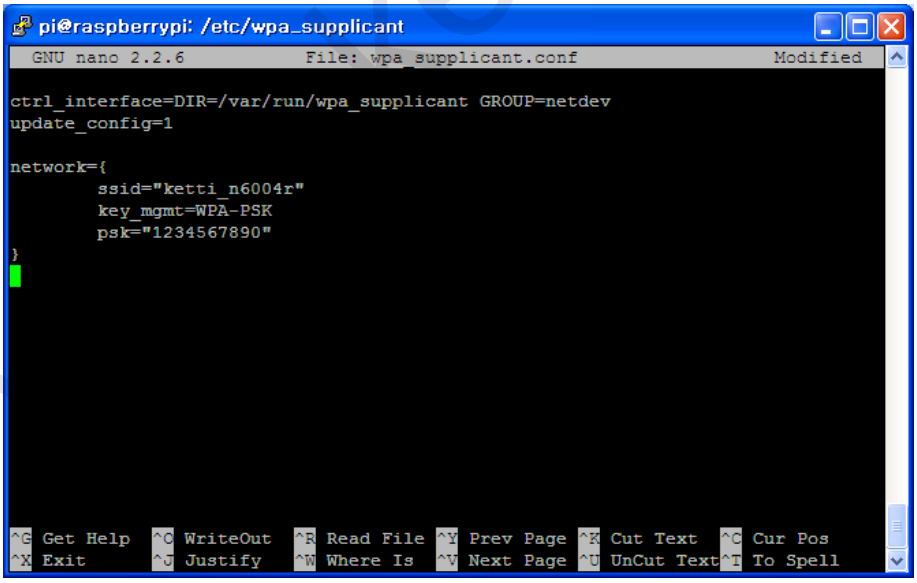
WPA/WPA2 암호화를 사용하는 경우

[Tab] key_mgmt=WPA-PSK

[Tab] psk="Your_WPA_Key"

}

그림 2-9 - WPA 암호화된 네트워을 위한 wpa_supplicant.conf 파일 편집



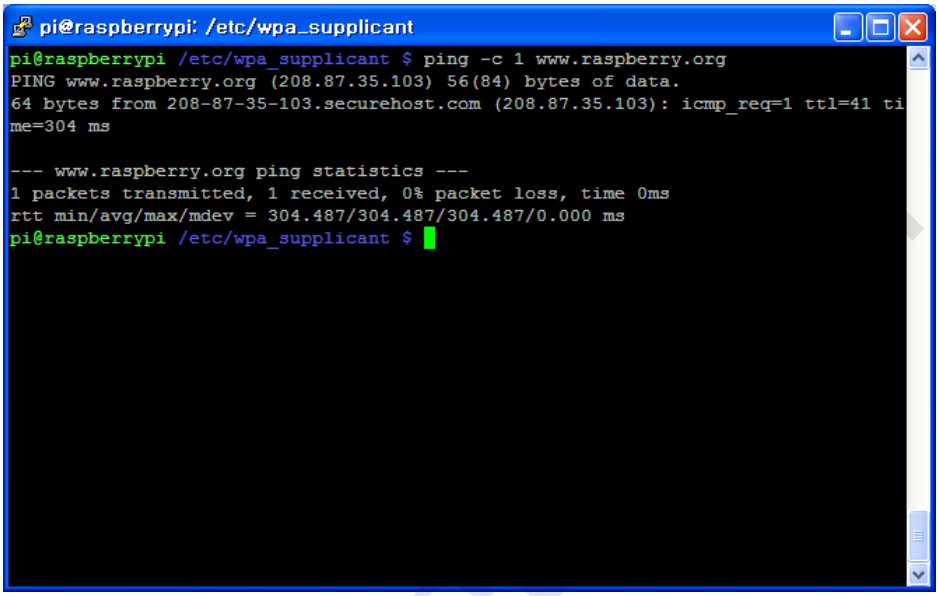
이제 라즈베리파이를 재부팅하면 무선 네트웍이 기동될 것이다. 재부팅하지 않고 무선네트웍만 활성화 하려면 다음과 같이 한다.

$ sudo ifup wlan0

무선랜이 동작하는 것을 확인하려면, 유선랜 케이블이 연결되어 있는 경우 랜 케이블을 빼고 다음과 같이 쳐본다.

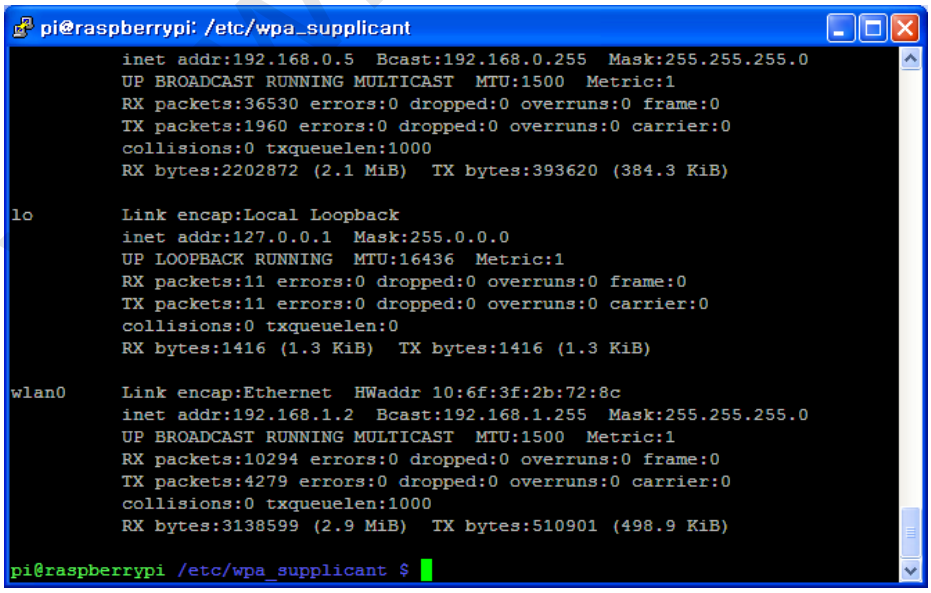
$ ping –c 1 www.raspberrypi.org

그림 2-10 - 무선랜 연결 확인



ifconfig명령을 이용하여 무선랜카드의 IP부여 여부를 확인해볼 수 있다.

그림 2-11 - 무선랜 카드의 IP 부여 여부 확인



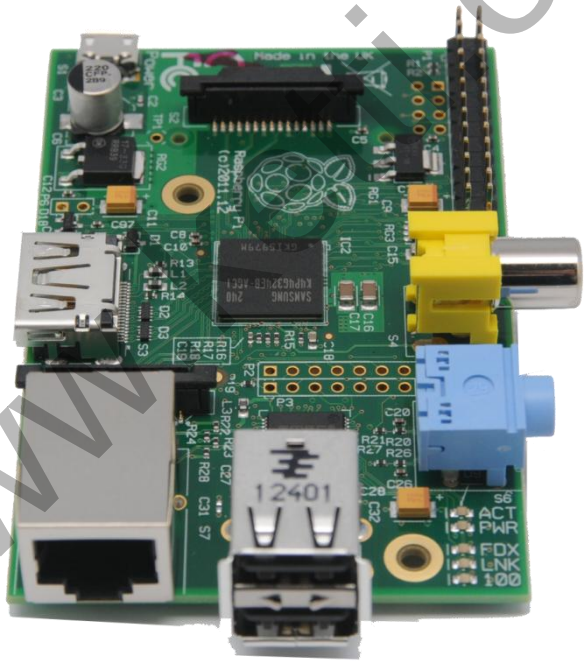
경고: 무선랜 설치가 완료된 후에 다시 문제가 발생하는 경우 다른 USB 장치와의 호환성 문제일 가능성도 있다. 일부 무선랜 어댑터들은 특정 USB 키보드들과 호환성에 문제가 있음이 알려져 있다. 문제가 없는 어댑터 목록: http://www.element14.com/community/docs/DOC-44703/l/raspberry-pi-wifi-adapter-testing 과 문제가 있는 어댑터 목록: http://elinux.org/RPi_VerifiedPeripherals#Working_USB_Wifi_Adapters 을 참조할 것.

본 문서는 자유롭게 배포 가능합니다.
다만 상업적 목적 및 문서 수정 배포시
한국전자통신교육원(kks@ketti.or.kr)로 문의 부탁드립니다.

www.ketti.or.kr

라즈베리파이를 활용한 오픈 소스 R&D 플랫폼

(6,7,8장)



www.ketti.or.kr

한국전자통신교육원

본 문서는 자유롭게 배포 가능합니다.
다만 상업적 목적 및 문서 수정 배포시
한국전자통신교육원(kks@ketti.or.kr)로 문의 부탁드립니다.

목차

목차 1

Chapter 1. 라즈베리파이 소개 6

1. 프로그래밍의 재미 6

2. 라즈베리파이의 역사 6

3. 라즈베리파이란? 7

4. 라즈베리파이로 무엇을 할 수 있나? 8

Chapter 2. 라즈베리파이와의 첫 만남 10

1. ARM 대 x86 10

2. 리눅스 대 윈도우 11

3. 라즈베리파이에 포팅 된 운영체제 11

4. 주변기기 연결 12

5. SD카드에 운영체제 이미지 써넣기 15

6. USB저장 장치 연결 16

7. 네트워 연결 16

8. 전원 연결 17

Chapter 3. 리눅스 시스템 관리 18

1. 리눅스 개요 18

2. 리눅스 기초 18

3. 데비안(Debian) 소개 19

4. 외부 저장 장치 사용 22

5. 사용자 계정 만들기 23

www.ketti.or.kr

6. 파일 시스템 구성 24

7. 소프트웨어 설치 및 제거 26

Chapter 4. 문제 해결 30

1. 키보드 및 마우스 진단 30

2. 전원 진단 30

3. 디스플레이 진단 32

4. 부팅 진단 32

5. 네트웍 진단 33

6. 비상용 커널 35

Chapter 5. 네트웍 설정 37

1. 유선 네트워킹 37

2. 무선 네트워킹 40

Chapter 6. 파티션 관리 50

1. 새 파티션 작성 50

2. 파티션 크기 변경 53

3. 수작업으로 파티션 크기 변경 54

4. 더 큰 SD카드로 옮기기 58

Chapter 7. 하드웨어 구성 설정 61

1. 하드웨어 설정 – config.txt 61

2. 디스플레이 설정 변경 62

3. 부팅 옵션들 65

4. 라즈베리파이 오버 클럭킹 65

www.ketti.or.kr

5. 메모리 파티셔닝 – start.elf 68

6. 소프트웨어 설정들 – cmdline.txt 68

Chapter 8. 홈씨어터로 활용 70

1. Rasbmc를 이용한 전용 HTPC 구축 70

2. 인터넷 미디어 스트리밍 71

3. 로컬 네트웍 미디어 스트리밍 73

4. Rasbmc 설정 변경 75

Chapter 9. 업무용으로 활용 77

1. 클라우드 기반 앱 사용 77

2. 오픈오피스 사용 78

3. Gimp를 이용한 이미지 편집 79

Chapter 10. 웹 서버로 활용 81

1. LAMP 스택 설치 81

2. WordPress 설치 84

Chapter 11. 스크래치로 프로그래밍 89

1. 예제 1: Hello World 89

2. 예제 2: 애니메이션 및 사운드 93

3. 예제 3: 간단한 게임 95

4. 로보틱스 및 센서들 100

5. 더 읽어보기 101

Chapter 12. 파이썬 프로그래밍 102

1. 파이썬 소개 102

www.ketti.or.kr

2. 예제 1: Hello World 102

3. 파이썬 프로그램 실행 가능하게 만들기 106

4. 예제 2: 주석, 입력, 변수 및 반복문 107

5. 예제 3: pygame을 이용하여 게임 만들기 110

6. 파이썬과 네트워킹 119

7. 더 읽어보기 124

Chapter 13. 하드웨어 해킹 126

1. 전자 부품 126

2. 저항 값 읽기 128

3. GPIO 포트 129

4. 파이썬에서 GPIO 사용하기 132

5. GPIO 출력: LED 밝히기 134

6. GPIO 입력: 버튼 읽기 138

7. 브레드보드 벗어나기 141

8. 납땜에 대한 간단한 가이드 144

참조 A: Python Recipes 149

1. Raspberry Snake (Chapter 11, Example 3) 149

2. IRC User List (Chapter 11, Example 4) 152

3. GPIO Input and Output (Chapter 12) 155

참조 B: HDMI 디스플레이 모드 목록 156

참조 C: 참고 문헌 163

1. 참고 문헌 163

www.ketti.or.kr

2. 참고 사이트.....163

www.ketti.or.kr

Chapter 6. 파티션 관리

SD카드에 운영체제 이미지가 담겨서 제공된 라즈베리파이를 사용하는 것은 매우 편리하다. 하지만 다소 유연성이 떨어진다. 대부분의 배포 이미지는 2GB 또는 4GB면 충분하다. 즉, 8GB 이상의 SD 카드를 가진 사용자는 대부분의 나머지 공간을 낭비하게 되는 것이다.

1. 새 파티션 작성

SD카드의 뒷 부분에 남아 있는 빈 공간을 활용하는 방법 한 가지는 그 곳에 새로운 파티션을 만드는 것이다.

모든 작업을 SD카드를 PC에 옮겨 꽂을 필요 없이 라즈베리파이에서 직접 할 수 있다.

경고: 디스크의 파티션을 수정하는 도구들을 사용할 때에는 기존에 존재하는 파티션들을 지우지 않도록 주의해야 한다. 특히 /boot 또는 루트 파일 시스템을 지우지 않도록 주의한다. 지울 경우 부팅이 불가능해질 수 있다.

SD카드에 새 파티션을 만드는 것은 다음과 같은 순서로 작업한다.

1. 터미널 창에서 다음을 입력

   $ sudo fdisk –l

   라즈베리파이에 연결된 저장장치와 그 안의 파티션 목록이 표시된다. SD 카드는 /dev/mmcblk0로 표시되고 파티션들은 p0, p1 으로 번호가 매겨져 표시된다.

2. 메뉴 기반 파티션 관리 도구인 cfdisk를 실행

   $ sudo cfdisk /dev/mmcblk0

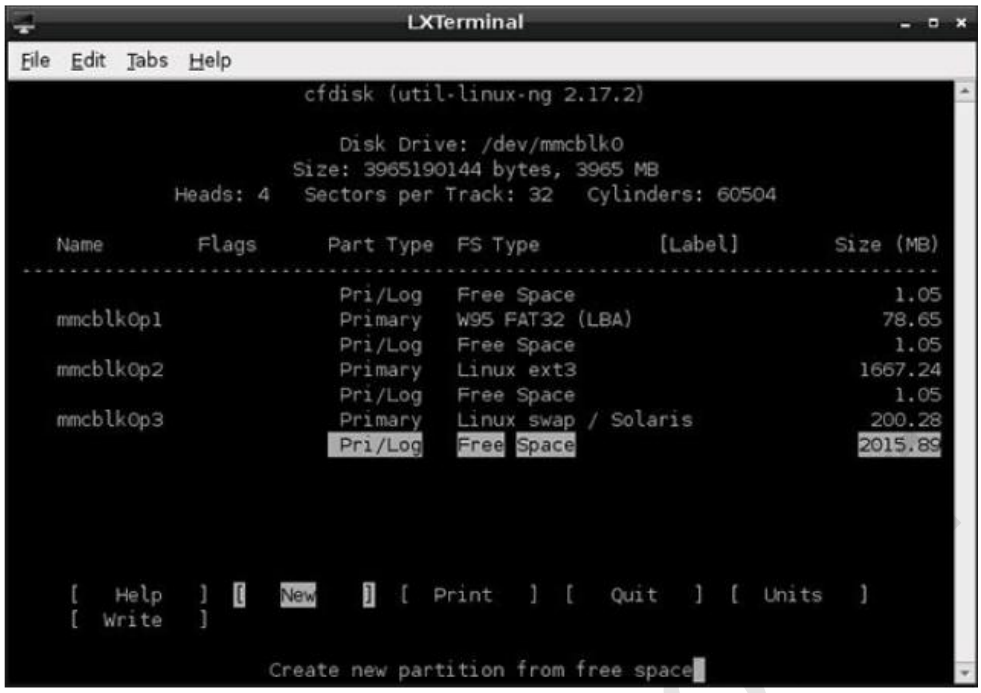
   타겟 장치로 SD카드를 지정하였다. 만약 USB 저장 장치를 사용하려고 하는 경우에는 장치명을 해당 장치로 지정하면 된다. (예를 들면 sda)

3. 상하 커서 키로 반전 막대를 움직여 맨 아래의 Free Space 행을 선택한다.

4. 좌우 커서 키로 화면 하단의 [New]를 선택하고 Enter 키를 입력한다.

그림 6-1 - cfdisk를 이용하여 새 파티션 만들기

www.ketti.or.kr



5. Primary 파티션인지 Logical 파티션인지를 묻는다. 기본값은 Primary 파티션이다. Enter키를 누른다.

6. 파티션 크기를 몇 메가바이트(MB)로 할 것인지 묻는다. 기본은 남은 여유 공간을 모두 차지하는 크기로 만드는 것이다. Enter 키를 누른다.

7. 좌우 커서키를 움직여 [Write] 옵션을 선택하고 Enter 키를 누르면 파티션 테이블이 SD카드에 기록된다.

8. 변경 사항이 맞는지 재차 확인한다. 맞는지 확인 후 yes를 입력하고 Enter 키를 친다.

9. 좌우 커서키를 움직여 [Quit] 옵션을 선택하고 Enter 키를 쳐 cfdisk를 빠져 나온다.

SD카드에 새로운 파티션이 만들어 졌지만 재부팅할 때까지는 운영체제에 인식되지 않는다. 인식시키는 가장 쉬운 방법은 재부팅을 하는 것이다.

```
$ sudo reboot
```

재부팅 후 다시 로그인 하고 fdisk 명령으로 새로 생긴 파티션을 확인해 볼 수 있다.

```
$ sudo fdisk –l
```

새 파티션을 사용하기 위해서는 포맷팅을 통해 파일 시스템을 만들어 주어야 한다.

www.ketti.o

$ sudo mkfs.ext4 /dev/mmcblk0p3

mkfs.ext4는 리눅스의 기본 파일시스템 타입인 ext4타입을 파일 시스템을 만드는 명령이다.

새로운 파티션 명칭은 mmcblk0p3이기 쉽지만 위의 fdisk –l 명령으로 확인한 후 지정한다.

mkfs명령은 파티션상의 모든 데이터를 삭제하므로 주의하여 실행한다.

새로운 파일 시스템을 기존 사용중인 파일시스템상의 어느 한 지점에 마운트(연결)되어야 한다. 다음과 같이 빈 디렉터리를 만들고 마운트 한다.

$ sudo mkdir /storage

$ sudo mount /dev/mmcblk0p3 /storage

chown 명령을 사용하면 디렉터리의 소유자를 다른 사용자로 바꿀 수 있다

$ sudo chown –R <userid> <directory>

chgrp 명령을 사용하면 그룹에 속한 모든 사용자가 디렉터리를 액세스 할 수 있다.

$ sudo chgrp –R users /storage

chmod 명령을 사용하면 그룹에 속한 모든 사용자가 디렉터리에 쓰기 권한을 가지도록 할 수 있다.

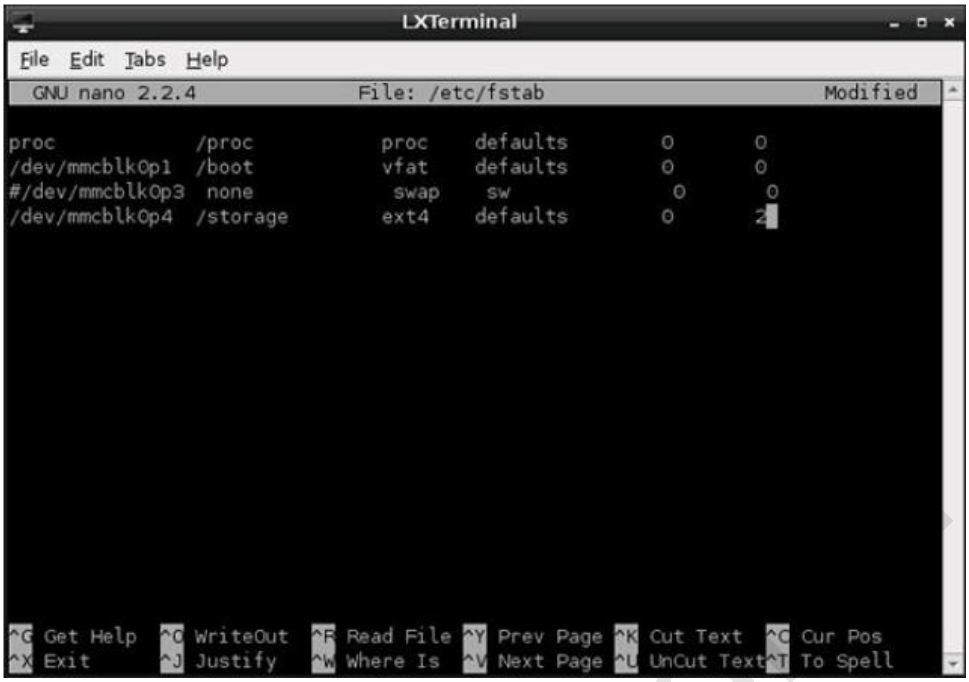
$ sduo chmod –R g+w /storage

마운트된 파티션은 라즈베리파이가 재부팅 되면 다시 언마운트(연결 해제)된다. 부팅 때 마다 자동으로 마운트 되도록 하기 위해서는 /etc/fstab 파일에 다음과 같은 형식으로 마운트할 파티션을 지정해 주어야 된다.

/dev/mmcblk0p3 /storage ext4 defaults 0 2 ← 각 필드 사이는 탭 키

그림 6-2 - 새 파티션 자동 마운트를 위한 fstab파일 편집

www.ketti.or.kr



재부팅 후 mount 명령으로 마운팅 여부를 확인 할 수 있다.

```
$ sudo mount
```

2. 파티션 크기 변경

SD카드의 빈 공간을 이용하는 더 좋은 방법은 기존 파티션의 크기를 변경하는 방법이다. 이 방법을 안전해가 하려면 라즈베리파이에서 SD카드를 꺼내어 PC에 연결하고 해야 된다.

자동 크기 변경

라즈베리파이를 위한 데비안 리눅스 배포 이미지에는 raspi-config라고 하는 유틸리티가 포함되어 있다. 이 유틸리티는 처음 부팅되었을 때 자동으로 실행되는데 콘솔 또는 터미널 창에서 sudo raspi-config 라고 치면 언제든지 실행 할 수 있다. 이 도구는 여러가지 설정작업을 할 수 있도록 하는 기능이 제공되는데 SD카드의 남은 공간을 모두 루트 파일 시스템이 사용하도록 자동으로 확장할 수 있는 기능을 제공한다.

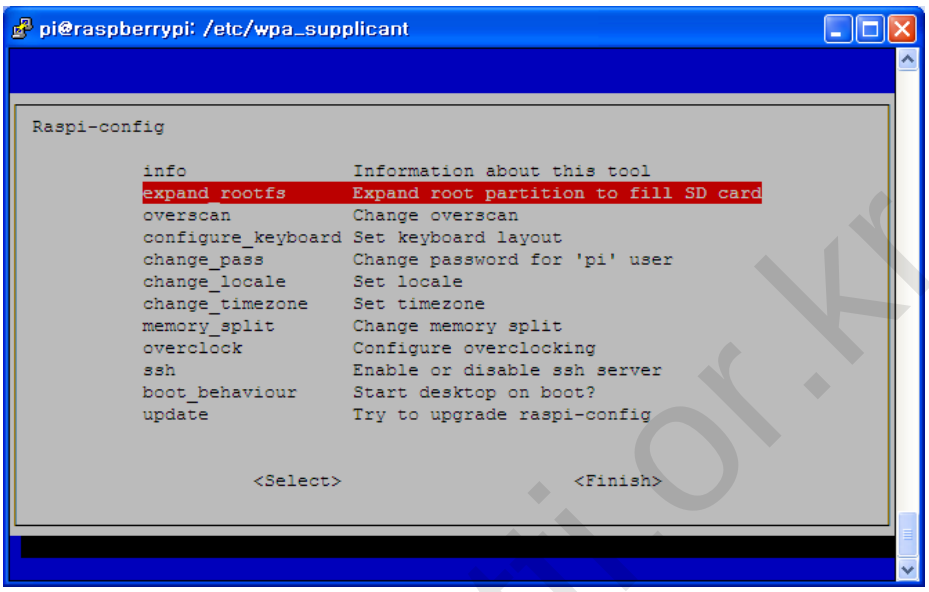
주의: raspi-config 를 사용하여 자동 루트 파일 시스템을 확장하는 기능은 간혹 데이터를 망가뜨리는 수 가 있다. 중용한 데이터가 있는 경우에는 반드시 백업을 하고, 되도록이면 이 장의 뒤에서 소개하는 수작업에 의한 방법을 사용하도록 한다.

터미널 창 또는 콘솔창에서

$ sudo raspi-config

상하 커서키로 expand_rootfs를 선택하고 Enter 입력

그림 6-3 - raspi-config 도구의 메뉴 화면



파티션 크기 변경 작업이 수 초간 진행된 후 라즈베리파이 재부팅 후 작업이 완결된다는 메시지가 나오면 Enter 키를 입력한다.

탭키를 두 번 쳐서 <Finish>를 선택하고 Enter키를 입력하여 빠져 나온다.

재부팅 한다.

$ sudo reboot

파티션 크기 변경작업 완료 후 재부팅이 완료된다.

변경된 파티션 크기를 확인한다. 콘솔 또는 터미널 창에서

$ df –h ← -h 옵션은 human friendly 의미

3. 수작업으로 파티션 크기 변경

가장 안정적인 방법은 PC에서 부팅 가능한 무료 CD인 Parted Magic 을 사용하는 방법이다. PC와 맥에서 모두 동작하며 온전히 메모리 상에서 동작한다. 또한 데비안 종속적인 raspi-config 도구와는 달이 모든 라즈베리파이 배포판과 호환된다.

www.kett.or.kr

팁: 리눅스 사용자는 CD를 사용하여 부팅하는 대신 Parted Magic에 사용되는 그래픽 파티셔닝 도구인 gparted를 설치하여 사용할 수 있다.

```
$ sudo apt-get install gparted

$ sudo gparted
```

PC에서 Parted Magic ISO 이미지를 다운로드한다:

http://partedmagic.com

CD 버닝 프로그램을 이용하여 빈 CD에 기록한다.

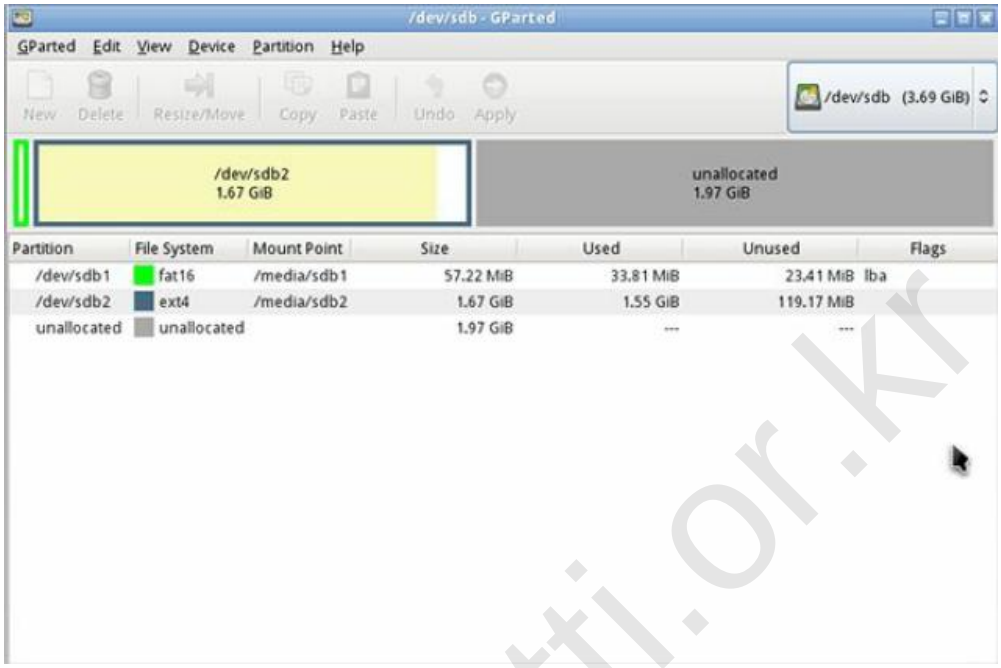
구워진 CD를 드라이브에 넣은 상태에서 PC를 리부팅하면 Parted Magic 메뉴 시스템이 로드 되고 Standard Settings를 선택한다.

Parted Magic는 커스터마이즈된 리눅스 운영체제로써 저장장치 관리를 위해 고안된 것이다. SD카드를 컴퓨터에 넣고 바탕화면의 Partition Editor 아이콘을 더블 클릭하여 실행한다.

그림 6-4 - Parted Magic 데스크톱 화면

Partition Editor는 기본적으로 시스템의 첫 번째 드라이브(주로 하드드라이브)를 찾는다. 오른쪽 위에 있는 장치 선택 버튼을 클릭하여 SD 카드를 선택(주로 /dev/sdb) 한다.

그림 6-5 - Parted Magic의 파티션 편집 도구 화면



주의: 파티션을 옮기거나 크기를 변경하는 작업은 매우 위험한 작업이다. 작업 도중 SD카드가 제거되거나 전원이 꺼지면 카드의 데이터가 깨질 수 있다. 파티션을 수정하기 전에 중요한 데이터는 반드시 백업을 해두어야 한다.

배포판에 따라 작업할 정확한 파티션을 선택한다. 데비안의 경우, 두 번째 파티션의 크기를 변경하고자 할 것이다(일반적으로 sdb2). 파티션 에디터가 로드된 상태에서 다음 순서로 작업한다.

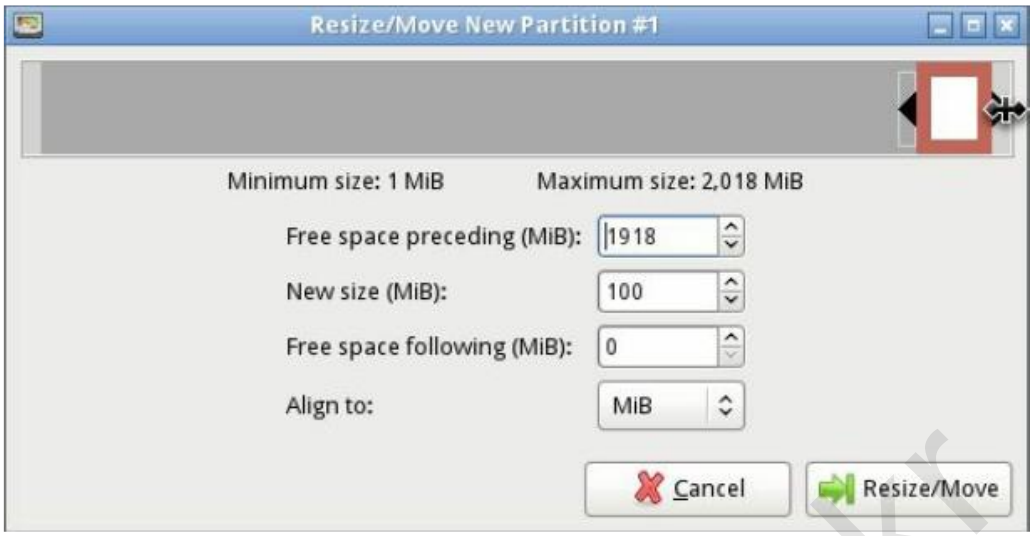
1. 어떤 리눅스 배포판들은 이미지의 맨 마지막 부분에 스왑 파티션을 가지고 있다. 파티션 에디터에서는 linux-swap 타입의 작은 파티션으로 표시된다. 스왑 파티션이 없으면 5번으로 건너뛴다.

2. 목록의 마지막에 있는 스왑 파티션을 클릭한 후 도구 막대에서 Resize/Move를 선택한다.

3. 대화상자가 표시되면 왼쪽 위의 박스를 선택하여 오른쪽 위 끝으로 끌어다 놓는다. Free Space Flowing 상자에는 있는 값이 0이 된다.

그림 6-6 - 파티션 에디터에서 스왑 파티션 이동

www.ketti.or.kr



4. 오른쪽 아래단의 Resize/Move 버튼을 클릭한다. 변경된 새로운 파티션 테이블은 부팅에 지장을 초래할 수 도 있다는 경고메시지가 나오지만 무시해도 된다. 부팅 파티션에 손을 댄 것이 아니기 때문이다. 그냥 OK를 클릭한다.

5. 목록에서 가장 큰 파티션을 클릭한다. sdb2라고 되어 있는 파티션이다. 도구 모음에 Resize/Move 를 다시 클릭한다.

6. 이번에는 파티션의 오른쪽에 있는 화살표를 클릭하여 맨 오른쪽까지 끌고 간다. 이렇게 하면 파티션의 크기를 최대로 늘리게 된다.

그림 6-7 - 파티션 에디터에서 루트 파일 시스템 크기 변경

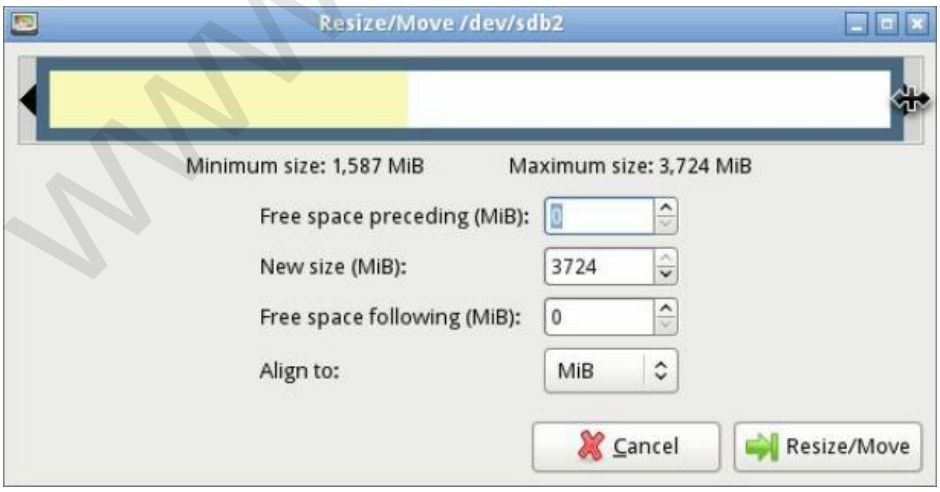


그림 6-8 - 파티션 편집 도구에서 파티션 크기가 변경된 후 화면

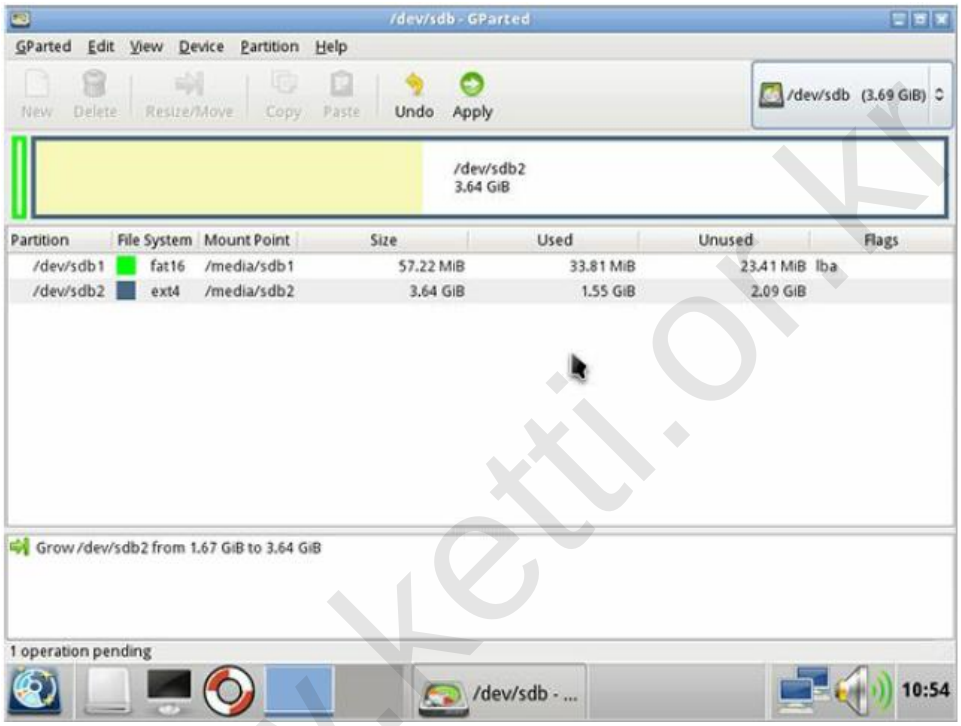
7. Resize/Move 버튼을 클릭한다. 경고 박스가 나타나면 OK를 클릭한다.

www.ketti.or.kr

8. 도구 막대에서 Apply를 클릭한다. 경고박스가 나타나면 다시 한번 Apply 클릭. SD카드의 속도에 따라 파티션 크기 변경 작업에 수 분이 소요된다.

9. 작업이 완료되면 PC를 꺼도 되고 SD카드를 꺼내어 다시 라즈베리파이에 넣는다.

이제 파티션 에디터를 이용하여 대부분의 리눅스 배포판에서 시스템 파일들과 사용자 데이터를 모두 담는데 사용되는 SD카드의 주 파티션을 최대로 활용할 수 있게 되었다.



4. 더 큰 SD카드로 옮기기

라즈베리파이를 어느 정도 사용했다면 처음에는 충분하리라 여겼던 4GB SD카드가 꽉 차가고 있을 것이다. 8GB나 16GB 또는 더 큰 SD카드도 충분히 싼 가격에 구입할 수 있다. 하지만 사용하던 파일들을 잃고 싶지는 않을 것이다.

SD카드에 있는 컨텐츠를 새로운 SD카드로 옮기는 작업은 아주 간단하다. 작업의 뒤쪽 반은 "라즈베리파이 만나보기" 장에서 이미 경험했던 SD카드에 리눅스 배포판 이미지를 굽는 것과 동일하다. 나머지 다른 점은 사용하던 SD카드의 컨텐츠를 이미지 파일로 복제하는 작업이다.

경고: SD카드를 복제하는 작업은 기존의 카드의 내용을 훼손하지는 않는다. 제 3자에게 사용하던 SD카드를 건네줄 경우에는 모든 데이터를 삭제할 필요가 있다. 새 SD카드에서는 이 장 앞에서 살펴본 것처럼 파티션 크기를 변경할 필요가 있다.

www.ketti.or.kr

더 큰 SD카드로 옮겨가는 과정의 첫 번째 작업은 기존 카드의 이미지를 추출하는 것이다. 두 개의 SD카드를 동시에 사용할 수 있는 카드리더기를 가지고 있다면 후 반 작업에서 데이터 소스 부분에 이미지 파일 대신 사용하던 SD카드를 지정하면 된다.

윈도우 상에서 이미지 추출

"라즈베리파이 만나보기" 장에서 사용했던 Win32DiskImager 유틸리티를 사용하거나 Parted Magic CD에서 제공하는 dd 유틸리티를 사용한다. dd를 사용하는 방법은 다음과 같다.

1. Parted Magic CD를 PC에 넣고 재부팅한다. Standard Settings를 선택한다.
2. 바탕화면의 세 번째에 있는 모니터 모양으로 생긴 아이콘을 더블 클릭하여 터미널 창을 실행한다.
3. 하드디스크와 SD카드 장치 이름을 확인한다.

   $ fdisk –l

   하드디스크는 /dev/sdXN, SD카드는 /dev/mmcblkX 형태로 나타난다. 하드디스크는 크기로 유추한다.
4. 하드디스크를 마운트 한다.

   $ mkdir /media/harddrive

   $ mount /dev/sdXN /media/harddrive –o=rw
5. 사용하던 SD카드를 카드리더기에 넣고 다음 명령으로 디바이스 노드를 확인한다.

   $ fdisk –l

   /dev/mmcblkX 또는 /dev/sdY 형태로 나타난다.
6. SD카드의 모든 데이터를 이미지 파일로 추출하여 저장한다.

   $ dd of=/media/harddrive/temporaryimage.img if=/dev/sdY bs=2M

그림 6-9 - Parted Magic 의 terminal 아이콘

www.ketti.or.kr

이제 사용하던 SD카드의 이미지 파일을 만들었다. 다음은 "라즈베리파이 만나보기"에서 했던 것처럼 이미지를 새 SD카드에 써 주면 된다.

작업이 완료되면 원래 카드와 새 카드가 동일한 내용을 가지게 된다. 그 의미는 원래 카드가 2GB이면 새 카드가 16GB 카드이더라도 라즈베리파이는 2GB까지만 접근할 수 있다는 뜻이다. 나머지 용량을 모두 사용하기 위해서는 이 장의 앞부분에서 했던 것처럼 파티션 크기 변경 작업을 해 줄 필요가 있다.

www.ketti.

Chapter 7. 하드웨어 구성 설정

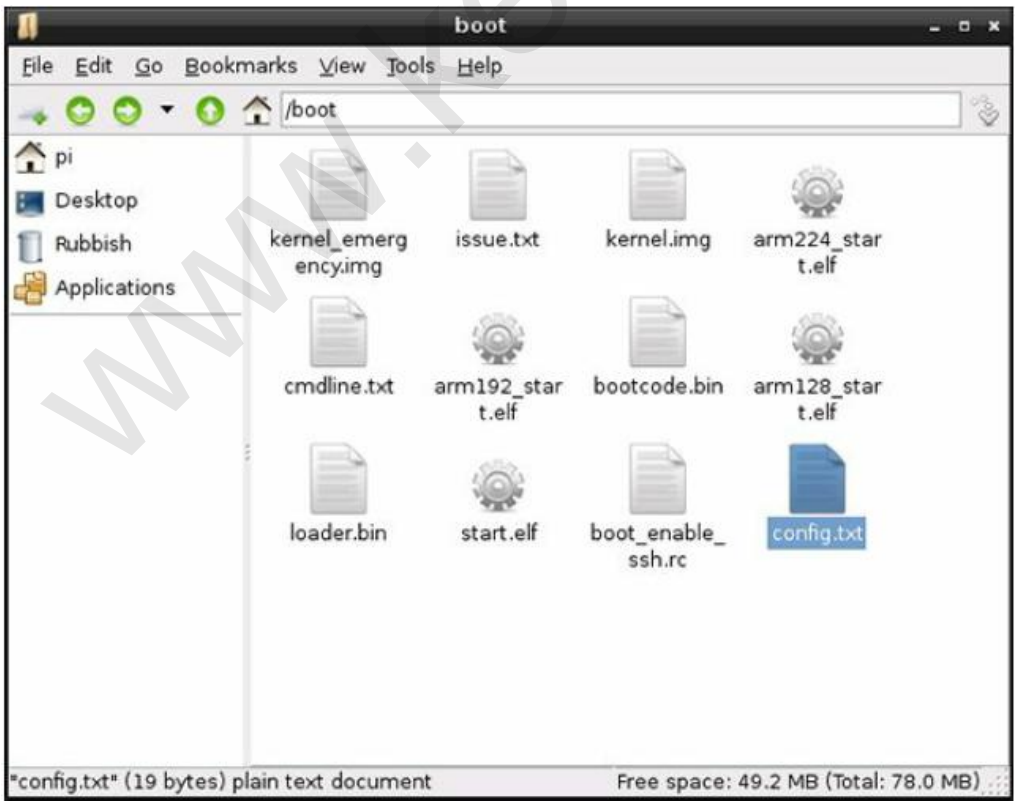
라즈베리파이의 심장부에 있는 BCM2835 칩은 임베디드 컴퓨팅 분야에 사용되던 것이기 때문에 일반 PC에서처럼 다양한 저수준 시스템 설정 기능을 가지고 있는 BIOS 메뉴와 같은 것을 가지고 있지 않다. 대신 전원이 켜졌을 때 칩에 로드되는 설정 값들을 담고 있는 텍스트 파일에 의존한다.

config.txt, cmdline.txt, start.elf와 같은 파일들을 살펴보기 전에 한마디 경고를 덧붙이자면 이 파일들의 설정 값들을 잘못 바꾸는 경우 라즈베리파이가 부팅되지 않거나 최악의 경우에는 시스템에 물리적인 손상을 가할 수도 있다.

1. 하드웨어 설정 – config.txt

라즈베리파이의 하드웨어는 /boot 디렉터리에 있는 config.txt파일에 담겨있는 설정 값들에 의해 제어된다. 이 파일은 라즈베리파이가 다양한 입/출력들을 어떻게 설정하고 BCM2835 칩과 연결된 메모리 모듈이 실행되어야 하는 속도를 알려준다.

그림 7-1 - /boot 디렉터리의 안의 내용, config.txt 파일 선택



www.kettl.or.kr

그래픽 출력이 나오지 않거나 이미지가 스크린에 꽉 차지 않거나 넘치는 문제가 있다면 config.txt 파일을 수정하면 된다. 어떤 배포판에서는 이 파일이 비어 있거나 없는 경우도 있는데 그것은 라즈베리파이가 사전 설정된 기본값을 사용하고 있다는 뜻이다. 값을 변경하고 싶은데 파일이 없으면 그냥 config.txt 라는 이름의 새 텍스트 파일을 만들어주면 된다.

config.txt파일에서는 CPU와 GPU의 메모리 사용비율을 지정하는 부분을 제외한 대부분의 하드웨어 관련 부분을 제어할 수 있다. 메모리 사용 비율 부분은 이장의 뒷부분에 있는 "메모리 파티셔닝 – start.elf"에서 배우게 될 것이다.

config.txt파일은 시스템이 처음 시작될 때 단 한번 읽혀진다. 실행 중에 변경된 내용은 시스템이 재시작할 때까지 반영되지 않는다. 변경된 내용을 되돌리고자 할 경우 단순히 /boot에 있는 config.txt파일을 삭제하면 된다. 값을 새로 변경한 후 라즈베리파이가 부팅되지 않으면 SD카드를 PC에 가져가서 config.txt파일을 지운 후 다시 라즈베리파이에 넣고 부팅하면 된다.

2. 디스플레이 설정 변경

일반적으로 라즈베리파이는 연결된 디스플레이를 감지하여 설정을 그에 따라 자동으로 변경한다. 하지만 때대로 이런 자동 감지가 작동하지 않을 때가 있다. 라즈베리파이가 한 나라의 구형 TV와 연결되어 사용되다가 다른 나라로 가져갔을 때 이런 일이 종종 일어난다. 라즈베리파이를 TV에 연결했는데 아무것도 표시되지 않으면 디 기본값들을 바꿔줘야 할 필요가 있다.

config.txt의 다양한 설정값들이 이러한 비디오 출력 문제를 개선하는데 사용될 수 있다.

경고: HDMI 또는 컴포지트 출력 설정을 수동으로 조정하는 것은 라즈베리파이가 모니터와 소통하는 것을 방해할 수 있다. 화면에 아무것도 출력되지 않을 때 일반적으로 가장 좋은 방법은 라즈베리파이가 설정 값을 자동으로 감지하도록 내버려 두는 것이다.

- overscan_left – 왼쪽 화면을 지정한 픽셀 수만큼 안쪽으로 표시. 표시되는 텍스트가 경계면에서 잘려서 표시된다면 지정된 픽셀 수만큼 지나서 표시되도록 하여 문제를 해결한다.
- overscan_right – 오른쪽
- overscan_top – 윗쪽
- overscan_bottom – 아랫쪽
- disable_overscan – HDMI를 통해 모니터나 TV를 사용하는 경우 표시되는 이미지 주위에 검은색 여백이 생길 수 있다. 이 항목에 값을 1로 지정하여 기본 설정된 오버스캔을 비

www.kotti.or.kr

활성화 함으로써 이 여백을 없앨 수 있다.

- framebuffer_width – 콘솔 화면의 가로 크기를 픽셀 단위로 지정. 콘솔 화면이 디스플레이보다 작게 표시되면 이 값을 더 크게 바꿔준다. 실제 표시되는 콘솔 화면의 크기는 이 값에서 오버스캔 값을 뺀 값이다.
- framebuffer_height – 콘솔 화면의 세로 크기
- framebuffer_depth – 콘솔 화면의 색상수를 픽셀 당 비트수로 지정. 기본 값은 16 비트/픽셀 (65,536 색상) 이다. 8 비트/픽셀(256 색상), 24 비트/픽셀(16.7 백만 색상), 32 비트/픽셀(약 10억 색상)을 지정할 수 있다. 32 비트/픽셀을 지정하면 실제 이를 지원하는 디스플레이가 아직 흔치 않으므로 화면이 깨질 수 있다.
- framebuffer_ignore_alpha – 1로 설정하면 콘솔 화면의 투명도를 제어하는 알파 채널을 비활성화 한다. 알파 채널 비활성화는 일반적으로 필요하지 않지만 framebuffer_depth를 32 비트/픽셀로 지정했을 때 발생하는 화면 깨짐을 해소할 수 있다.
- sdtv_mode – 이 값은 아날로그 컴포지트 출력이 다양한 국가에 맞추어 동작할 수 있도록 해준다. 기본값으로 라즈베리파이는 NTSC 북미 비디오 표준을 사용한다. 다른 국가에서는 아날로그 TV를 이용하기 위해 이 값을 바꿀 필요가 있을 것이다(한국은 NTSC 사용). 설정 값들은 다음과 같다:
  - 0 – NTSC, 북미 비디오 표준
  - 1 – NTSC-J, 일본 비디오 표준
  - 2 – PAL, 영국 및 다른 국가 (주로 유럽) 비디오 표준
  - 3 – PAL-M, 브라질 비디오 표준
- sdtv_aspect – 아날로그 컴포지트 출력의 가로:세로 출력비를 조절. 콘솔 화면이 가로 또는 세로로 늘어나거나 찌그러진 것처럼 보이면 사용주인 TV의 가로:세로 비율에 맞추어 이 값을 조절. 가능한 설정 값들은 다음과 같다:
  - 1 – 4:3, 일반적인 구형 기기
  - 2 – 14:9, 비교적 작은 와이드스크린 TV
  - 3 – 16:9, 일반적인 근래의 와이드스크린 TV

www.kotl.or.kr

- hdmi_mode – 아날로그 컴포지트 출력뿐만 아니라, HDMI 포트에 대한 자동 감지 해상도도 수동으로 설정 가능. 좀 더 먼 거리에서 화면을 읽기 쉽도록 해상도를 낮춰 설정하고자 할 경우에 간편. 참조 B, "HDMI Display Modes"에 설정 가능한 값들의 목록이 있다.
- hdmi_drive – HDMI 포트의 출력 전압을 변경할 수 있다. 이 값은 HDMI-DVI 어댑터를 사용할 때 중요하다. HDMI와 DVI의 전압이 약간 다르기 때문이다. 화면이 너무 밝으면서 허옇거나 바람에 날아간 것 같으면 이 설정을 바꿔본다. 가능한 값은 다음과 같다:
  - 1 – DVI 출력 전압. 이 모드에서는 HDMI 케이블에 음성 신호가 출력되지 않는다.
  - 2 – HDMI 출력 전압. 이 모드에서는 HDMI 케이블에 음성 신호가 포함되어 출력.
- hdmi_force_hotplug – 라즈베리파이가 디스플레이를 감지하지 못했더라도 HDMI 포트를 사용하도록 강제한다. 0으로 설정하면 라즈베리파이가 디스플레이 감지 시도를 할 수 있도록 허용한다. 반면에 1로 설정하면 무조건 HDMI를 사용하도록 한다.
- hdmi_group - HDMI그룹 모드를 CEA 또는 DMT로 설정한다. hdmi_mode를 이용하여 출력할 해상도와 주파수를 설정하려고 하기 전에 사용하려는 디스플레이에 맞게 이 설정을 해주어야 한다. 두 개의 설정값이 가능하다:
  - 1 – HDMI 그룹을 미국 가전 협회(CEA; Consumer Electronics Association of America)가 정의한 모드들의 그룹으로 설정
  - 2 - HDMI 그룹을 영상기기 표준 협회(VESA; Video Electronics Association of America)가 디스플레이 모니터 타이밍(DMT; Display Monitor Timings) 사양에서 정의한 모드들의 그룹으로 설정
- htmi_safe – 라즈베리파이가 HDMI 포트에 연결된 디스플레이와 호환되는 최대한의 호환성을 제공하도록 디자인 된 HDMI 설정의 사전 설정 집합을 사용하도록 강제한다. 이 값을 1로 설정하면 hdmi_forces_hotplug=1, config_hdmi_boost=4, hdmi_group=1, hdmi_mode=1, disable_overscan=0으로 설정한 것과 같다.
- config_hdmi_boost – 어떤 모니터들은 HDMI 출력에 좀 더 많은 전력을 공급해야 정상적으로 동작한다. 화면에 눈덮인 것 같은 모습이면 이 값을 1(케이블이 짧을 때)에서 7(케이블이 길 때)까지 줄 필요가 있다.

config.txt파일의 각 옵션들은 한 행에 기술되어야 하며 각 옵션명칭 뒤에는 등호(=)를 붙이고 값을 쓴다.

www.kett.or.kr

3. 부팅 옵션들

리눅스 커널을 로딩하는 방법을 제어하는 것은 주로 cmdline.txt 파일에서 설정하지만 config.txt 파일에서도 설정이 가능하다.

disable_commandline_tags – start.elf 모듈이 리눅스 커널이 로드되기전에 0x100번지 뒤로 채워지는 것을 건너 뛴다. 이 옵션은 비활성화되면 리눅스가 비정상적으로 로드되어 크래시가 발생한다.

cmdline – 리눅스 커널에 전달될 커맨드 라인 파라메터들. /boot/cmdline.txt파일을 대신해서 사용될 수 있다.

kernel – 로딩될 커널파일 이름. 비상용 커널을 로딩하는 수단으로 사용될 수 있다. "무제 해결" 장 참조

ramfsfile – 로딩 될 초기 RAM 파일 시스템(RAMFS) 이름. 실험용으로 초기 RAM 파일 시스템을 만든 경우가 아니라면 거의 사용할 일이 없다.

init_uart_baud – 직렬 콘솔의 속도, 비트/초. 기본 값은 115200이지만 구형 하드웨어 직렬 터미널을 사용하고 있다면 속도를 낮춰서 연결해보는 것이 좋다.

4. 라즈베리파이 오버 클럭킹

config.txt 파일을 이용하여 라즈베리파이의 BCM2835 프로세서의 그래픽 출력을 제어할 수도 있지만 다른 부분들도 제어가 가능하다. 특히 부품들의 수명 단축을 감수한다면 칩이 실행되는 속도를 바꾸어 시스템의 성능을 높일 수도 있다 – 이러한 과정을 오버클럭킹이라고 한다.

경고: 이 섹션에 나열된 설정 값들을 변경하는 것은 라즈베리파이에 손상을 입힐 수 있다. 특히 메모리, CPU, GPU의 전압과 관련된 설정을 바꾸는 것은 실제 손상을 입히기 전에 원상 복귀하더라도 라즈베리파이의 사후 지원 보장을 무력화하는 퓨즈를 설정할 수 있다. 이러한 설정들을 사용하여 야기되는 손상은 라즈베리파이 재단 또는 사용자가 제품을 구매한 소매상에 의해 보상되지 않을 것이다. 확신이 없으면 이 설정 값들을 바꾸지 말기 바란다. 오버클럭킹을 통한 성능 향상 이득은 위험성에 비해 그다지 크지 않다.

라즈베리파이에 사용되는 BCM2835 멀티미디어 프로세서는 시스템온칩(SoC)으로 크게 두 부분으로 나누어진다. 일상적인 프로세싱을 담당하는 CPU와 2D 혹은 3D 그래픽을 처리하는 그래픽 프로세서(GPU)이다.

config.txt 파일을 이용하면 두 부분을 모두 오버클럭킹할 수 있다. 또한 메모리 모듈이 동작하는

www.ketti.or.kr

속도도 높일 수 있다.

오버클럭킹 설정들

- arm_freq – CPU 클럭 설정. 기본 값은 700 MHz
- gpu_freq – GPU 클럭 설정. 기본 값은 250 MHz
  - core_freq – GPU의 코어 클럭 설정. 기본 값은 250 MHz
  - h264_freq – GPU의 하드웨어 비디오 디코더의 클럭 설정. 기본 값은 250 MHz
  - isp_freq – GPU의 이미지 센서 파이프라인 클럭 설정. 기본 값은 250 MHz
  - v3d_freq – GPU의 3D 렌더링 하드웨어 클럭 설정. 기본 값은 250 MHz
- sdram_freq – RAM의 클럭 설정. 기본 값은 400 MHz
- init_uart_clock – UART의 클럭 속도 설정. 기본 값은 3000000
- init_emmc_clock - SD카드 컨트롤러의 클럭 속도 설정. 기본 값은 80000000

예를 들어, CPU를 800 MHz, GPU를 280 MHz, RAM을 420 MHz로 오버클럭킹하려고 한다면 config.txt 파일에 다음 옵션들을 추가한다:

arm_freq=800

gpu_freq=280

sdram_freq=420

모든 설정 값들은 재부팅 후 적용된다. 원상 복귀를 원하면 config.txt파일 전체를 지우거나 해당 옵션행만 삭제한 후 재부팅하면 된다.

과전압(Overvoltage) 설정들

- over_voltage – BCM2835의 코어 전압을 조정. 기본 값(0)을 초과 또는 미달하는 전압의 0.025V의 배수(정수)로 지정. 최소 값은 -16, 최대 값은 8.
- over_voltage_sdram – 메모리 칩에 주어지는 전압을 조정. 기본값(0). 최소 값은 -16, 최대 값은 8. 또한 각 메모리 컴포넌트들에 주어지는 전압을 개별 조절할 수 있다.

www.kett.or.kr

- over_voltage_sdram_c – 메모리 컨트롤러에 주어지는 전압 조절. 범위는 over_voltage_sdram과 동일
- over_voltage_sdram_i – 메모리 입출력 시스템에 주어지는 전압 조절. 범위는 over_voltage_sdram과 동일
- over_voltage_sdram_p – 메모리 물리 레이어(PHY) 컴포넌트에 주어지는 전압 조절. 범위는 over_voltage_sdram과 동일

예를 들어, BCM2835에 0.05V을 더하여 1.25V를 가하고 메모리 칩에 0.1V를 더하여 1.3V를 가하려 한다면 config.txt파일에 다음 옵션 행을 추가한다:

over_voltage=2

over_voltage_sdram=4

L2 캐시 비활성화

라즈베리파이의 BCM2835 SoC 프로세서는 128KB의 Layer 2 캐시메모리를 가지고 있다. 이 메모리는 작지만 매우 빨라서 성능 향상을 위해 속도가 느린 메인 메모리와 프로세서 사이에서 데이터와 명령어들을 임시로 보관하는 역할을 한다.

BCM2835는 원래 셋톱박스를 목표로 고안된 멀티미디어 프로세서이기 때문에 L2 캐시는 오직 GPU에 의해 사용되도록 디자인 되었다. 일반적인 프로세서들과는 달리 CPU는 자신을 위한 L2 캐시를 가지고 있지 않다.

config.txt를 이용해서 BCM2835에게 CPU가 L2 캐시 메모리를 사용할 수 있도록 지시할 수 있다. 이렇게 하면 때때로 성능 향상을 가져오기도 하지만 L2 캐시가 물리적으로 CPU보다는 GPU에 비교적 가깝게 위치하고 있기 때문에 성능에 해를 끼치기도 한다.

L2 캐시 메모리를 이용하려면 캐시 메모리 사용을 전제로 컴파일 된 리눅스 배포판을 사용해야 된다. 캐시가 비활성화된(GPU가 L2 캐시 사용) 일반적인 라즈베리파이를 위한 배포판에서 L2 캐시를 활성화(CPU가 L2 캐시 사용)하면 비정상적인 동작을 일으킬 수 있다.

CPU가 L2 캐시를 사용하도록 하려면 config.txt 파일에 다음 옵션 행을 추가한다.

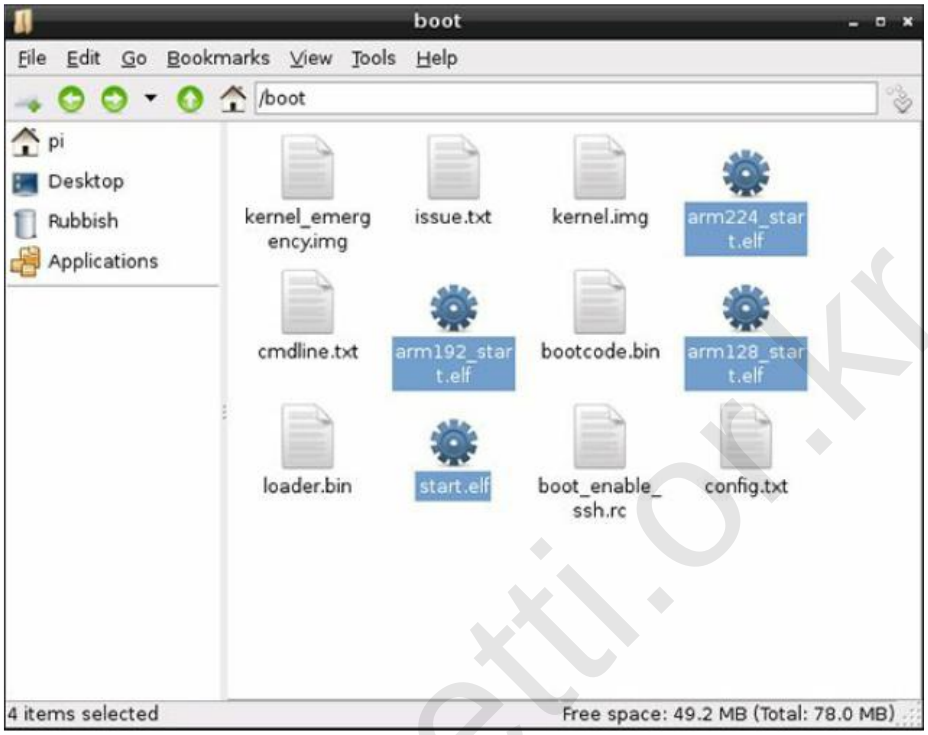
disable_l2cashe=0

CPU의 L2 캐시 사용을 비활성화 하려면 0을 1로 바꾸어 재부팅 한다.

www.ketii.or.kr

5. 메모리 파티셔닝 – start.elf

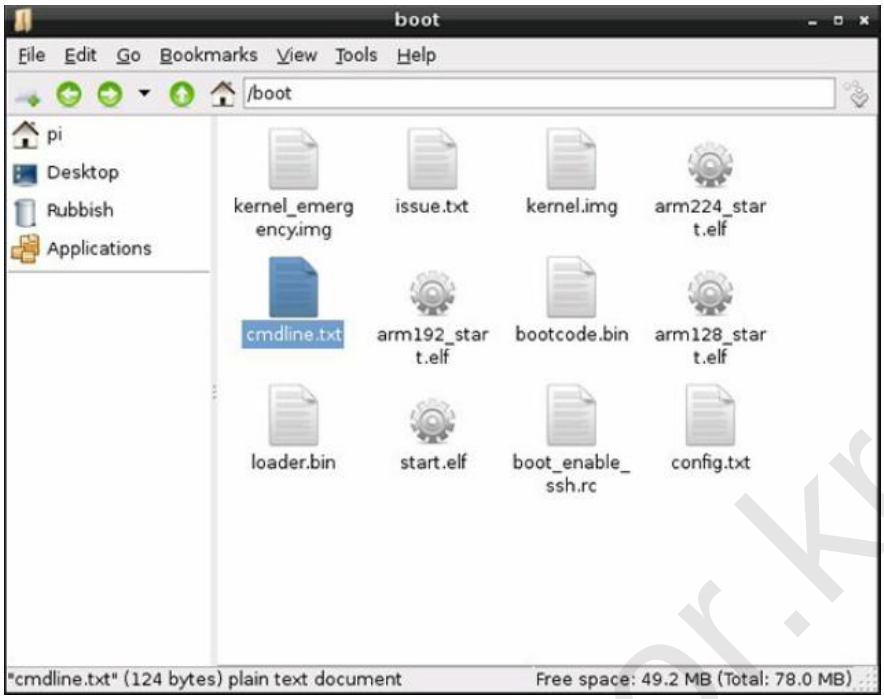
그림 7-2 - /boot 디렉터리의 다양한 start.elf 파일들



6. 소프트웨어 설정들 – cmdline.txt

그림 7-3 - /boot디렉터리의 cmdline.txt 파일

www.ketti.or.kr



www.ketti.or.kr

Chapter 8. 홈씨어터로 활용

라즈베리파이를 활용하는 가장 보편적인 방법은 홈씨어터 PC 또는 HTPC로 활용하는 것이다. 라즈베리파이에 사용되는 브로드컴 BCM2835는 원래 HTPC에서 멀티미디어 기능을 담당하도록 하기 위해 개발된 것이다.

BCM2835 시스템온칩(SoC)의 그래픽 부분인 브로드컴 VideoCore IV는 인기 있는 H.264 포맷을 사용하여 full-speed high-definition 비디오를 재생할 수 있는 능력을 가지고 있다. 또한 동시에 3.5mm 아날로그 오디오 출력과 HDMI 디지털 출력을 통해 다양한 포맷의 오디오 파일을 재생할 수 있다.

홈 씨어터 애호가들에게는 라즈베리파이의 작은 크기, 저전력 소모, 조용한 동작과 같은 특성들은 아주 매력적인 것이다. 다양한 배포판들과 소프트웨어 패키지들이 라즈베이 파이를 사용자 친화적인 홈 씨어터 PC로 변신시킬 수 있도록 디자인되어 있을 뿐만 아니라 그 것들을 사용해보기 위해서 기존의 운영체제를 포기할 필요도 없다.

1. Rasbmc를 이용한 전용 HTPC 구축

라즈베리파이에서 음악을 재생하는 것 이외에 BCM2835는 그 이상을 할 수 있는 능력을 가지고 있다. 라즈베리파이는 VideoCore IV GPU를 사용하여 Full HD 1080p H264 비디오를 디코딩하고 재생할 수 있어 조그만 패키지와 믿을 수 없을 만큼 적은 전력 소모 만으로 라즈베리파이를 강력한 미디어 센터로 만들 수 있다.

라즈베리파이를 홈 씨어터 PC로 최대한 활용하기 위해서는 약간의 소프트웨어가 더 필요하다. 이 소프트웨어는 데비안 배포판 위에 설치할 수 있지만 Rasbmc 배포판을 사용하는 것이 더 쉽다.

Rasbmc는 라즈베리파이를 비디오 및 음악 재생, 사진 보기와 인터넷 스트리밍 능력을 갖춘 온전한 기능의 미디어 센터 시스템으로 변신시키기 위해 Sam Nazarko에 의해 만들어진 배포판이다. 이는 몇몇 장비 제조 회사들이 상용 셋톱박스 제작에 사용한 적이 있는 인기 있는 Xbmc 배포판에 기반하고 있다.

라즈베리파이를 사용하여 고선명 비디오 출력과 H.264 디코딩 능력을 가진 홈 씨어터를 구축하려고 한다면 다른 어떤 리눅스 배포판보다 Rasbmc야말로 탁월한 선택이며 설치하고 실행하기도 매우 쉽다. 공식 웹사이트인 http://www.raspbmc.com/download/에서 설치파일을 다운로드 할 수 있다. 리눅스, OS X, 윈도우를 위한 설치 프로그램을 이용하면 이미지 파일을 자동으로 다운로드하여 연결된 SD카드에 기록해준다. 직접 이미지 파일을 다운로드하여 "라즈베리파이 만나보기"장

www.ketti.or.kr

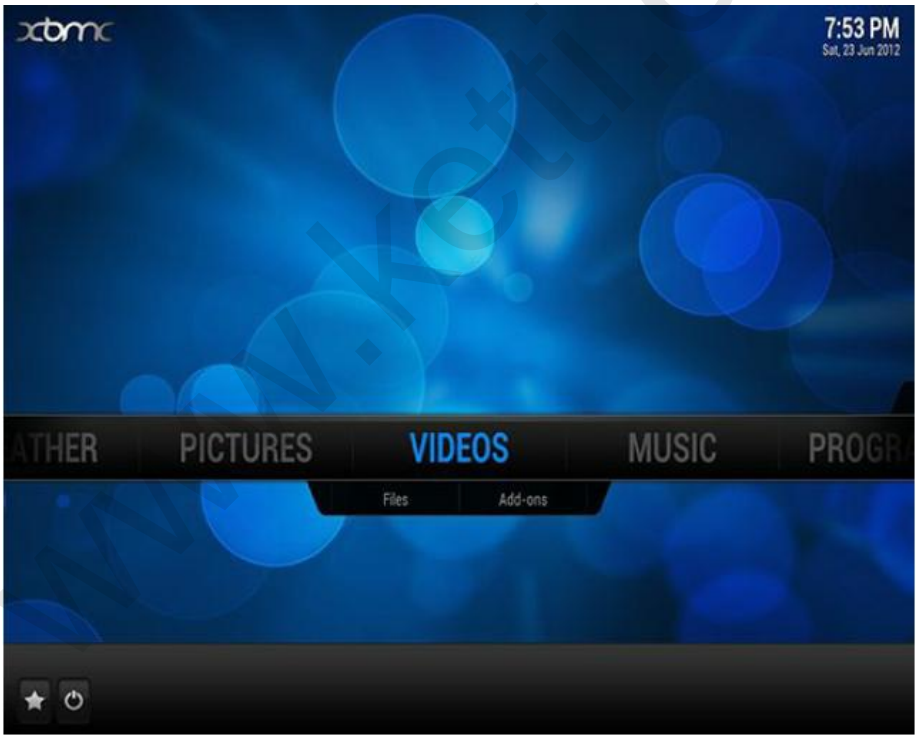
에서 해 본 방법으로 SD카드에 기록할 수도 있지만 그럴 필요는 없다.

경고: 이미 사용 중인 SD카드에 Raspbmc를 설치하면 기존의 모든 내용이 삭제된다. 보관을 원하는 파일을 별도로 복사를 해두어야 하며 두 배포판 사이를 오가며 사용하길 원한다면 별도의 SD카드를 따로 구매하기 바란다.

Raspbmc 설치 프로그램이 완료되면 SD카드를 라즈베리파이에 넣고 전원 플러그를 다시 연결한다. 이 때 이더넷 케이블은 연결이 되어 있어야 한다. Raspbmc가 처음 기동될 때 인터넷을 통해 필요한 데이터를 다운로드 받기 때문이다. 약 10분에서 15분 정도가 소요된다. 파티션 포맷 단계에서 멈춰서 있는 시간이 길어지더라도 놀라지 말자. 원래 좀 시간이 걸린다.

기동이 되면 Raspbmc는 자동으로 XBMC서비스를 시작한다. USB수신기가 있는 리모콘을 구입하여 연결하면 키보드와 마우스를 대신할 수도 있다.

그림 8-1 - Rasbmc에 의해 실행된 Xbmc 홈 스크린



2. 인터넷 미디어 스트리밍

기본적으로 Xbmc는 라즈베리파이 자체에 있는 파일만을 재생하도록 구성되어 있다. 하지만 비디오 메뉴 아래의 Add-ons를 선택하면 다양한 TV 채널들과 인터넷 스트리밍 서비스들을 포함한 스트리밍 기능을 추가할 수 있다. 호환 가능한 플러그인 목록을 보려면 Add-ons를 클릭한 후

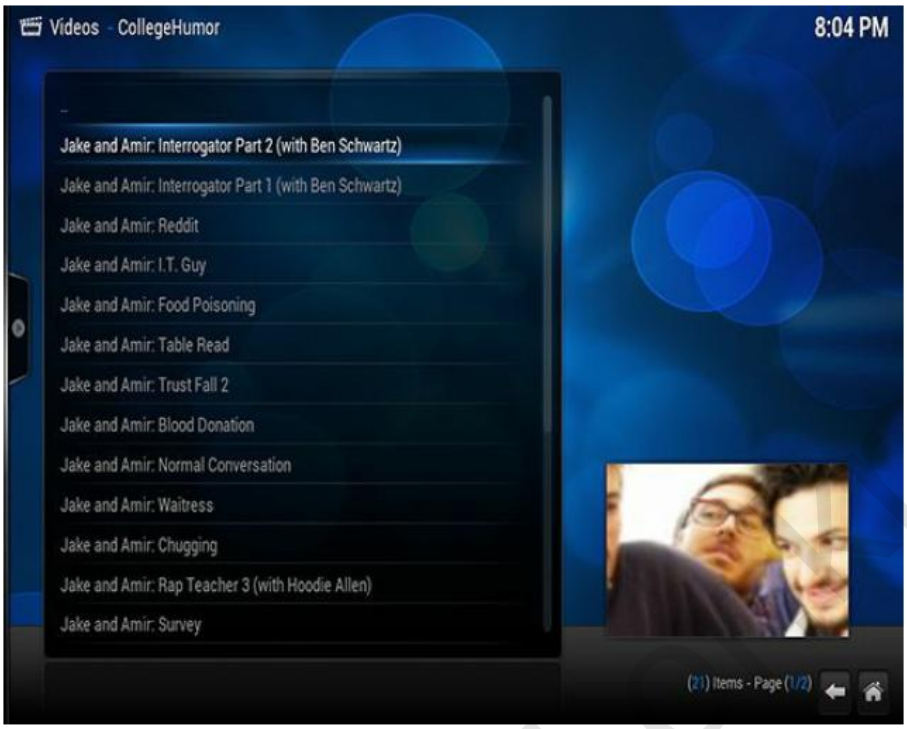
Get More를 선택한다.

그림 8-2 - Rasbmc에 새로운 비디오 애드온을 추가

마우스나 커서 키를 이용해서 스크롤을 할 수 있고 원하는 항목을 클릭하거나 Enter키를 입력해서 자세한 정보를 볼 수 있다. 애드온을 설치하려면 목록에서 해당 항목을 클릭했을 때 나타나는 팝업 메뉴에서 Install 을 클릭한다. 그러면 해당 애드온이 인터넷으로부터 다운로드되어 자동으로 Xbmc에 설치된다. 설치된 애드온이 Broken으로 표시되는지 주의해서 봐야 한다. 그런 애드온들은 아직 정상적으로 작동되지 않는 것들로써 개발자가 문제를 해결할 때까지 기다려야 한다.

비디오 애드온을 설치한 다음에는 스크린의 우하단에 있는 홈 버튼을 클릭하여 Xbmc의 메인 화면으로 돌아간 다음 화면 중앙의 Video를 선택하고, 표시되는 비디오 옵션들 중에서 설치된 애드온을 선택하면 된다. 표시되는 파일 목록 중에서 다운로드하고자 하는 것을 선택한다. 애드온이 여러가지 카테고리를 가지고 있으면, 카테고리가 먼저 다운로드 되고, 그 중의 한 항목을 선택하여 내용을 볼 수 있다.

그림 8-3 - CollegeHumor Xbmc 애드온을 통해 볼 수 있는 비디오 목록



Music에 있는 애드온들도 Video에 있는 애드온들과 동일한 방식으로 동작한다. 이러한 애드온들을 이용해서 Flickr, Picasa, The Big Picture, Grooveshark, Sky.fm, SoundCloud 등과 같은 사이트로부터 사진을 보고 오디오를 스트리밍해서 들을 수 있다.

3. 로컬 네트웍 미디어 스트리밍

XBMC 소프트웨어는 홈 네트웍 범위의 장비들에 접속할 수 있도록 해주는 UPnP(Universal Plug and Play) 미디어 스트리밍 표준을 지원한다. 휴대폰, 게임 콘솔, NAS(Network Attached Storage) 등 동영상, 음악, 사진을 공유하고 스트리밍하기 위한 근래의 대부분의 장비들이 UPnP를 지원한다. 또 많은 랩톱과 데스크톱들이 UPnP나 DLNA(Digital Living Network Alliance) 표준을 지원한다 – 가지고 있는 장비들에서 이러한 기능들을 활성화 하려면 어떻게 해야 하는지 관련 문서를 참조하기 바란다.

Xbmc는 UPnP뿐만이 아니라 유닉스 기반의 컴퓨터들이 일반적으로 사용하는 NFS(Network File System), 윈도우 계열의 컴퓨터들이 사용하는 SMB(Server Message Block), OS X 장비들에 사용되는 Zeroconf 표준을 지원하는 네트웍 서버들에도 접속할 수 있다. 미디어 컨텐츠를 저장하기 위해 어떤 종류의 네트웍 접속 장비를 사용하건 Xbmc가 지원하는 방법중의 하나는 최소한 그 장비를 지원할 것이다.

Xbmc가 홈 네트웍 상에 있는 서버에 접속하도록 하려면 미디어 종류 – Video, Music, 또는

Pictures – 를 선택한 후 Add Source 옵션을 선택한다. 창이 표시되면 Browse to retrieve a list of source types를 선택한다(그림 8-4 참조). 소스 종류에는 파란색 아이콘으로 표시된 라즈베리파이에 연결된 로컬 드라이브들과 빨간색 아이콘으로 표시된 네트웍 장비들도 포함된다. 소스 종류를 선택한 후 표시되는 서버들 중 원하는 서버를 선택한다.

선택한 서버에 여러 폴더가 있을 경우 예를 들어, 장르별, 아티스트별, 앨범별, 파일 종류별 폴더가 있을 경우 Xbmc에 연결하기를 원하는 폴더를 선택한 후 OK 버튼을 클릭한다. 그러면 필요한 정보가 채워진 후 Add Source screen 화면으로 돌아오게 된다 (그림 8-4 참조). 암호로 보호된 서버인 경우 사용자 이름과 패스워드를 입력한 후 OK를 클릭한다.

그림 8-4 - Xbmc에서 네트웍 소스 선택

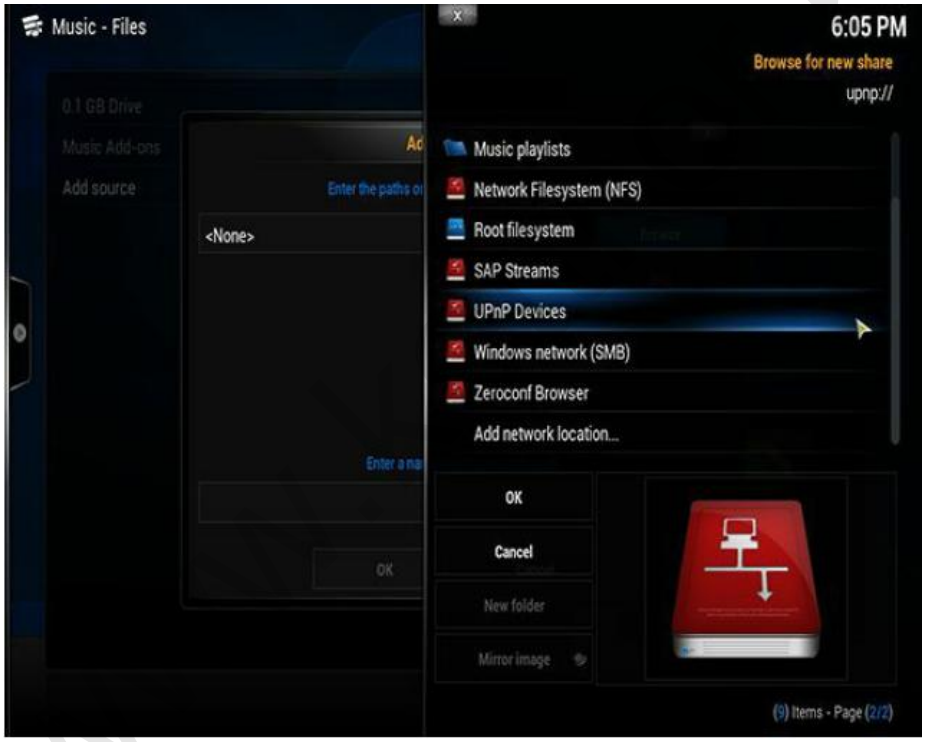
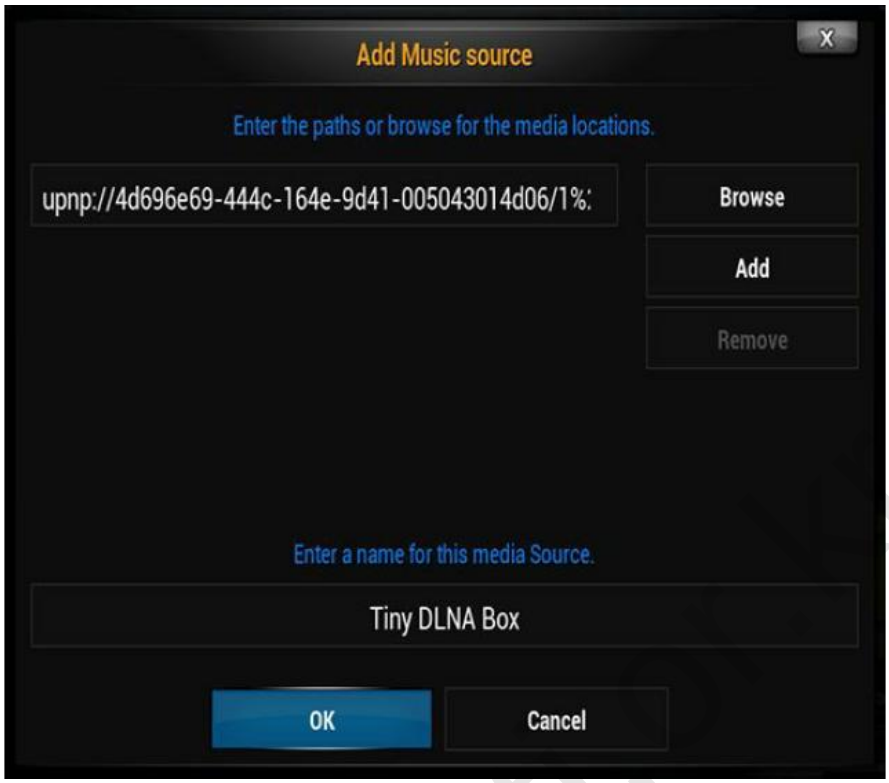


그림 8-5 - Xbmc에 UPnP 음악 소스 추가



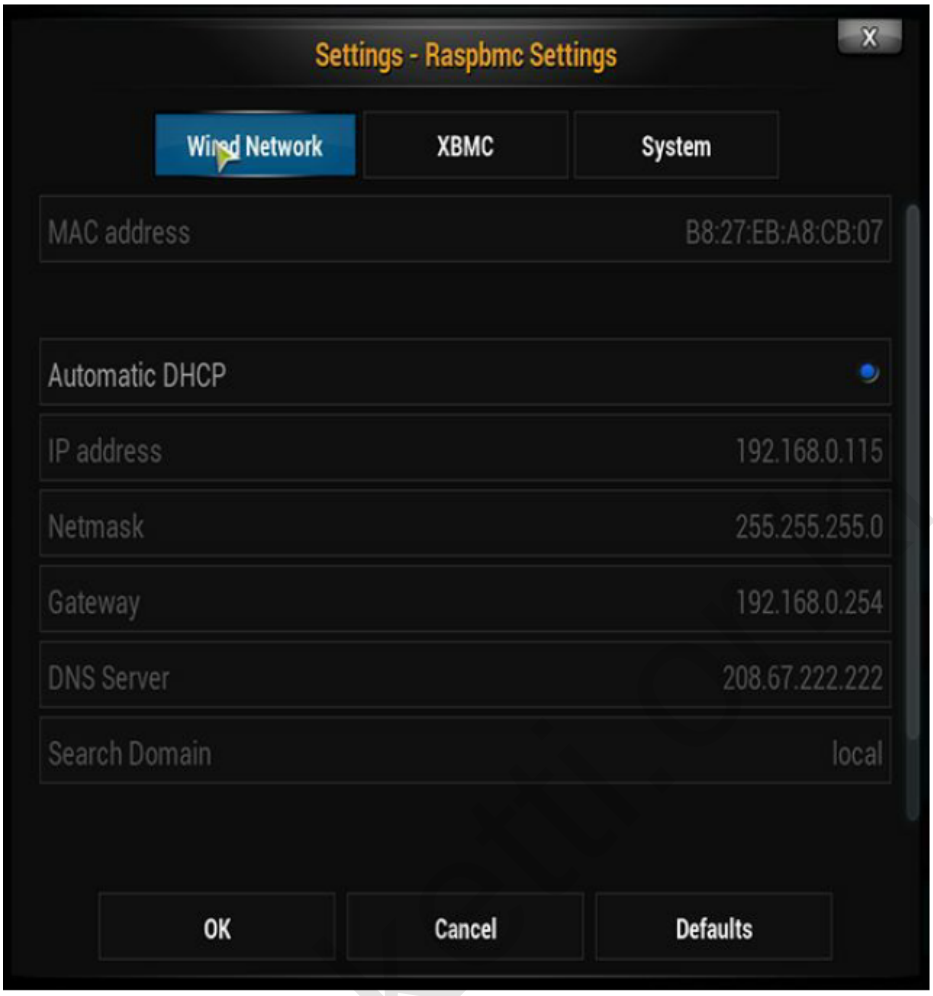
외장 하드 디스크를 소스로 추가할 때도 처음 목록에서 그 항목을 선택하여 같은 방법으로 추가할 수 있다. 대부분의 외장 드라이브는 자동으로 감지되어 목록에 나타나므로 별도로 추가할 필요가 없으며 Xbmc의 메뉴에 컨텐츠가 표시되지 않는 경우에만 별도로 추가하면 된다.

4. Rasbmc 설정 변경

홈스크린의 Programs메뉴를 통해 Rasbmc Settings 메뉴(그림 8-6 참조)에 접근할 수 있다. 이 곳에서는 네트웍 설정, 소프트웨어 업그레이드 설정, 여러 가지 시스템 설정을 할 수 있다. 대부분 신경 쓸 필요가 없지만 인터넷 접속에 문제가 있거나 Rasbmc 시스템이 지속적으로 멈추거나 할 때는 문제를 해결하는데 도움이 될 수 있다.

Programs밑의 Rasbmc Settings 옵션을 클릭하면 세 개의 탭을 가진 창이 열린다. 첫 번째 탭은 DHCP를 비활성화 한다거나 수동 네트웍 설정(Chapter 5. 네트웍 설정 참조)을 할 수 있는 곳이다. 다른 배포판에서 수동으로 네트웍 설정을 할 필요가 있었다면 Rasbmc에서는 같은 작업을 여기에서 할 수 있다.

그림 8-6 - 설정 메뉴에서 Rasbmc 설정 변경



XBMC로 표시되어 있는 두 번째 탭은 XBMC의 야간 빌드를 설치하도록 할 수 있는 곳이다. 야간 빌드는 매일 밤 자동으로 개발자의 최신 작업 결과가 반영된 프로그램 소스 코드로부터 빌드가 되므로 사용자가 겪었을 버그들을 위한 최신 수정 사항을 포함하고 있을 수 있다. 반면에 문제를 악화시키는 테스트되지 않은 변경사항도 포함하고 있을 수 있다. 같은 화면에 야간 빌드를 테스트해보고 문제가 발생하면 기본 버전으로 쉽게 되돌아갈 수 있도록 Xbmc의 다른 버전간의 전환을 지원하는 기능도 지원된다.

마지막 System 탭에서는 Rasbmc를 네트웍에서 안전하게 접근할 수 있도록 pi 사용자계정 암호를 변경할 수 있다. 범용 데비안 배포판에서는 이 암호는 기본값으로 raspberry로 설정되어 있다. System 탭에서는 또한 Rasbmc 소프트웨어의 자동 업그레이드를 비활성화 할 수 있다. 하지만 가급적이면 켜두는 것이 좋다. 최신 보안 및 버그 수정을 자동으로 받을 수 있기 때문이다. 그러나 인터넷 접속을 종량제 방식으로 사용하고 있다면 이 기능을 비활성화 하는 것이 나을 것이다.

본 문서는 자유롭게 배포 가능합니다.
다만 상업적 목적 및 문서 수정 배포시
한국전자통신교육원(kks@ketti.or.kr)로 문의 부탁드립니다.

라즈베리파이를 활용한
오픈 소스 R&D 플랫폼
(9,10장)

한국전자통신교육원

본 문서는 자유롭게 배포 가능합니다.
다만 상업적 목적 및 문서 수정 배포시
한국전자통신교육원(kks@ketti.or.kr)로 문의 부탁드립니다.

www.ketti.or.kr

목차

Chapter 1. 라즈베리파이 소개 6

프로그래밍의 재미 6

라즈베리파이의 역사 6

라즈베리파이란? 7

라즈베리파이로 무엇을 할 수 있나? 9

Chapter 2. 라즈베리파이와의 첫 만남 10

ARM 대 x86 10

리눅스 대 윈도우 11

라즈베리파이에 포팅 된 운영체제 11

주변기기 연결 12

SD카드에 운영체제 이미지 써넣기 15

USB저장 장치 연결 16

네트웍 연결 16

전원 연결 17

Chapter 3. 리눅스 시스템 관리 18

리눅스 개요 18

리눅스 기초 18

데비안(Debian) 소개 19

외부 저장 장치 사용 22

사용자 계정 만들기 23

www.ketti.or.kr

파일 시스템 구성 24
소프트웨어 설치 및 제거 26
Chapter 4. 문제 해결 30
키보드 및 마우스 진단 30
전원 진단 30
디스플레이 진단 32
부팅 진단 32
네트웍 진단 33
비상용 커널 35
Chapter 5. 네트웍 설정 36
유선 네트워킹 36
무선 네트워킹 39
Chapter 6. 파티션 관리 49
새 파티션 작성 49
파티션 크기 변경 52
수작업으로 파티션 크기 변경 53
더 큰 SD카드로 옮기기 57
Chapter 7. 하드웨어 구성 설정 60
하드웨어 설정 – config.txt 60
디스플레이 설정 변경 61
부팅 옵션들 64
라즈베리파이 오버 클럭킹 64

www.ketti.or.kr

메모리 파티셔닝 – start.elf 67

소프트웨어 설정들 – cmdline.txt 68

Chapter 8. 홈씨어터로 활용 71

콘솔에서 음악 플레이 71

Rasbmc를 이용한 전용 HTPC 구축 73

인터넷 미디어 스트리밍 74

로컬 네트웍 미디어 스트리밍 76

Rasbmc 설정 변경 78

Chapter 9. 업무용으로 활용 80

클라우드 기반 앱 사용 80

OpenOffice.org 사용 81

Gimp를 이용한 이미지 편집 82

Chapter 10. 웹 서버로 활용 84

WordPress 설치 87

Chapter 11. 스크래치로 프로그래밍 92

예제 1: Hello World 92

예제 2: 애니메이션 및 사운드 96

예제 3: 간단한 게임 98

로보틱스 및 센서들 103

더 읽어보기 104

Chapter 12. 파이썬 프로그래밍 105

파이썬 소개 105

www.ketti.or.kr

예제 1: Hello World 105

파이썬 프로그램 실행 가능하게 만들기 109

예제 2: 주석, 입력, 변수 및 반복문 110

예제 3: pygame을 이용하여 게임 만들기 113

예제 4: 파이썬과 네트워킹 121

더 읽어보기 127

Chapter 13. 하드웨어 해킹 128

전자 부품 128

저항 값 읽기 130

GPIO 포트 131

파이썬에서 GPIO 사용하기 134

GPIO 출력: LED 밝히기 136

GPIO 입력: 버튼 읽기 140

브레드보드 벗어나기 144

납땜에 대한 간단한 가이드 146

참조 A: Python Recipes 151

Raspberry Snake (Chapter 12. 예제 3) 151

IRC User List (Chapter 12. 예제4) 155

GPIO Input and Output (Chapter 12) 157

참조 B: HDMI 디스플레이 모드 목록 159

참조 C: 참고 문헌 166

참고 문헌 166

www.ketti.or.kr

참고 사이트 166

www.ketti.or.kr

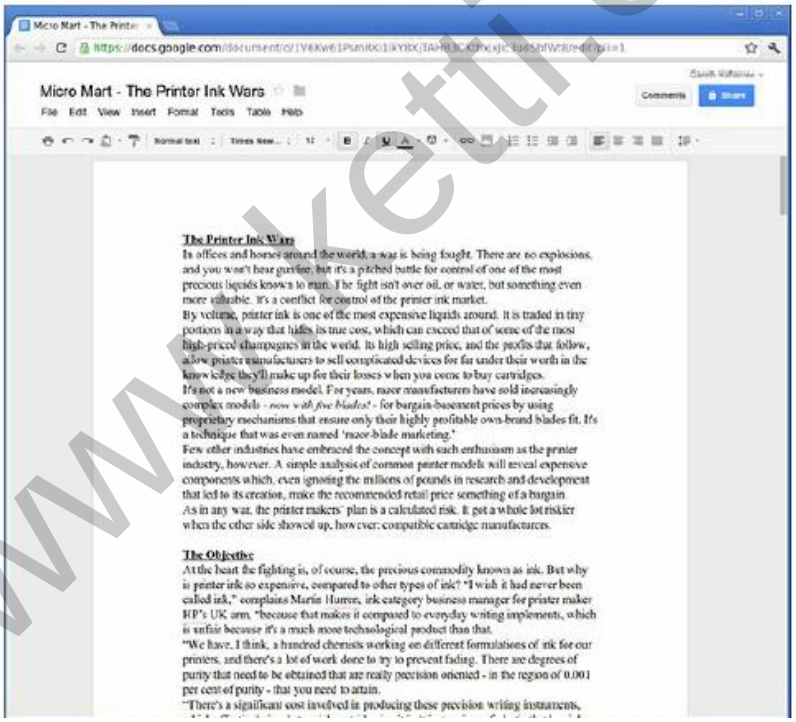
Chapter 9. 업무용으로 활용

라즈베리파이는 매우 유연성이 있어 저전력의 범용 데스크톱 컴퓨터로 사용하기에도 좋다. 일반적인 데스크톱이나 랩톱 컴퓨터들만큼의 성능을 내기는 어렵겠지만 가끔 버벅거리는 성능을 보이더라도 저비용의 환경 친화적인 전력소모량을 감안하면 충분히 눈감아 줄 만하다.

1. 클라우드 기반 앱 사용

- 구글 드라이브 – 구글 드라이브(구 구글 독스)는 워드프로세서, 스프레드시트 및 프레젠테이션 도구(그림 9-1 참조) 포함. 기업 고객은 구글 앱스 계정을 구매하면 더 향상된 기능 제공. 웹 기반 GMail 계정이 있으면 자동으로 사용 가능. http://docs.google.com을 통해 접근 가능.

그림 9-1 - 라즈베리파이의 크로미움에서 실행되는 구글 드라이브



- Zoho – Zoho는 등록 사용자가 5백만명에 이르는 인기 있는 서비스이다. 구글 드라이브처럼 워드프로세서, 스프레드시트 및 프레젠테이션 패키지가 포함되어 있지만 위키 기반의 지식베이스 시스템, 웹 컨퍼런싱, 재정관리 및 심지어는 고객관계 관리 시스템 같은 업무 위주의 기능들도 제공한다. 하지만 고급 기능들은 대부분 유료 계정을 구입해야 한다. http://www.zoho.com을 통해 접근할 수 있다.

www.kettl.or.kr

- Office365 – 마이크로소프트 오피스 사용자에게는 Office365가 최적이다. 데스크톱용 최신 마이크로소프트 오피스와 같은 사용자 인터페이스에 기반하며 강력하고 유연하다. Zoho나 Google Drive와는 달리 무료 계정이 없으며 월 사용료를 지불해야 한다. 더구나 리눅스 컴퓨터에서 접근할 때는 동작하지 않는 기능이 더러 있다. http://office365.microsoft.com에서 등록할 수 있다.

- ThinkFree 온라인 – 한컴 씽크프리 오피스 소프트웨어의 웹 기반 인터페이스이다. 씽크프리 온라인은 워드프로세싱, 스프레드시트, 프레젠테이션 기능을 제공하며 1GB의 저장공간이 무료이다. 기업들을 대상으로 제공되는 씽크프리 서버 소프트웨어 및 태블릿과 스마트폰용의 씽크프리 모바일과도 결합된다.

이들 웹기반의 서비스들을 사용하려면 데비안 배포판에 기본적으로 포함되어 있는 미도리 브라우저로는 역부족이다. 오픈 소스 프로젝트인 구글의 크로미움 브라우저가 좋은 선택이 될 것이다.

```
$ sudo apt-get install chromium-browser
```

2. 오픈오피스 사용

터미널 창에서 다음과 같이 입력하여 설치한다.

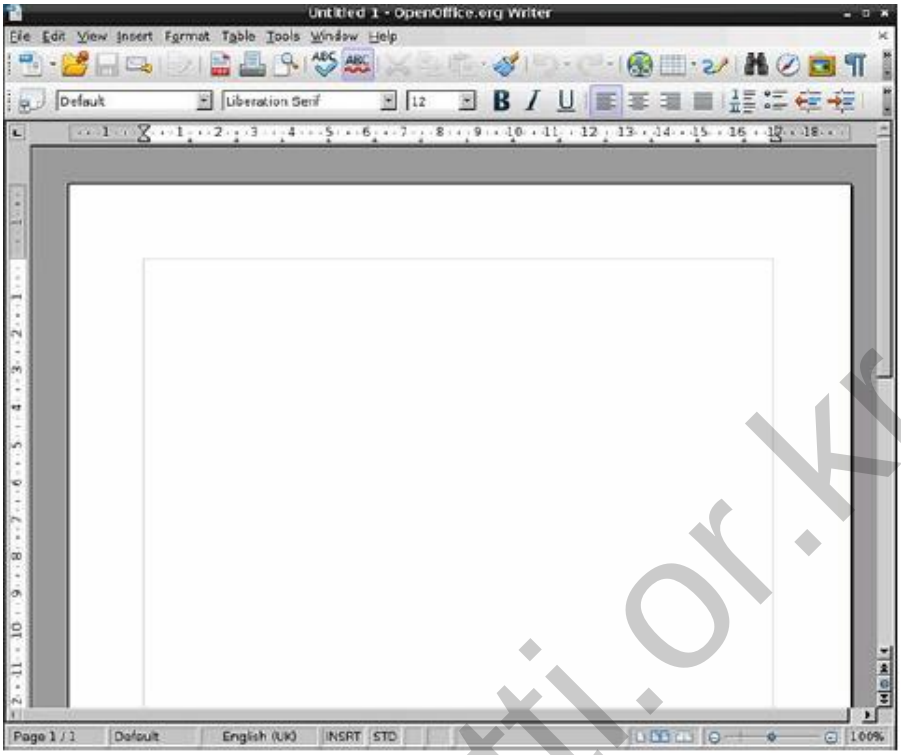
```
$ sudo apt-get update

$ sudo apt-get install openoffice.org
```

OpenOffice.org는 다음과 같이 구성되어 있다.

- OpenOffice.org – 주 애플리케이션 각 부분을 선택할 수 있는 링크를 제공한다.
- OpenOffice.org Calc – 스프레드시트
- OpenOffice.org Draw – 벡터 기반 일러스트레이션 애플리케이션
- OpenOffice.org Impress – 프레젠테이션 애플리케이션
- OpenOffice.org Math – 수식 편집기
- OpenOffice.org Write – 워드프로세서(그림 9-2 참조)

www.ketti.or.kr

그림 9-2 - 라즈베리파이에서 실행되는 OpenOffice.org Writer



기본적으로 OpenOffice.org에서는 ODF(Open Document Format) 형식으로 파일을 저장한다. 표준에 기반한 로열티 없는 파일 포맷이며 마이크로소프트 오피스의 최신 버전을 비롯한 대부분의 오피스 패키지에서 지원한다. 저장 대화창에서 다른 포맷을 선택하여 문서를 저장할 수 있다. 예전 버전의 마이크로소프트 오피스 문서 포맷도 지원한다. OpenOffice.org는 리눅스, 윈도우 및 OS X용의 무료 버전이 제공된다.

3. Gimp를 이용한 이미지 편집

사진 편집과 같은 비트맵 이미지 편집용으로는 Gimp(GNU Image Manipulation Program)를 사용할 수 있다. 상업용 이미지 편집 패키지인 어도비 포토샵 패키지와 인터페이스와 강력한 비트맵 이미지 편집 기능을 제공한다(그림 9-3 참조)

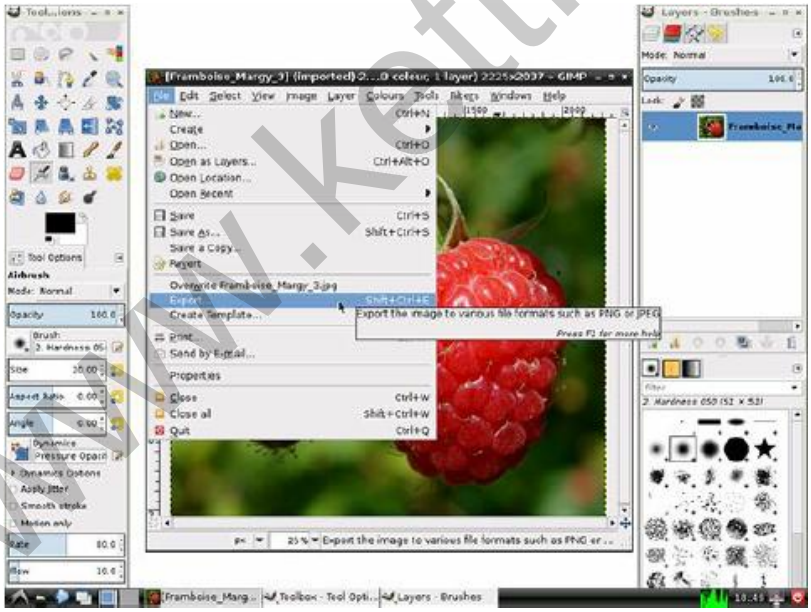
그림 9-3 - 라즈베리파이에서 실행되는 Gimp

www.ketti.or.kr

Gimp를 설치하려면 터미널 창에서 다음과 같이 입력한다.

```
$ sudo apt-get install gimp
```

그림 9-4 - Gimp에서 파일 탐색



www.ketti.or.kr

Chapter 10. 웹 서버로 활용

라즈베리파이는 데이터 센터에서 볼 수 있는 대부분의 장비보다는 훨씬 덜 강력하지만 그렇다고 해서 가정이나 업무 환경에서 유용한 서버로 활용되지 말란 법은 없다. 비록 적은 메모리와 상대적으로 열악한 프로세스 파워를 가지고 있지만 적은 전력 소모와 저소음은 트래픽이 적은 간단한 페이지들을 로컬 네트웍이나 인터넷에 서비스하기에는 아주 좋은 장점을 제공한다.

근래 상당수의 웹 서버들이 리눅스, 아파치, MySQL 및 PHP의 조합으로 운영되고 있으며 통상 LAMP 스택이라고 부른다. LAMP 기반의 서버에서 컨텐츠 관리 시스템인 워드프레스로부터 대화형 포럼인 phpBB에 이르는 상당히 복잡한 패키지들을 실행할 수 있다. 강력한 상업용 서버의 성능을 기대하지 않는 한 라즈베리파이로도 모두 가능한 것들이다.

팁: 웹 서버는 많은 양의 메모리를 사용한다. 메모리 사용 비율을 224/32 MB 비율로 해주고 (Chapter 7. 하드웨어 구성 설정 참조) GUI는 실행하지 않는 것이 좋다.

1. LAMP 스택 설치

$ sudo apt-get update –fix-missing

$ sudo apt-get install apache2 php5 php5-mysql mysql-server

MySQL 설치가 진행되는 도중 패스워드 입력을 요구하면 적절한 패스워드를 입력해준다. (그림 10-2 참조) 타이핑 실수를 방지하기 위해 같은 패스워드를 한 번 더 입력해주어야 한다.

그림 10-1 – 데비안에 LAMP 스택 설치

www.ketti.or.kr

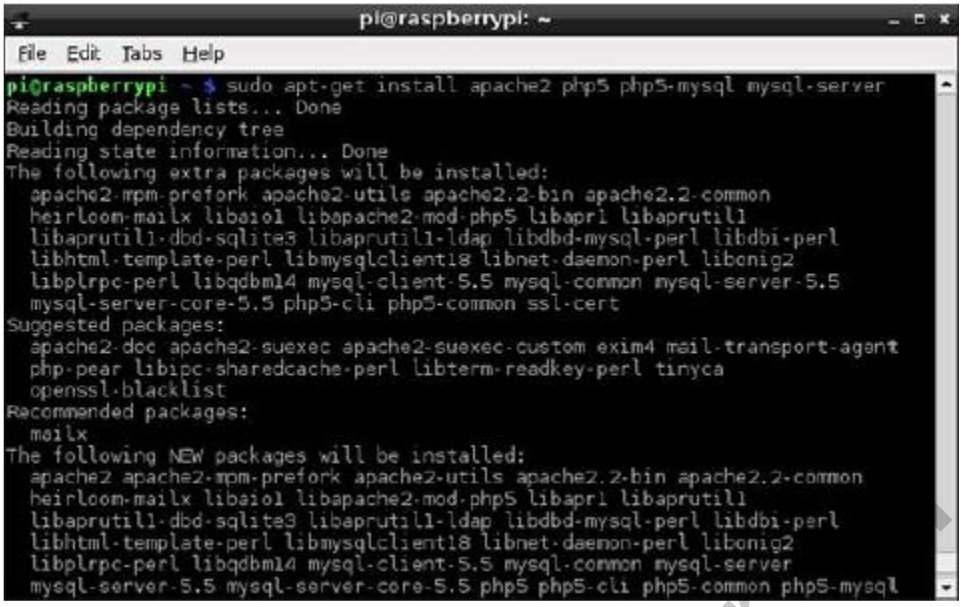
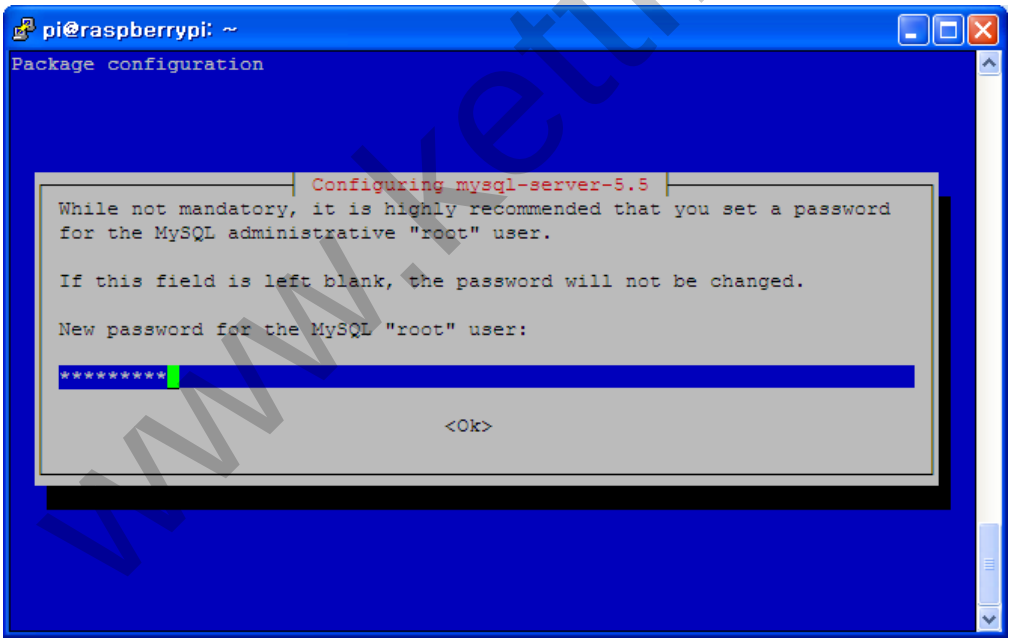
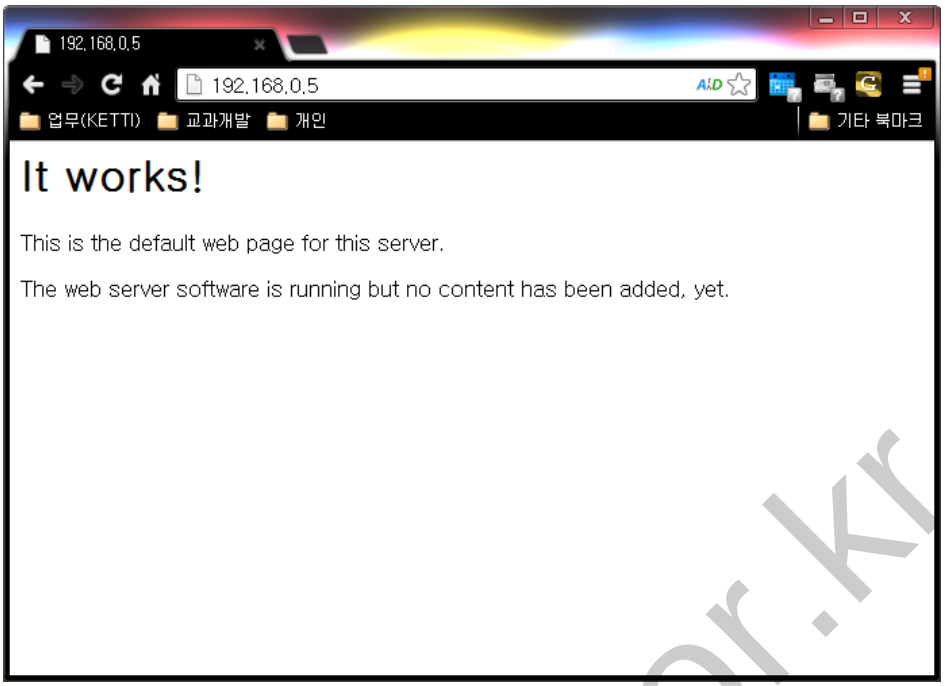


그림 10-2 - MySQL 패스워드 입력



설치가 완료되고 나면 아파치 서버와 MySQL서버가 백그라운드에서 실행되고 있을 것이다. 정상적으로 실행되고 있는지를 확인해 보려면 네트웍 상의 다른 컴퓨터에서 웹 브라우저로 라즈베리파이에 접속을 시도해 본다(그림 10-3 참조).

그림 10-3 - 웹 브라우저를 이용하여 아파치 서버에 접속



마지막 단계는 PHP 스크립팅 모듈이 아파치에 제대로 로딩되어 있는지를 확인해보는 것이다. PHP스크립트는 동적인 컨텐츠를 생성하기 위한 것으로써 PHP 스크립팅 모듈은 아파치 웹 서버가 동적인 컨텐츠를 지원하기 위해서는 없어서는 안될 중요한 모듈이다. PHP모듈이나, 유사한 다른 모듈인 파이션 모듈이 없다면 아파치는 단순히 정적인 페이지들만을 서비스할 수 있을 것이다. 아파치 PHP모듈을 시험해보기 위해서 다음 내용을 입력하여 PHP스크립트 파일을 작성한다.

$ cd /var/www

$ sudo nano phptest.php'

<?php phpinfo(); ?>

저장 후 빠져 나온다.

다른 컴퓨터에서 웹 브라우저를 실행한 후 주소 창에 다음 주소를 입력한다.

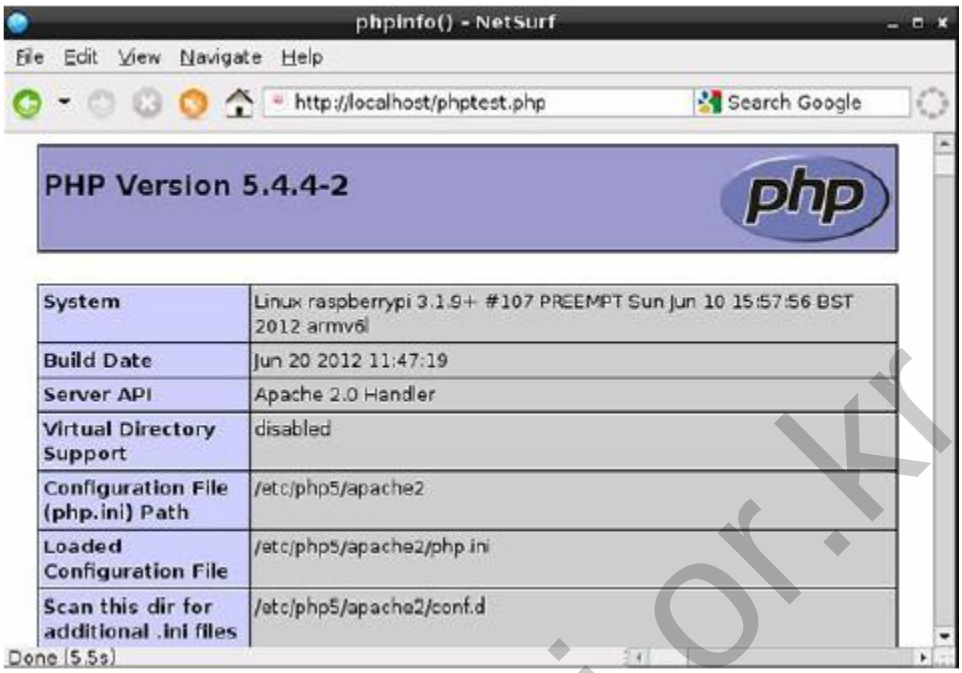
http://192.168.0.5/phptest.php ← IP주소는 자신의 라즈베리파이 주소

또는 라즈베리파이 자체 GUI의 웹 브라우저에서는 주소 창에 다음 주소를 입력한다(그림 10-4 참조).

http://localhost/phptest.php

www.ketti.or.kr

그림 10-4 - 라즈베리파이에서 아파치의 PHP 모듈 테스트



시험이 완료된 후에는 phptest.php파일을 삭제한다.

```
$ sudo rm /var/www/phptest.php
```

팁: 아파치가 가장 보편적인 웹 서버이기는 하지만 너무 느리게 느껴진다면 다른 웹 서버를 시험해 보는 것도 좋다. lighthttpd는 아파치 보다는 훨씬 적은 메모리를 사용하도록 설계된 웹 서버다. sudo apt-get install lighttpd 명령으로 설치할 수 있다.

기본적으로 웹 페이지들은 /var/www 폴더에 저장된다. 이 폴더는 슈퍼유저만 쓰기 가능하다. 아파치가 파일을 찾는 위치를 변경하거나 웹사이트를 여유 공간이 넉넉한 외부 저장장치로 옮기려면 /etc/apache2/sites-enabled 폴더에 있는 000-dedfault 파일을 수정한다. 아파치, PHP, MySQL에 대한 더 자세한 정보가 필요하면 터미널 창이나 콘솔에서 다음 명령을 쳐서 참조한다.

```
$ man apache2

$ man php5

$ man mysql
```

2. WordPress 설치

가장 인기 있는 블로깅 플랫폼 중의 하나인 워드프레스는 사용자들에게 매력적인 대화형 웹사이

www.ketti.or.kr

트를 만들 수 있도록 돕기 위한, 전에 없이 간단하고 강력한 플랫폼을 목표로 하는 오픈 소스 프로젝트이다. 워드프레스는 PHP와 Javascript를 기반으로 한다. 예를 들어 가장 있기 있는 몇몇 뉴스 사이트들도 워드프레스 플랫폼을 커스터마이징하여 구축되었다. 워드프레스를 설치하기 위해서는 SD카드에 37MB 가량의 여유 공간이 있는지 확인한다. LAMP스택 설치를 위한 113MB의 여유 공간도 필요하다. 다음 명령으로 워드프레스를 설치할 수 있다.

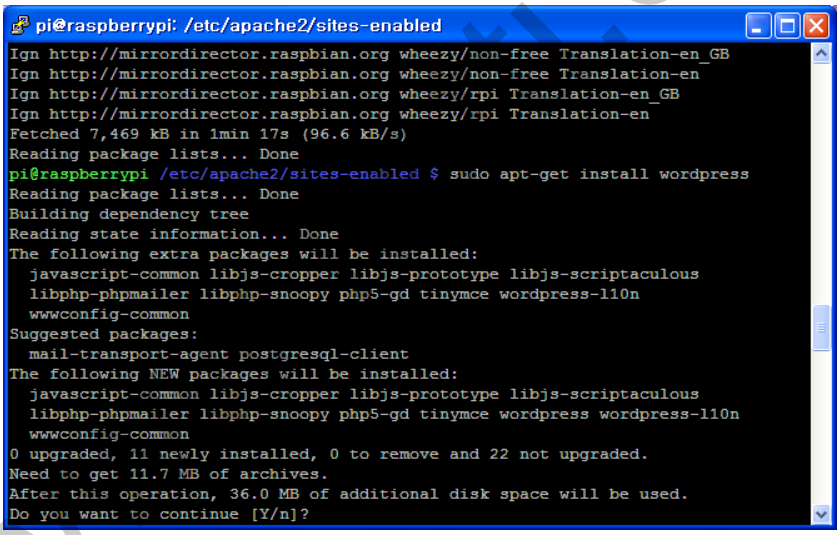
```
$ sudo apt-get install wordpress
```

Y를 입력하여 설치 계속

기본 설치 디렉터리는 /usr/share/wordpress이다. 아파치가 인식할 수 있도록 /var/www 디렉터리로 링크를 걸어 주어야 한다.

```
$ sudo ln –s /usr/share/wordpress /var/www/wordpress
```

그림 10-5 - 라즈베리파이에 워드프레스 설치



워드프레스 MySQL 설정 스크립트를 실행하여 LAMP 스택의 일원으로 새로운 데이터베이스를 생성한다.

```
$ sudo gunzip /usr/share/doc/wordpress/examples/setup-mysql.gz
```

```
$ sudo bash /usr/share/doc/wordpress/examples/setup-mysql –n wordpress localhost
```

생성된 데이터베이스는 사용자 계정, 게시물, 코멘트 및 기타 사항들을 저장하는데 사용된다. 스크립트가 완료되면 http://localhost에 접속하라는 안내를 받게 되는데 주소가 약간 틀렸다. 웹 브라우저로 http://localhost/wordpress에 접속해 본다(그림 10-6 참조).

그림 10-6 - NetSurf 웹 브라우저로 워드프레스 설정

웹 브라우저에 표시된 폼에서 사이트를 설명할 수 있는 이름을 입력하고 사이트를 이용할 사용자를 위한 패스워드를 설정한다. 보안을 위해 사용자 이름 필드를 admin 대신 다른 이름으로 바꾸어 준다. 모든 필드에 정보를 입력한 후 페이지 맨 밑에 있는 Install WordPress 버튼을 클릭한다.

몇 분 후 설치가 완료되고 완료 페이지가 웹 페이지에 로드 된다. 앞서 입력했던 사용자 이름과 패스워드를 입력하고 페이지 맨 아래의 Log In 버튼을 클릭하여 워드프레스에 로그인 한다(그림 10-7 참조).

그림 10-7 - 라즈베리파이에서 워드프레스에 로그인

www.keti.or.kr

다른 컴퓨터에서 워드프레스에 접근하기 위해서는 추가적인 설정 파일을 작성해야 된다.

로컬 액세스를 위한 설정 파일을 다음과 같이 링크로 걸어 주면 된다.

```
$ sudo ln –s /etc/wordpress/config-localhost.php /etc/wordpress/config-ipaddress.php
```

ipaddress 부분에 라즈베리파이의 ip 주소 또는 호스트명을 써준다.

외부로부터의 접근을 위한 설정을 마치기 위해서는 워드프레스 대시보드의 왼편에 있는 설정 메뉴에서 General Settings를 선택한 후 라즈베리파이의 ip 주소 또는 호스트명에 맞도록 URL을 수정해 주어야 한다(그림 10-8 참조).

그림 10-8 - 워드프레스 Genaral Settings 메뉴의 URL 변경

www.ketti.or.kr

팁: 워드프레스는 메모리 매우 많이 사용하기 때문에 되도록이면 라즈베리파이를 GUI 없이 실행하는 것이 좋다. 대시보드에 접근할 때는 다른 컴퓨터에서 웹 브라우저를 이용하여 http:;/ipaddress/wordpress/wp-login.php로 접속하도록 한다. 라즈베리파이에서 직접 대시보드에 접속하면 굉장히 시간이 오래 걸린다.

워드프레스는 보안 취약성 보완을 위해 자주 업데이트해주는 것이 좋다. APT를 사용하여 업데이트하는데 필요한 권한이 결여되어 있다. 이를 수정하려면 다음과 같이 한다.

```
$ sudo chown –R www-data /usr/share/wordpress
```

이 명령은 아파치 웹 서버에 의해 사용되는 www-data 사용자에게 /usr/share/wordpress 아래에 있는 파일들을 수정할 수 있는 권한을 부여한다. 이렇게 해주어야 워드프레스 대시보드에서 자동 업그레이드를 선택했을 때 제대로 동작하게 된다.

워드프레스에 대해 더 자세한 정보가 필요하면 http://www.wordpress.org 사이트를 방문해보기 바란다.

본 문서는 자유롭게 배포 가능합니다.
다만 상업적 목적 및 문서 수정 배포시
한국전자통신교육원(kks@ketti.or.kr)로 문의 부탁드립니다.

www.ketti.or.kr

라즈베리파이를 활용한
오픈 소스 R&D 플랫폼
(11,12,13장)

www.ketti.or.kr

한국전자통신교육원

본 문서는 자유롭게 배포 가능합니다.
다만 상업적 목적 및 문서 수정 배포시
한국전자통신교육원(kks@ketti.or.kr)로 문의 부탁드립니다.

목차

Chapter 1. 라즈베리파이 소개 6

프로그래밍의 재미 6

라즈베리파이의 역사 6

라즈베리파이란? 7

라즈베리파이로 무엇을 할 수 있나? 9

Chapter 2. 라즈베리파이와의 첫 만남 10

ARM 대 x86 10

리눅스 대 윈도우 11

라즈베리파이에 포팅 된 운영체제 11

주변기기 연결 12

SD카드에 운영체제 이미지 써넣기 15

USB저장 장치 연결 16

네트웍 연결 16

전원 연결 17

Chapter 3. 리눅스 시스템 관리 18

리눅스 개요 18

리눅스 기초 18

데비안(Debian) 소개 19

외부 저장 장치 사용 22

사용자 계정 만들기 23

www.ketti.or.kr

파일 시스템 구성 24
소프트웨어 설치 및 제거 26
Chapter 4. 문제 해결 30
키보드 및 마우스 진단 30
전원 진단 30
디스플레이 진단 32
부팅 진단 32
네트웍 진단 33
비상용 커널 35
Chapter 5. 네트웍 설정 36
유선 네트워킹 36
무선 네트워킹 39
Chapter 6. 파티션 관리 49
새 파티션 작성 49
파티션 크기 변경 52
수작업으로 파티션 크기 변경 53
더 큰 SD카드로 옮기기 57
Chapter 7. 하드웨어 구성 설정 60
하드웨어 설정 – config.txt 60
디스플레이 설정 변경 61
부팅 옵션들 64
라즈베리파이 오버 클럭킹 64

www.ketti.or.kr

메모리 파티셔닝 – start.elf 67

소프트웨어 설정들 – cmdline.txt 68

Chapter 8. 홈씨어터로 활용 71

콘솔에서 음악 플레이 71

Rasbmc를 이용한 전용 HTPC 구축 73

인터넷 미디어 스트리밍 74

로컬 네트웍 미디어 스트리밍 76

Rasbmc 설정 변경 78

Chapter 9. 업무용으로 활용 80

클라우드 기반 앱 사용 80

OpenOffice.org 사용 81

Gimp를 이용한 이미지 편집 82

Chapter 10. 웹 서버로 활용 84

WordPress 설치 87

Chapter 11. 스크래치로 프로그래밍 92

예제 1: Hello World 92

예제 2: 애니메이션 및 사운드 96

예제 3: 간단한 게임 98

로보틱스 및 센서들 103

더 읽어보기 104

Chapter 12. 파이썬 프로그래밍 105

파이썬 소개 105

www.ketti.or.kr

예제 1: Hello World 105

파이썬 프로그램 실행 가능하게 만들기 109

예제 2: 주석, 입력, 변수 및 반복문 110

예제 3: pygame을 이용하여 게임 만들기 113

예제 4: 파이썬과 네트워킹 121

더 읽어보기 127

Chapter 13. 하드웨어 해킹 128

전자 부품 128

저항 값 읽기 130

GPIO 포트 131

파이썬에서 GPIO 사용하기 134

GPIO 출력: LED 밝히기 136

GPIO 입력: 버튼 읽기 140

브레드보드 벗어나기 144

납땜에 대한 간단한 가이드 146

참조 A: Python Recipes 151

Raspberry Snake (Chapter 12. 예제 3) 151

IRC User List (Chapter 12. 예제4) 155

GPIO Input and Output (Chapter 12) 157

참조 B: HDMI 디스플레이 모드 목록 159

참조 C: 참고 문헌 166

참고 문헌 166

www.ketti.or.kr

참고 사이트 166

www.ketti.or.kr

Chapter 11. 스크래치로 프로그래밍

라즈베리파이의 주 사용 대상자는 어른만 있는 것이 아니다. 어린이들도 포함된다. 어린이들이 단순히 다른 사람들이 만들어준 것만을 소비하는 소비자가 아닌 주체적인 생산자가 되기 위해서는 프로그래밍을 할 수 있어야 한다. 그 방법을 제공해 주는 것이 바로 스크래치(Scratch)이다.

공식 스크래치 웹사이트 http://scratch.mit.edu에 가면 윈도우와 애플 OS X 버전도 다운로드 할 수 있다.

1. 예제 1: Hello World

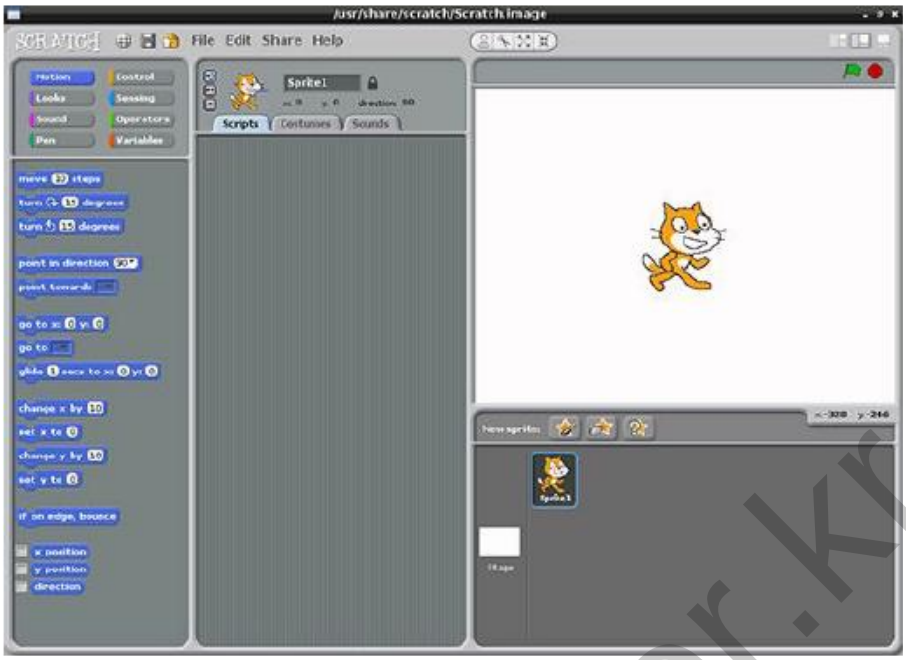
전통적인 프로그래밍 언어들과는 달리 스크래치는 사용자들이 print 또는 inkey$ 같은 명령어를 외우도록 하지 않는다. 대신 거의 모든 것들이 코드 블록을 논리 패턴 주위에 끌어다 놓는 방식으로 이루어진다.

스크래치를 시작하기 위해서는 데스크톱에 있는 아이콘을 더블 클릭하거나 Programming 메뉴에 있는 스크래치 항목을 선택한다. 잠시 후에 스크래치 주 인터페이스가 나타날 것이다(그림 11-1 참조).

스크래치 인터페이스는 여러 개의 창으로 나뉘어 있다. 왼편에는 프로그램을 작성하는데 사용하는 여러 가지 코드 블록들을 담고 있는 블록 팔레트가 있다. 스프라이트라고 불리는 프로그램 내의 개체들의 목록이 오른쪽 아래에 스프라이트가 나타날 스테이지의 컨트롤과 나란히 표시되어 있다. 스테이지는 오른편 위에 있는데 이곳에서 실행되는 프로그램을 볼 수 있다. 가운데는 프로그램을 작성하는 곳이다.

그림 11-1 - 라즈베리파이에서 실행되는 스크래치 주화면

www.ketti.or.kr

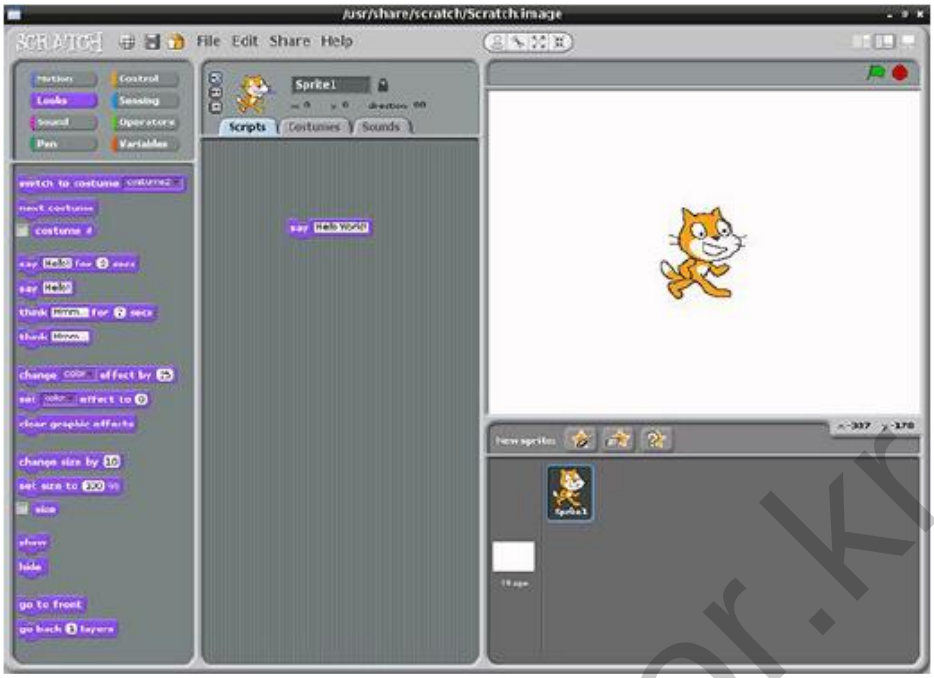


사용자가 시작을 쉽게 할 수 있도록 새 스크래치 프로젝트는 이미 비어 있는 스테이지와 한 개의 스프라이트를 포함하고 있다. 하지만 프로그램이 작성되어 있지 않기 때문에 초록색 깃발 모양 아이콘을 클릭하여 프로그램을 실행해도 아무 일도 일어나지 않을 것이다.

Hello World 프로그램을 작성하기 위해서는 화면의 좌측에 있는 블록 팔레트를 Looks 버튼을 클릭하여 Looks 모드로 변경할 필요가 있다. 표시된 Looks블록들을 읽어 내려가다 보면 중간에 say Hello! 블록이 있다. 이 블록을 끌어서 Scripts라는 라벨이 붙어 있는 창의 가운데 부분에 가져다 놓는다. 그런 다음 블록의 한 가운데 있는 Hello!라는 문구를 Hello World!로 변경한다(그림 11-2 참조).

그림 11-2 - 스크래치 프로그램에 처음 블록 놓기

www.ketti.or.kr

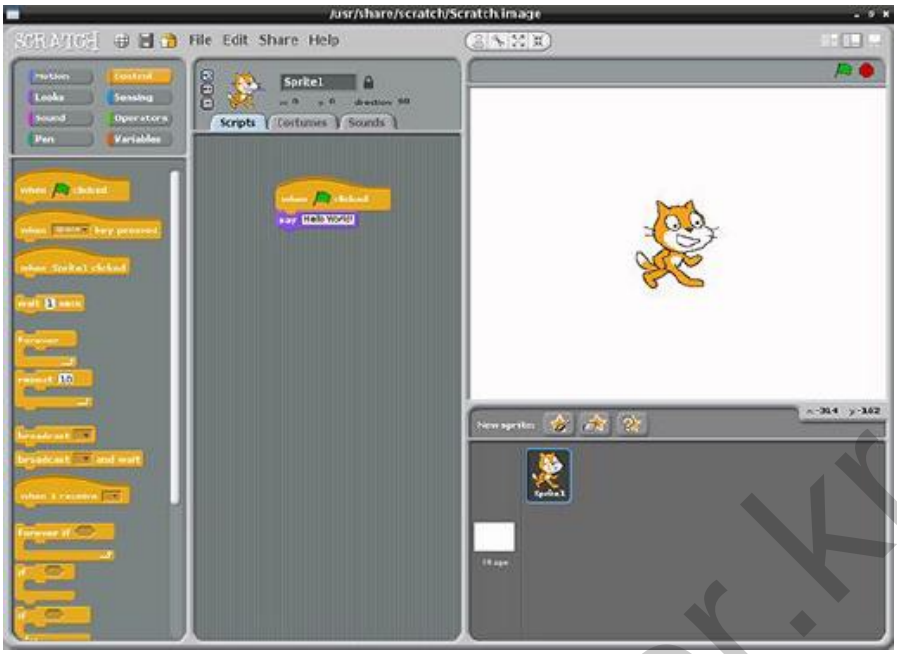


초록색 깃발을 다시 클릭해보아도 프로그램은 아무것도 하지 않을 것이다. 왜냐하면 프로그램은 고양이 스프라이트가 무엇을 해야 하는지는 알지만 언제 해야 할지를 모르기 때문이다. 이벤트는 trigger 블록을 필요로 한다. trigger블록은 블록 팔레트의 Control섹션에 있다.

Control모드 버튼을 클릭하여 Control 모드로 바꾼 후 when (flag icon) clicked 로 표시되어 있는 항목을 드래그하여 보라색 say 블록의 바로 위에 놓는다(그림 11-3 참조). 충분히 가깝게 놓으면 자동으로 그 위치에 있는 블록에 직소 퍼즐처럼 달라 붙는다

그림 11-3 - 스크래치에서 Looks 블록에 결합된 제어 블록

www.kettl.or.kr



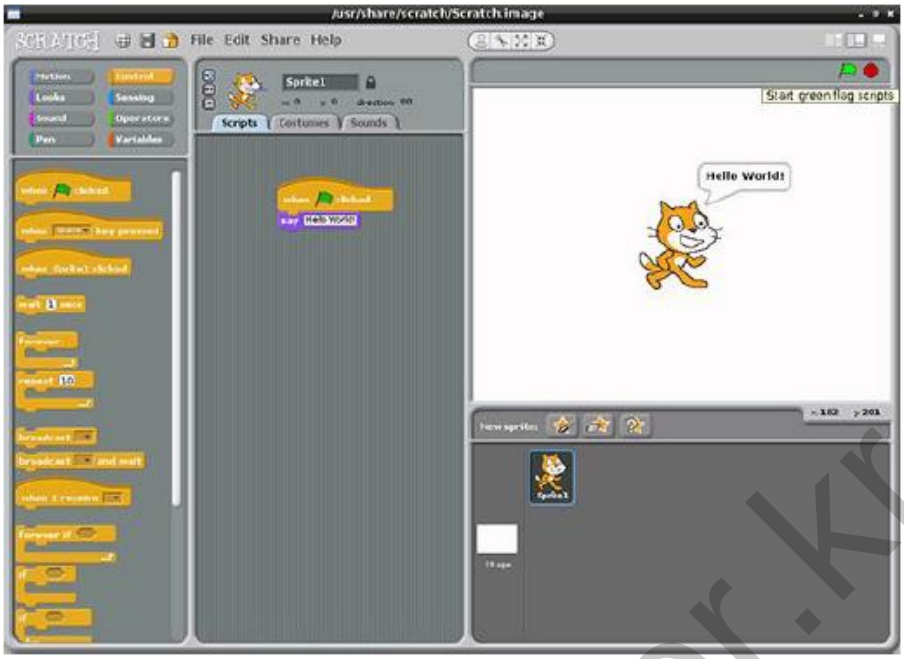
여러 개의 블록들을 결합한다는 것이 스크래치의 핵심 개념이다. 방금 가져다 놓은 Control 블록의 위쪽에는 연결 구멍이 없는 것을 알 수 있다. 이 Control 블록은 다른 연속된 블록들을 격발하도록 만들어져 있고 다른 블록에 의해 격발되지 않으므로 위쪽이 다른 블록을 연결할 수 있도록 되어 있지 않다. 반면에 say 블록의 아래쪽은 다른 블록의 위쪽과 결합될 수 있는 구멍을 가지고 있다.

두 블록만 있는 상태에서 오른쪽 위에 있는 초록색 깃발 아이콘을 클릭하면 이번에는 고양이 입에서 말 풍선이 표시될 것이다(그림 11-4 참조).

다음 예제를 진행하기 전에 File 메뉴로 가서 현재 프로젝트를 저장한다. 스크래치는 한 번에 하나의 프로젝트만을 열 수 있다.

그림 11-4 - 스크래치에서 Hello World 프로그램 실행

www.ketti.or.kr



2. 예제 2: 애니메이션 및 사운드

스크래치의 File 메뉴에서 New를 선택하여 새로운 프로젝트를 만든다. 기본 스프라이트가 하나 제공된다. 이 스프라이트를 제어해 볼 것이다.

스크래치에서 간단한 애니메이션 제어를 위해, 블록 팔레트의 Motion 섹션을 사용해 볼 것이다. 새 프로젝트를 시작했을 때 Motion 섹션이 기본 팔레트이다. Move 10 steps 블록을 스크립트 영역에 끌어다 놓는다. 이름이 시사하는 것처럼 이 블록은 선택된 스프라이트가 현재 바라보고 있는 방향으로 10 걸음 움직이도록 한다. 스크래치의 기본 스프라이트는 언제나 오른쪽을 향하고 있으므로 오른쪽으로 10 걸음 움직이도록 하는 역할을 하게 된다.

10 걸음은 너무 작으므로 10을 클릭하여 30으로 바꿔준다. 이번에는 Sound 블록 팔레트로 바꾼 후 play sound meow 블록을 스크립트 영역으로 끌어 기존의 move 블록 아래에 결합시킨다. 고양이가 이 위치에 잠시 서 있도록 Control 블록 팔레트에서 wait 1 secs 블록을 끌어다 붙인다. 이 블록이 없으면 마지막 위치에 도달했을 때 처음 위치로 순간이동을 해버린다.

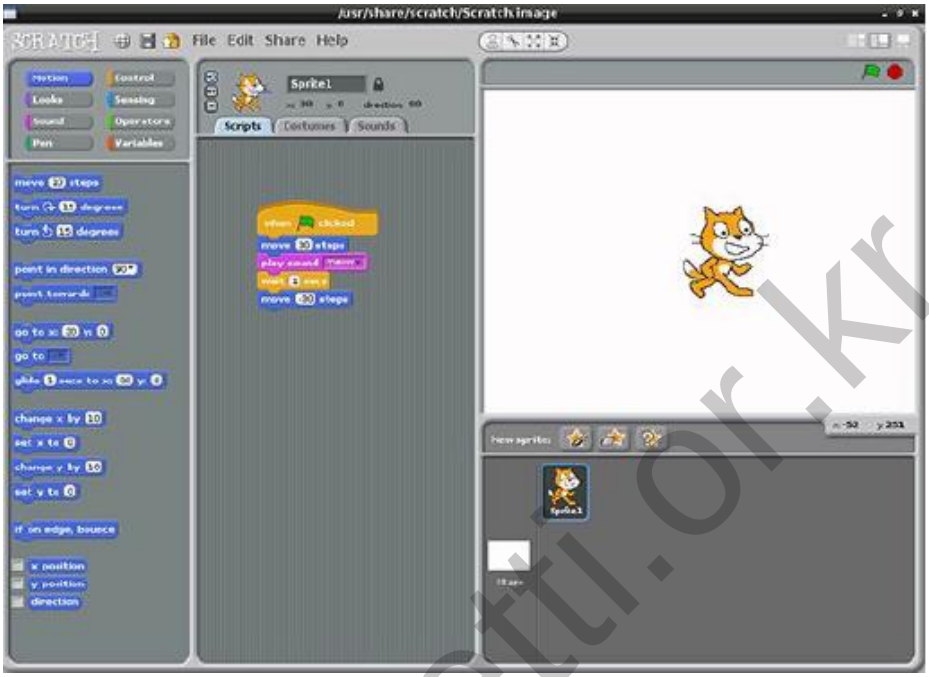
고양이 스프라이트가 Stage 가장자리에서 사라지지 않고 여러 번 반복해서 실행될 수 있도록 play sound 블록 아래에 move 10 steps 블록을 추가하고 10을 -30으로 수정한다. -30은 30과는 반대 방향으로 스프라이트를 움직이게 한다.

마지막으로 Control 블록 팔레트에서 when (flag icon) clicked 블록을 끌어다 스크립트 영역의 블록들 맨 위에 연결한다(그림 11-6 참조). 스크린 오른쪽 위의 초록 깃발 아이콘을 클릭하면 프로

www.ketti.or.kr

그램이 실행될 것이다. 소리를 들으려면 라즈베리파이에 스피커 또는 헤드폰을 연결하는 것을 잊지 말 것!

그림 11-5 - 스크래치에서의 완전한 애니메이션 프로그램

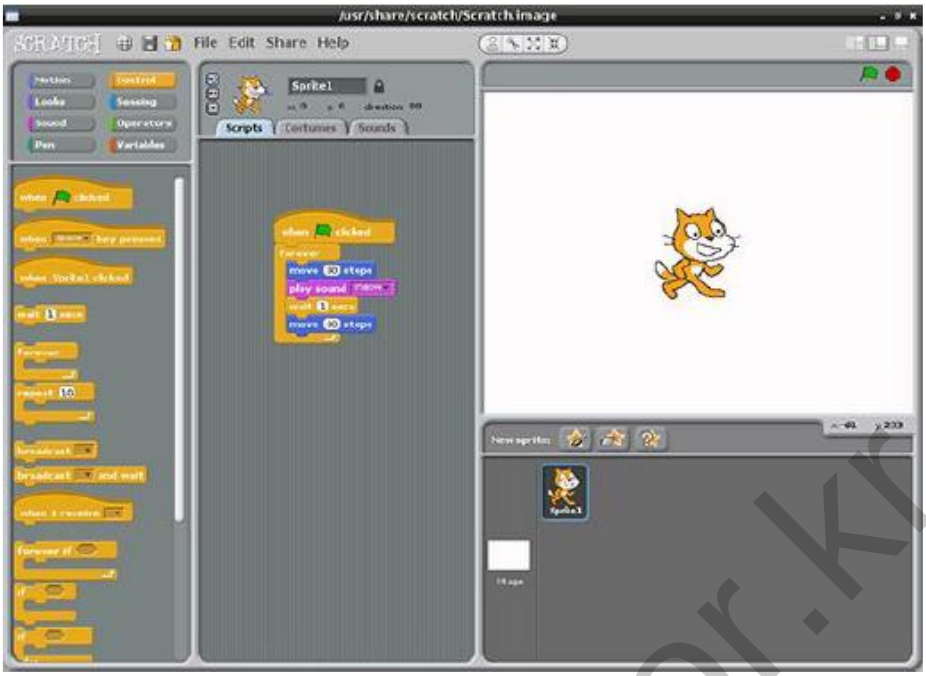


이 간단한 애니메이션 프로그램은 다양한 방식으로 확장이 가능하다. 우측의 stage 바로 밑에 있는 New Sprite 옵션을 사용하면 프로그래머가 독립적으로 움직이고 소리를 낼 수 있는 스프라이트를 더 추가할 수 있다. 처음 예제에서 say 블록을 추가했던 것 대신에 think 블록을 추가하면 말 풍선이 아닌 생각 풍선이 표시된다. 애니메이션 되는 만화를 만들 수 있을 것이다.

더욱 중요한 것은 겨우 다섯 개의 블록만으로 프로그래밍의 주요 개념을 알려준다는 것이다. 스프라이트가 전진, 후진하고, 사운드를 재생하고, 프로그램의 지연 개념도 다루고 있다. 여기에 Control 블록 팔레트에서 forever 블록을 추가해 보자(그림 11-5 참조). 그러면 프로그램은 루프를 가지게 되어 루프에 속한 목록들을 무한 반복하게 된다. stage의 위에 있는 빨간색 stop버튼을 클릭하면 프로그램을 멈출 수 있다.

그림 11-6 - 간단한 스크래치 애니메이션에 무한 반복 추가

www.ketti.or.kr



3. 예제 3: 간단한 게임

스크래치를 이용하면 간단한 애니메이션을 실행할 수 있는 한편 키보드 입력을 읽게 하여 사용자와의 상호작용성도 부여할 수 있다. 앞에서 해 본 간단한 애니메이션 제어와 결합하면 간단한 게임을 만들 수도 있다. 동시에 스프라이트 충돌과 if 문의 개념도 도입하기로 하자.

새 스크래치 프로젝트를 시작한 후 move 10 steps 블록을 스크립트 영역에 끌어 놓는다. Control 블록 팔레트로 가서 move블록 위에 있는 when space key pressed 블록을 끌어다 위에 붙인다.

space 단어 뒤의 아래 화살표 버튼을 클릭하여 키 목록을 펼친 후 right arrow를 선택한다.

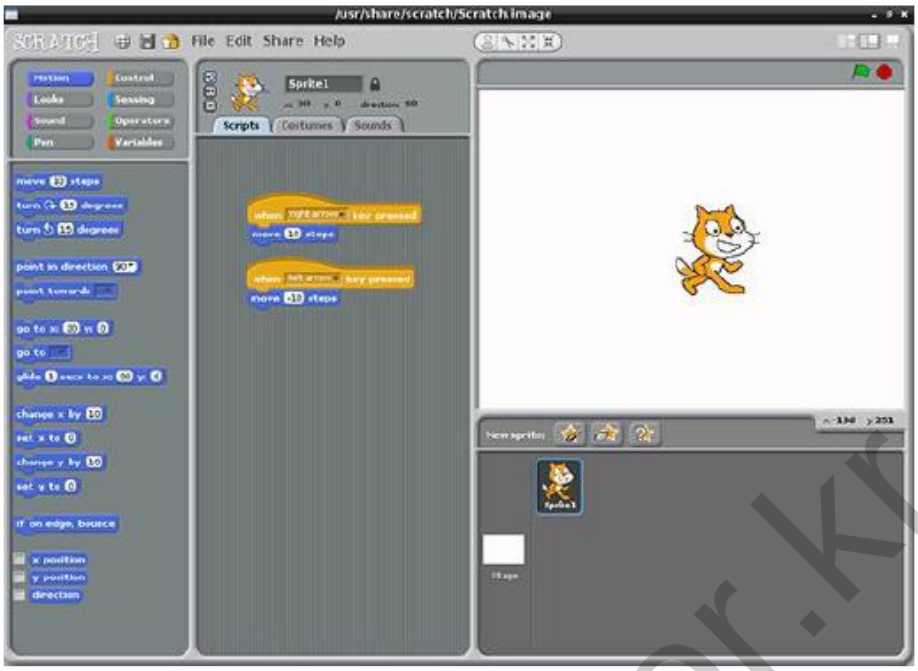
다시 Control 블록 팔레트에서 when space key pressed 블록을 끌어다 기존 블록들 조금 아래에 놓은 후 space를 left arrow로 바꾼다.

Motion블록 팔레트에서 move 10 steps 블록을 끌어다 when left arrow key pressed 블록아래에 붙이고 move 10 steps를 move -10 steps 바꾼다.

이제 왼쪽 또는 오른쪽 화살표 키를 누르면 고양이가 키에 따라 좌/우로 움직이는 것을 볼 수 있다(그림 11-7 참조).

그림 11-7 - 스크래치에서 스프라이트의 움직임을 제어하기 위한 입력 블록 사용

www.kerti.or.kr



스크린 오른쪽 아래에 있는 스프라이트 팔레트의 위, 세 개의 버튼 중 가운데에 있는 New Sprite From File 버튼을 클릭한다. 대화 상자가 나타나 스프라이트를 선택하도록 할 것이다. Things 폴더를 더블 클릭한 후 Cheesy-Puffs를 더블 클릭한다(그림 11-8 참조). 이렇게 하면 스프라이트 팔레트에 새로운 스프라이트가 나타나고 이 개체는 게임 내에서 제어가 가능한 새로운 개체로 사용할 수 있다.

팁: 스크래치 언어는 원래 멀티-쓰레드 기반이며 부분적으로 개체 지향적이다. 이 말의 의미는 프로그램 내의 스프라이트를 포함한 각 개체는 각자의 코드를 가지고 있으며 코드의 각 섹션은 다른 블록과는 동시적이고 독립적으로 실행된다는 의미이다. 적절히 사용되기만 하면 이러한 특성들을 이용하여 상당히 복잡한 프로그램들도 만들 수 있다.

그림 11-8 - 스크래치 게임 프로젝트에 Cheesy-Puffs 스프라이트 추가

www.kett.or.kr

새로 추가된 스프라이트는 기본적으로 스테이지의 가운데에 나타난다. 결과적으로 기존의 고양이 스프라이트를 가리게 된다. 새 스프라이트를 클릭하여 고양이의 오른쪽에 끌어다 놓는다.

Cheesy-Puffs 그릇이 우리의 고양이가 먹기에는 너무 크므로 스테이지의 오른쪽 위에 있는 Shrink Sprite버튼(안을 향하는 네 개의 화살표 모양 아이콘)을 클릭한다.

그러면 커서의 모양이 클릭한 버튼 아이콘과 같은 모양으로 변한다(Grow Sprite 버튼을 클릭해도 기능만 반대이고 같은 식으로 마우스 커서가 변한다). 이 상태에서 Cheesy-Puffs 스프라이트를 클릭하여 크기를 줄인다. 여러 번 클릭하면 크기가 계속 줄어든다. 적당한 크기로 줄었으면 스테이지 밖의 아무 곳이나 클릭하면 마우스 커서가 원래 형태로 되돌아 온다. 그런 다음 Cheesy-Puffs 그릇을 클릭하여 스테이지의 오른쪽 가장자리에 끌어다 놓는다.

화살표 키를 사용하여 고양이 스프라이트를 Cheesy-Puffs 스프라이트 쪽으로 움직여 보자. 지금은 두 스프라이트가 겹쳐지더라도 아무 일도 일어나지 않는다. 두 스프라이트가 만났을 때 (스프라이트 충돌) 무언가를 해주기 위해 Sensing 블록을 사용해보자.

Cheesy-Puffs 스프라이트가 활성화된 상태(스크립트 창의 위에서 이미지가 나타나 있다, 없으면 스테이지에 있는 스프라이트를 더블 클릭하면 된다)에서 Sensing 블록 팔레트를 클릭한 후 touching ? 블록을 끌어다 스크립트 창에 가져다 놓는다.

물음표 뒤의 아래 화살표를 클릭하여 Sprite1(고양이)을 선택한다. 이제 이 블록은 두 스프라이트가 충돌했을 때 실행될 것이다.

www.ketri.or.kr

팁: 스크립트 창에 있는 이미지 뒤에 있는 상자를 클릭하여 스프라이트의 이름을 변경할 수 있다. Cat, CheesyPuffs와 같이 스프라이트와 관련된 이름을 주면 프로그램을 이해하기가 훨씬 쉽게 된다.

touching Sprite1 블록을 보면 위나 아래쪽에 다른 블록을 연결할 수 있는 연결 구멍이 없는 것을 알 수 있다. 또한 다이아몬드 모양으로 생겨서 순서도의 의사결정 지점 모양과 유사한 것을 알 수 있다. 우연히 그런 것이 아니라 Sensing 블록의 대다수는 Control 블록에 내장되어 동작하기 때문이다.

Control 블록 팔레트로 가서 if 블록을 보자. 마치 찌그러진 C 자처럼 생겼다. if 블록 내부에 touching Sprite1? 블록처럼 생긴 부분이 있는 것을 주목하자. if 블록을 스크립트 창에 끌어다 놓는다. 그 다음 touching Sprite1? 블록을 끌어 if 블록 내부의 다이아몬드 형태로 빈 곳에 가져다 붙인다. 결과는 두 가지 색깔의 블록을 합쳐 if touching Sprite1?로 읽을 수 있다.

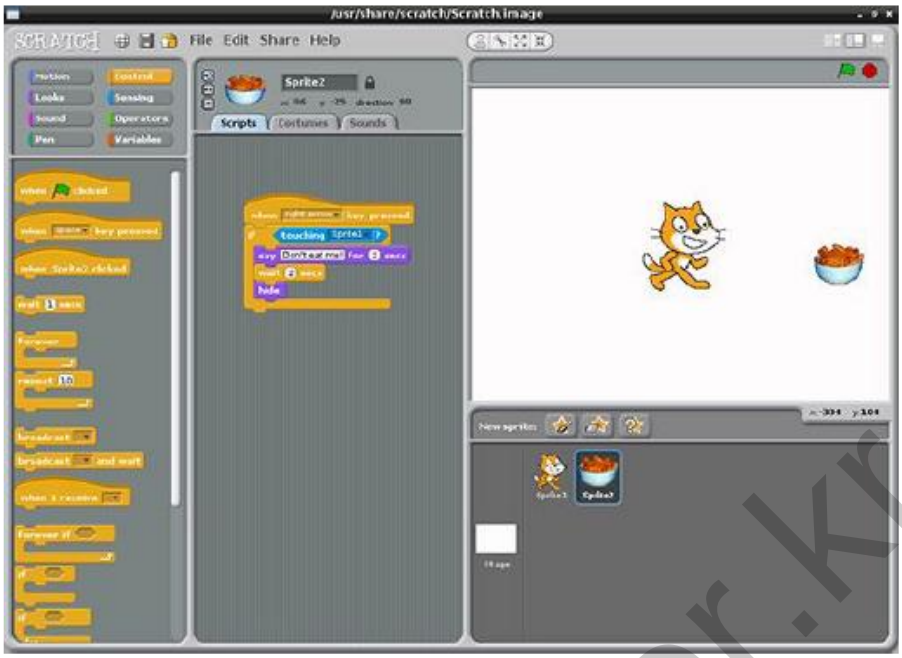
이 것은 프로그램에서 if 조건 문을 나타낸다. 즉 이 지점이 실행되면 if 문의 조건이 참이 될 때 그 안의 블록들이 실행된다. 이 경우에 조건은 Sprite2가 Sprite1과 접촉했을 경우를 말한다. Operators 블록 팔레트에 있는 and, or, not 논리 블록들을 사용하면 좀 더 복잡한 시나리오도 수용이 가능하다.

Looks 블록 팔레트에서 say Hello! For 2 secs 블록을 끌어다가 if touching Sprite2 조건의 가운데에 놓는다. 텍스트를 Don't eat me!로 바꾸고 Control 블록에서 wait 1 secs 블록을 끌어다가 그 밑에 붙인 후 값을 2로 변경한다.

when space key pressed 블록을 가져다가 맨 위에 붙이고, 값을 when right arrow key pressed 로 바꾼다. 마지막으로 Looks 팔레트에서 hide 블록을 끌어다가 그림 11-9와 같이 블록 리스트의 끝을 마무리 한다.

그림 11-9 - 스크래치 if 블록으로 Cheesy-Puffs 제어

www.ketti.or.kr

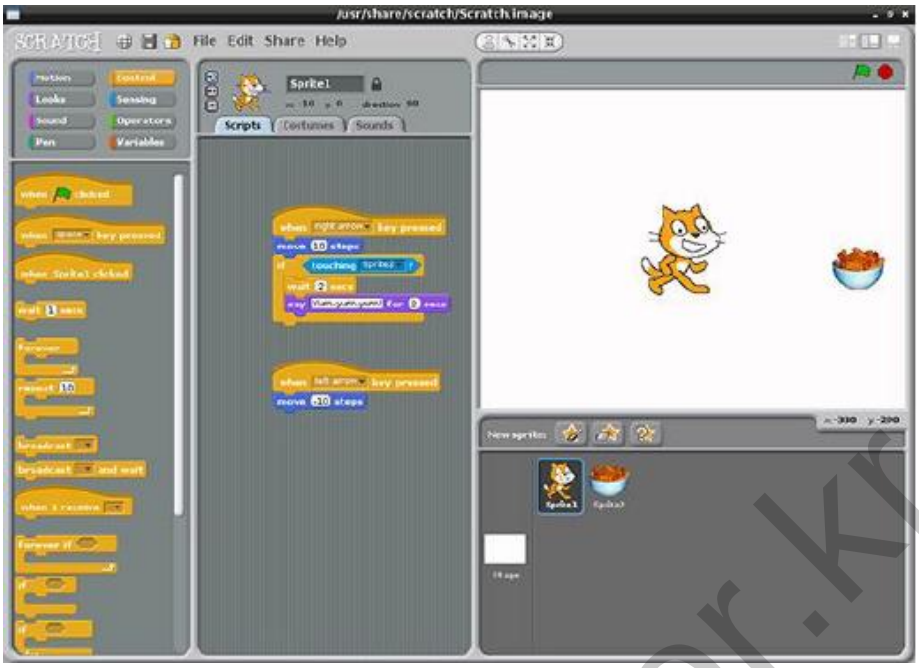


고양이의 스크립트를 편집하기 스테이지에 있는 고양이 스프라이트를 더블 클릭한다. Cheesy-Puffs 스프라이트를 위한 스크립트가 사라지지만 염려할 필요는 없다. 사라진 것이 아니라 따로 저장이 되어 있으며 해당 스프라이트를 편집할 때만 나타나게 되어 있다.

Control 팔레트에서 또 다른 if 블록과 touching ? 센싱 블록을 끌어다가 if touching Sprite2?가 되도록 붙여 놓는다. 이 블록 안에 wait 1 secs Control 블록을 가져다 넣고 값을 2 로 바꾼다. 그 밑에 Looks 블록에서 say Hello! for 2 secs 블록을 가져다 붙이고 메시지를 Yum-Yum-Yum!으로 바꾼다. 마지막으로 층층이 쌓인 블록 묶음 전체를 끌어 올려 기존의 when right arrow key pressed 블록의 move 10 steps 밑에 붙인다. 완료된 스크립트의 형태는 그림 11-10과 같다.

그림 11-10 - 간단한 스크래치 게임의 고양이 스프라이트를 위한 완성된 코드

www.ketti.or.kr



키보드의 오른쪽 화살표 키를 눌러 고양이를 Cheesy-Puffs 쪽으로 움직이기 시작하면 게임이 시작된다. 고양이가 Cheesy-Puffs에 닿으면 말풍선이 나타나고 그릇이 사라질 것이다.

이 예제는 몇 가지 중요한 프로그래밍 개념들을 소개하는데 효과적이기는 하지만 게임을 작성하는 최선의 방법이라고 하기는 어렵다. 스크래치에는 하나의 개체에 부착된 코드가 다른 개체에 부착된 코드와 대화할 수 있도록 해주는 메시지 브로드캐스트 시스템을 포함하고 있어서 이를 이용하면 개체의 접촉 시에 주의 깊게 지연시간을 넣지 않아도 훨씬 간편한 방법으로 개체 접촉 결과를 만들 수 있다.

4. 로보틱스 및 센서들

순수한 소프트웨어 기반의 입력과 출력 이외에도, 스크래치를 센서보드와 로보틱스 시스템들을 사용하는 외부 하드웨어와 묶는 것도 가능하다. 멀티 쓰레디드 속성과 강력한 프로세스간 메시징 시스템 덕분에 스크래치는 놀라울 정도로 진보된 로보틱스 엔진을 만드는데 사용될 수 있으며 라즈베리파이를 간단한 로봇의 심장부에 가져다 놓을 수 있는 다른 무엇보다도 가장 간단한 방법이다.

훨씬 강력하지만 상당히 복잡한 프로그래밍 언어인 파이썬과는 달리 스크래치는 외부 하드웨어와의 커뮤니케이션을 위해 라즈베리파이의 GPIO 포트를 이용할 수 없다. 따라서 물리적인 세계와의 대화를 위해서는 약간의 부가적인 하드웨어를 필요로 한다.

PicoBoard를 이용한 센싱

www.kett.or.kr

SparkFun에 의해 디자인된 PicoBoard는 스크래치를 실행하는 컴퓨터 연결할 수 있는 애드온 모듈로써 다양한 슬라이더 입력, 광센서, 버튼, 소리 센서, 외부 전자 장비를 모니터 할 수 있는 4쌍의 악어 클립을 제공한다.

직렬 통신을 사용하는 오리지널 모델과 USB통신을 사용하는 변형 모델, 두 가지 종류의 PicoBoard가 있다. 라즈베리파이는 후자를 지원한다. 단순히 라즈베리파이의 USB포트에 연결하고 스크래치를 재시작 하기만 하면 스크래치 인터페이스에서 다양한 기능을 사용할 수 있다.

자세한 내용은 http://www.sparkfun.com/products/10311 에서 확인하기 바란다.

LEGO를 이용한 로보틱스

애드온 센서 모듈이외에 스크래치를 통해서 외부 로보틱스 시스템을 제어할 수도 있다. LEGO education WeDo 로보틱스 킷은 모터, 거리 센서, 속도 센서 등을 컴퓨터에 연결할 수 있도록 디자인 되어있으며 이 센서들은 스크래치에 있는 전용 블록들을 이용해서 제어할 수 있다.

어린이들에게 WeDo 킷은 로보틱스를 시작할 수 있는 뛰어난 방법을 제공한다. 정규 LEGO처럼 컴포넌트들을 같이 끼우면 되고 잘못 끼워질 염려가 없고 전기 쇼크를 걱정하지 않아도 된다. WeDo킷은 정규 LEGO 및 LEGO 테크닉 킷과도 호환되므로 쉽고 빠르게 대규모 프로젝트를 빌드하는 것도 가능한다. 스크래치와 LEGO education WeDo 킷에 대한 더 자세한 정보는 http://info.scratch.mit.edu/WeDo 사이트를 참고하기 바란다.

5. 더 읽어보기

http://info.scratch.mit.edu/support

http://scratch.mit.edu/forums/

http://scratch.mit.edu/latest/shared

www.ketii.or.kr

Chapter 12. 파이썬 프로그래밍

라즈베리파이란 이름의 앞의 절반은 새로운 컴퓨팅 시스템의 이름에 과일을 이용하는 오래된 관행(클래식 마이크로 컴퓨터 이름인 Acorn, Apricot, Tangerine, 좀 더 인지도가 있는 최근의 브랜드로는 Apple, BlackBerry 가 있다)으로부터 유래했고 나머지 반은 파이썬 프로그래밍 언어에 대한 예우에서 비롯되었다.

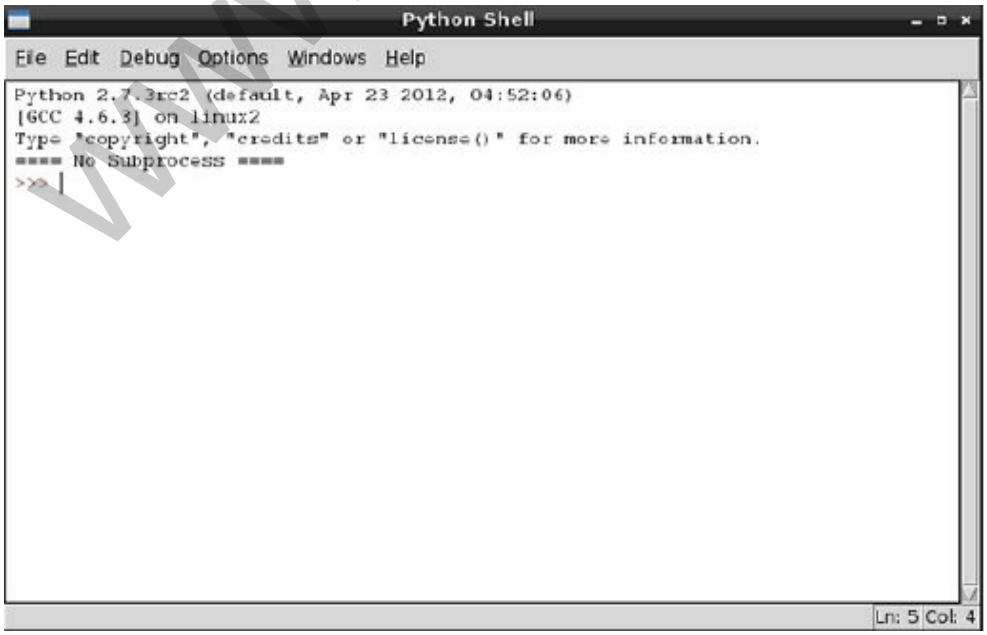
파이썬 소개

유연하면서도 강력한 파이썬은 원래 1980년대에 컴퓨터 과학과 수학을 위한 국립 연구소(National Research Institute for Mathematics and Computer Science)의 Guido van Rossum에 의해 ABC언어의 뒤를 잇는 후계자로 개발되었다. 처음 소개된 이후 파이썬은 가독성을 높이는데 초점을 두고 개발된 명확하고 표현력 있는 문법으로 사용자를 증가시켜 왔다.

예제 1: Hello World

데비안 배포판의 데스크톱 환경에 있는 Programming 메뉴에서 IDLE를 실행한다. IDLE는 기본적으로 파이썬 쉘 모드(그림 12-1 참조)로 열린다. 따라서 초기 창에 입력한 겉은 무엇이던지 즉시 실행된다. 나중에 실행할 수 있는 새 파이썬 프로젝트를 만들려면 File 메뉴를 클릭하여 New Window를 선택하여 새 창을 연다.

그림 12-1: IDLE 파이썬 쉘 창



www.ketti.or.kr

팁: IDLE 대신 IDLE3를 선택하면 파이썬 3.0 버전이 로드 된다. 이 버전에는 IDLE에 포함된 파이썬에는 포함되지 않은 기능이 포함되어 있다. 하지만 그 기능들은 이 장에서는 사용되지 않는다. 어느 버전을 실행해도 상관없으며 이 예제는 양쪽 모두에서 실행된다.

파이썬 프로그램을 시작할 때는 셔뱅으로 알려진 줄로 시작하는 것이 좋은 습관이다. 셔뱅이란 이름은 행 처음에 오는 # 과 ! 글자로부터 유래했다. 이 행은 운영체제에게 파이썬 파일들을 어디에서 찾아야 하는지를 알려준다. 이 행은 IDLE 내에서 실행되거나 터미널에서 별도로 파이썬을 호출하는 프로그램인 경우에는 전혀 필요가 없다. 프로그램의 파일명을 직접 호출하여 실행되는 경우에 필요한 것이다.

파이썬 실행파일이 어디에 설치되었는지 상관없이 프로그램이 실행될 수 있도록 하기 위해 사용자 프로그램의 첫 줄은 다음과 같이 시작해야 된다.

```
#! /usr/bin/env python
```

이 행은 운영체제가 파이썬의 위치를 찾을 때 라즈베리파이에 어떤 리눅스 배포판을 사용하더라도 작동할 수 있도록 $PATH 환경변수(리눅스가 프로그램으로 실행될 수 있는 파일들의 위치를 저장해두는 곳)을 참조하도록 지시한다. $PATH 변수는 실행 가능한 파일들이 저장되어 있는 디렉터리들의 목록을 가지고 있어 사용자가 콘솔이나 터미널 창에서 실행할 프로그램명을 쳤을 때 그 프로그램의 위치를 찾는데 사용된다.

메시지를 출력하려면 파이썬의 명령을 사용해야 된다. 명령어가 의미하듯 print 명령은 출력 장치에 텍스트를 출력한다. 기본적으로 프로그램이 실행되고 있는 터미널 창이나 콘솔로 출력된다. 사용법은 간단하다. print 명령어 뒤에 따라오는 이중 따옴표 사이에 있는 모든 텍스트가 표준 출력 장치로 출력된다. 사용자의 새 프로젝트에 다음 행을 추가한다.

```
print “Hello, World!”
```

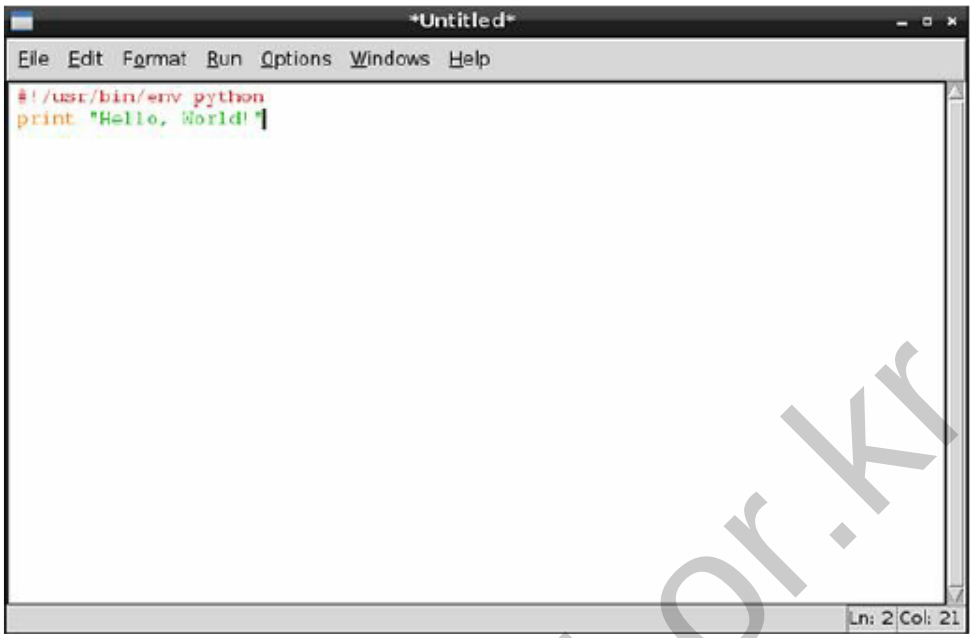
최종 프로그램은 다음과 같은 모양이 될 것이다.

```
#! /usr/bin/env python

print “Hello, World!”
```

IDLE는 문법 강조 표시 기능을 지원하므로 그림 12-3에서 보듯이 문법에 따라 각 단어의 색을 다르게 표시해준다. 지금처럼 간단한 예제에서는 그다지 필요가 없지만 복잡한 프로그램을 작성할 때는 오류가 있는 지점을 발견하는데 매우 유용한 기능이다.

www.ketu.or.kr

그림 12-2: IDLE에서의 문법에 따른 강조 표시

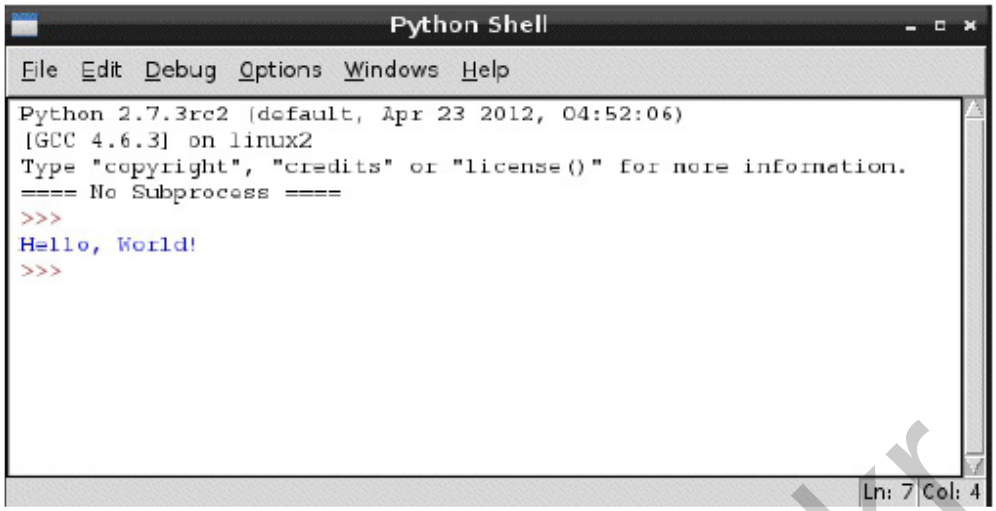


프로그램을 실행하기 전에 File 메뉴를 이용하여 프로그램을 helloworld.py란 이름으로 저장한다. IDLE는 파일 확장자에 .py를 자동으로 부여한다. 다른 확장자를 사용하더라도 파이썬이 프로그램을 실행하는 데는 상관이 없지만 .py라는 이름을 사용하면 파이썬 프로그램이라는 것을 쉽게 구분할 수 있다.

Run 메뉴에서 Run Module 을 선택하거나 F5키를 눌러 프로그램을 실행한다. 그러면 IDLE는 파이썬 쉘 창으로 되돌아가 프로그램을 실행한다. Hello, World! 메시지가 파란색으로 표시될 것이다 (그림 12-3 참조). 표시가 되지 않으면 프로그램을 학인 해봐야 된다. print 문 뒤에 있는 텍스트 메시지의 앞과 뒤에 이중 따옴표가 제대로 있는지 특히 주의해서 본다.

그림 12-3: IDLE에서 helloworld.py 실행

www.keti.or.kr



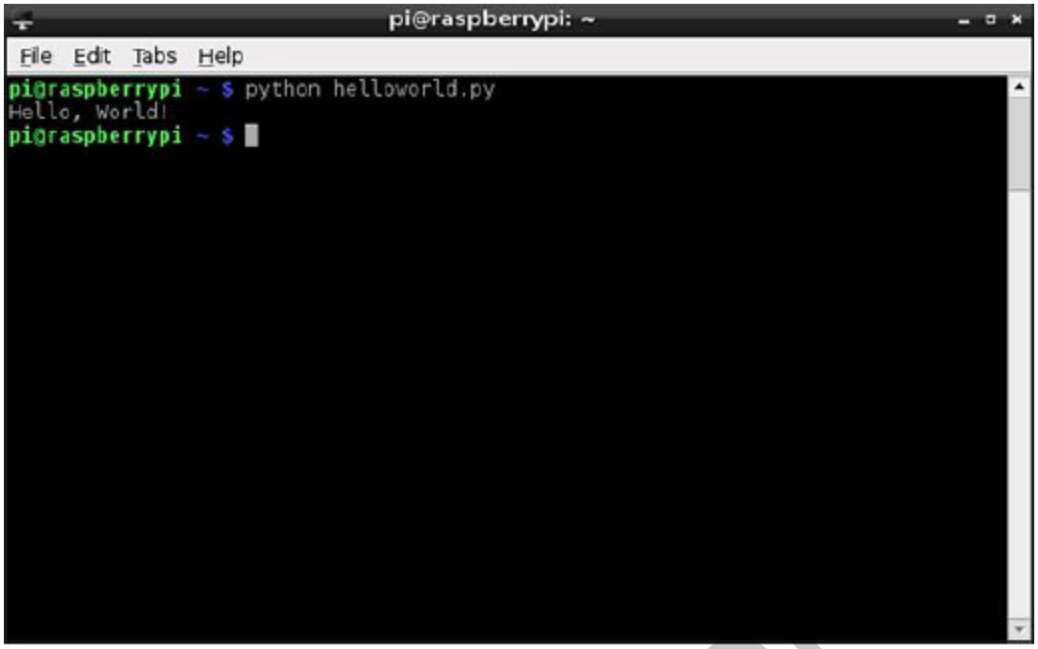
텍스트 에디터를 사용해서 프로그램을 작성하는 경우에는 터미널 창을 열고 cd 명령어를 이용하여 프로그램을 저장한 디렉터리로 이동한다(Chapter 3. 리눅스 시스템 관리 참조). 다음 명령어를 쳐서 사용자 프로그램을 실행한다.

$ python helloworld.py

이 명령어는 운영체제가 파이썬을 실행한 후 helloworld.py 파일을 로드하여 실행하도록 한다. IDLE의 파이썬 쉘과 달리 프로그램이 파일 끝까지 실행된 후 파이썬은 종료가 되고 터미널 창으로 제어가 되돌아 온다. 결과는 동일하다. Hello, World!가 표준 출력으로 출력된다(그림 12-4 참조).

그림 12-4: 터미널에서 helloworld.py 실행

www.ketti.or.kr



파이썬 프로그램 실행 가능하게 만들기

일반적으로 파이썬 프로그램을 실행하는 유일한 방법은 파이썬이 그 프로그램 파일을 열어서 실행하도록 하는 것이다. 하지만 프로그램 첫 줄에 셔뱅 행을 써두었다면 파이썬을 호출하지 않고도 프로그램을 직접 실행할 수 있다. 이 방법은 사용자가 작성한 자신의 프로그램을 터미널에서 직접 실행시킬 수 있게 만드는 유용한 방법이다. 일단 프로그램을 시스템의 $PATH 환경변수에 있는 위치로 복사한 다음에는 프로그램명을 타이핑하는 것 만으로 실행이 가능하다.

우선 파이썬 파일에 실행 가능 속성을 부여한다.

```
$ chmod +x helloworld.py
```

직접 실행 해 본다.

```
$ ./helloworld.py
```

전체 경로를 입력하지 않고 실행하려면 $PATH에 등록되어 있는 디렉터리로 복사를 해준다.

```
$ sudo cp helloworld.py /usr/local/bin/
```

이제 경로 없이 파일명 만으로 실행이 가능하다.

```
$ helloworld.py
```

www.kett.or

좀 더 기본 유틸리티처럼 사용하려면 확장자도 없애주면 된다.

```
$ sudo mv /usr/local/bin/helloworld.py /usr/local/bin/helloworld
```

이제는 helloworld라고 치는 것 만으로 실행이 가능하다.

예제 2: 주석, 입력, 변수 및 반복문

이제 좀 더 흥미로운 대화형 프로그램을 작성해 보자.

첫 줄은 셔뱅 라인을 입력한다.

```
#! /usr/bin/env python
```

둘째 줄은 코멘트를 입력한다.

```
# Example 2: A Python program
```

파이썬에서는 셔뱅라인을 제외한 # 으로 시작하는 행은 코멘트로 처리된다. 행 중간에 # 기호가 있으면 그 앞까지만 실행된다.

다음 행은 사용자에게 이름을 입력하도록 요청한다.

```
userName = raw_input("What is your name? ")
```

userName이라는 변수가 생성된다. raw_input 명령은 괄호 안의 텍스트 메시지를 표준 출력에 표시하고 키보드로부터 표준 입력을 받는다. 등호(=)는 우측의 값이나 명령어 실행 결과를 왼쪽의 변수에 대입하는 기능을 한다. 텍스트 메시지 끝에 공백을 두어야 사용자가 입력한 텍스트가 질문 텍스트에 달라붙어 표시되는 것을 방지할 수 있다.

주의: 사용자에게 입력을 요구할 때 input 명령어를 사용하면 사용자가 자신의 프로그램을 주입하여 보안상의 위협을 초래할 수 있다. 항상 raw_input 명령어를 사용해야만 의도치 않은 동작을 막을 수 있다.

사용자 이름을 username 변수에 받았으면 이제 다음 행을 추가한다.

```
print "Welcome to the program, ", userName
```

이 print 명령은 텍스트 메시지를 출력하고 뒤이어 콤마(,)로 나뉘어진 뒷부분을 출력한다. 콤마는 출력할 부분이 뒤이어 더 있음을 의미한다. 이중 따옴표로 둘러싸인 앞부분은 그대로 출력되지만

www.ketti.or.kr

그렇지 않은 뒷부분은 변수로 인식되어 그 안에 든 값이 출력된다.

팁: 출력될 문자열을 간편하게 포맷팅하는 방법은 print명령어 끝에 .format 지시자를 사용하는 것이다. .format을 써서 위의 print명령을 다시 써보면 다음과 같이 쓸 수 있다.
print "Welcome, {0}, to this program.".format(userName)

이 예제 프로그램은 점차 간단하지만 친숙한 계산기 프로그램 형태를 갖춰갈 것이다. 첫 번째 예제와 달리 이 예제는 사용자가 멈추게 하기 전까지 계속 실행되도록 할 것이다. 그렇게 하기 위해서 스크래치에서 했던 것처럼 반복문을 사용한다. 다음 두 행을 입력해서 반복문을 만든다.

```
goAgain = 1

while goAgain == 1:
```

첫 번째 행은 goAgain 변수를 만들고 그 값을 1로 설정한다. 두 번째 행은 반복문을 시작하되 goAgain이 1과 같은 동안 뒤에 오는 일련의 코드들을 반복한다. 뒤에 오는 반복될 코드들은 들여쓰기 되어야 한다.

가장 기본적인 계산기 형태로써 두 개의 숫자를 입력으로 받아 두 숫자를 사용하여 계산을 해보자. 일단 두 개의 숫자를 입력 받는다.

```
firstNumber = int(raw_input("Type the first number: "))

secondNumber = int(raw_input("Type the second number: ")
```

여기서는 두 개의 숫자를 입력 받기 위해 raw_input을 사용할 뿐만 아니라 int를 사용한다. int는 integer의 약자로 입력 받은 문자열을 숫자로 변환하도록 한다. 계산기가 단어들을 가지고 계산할 수는 없으니 분명히 중요한 부분이다.

변수들에 두 개의 숫자를 저장했으므로 이제 계산을 할 수 있다. 덧셈, 뺄셈, 곱셈, 나눗셈을 한 후 사용자에게 결과를 출력해보자.

```
print firstNumber, " added to ", secondNumber, " equals ", firstNumber + secondNumber

print firstNumber, " minus ", secondNumber, " equals ", firstNumber – secondNumber

print firstNumber, " multiplied by ", secondNumber, " equals ", firstNumber * secondNumber
```

덧셈과 밸셈은 당연히 덧셈기호와 뺄셈기호를 사용하지만 곱셈은 애스터리스크(*)를 사용한다. 문자열에 사용된 이중따옴표 사이에 공백이 없는 점을 주목하자. 파이썬은 숫자들과 문자열을 섞어

www.ketti.or.kr

서 출력할 때는 자동으로 공백을 넣어 띄워 준다. 나눗셈은 빼놓았는데(/ 기호를 연산기호로 사용한다) 이 계산기 예제는 정수만을 사용하고 있기 때문이다.

계산 부분은 완성이 되었지만 언제 반복문을 빠져나갈지 지시하는 부분이 없으므로 이 프로그램은 무한 반복을 벗어나지 못한다. 빠져나갈 수 있도록 하기 위해 다음 행들을 추가하자.

goAgain = int(raw_input("Type 1 to enter more numbers, or any other number to quit: "))

이제 사용자가 goAgain의 값을 바꿀 수 있어 while 루프를 제어할 수 있게 되었다. 사용자가 1을 입력하면 goAgain 변수는 계속 1이므로 루프는 다시 반복될 것이고, 다른 숫자를 입력하면 조건식의 결과는 더 이상 참이 아니므로 루프는 반복을 끝낼 것이다.

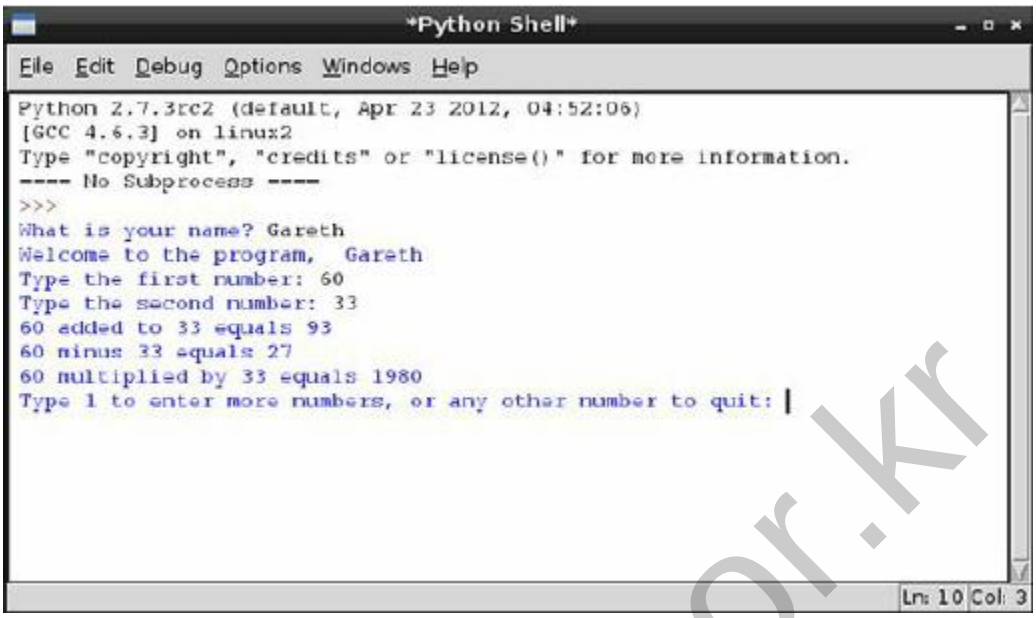
완성된 프로그램의 전체 코드는 다음과 같다.

```
#! /usr/bin/env python

# Example 2: A Python program

userName = raw_input("What is your name? ")

print "Welcome to the program, ", userName

goAgain = 1

while goAgain == 1:

firstNumber = int(raw_input("Type the first number: "))

secondNumber = int(raw_input("Type the second number: ")

print firstNumber, " added to ", secondNumber, " equals ", firstNumber + secondNumber

print firstNumber, " minus ", secondNumber, " equals ", firstNumber – secondNumber

print firstNumber, " multiplied by ", secondNumber, " equals ", firstNumber * secondNumber

goAgain = int(raw_input("Type 1 to enter more numbers, or any other number to quit: "))
```

프로그램을 calculator.py로 저장하고 실행해 본다. 이름을 입력하라는 메시지가 나오면 이름을 입력하고 두 개의 숫자를 입력해준다(그림 12-5 참조). 프로그램을 끝내고 싶으면 1이 아닌 숫자를 입력하여 프로그램을 끝낸다.

www.ketti.or.kr

그림 12-5: IDLE에서 calculator.py 실행



파이선의 주요 개념을 소개가 위한 짧은 프로그램 예제들을 더 원하면 다음의 공식 파이썬 예제 위키 페이지를 방문해 보기 바란다: http://wiki.python.org/moin/SimplePrograms.

예제 3: pygame을 이용하여 게임 만들기

파이썬의 강력함을 보이기 위해서 이번 예제에서는 고전 게임인 스네이크(또는 니블즈)에 기반한 완전한 기능을 가지는 아케이드 게임을 만들어 보자. 이 예제에서는 게임을 위한 외부 파이썬 라이브러리인 pygame을 사용할 것이다.

pygame은 원래 Pete Shinners에 의해 개발되었으며 파이썬에서 게임을 쉽게 작성할 수 있도록 디자인 된 기능들을 더한 파이썬 모듈들의 모음이다. pygame 모듈은 근래의 게임들에 필요한 사운드, 그래픽 및 심지어는 네트웍 지원 기능 등을 제공한다. pygame을 사용하지 않고도 파이썬에서 게임을 작성할 수는 있겠지만 이미 pygame 라이브러리에서 제공하는 코드들을 이용하는 것이 훨씬 쉽다.

pygame 프로그램을 작성하기 전에 pygame 라이브러리를 설치해야 한다. 데비안 배포판을 사용중이면 콘솔 또는 터미널 창에서 다음 명령어를 치기만 하면 된다.

```
$ sudo apt-get install python-pygame
```

다른 배포판인 경우 pygame 공식 사이트 http://www.pygame.org/download.shtml 에서 pygame 소스 파일을 다운로드 받을 수 있다. 이 곳에 설치 방법도 설명되어 있다.

www.ketti.or.kr

pygame 프로그램을 작성하는 것은 다른 파이썬 프로그램을 작성하는 것과 다를 바 없다. IDLE에서 새 문서 창을 열고 첫 줄에 다음 셔뱅 라인을 추가한다.

```
#! /usr/bin/env python
```

다음 줄에는 파이썬에게 이 프로그램이 pygame 모듈을 사용한다는 것을 알려주기 위해 import 문을 사용한다. import문은 필요한 외부 모듈들(또 다른 파이썬 파일들)을 로드하여 현재 프로그램에서 그 모듈들을 사용할 수 있도록 해준다. 다음 두 줄을 입력한다.

```
import pygame, sys, time, random

from pygame.locals import *
```

첫 줄은 pygame 모듈과, 함께 사용될 sys, time 및 random 모듈들을 임포트 한다. 전형적으로 임포트 된 모듈내의 명령은 모듈명.명령 형태로 호출한다. 반면에 둘 째 줄은 pygame.locals 모듈의 모든 명령들을 로드 하도록 지시한다. 그런 경우 이들 명령을 사용할 때는 앞에 모듈명을 붙일 필요 없이 명령만 사용하여 호출한다. 이외의 모듈들 – pygame.locals와는 별개인 pygame.clock 같은 –은 전체 이름을 사용해야 한다.

pygame을 예제에서 사용하기 위해서는 셋업이 필요하다. 다음 두 줄을 입력한다.

```
pygame.init()

fpsClock = pygame.time.Clock()
```

첫 줄은 pygame이 자신을 초기화하도록 한다. 둘째 줄은 게임의 속도를 조절하는데 사용할 fpsClock변수를 생성한다. 다음은 게임 내의 개체들이 그려질 새로운 게임 디스플레이 surface를 셋업한다.

```
playSurface = pygame.display.set_mode((640, 480))

pygame.display.set_caption('Raspberry Snake')
```

다음으로 프로그램에서 사용할 색상들을 정의한다. 색상정의가 반드시 필요한 것은 아니지만 같은 색을 여러 번 다시 사용할 때 색상값을 외울 필요가 없고 pygame.Color 명령을 매번 다시 호출하는 수고로움을 덜어준다. 다음 내용을 입력하자.

```
redColour = pygame.Color(255, 0, 0)

blackColour = pygame.Color(0, 0, 0)
```

www.ketti.or.kr

whiteColour = pygame.Color(255, 255, 255)

greyColour = pygame.Color(150, 150, 150)

다음의 몇 행은 몇 가지 변수들을 초기화한다. 각 변수들이 무엇을 위한 것인지는 일단 접어두고 우선 입력만 하자.

snakePosition = [100, 100]

snakeSegments = [[100,100], [80,100], [60,100]]

raspberryPosition = [300,300]

raspberrySpawned = 1

direction = 'right'

changeDirection = direction

세 개의 변수들이 – snakePosition, snakeSegments, raspberryPosition – 쉼표로 구분된 값들의 목록으로 설정되어 있는 것을 주목하자. 이것은 하나의 변수명에 여러 개의 값이 저장되는 list 변수를 생성하게 된다. 뒤에서 리스트에 저장된 개별 값들을 어떻게 접근하는지 보게 될 것이다.

다음은 새로운 함수를 정의해보자 – 일련의 파이썬 코드의 묶음으로 나중에 프로그램의 다른 부분에서 호출될 수 있다. 함수는 코드의 중복을 막고 프로그램을 이해하기 쉽게 해준다. 여러 곳에서 사용될 명령어 묶음이 있다면 이를 def 문을 이용해 함수로 정의한다 – 그러면 나중에 수정할 일이 생기더라도 한 곳만 수정하면 된다. gameOver함수를 다음과 같이 정의한다.

def gameOver():

gameOverFont = pygame.font.Font('freesansbold.ttf', 72)

gameOverSurf = gameOverFont.render('Game Over', True, greyColour)

gameOverRect = gameOverSurf.get_rect()

gameOverRect.midtop = (320, 100)

playSurface.blit(gameOverSurf, gameOverRect)

pygame.display.flip()

www.ketti.or.kr

time.sleep(5)

pygame.quit()

sys.exit()

반복문처럼 함수를 위한 코드는 반드시 들여쓰기 되어야 한다. 함수의 마지막 행 – sys.exit() – 다음 행은 들여쓰기를 하지 않는다.

gameOver함수는 스크린에 커다란 폰트로 Game Over 문자열을 쓰고 5 초간 멈췄다가 pygame과 프로그램 자체를 종료하기 위해 몇 가지 pygame명령들을 사용한다. 게임은 시작도 하지 않았는데 게임을 종료하기 위한 코드를 작성하는 것이 이상하게 느껴질 수도 있다. 그러나 함수는 항상 호출이 되기 전에 미리 정의가 되어야 하는 것이다. 새로 작성된 함수는 함수 이름 – gameOver – 를 호출하기 전까지는 실행되지 않는다.

이제 프로그램의 메인 부분을 작성할 차례이다. 메인 부분은 끝나지 않는 while 반복문(무한 반복) 내에 위치한다. 게임은 플레이어가 벽에 부딪치거나 자신의 꼬리에 닿아 죽을 때까지 계속 실행된다. 다음 행을 입력하여 메인 반복문을 시작한다.

while True:

True는 평가해보나 마나 언제나 참이므로 반복문은 중간에 gameOver 함수를 호출하여 프로그램을 끝내기 전까지는 무한히 반복될 것이다.

다음 행들을 계속 입력한다. 들여쓰기에 주의해야 한다.

for event in pygame.event.get():

if event.type == QUIT:

pygame.quit()

sys.exit()

elif event.type == KEYDOWN:

while문 뒤에 오는 첫 번째 행은 반드시 들여쓰기 되어야 한다. 여러 번의 중첩이 되게 되므로 들여쓰기에 주의를 기울인다.

if문에서는 발생한 이벤트가 QUIT인지(사용자가 Esc키를 눌렀을 때 발생하도록 아래에서 작성하였

www.ketti.or.kr

다)를 비교하여 pygame과 프로그램 자신을 종료한다.

elif는 else if를 줄인 것으로 앞의 if문에서의 평가 결과가 거짓일 때 실행되어 발생한 이벤트 종류가 KEYDOWN 종류인지를 확인한다 – 사용자가 키보드 상의 키를 눌렀을 때 발생한다. 사용자가 어떤 키를 눌렀을 때 실행되도록 다음 행들을 입력한다. 물론 들여쓰기는 필수이다.

```
if event.key == K_RIGHT or event.key == ord('d'):

changeDirection = 'right'

if event.key == K_LEFT or event.key == ord('a'):

changeDirection = 'left'

if event.key == K_UP or event.key == ord('w'):

changeDirection = 'up'

if event.key == K_DOWN or event.key == ord('s'):

changeDirection = 'down'

if event.key == K_ESCAPE:

pygame.event.post(pygame.event.Event(QUIT))
```

이 명령어들은 게임상에서 움직이고 있는 플레이어의 스네이크의 방향을 제어하는데 사용되는 changeDirection변수의 값을 변경한다. if 선언문안의 or 는 하나 이상의 조건을 사용할 수 있게 해준다. 이 경우 두 가지 방식으로 스네이크를 제어할 수 있게 되는데 플레이어는 커서 키들을 사용하거나 W, A, S, D키를 사용하여 스네이크를 상, 하, 좌, 우로 움직이게 할 수 있다. 프로그램 시작 시에는 changeDirection값이 right로 초기 설정되어 있어 스네이크는 오른쪽으로 움직인다.

프로그램의 시작 시에 초기화한 변수들을 되돌아 보면 direction이라는 또 다른 변수를 볼 수 있는데 이 변수는 현재 스네이크가 진행중인 방향을 가지고 있어서 changeDirection 변수와 더불어 플레이어의 방향전환이 유효한지를 결정하는데 사용된다. 현재 스네이크가 진행 중인 방향과 정반대로 진행하려고 하면 키 입력을 무시하도록 한다.

```
if changeDirection == 'right' and not direction == 'left':

direction = changeDirection
```

www.ketti.or.kr

```
if changeDirection == 'left' and not direction == 'right':

direction = changeDirection

if changeDirection == 'up' and not direction == 'down':

direction = changeDirection

if changeDirection == 'down' and not direction == 'up':

direction = changeDirection
```

사용자 입력이 모두 걸러진 뒤이므로 스네이크를 – 스크린 상에 일련의 블록들로 표시된다 – 움직일 수 있다. 매번 블록 크기만큼 이동시킨다. 다음 코드를 입력한다.

```
if direction == 'right':

snakePosition[0] += 20

if direction == 'left':

snakePosition[0] -= 20

if direction == 'up':

snakePosition[1] -= 20

if direction == 'down':

snakePosition[1] += 20
```

+=과 -= 연산자는 변수의 값을 오른쪽에 있는 값 만큼 증가시키거나 감소시킨다. snakePosition 변수 뒤의 브라켓 속의 숫자는 변수의 몇번째 항목을 참조할 것인지를 가리킨다. snakePosition[0]에는 스네이크의 X좌표 위치를 snakePosition[1]에는 Y좌표 위치를 담고 있다. 인덱스 값은 0에서부터 시작한다.

snakePosition 변수는 계속 2개의 값을 가지지만 프로그램의 처음에 생성한 snakeSegments변수는 그렇지 않다. 이 리스트 변수는 스네이크 머리 뒤의 몸통들의 위치를 저장한다. 스네이크는 라즈베리를 먹을 때 마다 길이가 늘어나므로 리스트 변수의 크기도 커져야 한다. 게임이 진행될 수록 몸통을 피하기가 점점 더 어려워진다. 스네이크의 머리가 몸통과 닿으면 게임은 끝난다. 스네이크의 몸통이 길어지도록 하기 위해 다음 코드를 입력한다.

www.ketti.or.kr

```
snakeSegments.insert(0, list(snakePosition))
```

snakeSegments 리스트에 새로운 값을 추가하기 위해 insert명령을 사용했다. 파이썬이 이 행에 도달할 때 마다 스네이크의 몸통을 한 칸씩 증가시킬 것이다. 추가되는 위치는 스네이크의 현재 머리 위치이다. 스네이크가 라즈베리를 먹을 때 마다 길이를 증가시키도록 다음과 같이 입력한다.

```
if snakePosition[0] == raspberryPosition[0] and snakePosition[1] == raspberryPosition[1]:

raspberrySpawned = 0

else:

snakeSegments.pop()
```

첫 번째 행은 스네이크의 머리 위치가 라즈베리 위치와 같은지를 비교한다. 일치하면 현재 라즈베리의 개수를 나타내는 raspberrySpawned 값을 0으로 설정하고 그렇지 않으면 pop 명령을 사용하여 snakeSegments의 제일 오래된 값을 꺼낸다. 결과적으로 머리와 가장 떨어져 있는 꼬리를 하나 짧아지게 한다. 플레이어 입장에서는 스네이크의 꼬리부분이 자라지 않았으므로 그냥 이동한 것으로 보이게 된다. 라즈베리를 먹었을 때는 리스트의 마지막 항목이 제거되지 않으므로 스네이크가 한 칸 자란 것이 된다.

게임에 라즈베리가 하나밖에 없으면 재미가 없으므로 플레이어가 라즈베리를 먹어버리면 새로운 라즈베리를 다시 만들어 놓도록 하자.

```
if raspberrySpawned == 0:

x = random.randrange(1, 32)

y = random.randrange(1, 24)

raspberryPosition = [int(x*20), int(y*20)]

raspberrySpawned = 1
```

raspberrySpawned가 0이면 라즈베리가 먹힌 것이므로 프로그램 시작부분에서 임포트한 random 모듈을 사용해서 플레잉 서피스의 임의의 위치 값을 생성한 후 몸통 한 조각의 크기인 20 픽셀을 곱해서 실제 스크린 상의 위치 값을 구해 raspberryPosition에 지정한다. raspberrySpawned값을 1로 설정해 1 개의 라즈베리가 있음을 표시해준다.

이제 스네이크가 움직이고, 자라고, 라즈베리가 먹히고, 생성되도록 해줄 차례이다. 이런 절차를

www.ketii.or.kr

게임개발에서는 respawning이라고 한다. 하지만 화면상에 실제로 그려진 것은 아직 아무것도 없다.

```
playSurface.fill(blackColour)

for position in snakeSegments:

pygame.draw.rect(playSurface, whiteColour, Rect(position[0], position[1], 20, 20))

pygame.draw.rect(playSurface, redColour, Rect(raspberryPosition[0], raspberryPosition[1], 20, 20))

pygame.display.flip()
```

이 코드들은 pygame이 플레잉 서피스의 배경을 검은색으로 칠하고 스네이크의 머리와 몸통을 흰색으로 그리고, 라즈베리를 빨간색으로 그리도록 한다. 마지막 줄 pygame.display.flip()은 pygame이 스크린을 갱신하도록 한다. 플레이 서피스에 그리는 모든 작업은 플레이어에게 보이지 않는 뒷면에 그려졌다가 pygame.display.flip()을 호출했을 때 전면에 노출되어 보이게 된다.

현재 상태에서는 스네이크가 영원히 죽지 않는다. 다음 코드를 추가하여 스네이크가 죽는 처리를 한다.

```
if snakePosition[0] > 620 or snakePosition[0] < 0:

gameOver()

if snakePosition[1] > 460 or snakePosition[1] < 0:

gameOver()
```

스네이크가 좌/우로 벽에 부딪치거나 상/하로 벽에 부딪치면 gameOver를 호출하고 이 프로그램의 초기에 정의한 것처럼 화면에 메시지를 출력하고 게임을 끝낸다. 스네이크의 머리가 자신의 몸통과 부딪쳤을 때도 죽도록 해야 한다.

```
for snakeBody in snakeSegments[1:]:

if snakePosition[0] == snakeBody[0] and snakePosition[1] == snakeBody[1]:

gameOver()
```

for문은 각 snakeSegments의 두 번째 항목부터 마지막 항목까지 반복하면서 위치를 꺼내어 snakeBody에 대입하고 for문 내에서는 snakeBody의 X, Y좌표가 snakePosition과 같으면

www.ketti.or.kr

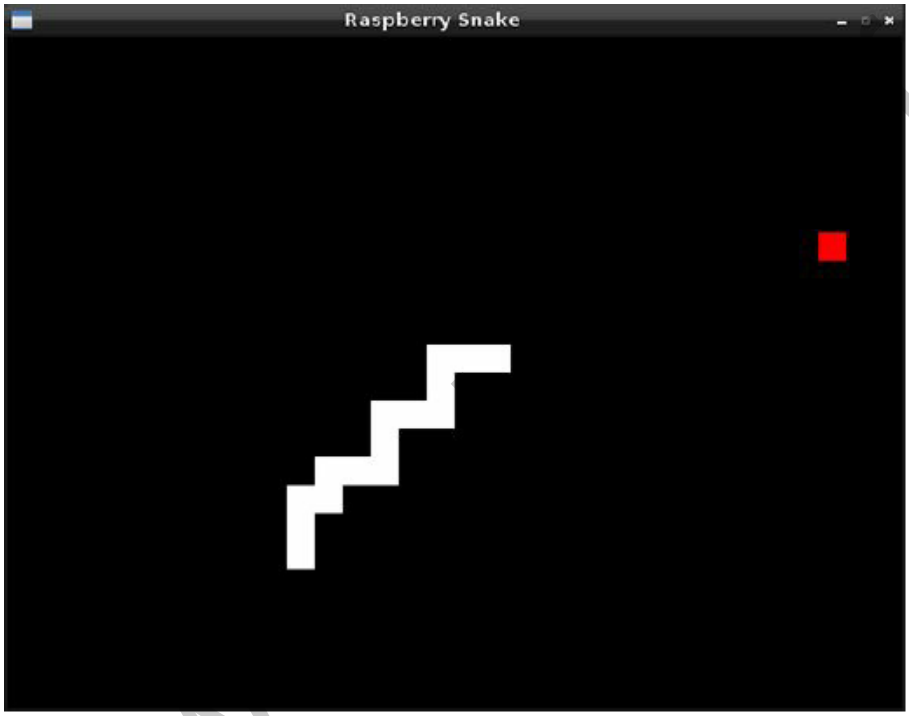
gameOver()를 호출하여 게임을 종료한다.

마지막으로 fpsClock 변수를 사용하여 게임의 속도를 조절한다. 이 변수가 없으면 게임이 너무 빨리 실행된다.

fpsClock.tick(20)

게임이 너무 느리다 싶으면 이 값을 증가시키고, 너무 빠르면 감소시키면 된다. 프로그램을 raspberrysnake.py파일에 저장하고 IDLE의 Run메뉴에 있는 Run Module을 이용하여 실행시켜 본다. 게임이 로드되는대로 시작될 것이다(그림 12-6 참조).

그림 12-6: 라즈베리파이에서 스네이크 게임 플레이



전체 소스코드는 참조 A 를 찾아보기 바란다. pygame은 이 예제에서 사용된 것뿐만 아니라 오디오 재생, 더 나은 그래픽을 위한 스프라이트 핸들링 및 마우스 제어 등 많은 기능을 제공한다. pygame의 고급 기능들을 배우기 위해 제일 좋은 곳은 공식 웹사이트인 http://www.pygame.org/wiki/tutorials이다. ,

예제 4: 파이썬과 네트워킹

파이썬으로 네트웍 커뮤니케이션을 해 보기 위해서 IRC(Internet Relay Chat) 채널에 접속하는 사용자를 모니터링 하는 도구를 만들어 보자.

우선 셔뱅라인과 프로그램의 목적을 설명하는 행을 입력한다.

#! /usr/env python

IRC Channel Checker, written for the Raspberry Pi User guide by Tom Hudson

그 다음 프로그램에 필요한 모듈들을 임포트한다.

import sys, socket, time

앞의 예제에서 sys와 time모듈은 사용을 해 봤지만 socket모듈은 처음이다. socket모듈은 파이썬에서 네트웍 소켓을 열고, 닫고, 읽고, 쓸 수 있도록 해주는 모듈이다. 이 예제에서는 socket 모듈을 통해 원격지의 IRC서버에 접속 한다.

몇 개의 상수를 정의한다. 상수는 변수처럼 자신에게 할당된 값을 가진다. 하지만 변수와 달리 처음에 할당된 값을 변경할 수 없다. 상수는 변수와 구별하기 쉽도록 모두 대문자를 사용하여 정의하는 것이 좋은 습관이다. 다음 두 줄을 프로그램에 추가한다.

PRL_NAMREPLY = '353'

RPL_ENDOFNAMES = '366'

이 값들은 IRC 상태코드들로, 서버에 의해 언제 특정 동작이 완료되었는지를 알려줄 목적으로 제공된다. 다음으로 서버 접속을 위한 변수들을 정의해보자.

```
irc = {
        'host' : 'chat.freenode.net',
        'port' : 6667,
        'channel'     : '#raspiuserguide',
        'namesinterval'       : 5
}
```

첫 번째 행은 dict 데이터 형을 생성하도록 한다. dictionary의 약자로 하나의 주 변수 – 이 경우에는 irc – 여러 개의 변수를 저장할 수 있도록 해준다. 개별 변수들은 나중에 프로그램에서 불러 쓸 수 있다. dict 데이터 형을 사용하지 개별 변수들을 써서 프로그램을 작성할 수도 있지만 사용하는 것이 훨씬 프로그램을 읽기 쉽다. dict 형은 첫 행에 좌 대괄호를 사용해서 열고 마지막 행에 우 대괄호를 사용해서 닫는다.

www.ketti.or.kr

host변수에는 프로그램이 접속할 서버의 FQDN(fully-qualified domain name)을 설정해주어야 한다. 이 예제에서는 chat.freenode.net이 사용되었다. 다른 서버를 사용하고 싶다면 이 도메인명을 바꿔주면 된다. port변수는 IRC가 사용하고 있는 네트웍 포트 번호를 지정한다. 일반적으로 IRC서버는 6667을 사용한다. channel변수는 사용자를 모니터하기 위해 어느 채널에 참여하지를 지정한다. nameinterval은 사용자 목록을 갱신하기 위해 프로그램이 얼마나 대기할지를 초단위로 지정한다.

사용자와 관련된 변수들을 정의하기 위해 두 번째 dict를 정의한다.

```
user = {
        'nick'     : 'botnick',
'username'         : 'botuser',
'hostname'         : 'localhost',
'servername'       : 'localhost',
'realname'         : 'Raspberry Pi Names Bot'
}
```

nick 변수에는 프로그램이 사용할 IRC 닉네임을 설정한다. 이 IRC서버에 직접 접속할 생각이면 이곳에 자신의 닉네임을 사용하면 안 된다. 대신 자신의 닉네임 뒤에 –bot을 붙여서 사람이 아니라 프로그램이라는 것을 나타내는 것이 명확하다. username과 realname에도 이 사용자가 누구인지를 설명할 수 있는 설명을 붙이도록 한다. hostname과 servername변수는 localhost로 두든지 자신의 인터넷 주소로 변경해도 된다.

socket모듈은 사용자가 socket 개체를 생성하도록 요구한다. 이 개체는 네트웍 연결을 제공한다.

```
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

다음 앞에서 정의한 변수들에서 지정한 IRC서버에 접속한다.

```
print 'Connecting to %(host)s:%(port)s...' % irc
try:
        s.connect((irc['host'], irc['port']))
```

www.ketti.or.kr

```
except socket.error:
        print 'Error connectiong to IRC server
        %(host)s:%(port)s' % irc
sys.exit(1)
```

try 및 except 명령이 오류 처리를 위해 사용되었다. 프로그램이 서버에 접속하는데 실패 – 라즈베리파이가 인터넷에 연결이 되어 있지 않거나 서버가 죽어 있거나 하는 경우 – 하는 경우 오류 메시지를 출력하고 프로그램을 정상 종료 한다. s.connect 행은 irc사전에 있는 host와 port 변수에 해당하는 IRC서버에 접속을 시도하도록 한다.

프로그램이 오류로 끝나지 않았으면 IRC서버에 잘 접속된 것이다. 채널에서 사용자 이름들을 조회하기에 앞서 프로그램 자신을 서버에 입증할 필요가 있다. socket모듈의 send명령을 사용하여 몇 가지 명령을 서버에 보낸다.

```
s.send('NICK %(nick)s)r]n' % user)
s.send('USER %(username)s %(hostname)s %(servername)s :%(realname)s)r]n' % user)
s.send('JOIN %(channel)s)r]n' % irc)
s.send('NAMES %(channel)s₩r₩n' % irc)
```

send 함수는 표준 출력 – 대개는 콘솔이나 터미널 창 – 대신 네트웍 접속으로 출력하는 것을 제외하면 print함수와 거의 동일하게 동작한다. 이 경우 프로그램은 문자열을 IRC서버에 보내 nick 변수에 있는 문자열을 닉네임으로 인식하고 username, hostname, servername 및 realname 변수에 있는 문자열들로 사용자 세부 정보를 인식하도록 한다. 그 다음 channel 변수에 있는 채널에 참석하는 명령을 보낸다. 그 다음 그 채널에 있는 사용자 목록을 받기 위한 명령을 보낸다. 이 프로그램은 조금만 변형하면 FTP서버에 있는 파일들의 목록을 받거나 POP3 서버의 읽지 않은 메일 목록을 받도록 하는데 사용할 수 있다.

소켓으로부터 데이터를 읽는 것은 조금 더 복잡하다. 우선 서버에서 수신된 데이터가 처리가능해질 때까지 보유하는 역할을 할 수신 버퍼로, 빈 문자열 변수를 만든다.

```
read_buffer = ''
```

이중 따옴표가 아니라 두 개의 홑 따옴표임에 주의한다.

www.ketti.or.kr

그 다음, 이름 목록을 저장할 빈 리스트 변수를 만든다.

names = []

리스트 형은 스네이크 게임에서 위치를 저장하기 위해 사용했던 것과 동일하다. 일반적인 변수와 달리 리스트 형은 여러 개의 값을 저장할 수 있다. 이 경우, IRC 채널에 있는 사용자 명을 저장한다.

다음은 계속해서 서버에 사용자명을 질의하고 화면에 출력하는 무한 루프를 만든다.

while True:

read_buffer += s.recv(1024)

소켓 모듈이 1024 바이트(1KB)를 IRC서버로부터 수신해서 read_buffer 변수에 덧붙인다. 1024는 임의로 정한 값이다.

버퍼의 문자열들을 행으로 나눈다.

lines = read_buffer.split('₩r₩n')

read_buffer = lines.pop();

첫 번째 줄은 수신 버퍼의 문자열에서 개행 제어 문자인 "₩r₩n"을 찾아내어 문자열을 행 단위로 나누어 lines 리스트 변수에 넣는다. 두 번째 줄은 lines에 들어 있는 맨 마지막 문자열만을 꺼내어 다시 read_buffer에 대입한다. 서버로부터 문자열을 수신할 때 1KB 단위로 받았기 때문에 마지막 문자열은 아직 개행이 되지 않은 미완성 행이기 때문에 다음 번에 수신되는 문자열의 앞부분에 합쳐져야 하기 때문이다.

이 시점에서 lines 변수에는 서버로부터의 완전한 행의 응답 메시지들이 담겨있다. 이 문자열들을 처리하여 채널 참석자들의 이름을 얻는다.

for line in lines:

response = line.rstrip().split(' ', 3)

response_code = response[1]

if response_code == RPL_NAMREPLY:

names_list = response[3].split(':')[1]

www.ketti.or.kr

```
names += names_list.split(' ')
```

lines변수의 매 행에 대해 서버가 보낸 숫자로 된 IRC 응답 코드를 찾는다. 여러 가지의 응답 코드가 있을 수 있지만 여기서는 오로지 앞서 정의한 두 개의 응답 코드 353(이름 목록이 뒤따른다는 의미)과 356(목록이 끝났다는 의미)만을 찾는다. if 문은 응답 행의 첫 부분으로 응답 코드를 확인한 뒤 이름들을 잘라내어 names 변수에 저장한다.

이제 names 리스트 변수에는 프로그램의 요청에 따라 서버가 응답한 모든 이름들이 담겨있다. 하지만 참석자 이름의 끝을 의미하는 366이 수신되기 전까지는 모든 참석자 이름이 수신된 것은 아니다. 마지막 행에서 새로 수신된 이름들을 names에 그냥 대입하지 않고 기존 이름들에 더하는 이유가 바로 그 때문이다. 이 부분이 실행될 때 마다 프로그램은 전체 참석자 이름 목록 중 일부분만을 수신할 것이다. 전체 참석자 목록이 수신되었을 때 무엇을 할 지는 다음 코드를 보자.

```
if response_code == RPL_ENDOFNAMES:

# Display the names

print '₩r₩nUsers in %(channel)s:' % irc

for name in names

print name

names = []
```

355 응답 코드를 받으면 이제 완성된 참석자 이름 목록을 표준 출력에 출력하고 names 리스트 변수를 비운다.

마지막으로 다음 코드를 입력하여 프로그램을 끝낸다.

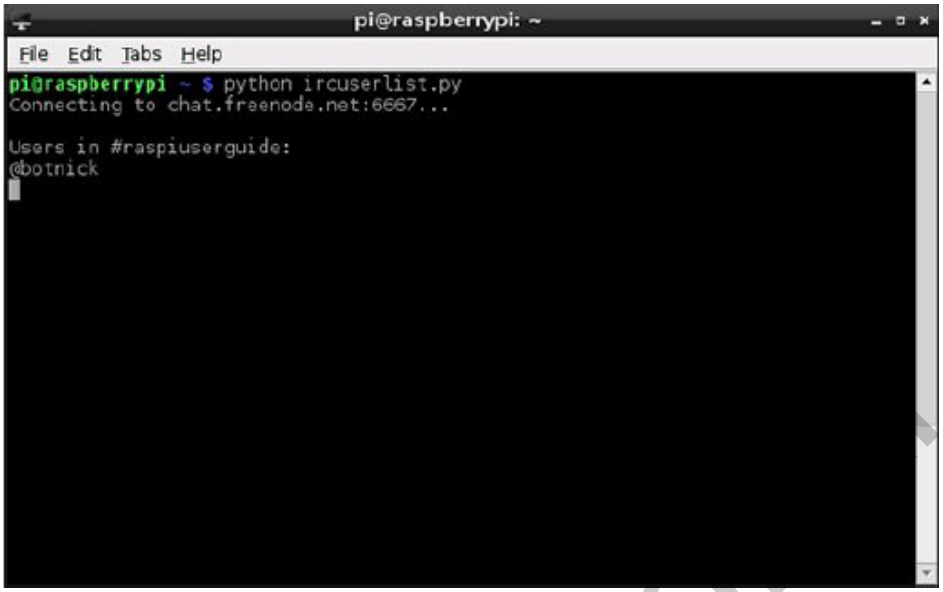
```
time.sleep(irc['namesinterval'])

s.send('NAMES %(channel)s₩r₩n' % irc)
```

일정 시간 쉬었다가 다시 참석자 이름 조회 명령을 보낸다. 이 시간 간격이 너무 짧으면 IRC 서버가 프로그램 접속을 강제로 끊을 수 있으므로 주의해야 한다.

프로그램을 ircuerlist.py로 저장하고 프로그램을 실행한다. 이 프로그램을 처음 실행하면 처음에는 서버에 접속하는데 좀 시간이 걸린다. 하지만 일단 접속이 되고 나면 참석자 이름 목록(그림 12-7 참조)이 빠른 속도로 출력된다. 프로그램을 끝내려면 CTRL + C를 입력한다.

www.ketti.or.kr

그림 12-7: 파이썬으로 IRC 채널에 있는 사용자 목록 출력



프로그램의 전체 리스트는 참조 A에 있다.

더 읽어보기

이 장에서는 파이썬을 맛보기 수준에 살펴보았다. 언어 하나를 충분히 다루려면 훨씬 많은 분량이 필요하기 때문에 이쯤에서 생략한다. 파이썬 언어에 대해 더 자세히 알고 싶은 분들은 다음 사이트들을 참고하기 바란다.

- 맛보기파이썬 공식 초보자 가이드: http://wiki.python.org/moin/BeginnersGuide
- 다운로드 가능한 무료 인터랙티브 튜토리얼: http://www.learnpython.org
- Zed A. Shaw의 Learn Python The Hard Way (Shavian Publishing 2012), 파이썬을 위한 베스트 프랙티스 코딩을 볼 수 있다. 책 이름과는 달리 초보자에게도 적합하다. 책을 구입할 수 도 있고 다음 사이트에서 무료로 읽을 수 있다: http://learnpythonthehardway.org/

www.ketti.o

Chapter 13. 하드웨어 해킹

앞에서는 라즈베리 파이를 다양한 소프트웨어를 실행하는 유연한 플랫폼으로 변신시킬 수 있다는 것을 배웠다. 하지만 그것이 다가 아니다. 일반 데스크톱과 랩톱들은 같은 소프트웨어를 대부분의 경우 라즈베리파이보다 더 빠르게 실행할 수 있다.

하지만 라즈베리파이는 표준적인 PC보다 더 나은 능력을 가지고 있는 것이 있다. 그 것은 회로기판의 왼쪽 위에 있는 26-핀 범용 입출력(GPIO; general-purpose input-output) 포트이다.

GPIO는 라즈베리파이가 다른 전자부품이나 회로와 의사소통을 할 수 있도록 해준다. 또한 더 큰 규모의 전자 회로에서 라즈베리파이가 제어기로써의 역할을 할 수 있도록 해준다. GPIO 포트를 통해서 라즈베리파이는 온도를 감지할 수 있고, 서보 모터를 돌릴 수 있고, 직렬 주변기기 인터페이스(SPI; Serial Peripheral Interface)와 집적회로연결(I^2C; Inter-Integrated Circuit) 같은 다양한 프로토콜들을 이용해서 다른 컴퓨팅 장비들과 의사소통을 할 수 있다.

하지만 라즈베리파이의 GPIO 포트를 이용해서 회로를 구축하기 전에 몇 가지 추가적인 장비들을 알아둘 필요가 있다.

전자 부품

라즈베리파이의 GPIO포트를 통해서 제어할 수 있는 회로를 만들어 보기 전에, 다양한 부품들과 공구들에 대해 알아둘 필요가 있다. 아래에 전자공학을 시작하기 위한 구매 목록을 적어보았다.

- 브레드보드(Breadboard) – 브레드보드는 2.54mm 간격의 구멍이 격자 모양으로 배치되어 있어 전자부품을 끼우고, 다시 뺄 수 있도록 되어 있다. 각 격자 밑에는 접점들이 있어 같은 행의 전자부품들이 별도의 전선 없이 연결되도록 되어 있다. 브레드보드는 납땜을 하거나 땜납을 제거하는 불편 없이 손쉽게 시험용 전자회로를 만들어보고 수정할 수 있어 전자 작업에 없어서는 안될 소중한 도구이다.

- 점퍼와이어(Jumper Wires) – 브레드보드가 전자부품들을 전선 없이도 연결할 수 있도록 해주기는 하지만 브레드보드 위의 행과 행 사이를 연결하려면 여전히 전선이 필요하다. 점퍼와이어로는 연선 보다는 단선이 좋다. 브레드보드의 구멍에 끼우기가 더 쉽기 때문이다. 여러 가지 색깔의 전선을 사용하면 용도에 따라 색깔로 구분할 수 있어 더 좋다.

- 저항기(Resistors) – 저항기 (또는 저항)은 전류의 흐름을 방해하여 전압을 떨어뜨리는 효과를 내는 전자부품으로 대부분의 전자회로에 사용된다. 이 번 예제에도 사용된다. 저항

www.ketti.or.kr

의 단위는 오옴(ohm)인데 기호로는 Ω로 표기한다. 많이 사용되는 값의 저항을 여러 개 갖춰두는 것이 편리한데 여러 소매점에서 판매하는 저항 모음 킷을 구매하는 것이 편리하다.

- 푸시버튼(Push-Buttons) – 매우 일반적인 입력 소자로 눌렸을 때 회로가 연결되게 한다. 기본적으로 키보드는 푸시버튼을 모아 놓은 것이라고 보면 된다. 눌리는 순간 연결되고 스위치를 놓으면 곧바로 연결이 끊어지므로 순간 접점 스위치(momentary contact switch)라고도 한다.

- 발광다이오드(LEDs; Light-emitting diodes) – 발광다이오드는 매우 자주 사용되는 일반적인 출력장치이다. LED는 전압이 가해졌을 때 빛을 내므로 라즈베리파이의 GPIO포트가 high인지 low인지를 나타낼 수 있다. 라즈베리파이의 GPIO 포트는 고휘도 백색이나 고휘도 푸른색 LED를 구동할 정도의 고전압, 고전력을 내지 못하므로 LED를 구매할 때 저전력 LED(일반적인 LED)를 구매해야 된다. 고전압, 고전력을 필요로 하는 소자를 구동할 때는 트랜지스터라고 하는 또 다른 소자가 필요하다.

브레드보드로 시제품을 만든 후 뭔가 영구적인 회로를 만들고 싶다면 다음과 같은 것들이 필요하다.

- 스트라이프보드(Stripboard) – 일회용 브레드보드라고 할 수 있다. 브레드보드처럼 2.54mm간격의 구멍이 되어 있지만 브레드보드와는 달리 부품들을 구멍에 끼운 후 납땜으로 고정한다. 납땜 후 영구적인 회로가 된다.

- 납땜인두(Soldering Iron) – 부품들을 회로에 영구적으로 고정하고자 할 때 납땜인두를 사용한다. 납땜인두에 너무 많은 돈을 투자할 필요는 없지만 예산에 여유가 있으면 온도조절이 가능한 제품을 장만해두는 것이 현명한 투자이다. 작고 뾰족한 팁을 가진 납땜용 인두를 구매해야 된다. 치즐팁(넓적하고 큰 목공예용 팁)을 가진 인두는 섬세한 전자 작업에는 적합하지 않다.

- 땜납(Solder) – 납땜 인두에는 당연히 땜납이 필요하다. 땜납은 전도성 금속과 플럭스라고 하는 세척물질이 포함되어 있다. 두꺼운 배관용 납을 구매하면 안 된다. 값은 싸지만 고열을 가해야 녹기 때문에 전자회로에는 어울리지 않는다.

- 인두받침대 및 스폰지(Stand and Sponge) – 사용하던 뜨거운 인두를 놓아둘 곳이 있어야 한다. 더불어 인두 팁을 청소할 것도 필요하다. 어떤 인두는 스탠드와 청소용 스폰지가 세트로 포함되어 있기도 하지만 별도로 구매하는 것이 일반적이다.

www.kett.or.kr

- 니퍼(Side Cutters) – 부품들을 납땜하고 나면 여분의 긴 다리를 잘라줄 필요가 있다. 이때 니퍼를 사용하면 납땜부위에 손상을 주지 않고 깨끗하게 잘라낼 수 있다.
- 핀셋(Tweezers) – 전자 부품들은 매우 작고 다루기 힘들기 때문에 핀셋 세트가 필요하다. 구멍에 끼우는 형태의(through-hole) 부품들은 비교적 다루기 쉽지만 표면 실장형(surface mount)부품들을 사용할 생각이라면 손가락의 화상을 예방하기 위해 핀셋이 필수품이다.
- 기판거치대(Work Stand) – 납땜 작업을 할 때 기판을 잡아주는 역할을 한다. 무거운 받침대와 클램프, 클립으로 구성되어 있고, 어떤 모델은 확대경과 조명등을 포함하고 있는 것도 있다.
- 회로시험기(Multimeter) – 멀티미터는 여러 가지 – 전압, 전류, 저항, 정전용량 등 – 을 측정하고 단선 및 소트를 시험할 수 있는 장비이다. 멀티미터가 반드시 필요한 것은 아니지만 회로를 진단하는데 매우 유용하다. 전문적인 멀티미터는 매우 비싸지만 간단한 모델은 값도 그다지 비싸지 않아 전자회로를 시작하는 초보자라면 누구나 투자할 만 하다.
- 땜납제거용 위크(Desoldering Wick) – 납땜을 하다 보면 실수도 하기 마련이다. 땜납제거용 위크는 금속선을 그물 모양으로 짠 테잎 형태인데 기판의 잘못 납땜된 부위에 대고 인두로 가열을 하면 납을 빨아들여 부품을 떼어낼 수 있다. 고장난 전자제품에서 부품들을 떼어내어 모으면 값싸게 여러 유용한 부품들을 모을 수 있다.

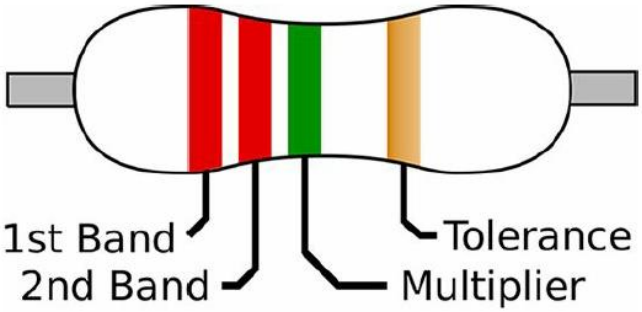
저항 값 읽기

대부분의 전자부품들은 각 부품의 값이 명확하게 표기되어 있다. 예를 들면 콘덴서에는 정전용량이 패럿 단위로 인쇄되어 있고, 크리스탈에는 주파수가 표기되어 있다.

하지만 저항은 예외이다. 저항 값은 표면에 그려져 있는 색 띠로 계산을 해야 된다. 어떤 저항의 저항 값을 알 수 있는 방법은 멀티미터로 측정을 하거나 색 띠로 알 수 밖에 없기 때문에 멀티미터로 일일이 측정해보지 않으려면 색 띠를 볼 줄 알아야 한다. 그림 13-1에 전형적인 4색 띠 저항 그림이 있다. 앞의 두 띠는 저항 값을 나타낸다. 세 번째 색 띠는 승수로써 앞의 두 자리 저항 값에 십의 몇 승을 곱할 것인지를 나타낸다. 마지막 네 번째 색 띠는 저항 값의 오차 허용치 내지는 정밀도를 나타낸다. 더 적은 오차 허용치를 갖는 저항일수록 더 표기된 저항 값에 일치할 가능성이 높다. 대신 가격이 더 비싸다.

그림 13-1: 4색 띠 저항 값 읽기

www.keti.or.kr



	1st/2nd Band	Multiplier	Tolerance
Black	0	0	-
Brown	1	$\times 10^{1}$	±1%
Red	2	$\times 10^{2}$	±2%
Orange	3	$\times 10^{3}$	-
Yellow	4	$\times 10^{4}$	-
Green	5	$\times 10^{5}$	±0.5%
Blue	6	$\times 10^{6}$	±0.25%
Violet	7	$\times 10^{7}$	±0.1%
Grey	8	$\times 10^{8}$	±0.05%
White	9	$\times 10^{9}$	-
Gold	-	$\times 10^{-1}$	±5%
Silver	-	$\times 10^{-2}$	±10%
None	-	-	±20%

예시된 그림의 저항 값을 읽어보자. 처음 두 색이 빨강, 빨강이므로 저항 값은 우선 22로 볼 수 있다. 그 다음 색 띠는 초록이므로 여기에 10의 5승을 곱해주어야 한다. 그러면 2,200,000 오옴이 된다.

1,000 오옴은 1킬로 오옴이고, 1킬로 오옴은 1메가 오옴이므로 보통 2.2 메가 오옴으로 쓰는 게 편리하다. 마지막 색 띠는 금색이므로 허용오차는 +-5%이다.

5색띠로 표시된 저항도 있다. 이 때는 첫 세 자리가 저항 값이고 네 번째는 역시 승수, 다섯 번째는 허용오차를 의미한다.

www.kett.co.kr

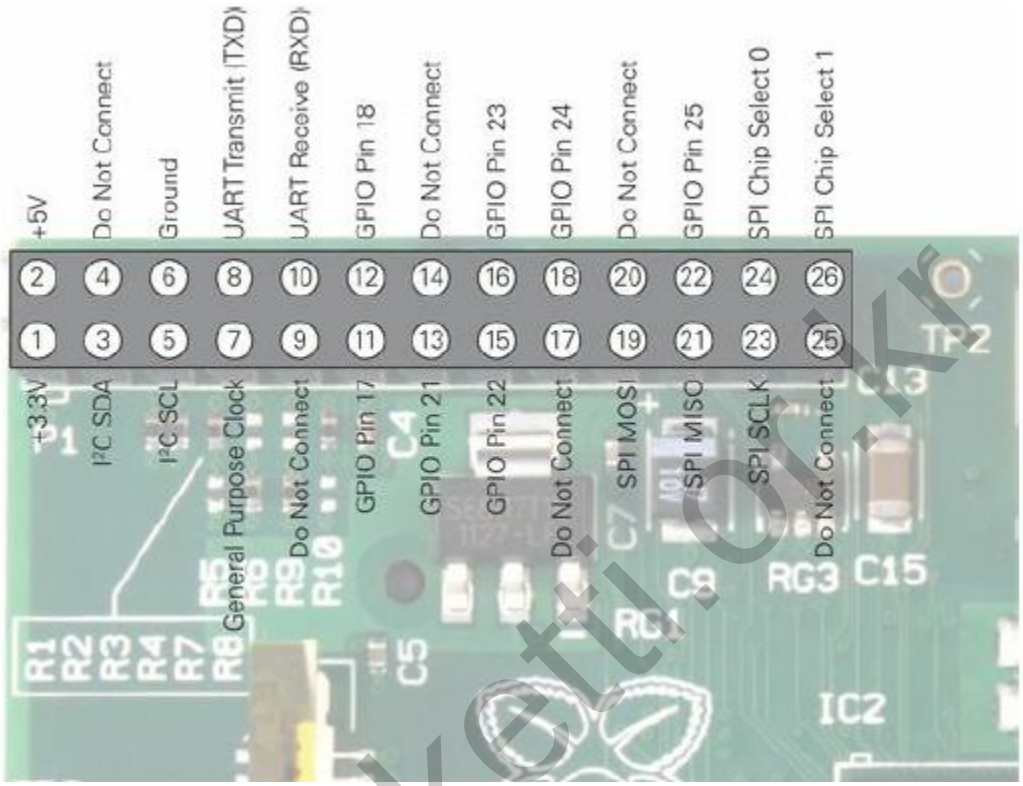
GPIO 포트

라즈베리파이의 GPIO포트는 인쇄회로기판의 왼쪽 위에 위치하고 있다. 2.54mm 간격 2행 13열의 숫놈 헤더로 되어 있는 26핀 포트이다. 헤더 사이의 간격은 특히 중요하다. 전자공학에서 2.54mm(0.1인치) 핀 간격은 자주 볼 수 있는 수치이며 스트라이프보드와 브레드보드를 포함한 프로토타이핑 플랫폼들을 위한 표준 간격이다.

GPIO포트의 각 핀들은 각각의 용도가 있으며 특정 회로를 형성하기 위해 몇몇 핀들이 같이 사용

되기도 한다. GPIO포트의 배치는 그림 13-2을 보면 알 수 있다.

그림 13-2: 라즈베리 파이의 GPIO 포트 핀 정의



GPIO포트의 핀들은 두 줄로 나뉘는데 아랫줄은 홀수 번호, 윗줄은 짝수 번호를 가진다. 라즈베리파이의 GPIO포트를 사용할 때는 이 점을 잘 기억하는 것이 좋다. 다른 대부분의 장비들과는 다른 번호 배치를 가진데다가 표시가 되어있는 것도 아니므로 어느 핀이 어느 핀인지 혼동하기 쉽다.

주의: Do Not Connect라고 표시되어 있는 핀에는 절대 아무것도 연결해서는 안 된다. 이 핀들은 라즈베리파이의 BCM2835 SoC의 내부 기능을 위해 유보되어 있다.

라즈베리파이의 GPIO포트가 5V 전원을 제공하기는 하지만, micro-USB단자를 통해 공급받은 전원을 핀 2를 통해 제공, 라즈베리파이의 내부 동작은 3.3V 회로에 기반하고 있다.이 말은 라즈베리파이의 부품들은 3.3V 전원으로 동작하고 있다는 뜻이다. 라즈베리파이의 GPIO포트를 통해 인터페이스 되는 회로를 만들 계획이라면 3.3V 로직 호환 부품을 사용하거나 정전압 회로를 거쳐 연결해야만 한다.

주의: 5V전원을 라즈베리파이의 GPIO핀에 연결하거나 두 개의 전원 공급 핀들(핀1과 핀2)을 다른 핀들에 연결하면 라즈베리파이를 손상시키는 결과를 초래한다. 포트의 핀들은 직접 브로

www.kotti.or.kr

드컴 BCM2835 SoC에 연결되어 있기 때문에 손상을 복구하기는 불가능할 것이다. GPIO포트 근처에서는 극도의 주의를 기울여야 한다.

GPIO포트는 기본적으로 범용 핀을 7개 제공한다. 핀 11, 12, 13, 15, 16, 18, 22이다. 핀 7은 기본적으로는 범용 클럭 신호를 제공하도록 되어 있지만 범용 핀으로 사용이 가능하기 때문에 도합 8개를 범용으로 사용할 수 있다. 이 핀들은 high상태(+3.3V) 또는 low상태(접지 또는 0V)를 가진다. 이 것은 이진 연산에서의 1과 0에 해당되며 다른 부품들을 켜거나 끄는데 사용될 수 있다. 이에 대해서는 이 장의 뒤에서 더 배울 것이다.

주의: 라즈베리파이의 내부 논리는 3.3V로 동작한다. 이는 주로 5V로 동작하는 아두이노 및 변형 제품들과 같은 대부분의 다른 마이크로 컨트롤러 장치들과는 아주 대조적이다. 아두이노를 위해 디자인된 장비들은 레벨 변환기나 광학 아이솔레이터를 사용하지 않는 한 라즈베리파이와 함께 동작시킬 수 없다. 마찬가지로 5V 마이크로 컨트롤러의 핀들을 라즈베리파이의 GPIO 포트에 연결하면 라즈베리파이에 영구적인 손상을 가할 수 있다.

이러한 범용 핀들 이외에도 GPIO 포트는 특별한 버스들을 위한 핀들을 가지고 있다. 버스들에 대해서는 설명이 이어진다.

UART 직렬 버스

UART(Universal Asynchronous Receiver/Transmitter) 직렬 버스는 간단한 2선 직렬 인터페이스이다. cmdline.txt파일에서 직렬 포트를 설정했을 때(60Chapter 7. 하드웨어 구성 설정에서 설명했듯이) 메시지를 출력하는데 사용되는 포트가 바로 이 직렬 버스이다. 라즈베리파이의 직렬 버스를 표시 장치에 연결하면 리눅스 커널의 메시지를 표시할 수 있다. 라즈베리파이가 부팅 과정에서 문제를 일으켜 디스플레이에 아무것도 표시되지 않는 경우 이 직렬 버스가 간편한 진단 도구가 되어줄 수 있다.

UART 직렬 버스는 핀 8과 10을 통해 접근할 수 있다. 핀 8은 발신 신호, 핀 10은 수신 신호선이다. 전송 송도는 cmdline.txt파일에서 설정할 수 있다. 통산 115,200 bps(Bits per second)이다.

I^2C 버스

이름이 의미하듯 I^2C(Inter-Integrated Circuit) 버스는 다수의 집적회로(ICs)간 통신을 제공하기 위해 디자인 되었다. 라즈베리파이의 심장부라고 할 수 있는 브로드컴 BCM2835 SoC도 이들 집적회로들 중의 하나이다. 이들 핀들은 풀업 저항을 포함하고 있어 별도의 풀업 저항을 달지 않아도 된다.

www.ketti.or.kr

I²C버스는 핀 3과 5를 통해 접근할 수 있다. 핀 3은 SDA(Serial Data Line)이고 핀 5는 SCL(Serial Clock) 신호선이다. 이들 핀으로 제공되는 I²C는 사실 BCM235 칩이 제공하는 두 개의 I²C 버스 중의 하나이다. 다른 하나는 보드 상에 저항으로 터미네이트되어 있어 범용으로 사용할 수 없도록 되어 있다.

SPI 버스

SPI(Serial Peripheral Interface)버스는 주로 마이크로 컨트롤러와 다른 기기의 ISP(in-system programming)을 위한 용도로 디자인 되었다. UART 또는 I²C와는 달리 하나 이상의 타겟 장비와 통신할 수 있도록 하기 위한 칩 선택 선을 포함하여 네 개의 선을 가지는 버스로 되어 있다.

라즈베리파이의 SPI 버스는 핀 19, 21 및 23과 핀 24, 26에 있는 칩 선택선을 통해 사용할 수 있다. 핀 19는 MOSI(SPI 마스터 출력과 슬레이브 입력), 핀 21은 MISO(마스터 입력과 슬레이브 출력), 핀 23은 SLCK(통신 동기화를 위한 직렬 클럭), 핀 24와 26은 두 개까지의 독립 슬레이브 장치를 지원하는 칩 선택 신호선이다.

라즈베리파이의 BCM2835 SoC 프로세서에는 다른 버스들이 더 있지만 GPIO포트로 연결되어 있지 않아 사용할 수 없다.

파이썬에서 GPIO 사용하기

GPIO 파이썬 라이브러리 설치하기

라즈베리파이가 출시된 이후 많은 개발자들이 라즈베리파이의 기능을 모두 활용하기 위해 라이브러리라고도 불리는 소프트웨어 모듈들을 만들어 냈다. 특히, 프로그래머들은 라즈베리파이 사용자들이 저수준 프로그래밍에 대한 지식이 없이도 GPIO 포트를 다룰 수 있게 해주었다.

이 라이브러리들은 우리가 Chapter 12. 파이썬 프로그래밍에서 다루었던 pygame처럼 기본 파이썬 언어의 기능을 확장하도록 디자인 된 것이다. 이러한 라이브러리들을 설치하면 라즈베리파이의 GPIO 포트를 쉽게 사용할 수 있지만 그것이 의미하는 것은 반면에 우리가 작성한 소프트웨어를 다른 사용자가 사용하려면 역시 그 라이브러리를 먼저 설치해야 된다는 것이다.

여러 가지 파이썬 라이브러리들이 있지만 이 장에서는 raspberry-gpio-python 라이브러리를 추천한다. 이 글을 쓰는 시점에는 0.4.1a가 최신 버전이다. 다음 구글 코드 웹사이트에서 이 라이브러리를 다운로드 할 수 있다. http://code.google.com/p/raspberry-gpio-python/

다음과 같이 설치한다.

www.ketti.or.kr

1. 라즈베리파이의 웹 브라우저를 열고 위의 링크에 접속하여 홈 폴더에 RPi.GPIO-0.4.1a.tar.gz파일을 다운로드 받는다.

2. 다음 명령을 사용하여 홈 디렉터리에 압축을 푼다.
   $ tar xvfz RPi.GPIO-0.4.1a.tar.gz

3. 라이브러리를 설치한다.
   $ cd RPi.GPIO-0.4.1a
   $ sudo apt-get install python-dev
   $ sudo python setup.py install

이제 라이브러리가 파이썬에 설치되기는 했지만 기본으로 로드가 되는 것은 아니다. 이를 사용하기 위해서는 pygame처럼 프로그램에서 명시적으로 임포트를 해주어야만 된다. 프로그램을 import RPi.GPIO as GPIO로 시작한다.

주의: 라즈베리파이의 GPIO 포트는 전압 스파크나 단락에 대한 어떠한 보호도 제공하지 않는다. 자신이 만든 회로를 라즈베리파이에 연결하기 전에 항상 이상이 없는지를 확인해야 된다. 가능하면 Gertboard(Chapter 13. 하드웨어 해킹 참조) 같은 보호기능을 제공하는 아이솔레이션 보드를 사용하는 것이 좋다.

전류 제한 저항 값 계산하기

LED는 과도한 전류가 흐르면 타버리기 때문에 전류를 제한할 저항이 필요하다. 저항이 필요하다는 점을 아는 것 이외에도 어떤 값을 가지는 저항을 써야 하는지도 알아야 한다. 저항 값이 너무 크면 빛의 밝기가 너무 어둡거나 아예 켜지지 않게 되고 너무 낮으면 LED를 태울 수 있다.

LED에 흐를 수 있는 순방향 최대 전류(mA)와 순방향 최대 전압(V)을 알아야 적정한 저항 값을 계산할 수 있다. 순방향 최대 전압은 3.3V이하여야 한다. 그 보다 큰 경우에는 트랜지스터 같은 스위칭 소자를 사용해야 한다.

얼마나 큰 저항 값을 가진 저항이 필요한 지를 알 수 있는 가장 쉬운 방법은 R = (V – F) / I 공식을 사용하는 것이다. 여기서 R은 오옴 단위의 저항 값이고, V는 LED에 가해지는 전압, F는 LED의 순방향 최대 전압, I는 암페어 단위의 LED의 순방향 최대전류이다.

전형적인 빨간색 LED의 순방향 최대 전류를 25mA, 순방향 최대 전압을 1.7V, 그리고 가해지는 전압을 라즈베리파이의 GPIO 포트의 출력전압인 3.3V라고 가정해보면 저항 값은 (3.3 – 1.7) / 0.026 = 64 오옴이다. 64 오옴 이상의 저항을 사용하면 LED를 안전하게 보호할 수 있다. 판매되는

www.kett.or.kr

저항 중에는 정확히 이 값을 가지는 저항이 흔치 않다. 따라서 그 이상 되는 저항 중 쉽게 구할 수 있는 것은 68 오옴 짜리가 있다.

사용하고자 하는 LED의 순방향 최대 전압과 전류를 모르는 경우(데이터시트가 없다거나 고장난 전자제품에서 뜯어낸 경우) 충분히 큰 저항부터 시작해 값이 작은 저항으로 바꿔가면서 밝기를 살펴보면 된다. 한번 타버린 LED는 수리할 수 없다.

GPIO 출력: LED 밝히기

첫 예제는 LED와 저항을 이용하여 간단한 회로를 구성해보자. 파이썬 프로그램을 작성하여 GPIO 포트에 명령을 내리면 LED로 동작을 확인해 볼 수 있다. 저항은 LED가 과전압으로 타는 것을 방지하기 위한 것이다.

회로를 구성하기 위해 브레드보드와 두 개의 점퍼케이블, LED, 저항이 필요하다. 브레드보드 없이 전선을 꼬아서 회로를 구성할 수도 있겠지만 브레드보드를 사용하는 것이 건전한 방법이다.

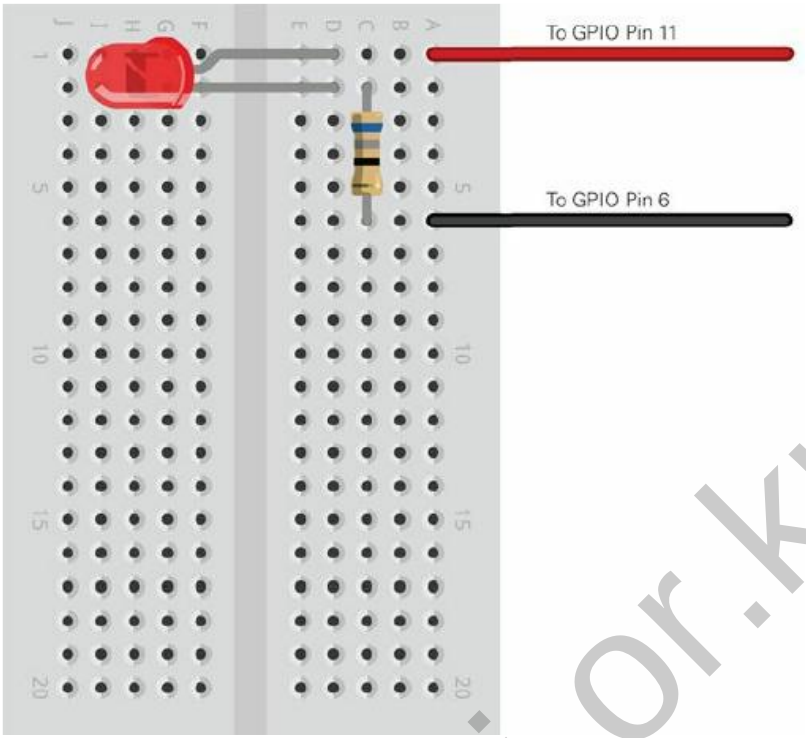
그림 13-3처럼 되도록 브레드보드에 회로를 구성해보자.

1. LED의 긴 다리(Anode, +)를 브레드보드의 한 행에 꽂고 짧은 다리(Cathode, -)를 다른 행에 꽂는다. 두 다리를 같은 행에 꽂으면 안 된다.
2. 저항의 한쪽 다리를 LED의 짧은 다리와 같은 행에 꽂고 반대편 다리는 빈 행에 꽂는다. 방향은 상관이 없다. 저항은 극성이 없는 소자이다.
3. 점퍼선을 이용해서 라즈베리파이 GPIO 포트의 핀 11을 LED의 긴 다리와 같은 행에 연결한다.
4. 다른 점퍼선을 이용해서 라즈베리파이 GPIO 포트의 핀 6을 저항의 홀로 꽂혀 있는 다리와 같은 행에 연결한다.

주의: 라즈베리파이의 GPIO 포트에 선을 연결할 때 매우 주의해야 한다. 앞서도 설명한 바 있지만 잘못된 핀에 연결하면 라즈베리파이에 심각한 손상을 줄 수 있다.

그림 13-3: 간단한 LED 출력을 위한 브레드보드 결선

www.ketii.or.kr



기본적으로 라즈베리파이의 GPIO 핀들은 꺼져 있기 때문에 이 시점에서는 지극히 당연하게도 아무 일도 일어나지 않는다. 핀 11에 연결된 선을 핀 1로 옮겨보면 LED가 켜질 것이다. 핀 2에 연결하지 않도록 주의한다. 저항 값이 3.3V에 맞추어져 있기 때문에 5V 핀에 연결하기에는 부적합하다. 불이 켜지는 것을 확인한 후에는 점퍼선을 다시 핀11에 되돌려 놓는다.

LED를 동작시키기 위한 새로운 파이썬 프로젝트를 시작한다. Chapter 12. 파이썬 프로그래밍에서 해보았던 것처럼 데비안 배포판에 포함되어 있는 IDLE를 이용하거나 그냥 텍스트 에디터를 이용해도 된다.

이 장의 앞에서 설치했던 GPIO 라이브러리를 사용하려면 임포트를 해주어야 한다. 따라서 맨 앞에 다음 행을 입력한다.

```
import RPi.GPIO as GPIO
```

파이썬은 대소문자를 구별하기 때문에 주의해서 그대로 입력해야 한다. LED가 주기적으로 반짝이게 하기 위해서는 시간 개념이 필요하다. time모듈을 임포트한다.

```
import time
```

GPIO 포트들은 사용하기 전에 먼저 입력 또는 출력으로 초기화를 해주어야 한다. 이 예제에서는 핀 11이 출력으로 사용되므로 초기화를 해준다.

www.ketri.or.kr

GPIO.setup(11, GPIO.OUT)

다른 장치들을 더 제어할 경우에는 더 많은 핀들을 초기화해야겠지만 우선은 하나만으로 충분하다.

출력으로 초기화 되었으므로 핀 11의 출력을 3.3V로 켜거나 끔으로써 간단한 이진 논리를 보일 수 있다.GPIO.output(11, True)명령은 핀을 켜고, GPIO.output(11, False) 명령은 핀을 끈다. 핀은 최근 상태를 기억한다. 핀을 켜는 명령을 내린 후 파이썬 프로그램을 끝내면 핀은 다른 명령을 다시 내리기 전까지는 계속 켜진 상태를 유지한다.

간단히 GPIO.out(11, True)를 입력할 수도 있지만 좀 더 재미있게 하기 위해 다음의 무한루프를 추가해보자.

while True:

그 다음, 다음 행들을 추가하여 핀을 켠 후 2 초간 기다렸다가 핀을 끄고 다시 2초를 기다리는 동작을 반복하도록 한다.

GPIO.output(11, True)

time.sleep(2)

GPIO.output(11, False)

time.sleep(2)

완성된 프로그램은 다음과 같다(그림 13-4 참조).

import RPi.GPIO as GPIO

import time

GPIO.setup(11, GPIO.OUT)

while True:

GPIO.output(11, True)

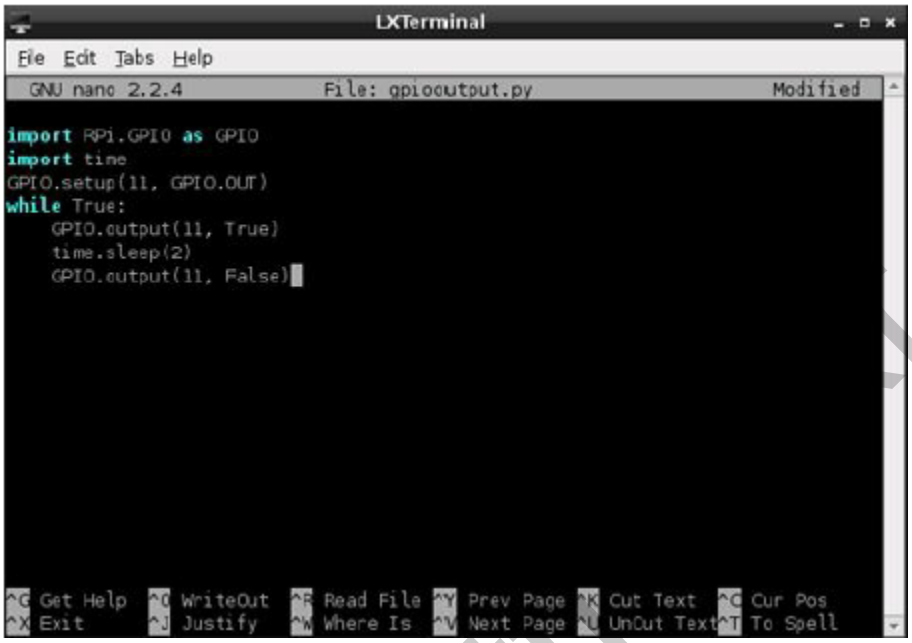
time.sleep(2)

GPIO.output(11, False)

www.ketti.or.kr

time.sleep(2)

그림 13-4: gpioout.py 프로그램



파일을 gpiooutput.py로 저장한다. SPE 같은 파이썬 개발환경을 사용할 경우 에디터 안에서 프로그램을 직접 실행하려고 시도하지 말라. 대부분의 라즈베리파이 리눅스 배포판들은 GPIO 포트를 슈퍼유저만 사용할 수 있도록 제한하고 있다. 따라서 터미널 창에서 sudo python gpiooutput.py 명령으로 실행해야 한다. 오류가 없다면 LED가 주기적인 간격으로 깜빡일 것이다. 나만의 최초의 홈메이드 라즈베리파이 장치가 작동하는 것이다.

동작을 하지 않더라도 실망할 필요는 없다. 우선 모든 연결들을 점검한다. 브레드보드의 구멍들은 아주 작아서 부품들을 생각과는 달리 다른 곳에 꽂았을 가능성이 있다. 그 다음 점퍼선을 GPIO포트의 제대로 된 핀 번호에 연결했는지를 점검한다. 라즈베리파이에 라벨링이 되어 있지 않기 때문에 잘못 꽂기가 쉽다. 그 다음 부품들을 재점검해 본다. LED가 3.3V 이상의 전압을 필요로 하거나 너무 큰 경우 불이 켜지지 않는다.

이 예제는 기초적이기는 하지만 근본 개념을 보이기에는 충분하다. 기능을 확장하여 LED대신 부저를 연결하면 경고음을 내도록 할 수 있고, 서보나 모터를 연결하면 로보틱스 플랫폼의 일부로 사용할 수도 있다. GPIO 핀들을 켜거나 끄는 코드는 새 이메일이 왔을 때 LED가 켜지게 하거나 IRC 채널에 친구가 입장했을 때 깃발을 들게 하는 등의 다른 프로그램의 일부로 통합될 수 있다.

GPIO 입력: 버튼 읽기

GPIO포트를 출력으로 사용하는 것도 유용하지만 하나 또는 그 이상의 입력과 같이 결합하여 사용하면 훨씬 더 유용해진다. 푸시 버튼 스위치를 다른 GPIO핀에 연결하여 파이썬에서 상태를 읽을 수 있는지 보자.

LED출력 프로그램과 마찬가지로 이 입력 예제도 파이썬 GPIO 라이브러리를 사용한다. 라이브러리는 이미 설치되어 있다고 가정하고, 이제 회로를 구성한다.

브레드보드에 이미 만들어 놓은 GPIO 출력 예제가 있으면 라즈베리파이와 연결된 점퍼선을 빼도 되고 그대로 두어도 상관없다. 이 예제는 다른 핀을 사용하기 때문에 같이 연결되어 있어도 괜찮다. 이전 예제를 그대로 두기로 결정했으면 이번 예제는 이전 예제와는 다른 행을 사용해서 구성한다.

다음과 같이 회로를 구성한다.

1. 푸시 버튼 스위치를 브레드보드에 꽂는다. 대부분의 스위치는 다리가 두 개이거나 네 개이다. 다리가 네 개인 경우 두 개씩 쌍으로 연결되어 있다. 어느 다리가 같이 연결되어 있는지 데이터시트를 참조하여 확인하고 서로 연결되지 않은 다리가 다른 행에 꽂히도록 해야 된다.

2. 10 KΩ 저항의 한 쪽 다리를 푸시 버튼 스위치의 한 쪽 다리와 같은 행에 꽂고 다른 쪽 다리는 사용되지 않은 행에 꽂는다. 이 저항은 풀업 저항으로 스위치가 눌리지 않았을 때 라즈베리파이의 입력 핀을 참조 전압(3.3V)에 연결되도록 함으로써 확실히 high상태로 감지되도록 하는 역할을 한다.

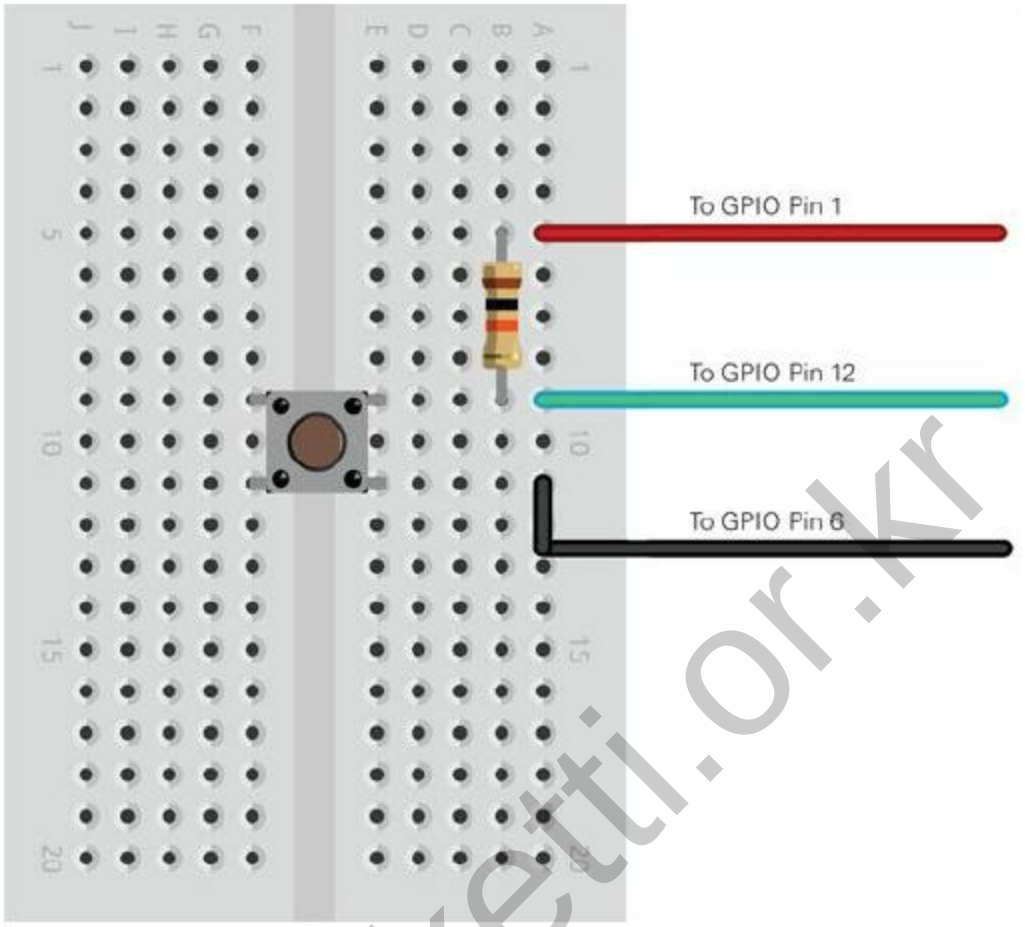
3. 풀업 저항의 연결되지 않은 쪽 다리를 라즈베리파이의 GPIO 포트 핀 1에 연결한다. 이 핀은 참조 전압(3.3V)을 제공한다.

4. 푸시 버튼 스위치의 사용되지 않은 다리를 라즈베리파이의 GPIO 포트 핀 6에 연결한다. 이 핀은 그라운드 연결을 제공한다.

5. 라즈베리파이의 GPIO 포트 핀 12를 10KΩ 저항과 푸시 버튼 스위치가 연결된 행에 연결한다. 브레드보드의 전체 모습은 그림 13-5과 같이 되어야 한다.

그림 13-5: 간단한 푸시버튼 입력을 위한 브레드보드 결선

www.keiti.or.kr



라즈베리파이의 GPIO 포트 핀 12는 3.3V 전압에 풀업 저항을 통해 연결된 덕분에 high 상태를 유지한다. 푸시 버튼이 눌리게 되면 핀 12는 그라운드에 연결되고 상태는 low로 바뀐다. 파이썬 프로그램은 버튼이 눌린 것을 알게 된다.

풀업 저항이 왜 필요한지 의아해 할 수 있다. 저항을 제거한 상태에서는 스위치를 누르기 전까지는 핀 12의 상태가 high인지 low인지가 불명확하다. 프로그램에서 이 핀의 상태를 읽으면 high로 읽힐 수도 있고 low로 읽힐 수도 있다. 이런 상태의 핀을 플로팅 핀이라고 한다.

새 파이썬 파일을 열어 GPIO 라이브러리 임포트 명령을 입력한다.

```
import RPi.GPIO as GPIO
```

이 예제에서는 시간과 관련된 명령이 없기 때문에 time라이브러리는 임포트할 필요가 없다. 핀 12을 입력으로 초기화한다.

```
GPIO.setup(12, GPIO.IN)
```

www.ketti.or.kr

다른 핀을 사용하려면 핀 번호를 바꾸어 주면 된다.

입력 핀이 low가 되는지를 체크하기 위해 (바꿔 말하면 스위치가 눌리는지) 무한 루프를 만든다.

while True:

입력 핀의 상태를 읽는 것은 결과 값을 받아 변수에 저장하는 것만 빼고는 출력 핀의 상태를 설정하는 것과 매우 유사하다. 다음의 명령어는 input_value라고 하는 변수를 생성하고 핀 12의 값을 대입한다.

input_value = GPIO.input(12)

지금이라도 프로그램을 실행하면 동작을 하기는 하지만 아무 유용한 일도 하지 않는다. 무슨 일이 일어나고 있는지를 알 수 있도록 다음 print 명령을 입력한다.

if input_value == False:

print("The button has been pressed.")

while input_value == False:

input_value = GPIO.input(12)

마지막 두 행 – 두 번째 루프와 두 번째 input_value, 내부 루프 – 은 중요하다. 라즈베리파이가 고성능의 데스크톱이나 랩톱보다는 낮은 성능을 가지고 있지만 그래도 파이썬 프로그램은 상당히 빠른 속도로 실행되고 있다. 내부 루프는 핀 12의 상태가 더 이상 low가 아닐 때까지 즉, 버튼이 놓일 때까지 체크를 반복한다. 이 내부 루프가 없으면 버튼이 눌려 있는 동안 메시지가 반복적으로 화면에 출력될 것이다.

최종 프로그램은 다음과 같은 모습니다.

import RPi.GPIO as GPIO

GPIO.setup(12, GPIO.IN)

while True:

input_value = GPIO.inut(12)

if input_value == False:

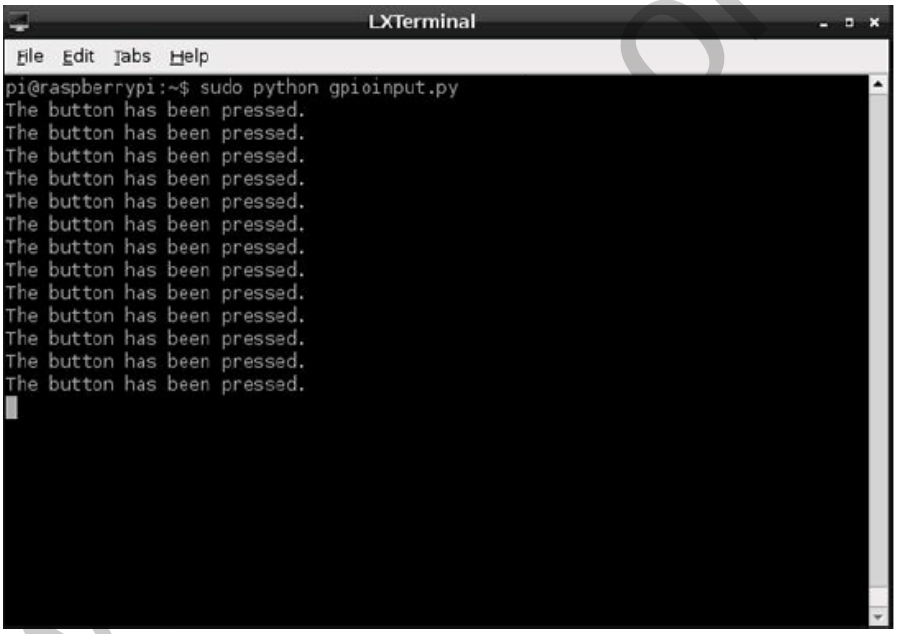
www.ketti.or.kr

```
print("The button has been pressed.")

while input_value == False:

input_value = GPIO.input(12)
```

파일을 pgioinput.py로 저장하고 터미널 창에서 sudo python gpioinput.py 명령으로 실행한다. 처음에는 아무 일도 일어나지 않는다. 버튼을 누르면 프로그램에서 메시지를 화면에 출력할 것이다 (그림 13-6 참조). 버튼을 눌렀다 놓았다 반복해보면 메시지가 반복해서 출력된다.

간단한 프로그램 이지만 여러 목적으로 사용될 수 있다. 스위치뿐만 아니라 센서 또는 외부 마이크로 컨트롤러 같은 별개의 장치가 연결되어 있을 때 이를 읽는데도 사용될 수 있다.

그림 13-6: gpioinput.py 프로그램의 출력 결과



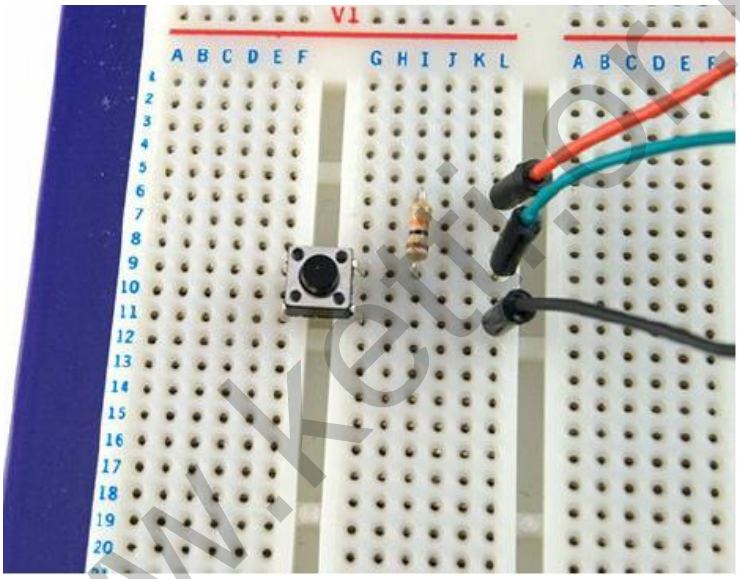
여러 개의 푸시 버튼을 읽도록 코드를 확장하면 4-버튼 게임 컨트롤러도 만들 수 있다. Chapter 12. 파이썬 프로그래밍의 스네이크 게임과 연결하면 라즈베리파이를 간단한 게임 콘솔로 변신시킬 수도 있다. 입력과 출력 예제를 결합하여 하나의 프로그램으로 만들 수도 있다. 버튼이 눌리기를 기다렸다가 출력 핀에 high를 보내어 LED를 켜는 것이다. 이번 내용을 제대로 이해했는지 확인해보기 위해 당장 시행에 옮겨보자. 막히는 부분이 있거나 자신의 방법을 비교해보고 싶으면 참조 A의 샘플 코드를 참조하라.

브레드보드 벗어나기

이 장의 앞에서 보인 예제들에서 보듯이 시험을 위한 프로토타입 회로를 빠르게 만들어 보는데 있어 브레드보드는 매우 뛰어나다. 브레드보드는 빠르고 재사용이 가능하며 부품들에 손상을 주지 않는다.

하지만, 브레드보드도 몇 가지 단점이 있다. 브레드보드는 부피가 크고, 비싸고, 이동 중에 부품이 빠지기 쉽다. 그림 13-7을 보면 푸시버튼이 느슨하게 꽂혀 있어서 조금만 거칠게 건드리면 빠져나올 것 같은 모습을 하고 있다.

그림 13-7: 브레드보드 위에 느슨하게 연결된 부품들



때문에 장치를 프로토타이핑하는 초기 시점에는 브레드보드가 사용되지만 나중에는 인쇄회로기판(PCB; Printed Circuit Board)으로 대체되기 마련이다. 가정에서도 인쇄와 에칭을 통해 자신만의 PCB를 만드는 것이 가능하다. 하지만 그전에 중간 단계로 스트라이프보드를 이용하면 좀 더 쉽게 영구적이고 독립적인 회로를 만들어 낼 수 있다.

일견 스트라이프보드는 브레드보드와 유사하다. 표면이 2.54mm 간격의 구멍으로 채워져 있기 때문이다. 하지만 브레드보드와는 달리 전자 부품의 다리를 고정하기 위한 정교한 메커니즘이 없다. 대신 스트라이프보드에 납땜으로 고정시켜야 한다. 스트라이프보드는 상표인 Veroboard로 불리기도 하는데 그것은 영국에서는 Vero Technologies, 캐나다에서는 Pixel Print사의 등록상표이다.

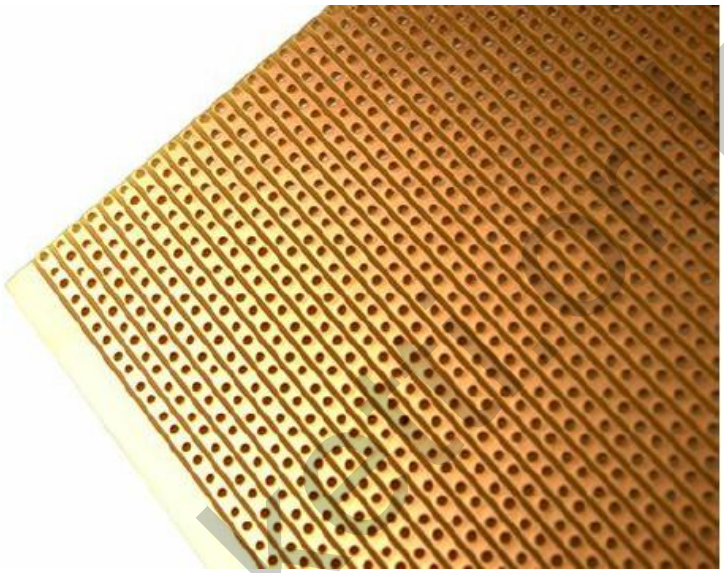
브레드보드에 비해 스트라이프보드를 사용하는 것은 많은 장점을 가지고 있다. 스트라이프보드는 같은 크기의 브레드보드보다 훨씬 싸고 좀 더 작게 집적이 가능하다. 큰 스트라이프보드를 잘라

www.kett.or.kr

여러 개의 작은 회로를 만드는 것도 가능하다.

부품들은 스트라이프보드에 땜납으로 고정되기 때문에 브레드보드에 비해 훨씬 강건하다. 스트라이프보드는 부품이 떨어져 분실될 위험 없이 이 곳 저 곳으로 옮겨 가지고 다닐 수도 있다.그림 13-8을 보면 스트라이프보드 뒷면의 동박 트랙을 볼 수 있다.

그림 13-8: 스트라이프보드 밑면의 동박 트랙들



스트라이프보드는 사용하기 쉽고 커스텀 회로 기판을 디자인하고 제조하기 위한 징검다리로써 매우 훌륭하다. 하지만 스트라이프보드를 구매하기 전에 알아둘 것이 있다.

- 시중에는 여러 종류의 스트라이프보드가 있다. 어떤 스트라이프보드는 밑면의 동박 트랙이 행 또는 열 한 방향으로만 되어 있고 반면에 다른 스트라이프보드는 브레드보드처럼 가운데 간격을 두고 두 개의 행들로 나뉘어 있다. 또 다른 종류의 스트라이프보드는(프로젝트 보드라고도 불리는) 구리 트랙이 전혀 없어 전선을 이용해서 연결해야 한다.

- 스트라이프보드는 각각 다른 두께와 다른 재질로 만들어 질 수 있어 어떤 종류는 어떤 프로젝트에 적합하고 다른 프로젝트에는 다른 종류가 적합할 수 있다. 예를 들어 내열성을 가진 스트라이프보드는 고온의 환경에서 동작하는데 적합하고 두꺼운 스트라이프보드는 거친 환경에 노출되는 곳에 사용될 수 있다.

- 스트라이프보드에 부품 배치를 간편하게 하기 위해 밑면의 트랙을 끊어 부품들을 전기적으로 분리시킬 수 있다. 이렇게 하면 보드 공간의 낭비를 피할 수 있고 상대적으로 복잡

한 회로에서는 불가결할 수도 있다. 어쨌든 최상의 결과를 얻기 위해서는 트랙 커터라고 불리는 조그만 수공구가 필요할 것이다. 스트라이프보드를 사용할 계획이라면 이 공구를 구매 목록에 추가해두는 것을 잊지 말기를 바란다. 또한 작은 드릴 날도 필요할 것이다.

무시하면 안될 몇 가지 스트라이프보드 작업 팁이 더 있다.

- 밑면의 구리 트랙들은 전혀 코팅이 되어 있지 않다. 트랙을 손으로 건드리면 구리 면에 변색을 초래해 납땜이 잘 안되게 된다. 지금 당장 사용할 것이 아니라면 구리 면을 건드리지 않는 것이 좋다. 이미 변색이 되었다면 납땜을 하기 전에 철 솜으로 구리 면을 부드럽게 문질러 변색된 표면을 제거한다.
- 인쇄회로기판과는 달리 스트라이프보드에는 납이 번지는 것을 막아주는 솔더 마스크가 없다. 따라서 PCB에서 보다는 납땜 작업이 조금 더 어렵다. 납땜 작업을 하다 보면 납이 번져 트랙과 트랙을 넘어 연결되는 수도 있다. 이럴 때는 납 제거용 위크를 이용하여 납을 제거한 후 다시 작업을 한다.
- 스트라이프보드의 구멍을 이용하면 보드를 원하는 크기로 자르기가 쉽지만 가장자리가 울퉁불퉁하게 남는다. 스트라이프보드를 잘라낸 뒤에 회로를 조립하기 전에 줄로 갈아낸다. 이 작업을 할 때는 마스크를 반드시 쓰는 것이 좋다. 스트라이프보드에서 발생하는 먼지는 건강에 특히 해롭다.

납땜에 대한 간단한 가이드

납땜 인두를 하나 준비한다고 다 되는 것이 아니라 납땜을 하는 방법에 대해 알아둘 필요가 있다. 다른 기술들과 마찬가지로 납땜도 능숙해지기 위해서는 연습을 많이 해 보는 수 밖에 없다. 여기서 이야기하는 요령을 숙지하고 연습을 직접 해 보면 얼마 지나지 않아 깨끗하고 간결한 납땜을 할 수 있게 될 것이다.

주의: 사족 같기는 하지만 납땜 인두는 사용 시 매우 뜨겁다. 팁은 물론 팁과 떨어진 부분이라도 노출되어 있는 금속 부분에 닿지 않도록 주의해야 하고 인두 끝을 항상 아래쪽으로 향하도록 두는 것이 좋다. 납땜 인두를 구매 시 스탠드와 같이 사거나 별도로 내열성을 보증하는 스탠드를 구매하기 바란다. 뜨거운 인두를 사람이 없는 곳에 방치하지 말고, 혹시 떨어뜨리는 경우 잡으려고 해서는 안 된다.

납땜은 두 가지 부품을 접합시키기 위하여 소량의 금속을 녹이는 작업이다. 라즈베리파이를 뒤집어 보면 이 것을 직접 눈으로 확인할 수 있다. 비교적 큰 모든 부품들은 이른바 쓰루-홀 납땜이라

www.ketti.or.kr

는 방식으로 부착되어 있다. 쓰루-홀 납땜 방식이란 부품의 다리가 인쇄회로 기판의 구멍을 통과하여 납땜으로 고정되는 것을 의미한다.

땜납은 순수한 금속은 아니고 플럭스라고 하는 물질을 포함하고 있는데 납땜이 될 표면의 오물을 녹여 청결하게 함으로써 가능한 접합이 잘 이루어지도록 하기 위한 것이다. 대부분의 전자회로용 땜납은 내부에 3 ~ 5개의 플럭스 코어를 포함하고 있다. 반죽이나 액체 형태의 플럭스를 별도로 구매할 수도 있는데 일반적인 취미 수준의 전자 작업에 있어서 꼭 필요한 것은 아니다.

납땜 프로젝트를 시작할 때, 청결하고 잘 정돈된 작업공간을 확보해야 한다. 또한 공기 배출이 원활해야 한다. 납땜 시 발생하는 연기는 건강에 매우 해롭다. 소량의 취미 수준의 납땜에서 발생하는 양일지라도 가능한 인체에 노출되는 것을 최소화하는 것이 좋다.

녹은 납이 떨어지면 작업대 표면을 태울 수 있기 때문에 보호를 위해 정전기 방지 매트를 까는 것이 좋다(그림 13-9 참조). 표지가 코팅된 잡지를 까는 것도 괜찮은 방법이다. 하지만 신문은 떨어지는 땜납에 의해 구멍이 난다.

그림 13-9: 정전기 방지 매트를 깐 납땜 작업 환경 예

세밀한 작업을 할 때는 보안경을 쓰는 것이 좋다. 땜납 속의 뜨거운 플럭스가 튀어 눈에 들어가면 돌이킬 수 없는 화상을 입을 수 있다.

인두가 뜨겁기는 하지만 두려움 때문에 납땜을 기피할 필요는 없다. 땜납은 빨리 식고 화상도 그리 흔한 것은 아니다. 장비를 믿되 두려워할 필요는 없다.

www.kett.or.kr

작업 공간이 마련되었으면 공구들을 정리할 차례다. 납땜 인두를 주로 사용하는 손 쪽에 두고 전원 코드가 작업 공간을 가로지르지 않도록 한다. 납땜 인두의 전원을 꽂기 전에 인두를 자유롭게 움직일 수 있는지를 확인해 본다. 전원 코드가 걸리적거리면 화상으로 연결되기 쉽다.

납땜 작업용 스펀지를 흥건하지 않을 정도로 물에 적신다. 인두 팁을 깨끗하게 닦는데 사용하는데, 말라버리게 되면 인두 팁에 녹아 붙어 팁을 상하게 할 수도 있다.

인두가 적정 온도로 가열되는 데는 몇 분이 걸린다. 온도 조절이 되는 인두를 구입했다면 일반적으로 불 빛이 켜져 적정 온도임을 알려주거나 숫자로 온도를 읽을 수 있도록 되어 있을 것이다(온도가 어떤 식으로 표시되는지 따라온 매뉴얼을 참조하기 바란다).

인두가 적정 온도에 도달하면 이제 인두 팁에 땜납을 입힐 차례이다.

1. 땜납의 끝을 인두 팁에 갖다 대어 소량의 납이 인두에 녹아 붙도록 한다. 너무 많은 양의 납이 녹지 않도록 주의한다. 땜납이 낭비되기도 하지만 작업대 위에 떨어질 수 있기 때문이다.

2. 인두 팁을 스폰지에 문질러서 인두 팁이 은색으로 매끄럽게 코팅이 되도록 한다(그림 13-10 참조). 필요하다면 땜납을 더 녹여주면 된다.

3. 인두 팁을 스폰지에 댔을 때 치익 소리가 나면서 물이 튄다면 물이 너무 많은 것이다. 스폰지의 물을 조금 짜낸 후 다시 해 본다.

그림 13-10: 납이 입혀진 납땜 인두 팁

www.keti.or.kr

인두 팁을 도금해 놓는 것은 팁의 손상을 막고 납땜이 될 표면으로 열이 효율적으로 전달될 수 있도록 하기 위한 것이다. 도금이 잘 안되면 납땜이 실패하는 주 원인이 된다. 납땜 작업을 많이 한 후에 팁의 반짝임이 가시면 도금 작업을 다시 해준다.

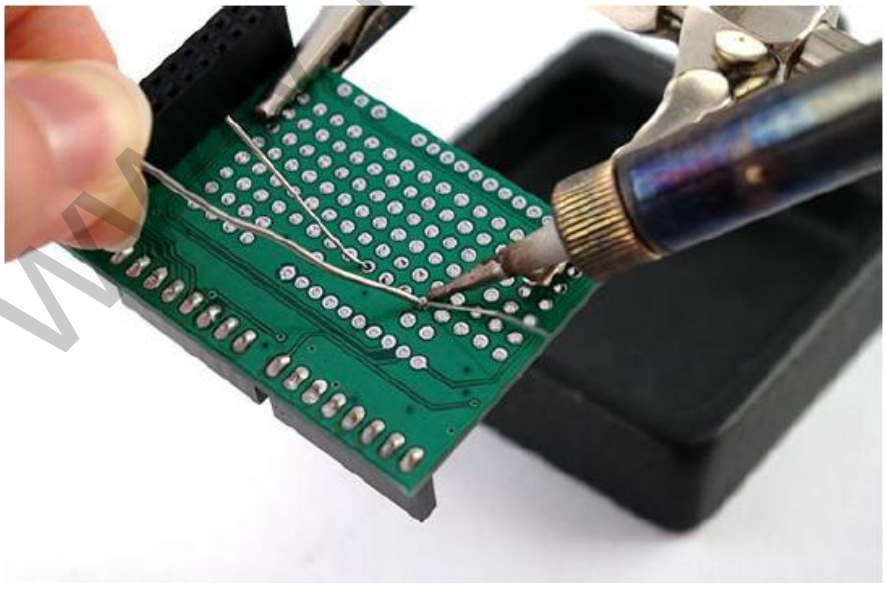
납땜 인두 준비가 끝났으면 이제 납땜을 시작할 차례이다. 인쇄회로기판과 부품들을 잘 보이는 곳에 두고 인두의 전원 코드 길이를 충분히 확보한다.

1. 인쇄회로기판이나 스트라이프보드 또는 유사한 쓰루-홀 보드에 부품을 납땜할 때는 부품의 다리를 구멍을 통과시킨 다음(납땜이 될 동박이 있는 반대편에서 부품을 꽂아서 동박 쪽으로 다리가 나오도록 꽂아야 한다) 다리를 바깥쪽으로 살짝 구부려서 기판을 뒤집었을 때 부품이 떨어지지 않도록 해준다.

2. 작업대에 보드가 안정되어 있는 상태에서 인두 팁을 부품의 다리와 기판의 동박이 동시에 닿도록 살짝 누르면서 접촉한다. 인두 팁을 양쪽에 동시에 접촉하는 것이 중요하다. 한 쪽만 접촉이 되면 납이 반대쪽으로만 붙어 결과가 좋지 않게 된다.

3. 인두 팁을 갖다 댄 지점이 몇 초면 충분히 가열된다. 셋을 센 후 납을 부품과 동박의 접점에 갖다 댄다(그림 13-11 참조). 납이 녹지 않으면 인두를 떼었다가 다시 대어본다. 코팅이 잘 된 부분을 찾아 위치를 바꿔가면서 대어보는 것이 좋다.

그림 13-11: 인쇄회로기판에 부품을 납땜하기



4. 납이 녹아 내리면서 구멍을 메우는 것을 확인할 수 있다. 납땜이 잘 될 만큼 그 부분이 충분히 가열되었다는 것을 의미한다.

www.ketti.or.kr

5. 납땜이 뾰족한 산 모양을 이루는 것과 동시에 납을 먼저 떼어내고 연이어 인두를 떼어낸다(인두를 먼저 떼어내면 납이 붙은 채로 굳어버릴 것이다).

납땜이 잘 되면 매끄럽고 광택이 나는 뾰족한 산 모양을 이루며 끝 부분이 부품의 다리를 따라 매끄럽게 연결된 모양이 된다. 잘 안되었을 때는 인두를 다시 대고 납을 조금 더 녹여서 납땜을 다시 해주면 된다. 납이 너무 많이 녹아 붙었을 때는 납 제거용 위크를 대고 인두로 녹여 납을 제거한 후 다시 작업한다.

인두를 부품에 접촉한 채 수 초 이상 두면 안 된다. 특히 집적 회로 같이 열에 약한 부품들은 열에 쉽게 손상을 입을 수 있으므로 주의해야 한다. 온도 조절이 가능한 인두를 사용하고 있으면 부품에 따라 적정 온도를 설정하여 사용하도록 한다(인두에 부속된 매뉴얼 및 부품들의 데이터시트를 참조하기 바란다).

작업을 마쳤을 때는 인두 팁을 도금해두는 것을 잊지 말기 바란다. 도금을 해두지 않으면 인두 팁이 쉽게 부식되고 수명보다 빨리 인두 팁을 갈아주어야 한다.

주의: 인두로 양 쪽을 동시에 가열하는 것을 잊지 말기 바란다. 한 쪽만 가열하면 흔히 말하는 냉땜이나 건땜을 유발하여 납이 한 쪽에만 붙게 된다.

모든 기술이 그렇듯이 납땜은 연습이 필요하다. 많은 전자부품 상점들이 인쇄회로기판과 필요한 쓰루-홀 부품들을 포함하는 전자회로 키트를 판매한다. 라즈베리파이를 위한 애드온 보드들 중에도 납땜을 필요로 하는 키트형태로 판매가 되는 것들이 많이 있다.

본 문서는 자유롭게 배포 가능합니다.
다만 상업적 목적 및 문서 수정 배포시
한국전자통신교육원(kks@ketti.or.kr)로 문의 부탁드립니다.

www.ketti.or.kr