KossLab
KOREA OPEN SOURCE SOFTWARE DEVELOPERS LAB
공개SW개발자 Lab

리눅스 및 안드로이드에서 사용되는 명령어 집합체

Toybox

공개 SW 개발자 Lab 오픈소스프론티어 3 기 이창환

KossLab

임베디드 디바이스 개발 시 커널로 리눅스를 많이 사용하는데, 그때 그 커널과 함께 리눅스 명령어도 필요하게 된다. 모든 명령어를 지원하기 위해서는 다수의 개별적인 패키지들이 필요하다. 한 프로젝트에서 그 많은 명령어를 한 번에 지원해 준다면 자원 제약적인 임베디드 환경에서 공간 효율성이 극대화될 것이다.
상대적으로 경량화된 리눅스 명령어 세트로 busybox(GPL)가 유명하고, 임베디드 업계에서 많이 사용되고 있다. 기능은 그보다 조금 적지만, 좀 더 자유로운 BSD라이센스를 사용하는 Toybox에 대해 알아보고자 한다.

[목차]

1 개요

리눅스 배포판에는 많은 리눅스 명령어들이 기본으로 탑재되어 있으며, 대부분의 명령어는 GPL 라이센스를 따른다. 아주 기본이 되는 명령어들은 GNU 코어 유틸리티(GNU coreutils) 프로젝트에 있는 명령어(ls, cp, mkdir 등)들이 사용되며, 그 외의 명령어들의 경우 각각 별도의 프로젝트들이 진행되고 있다. 그러나 의존성과 바이너리 크기 등의 문제로 임베디드 환경에서는 상대적으로 무겁고 같이 포팅(porting)되어야 하는 이런 프로젝트들이 많은 경우 부담스러운 것이 사실이다. 그래서 이런 대부분의 명령을 가볍고 간단하게 지원하고 싶은 요구 때문에 busybox라는 프로젝트가 생겨났고 유명해졌다. 하지만 이 프로젝트 또한 GPL 라이센스를 사용하고 있으므로 배포 시 엄격하게 소스 코드가 공개되어야 하는 의무를 지고 있다. 그래서 비교적 최근에 시작하는 프로젝트의 경우 도입 검토 시 피하게 되는 요소이기도 했다. 그래서 GPL보다 가벼운 BSD라이센스여서 소스 코드 공개가 자유로운 임베디드용 리눅스 명령어 집합체 프로젝트가 생기게 되었는데, 이것이 바로 Toybox프로젝트(https://landley.net/toybox)이다.

[그림 1] Toybox 프로젝트 로고

[그림 1]은 프로젝트 로고로 실제로 벽에다 콜라 캔을 Rob Landley 본인이 직접 쌓아서 찍은 것이다. 코카콜라 제로(0)와 펩시 원(1)을 이진 표기로 사용하여 프로젝트 이름인 toybox의 각 알파벳을 8bit ASCII 코드로 나타냈으며 맨 아래 제로만 있는 줄은 NULL(0) 처리로 표시되었다.

2 타 프로젝트에서 도입

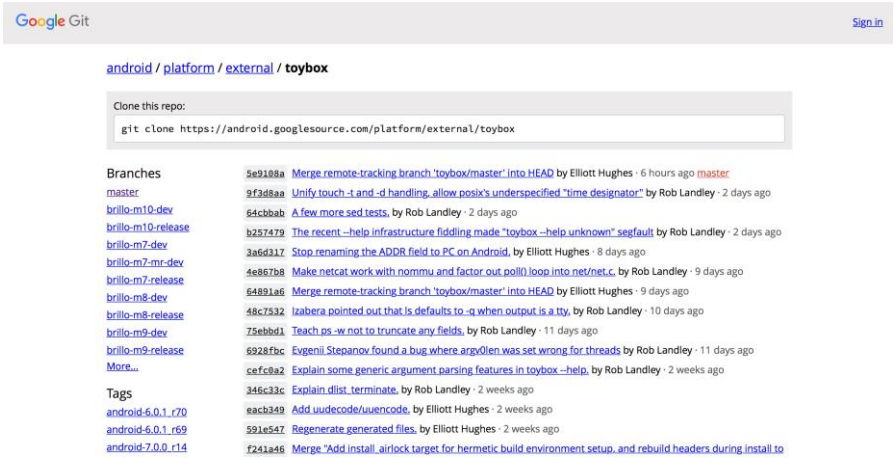
2006년부터 시작한 이 프로젝트는 비교적 최근에 유명한 프로젝트에 도입되고 있는데, 그 대표적인 프로젝트가 Android Open Source Project(이하 AOSP)와 Tizen이다. 프로젝트에 대한 통제권을 강화하기 위해 될 수 있으면 GPL을 피하는 프로젝트에서 많이 채택되는 추세인데, 구글의 경우 Android 프로젝트 메이저 버전 업을 할 때 진행 중인 모든 사항을 미리 공개하지 않고 정식 릴리즈가 된 이후에 코드를 공유한다. 이 경우 GPL을 사용하는 경우라면 진행 중인 경우라도 베타버전 바이너리를 배포할 때에도 코드가 공개되어야 하나 GPL을 사용하지 않는 것은 공개 의무가 없으므로 소스 공개에 대해 통제를 할 수 있는 장점이 있어 코드 공개에 대한 시기를 통제할 수 있다.

3 국내 기여자

국내에는 이 프로젝트에 삼성전자 엔지니어들이 일부 참여하고 있는데, 그 이유는 삼성전자가 참여하고 있는 Tizen 프로젝트가 Toybox를 리눅스 명령어로 채택하고 있어 국내 삼성전자 엔지니어들이 Toybox 코드에 기여하고 있다.

4 실행해보기

안드로이드 6.0 마시멜로 버전 이후부터 이 프로젝트가 도입되어 사용 되고 있으며, 해당 이상 버전의 안드로이드 단말을 가지고 있으면 이 프로젝트 명령어를 사용해 볼 수 있다.



[그림 2] AOSP의 Toybox 저장소 페이지

(https://android.googlesource.com/platform/external/toybox)

현재 안드로이드 7.0 Nougat 버전에서 지원되는 toybox 명령어는 [그림 3]과 같다.

```
flounder:/ $ toybox
acpi base64 basename blkid blockdev bunzip2 bzcat cal cat chattr chcon
chgrp chmod chown chroot cksum clear cmp comm cp cpio cut date dd
df dirname dmesg dos2unix du echo egrep env expand expr fallocate
false fgrep find flock free freeramdisk fsfreeze getenforce getprop
grep groups head help hostname hwclock id ifconfig inotifyd insmod
install ionice iorenice kill killall ln load_policy logname losetup
ls lsattr lsmod lsof lsusb makedevs md5sum mkdir mkfifo mknod mkswap
mktemp modinfo more mount mountpoint mv nbd-client nc netcat netstat
nice nl nohup od partprobe paste patch pgrep pidof pivot_root pkill
pmap printenv printf ps pwd pwdx readlink realpath renice restorecon
rev rfkill rm rmdir rmmod route runcon sed seq setenforce setprop
setsid sha1sum sleep sort split stat strings swapoff swapon switch_root
sync sysctl tac tail tar taskset tee time timeout top touch tr traceroute
traceroute6 true truncate tty ulimit umount uname uniq unix2dos uptime
usleep vconfig vmstat wc which whoami xargs xxd yes
```

[그림 3] Nexus 9 Android 7.0 Nougat 버전의 ADB shell에서 toybox 실행 결과

Android SDK를 설치하면 ADB가 설치되는데, ADB shell에서 위의 명령어를 입력하면 확인할 수 있다. 현재 만들어진 모든 명령어가 안드로이드에서 사용되지 않으면, 구글 엔지니어인 Elliot Hughes가 하나씩 추가해 넣고 있으며, 안드로이드용 개발 로드맵도 별도로 존재한다.

Tizen을 OS로 사용하는 단말인 삼성전자의 기어시리즈에서도 Android ADB와 유사한 SDB를 통해 해당 프로젝트가 사용되고 있는 것을 확인할 수 있다.

일반 리눅스 머신에서 사용해 보기 위해 아래와 같이 이미 빌드된 실행 파일을 가져와서 실행해 볼

수 있다.

```
$ wget https://landley.net/toybox/bin/toybox-x86_64
$ chmod +x toybox-x86_64
$ ./toybox-x86_64
```

5 저장소

메인테이너인 롭 랜들리의 개인 저장소에서 Mercurial로 코드가 관리되고 있다가 Github로 저장소를 옮기면서 현재 git으로 코드 관리를 하고 있으며, 그 저장소의 위치는 https://github.com/landley/toybox.git 이다. 그리고 약 1주에서 2주 간격으로 이 저장소에서 소스 코드를 가져와서 AOSP 저장소(https://android.googlesource.com/platform/external/toybox/)에 구글 엔지니어가 반영하고 있다. 그래서 안드로이드에서 발견된 Toybox버그 패치의 경우 원 저장소에 반영해야 안드로이드에도 같이 자연스럽게 반영이 된다.

6 사용언어

역사와 전통을 자랑하며 Linux Kernel의 개발언어인 C 프로그래밍 언어로 만들어져 있다.

7 소스 코드 다운로드 및 빌드 방법

```
$ git clone https://github.com/landley/toybox.git
```

```
$ make defconfig
$ make
$ ./toybox
```

```
lipi@ubuntu:~/src/toybox$ ./toybox
acpi base64 basename blkid blockdev bunzip2 bzcat cal cat catv chattr
chgrp chmod chown chroot chrt chvt cksum clear cmp comm count cp cpio
cut date df dirname dmesg dos2unix du echo egrep eject env expand
factor fallocate false fgrep file find flock free freeramdisk fsfreeze
fstype fsync getfattr grep groups halt head help hexedit hostid hostname
hwclock id ifconfig inotifyd insmod install ionice iorenice iotop
kill killall killall5 link ln login logname losetup ls lsattr lsmod
lspci lsusb makedevs md5sum mix mkdir mkfifo mknod mkpasswd mkswap
mktemp modinfo mount mountpoint mv nbd-client nc netcat netstat nice
nl nohup nproc nsenter od oneit partprobe passwd paste patch pgrep
pidof pivot_root pkill pmap poweroff printenv printf ps pwd pwdx readahead
readlink realpath reboot renice reset rev rfkill rm rmdir rmmod sed
seq setfattr setsid sha1sum shred sleep sort split stat strings su
swapoff swapon switch_root sync sysctl tac tail taskset tee time timeout
top touch true truncate tty tunctl ulimit umount uname uniq unix2dos
unlink unshare uptime usleep uudecode uuencode vconfig vmstat w wc
which who whoami xargs xxd yes
```

[그림 4] 디폴트 빌드 시 지원되는 Toybox 명령어

[그림 4]의 결과는 'defconfig' 옵션으로 디폴트 빌드를 할 경우 빌드된 바이너리가 지원하는 명령어 셋인데, 현재 안드로이드 7.0 Nougat 버전과 조금 다름을 알 수 있다. 디폴트 빌드를 할 경우 toys/pending에 있는 명령어들은 빌드가 되지 않는데, 이것은 외부에서 전달되고 메인테이너가 좀 더 수정이 필요하다고 판단되는 것들을 관리하는 디렉토리여서 디폴트 빌드에 포함되지 않는다. 그래서 빌드 에러나 발생하거나 경고가 많이 나오는 것을 볼 수 있다. 빌드 에러가 발생할 경우 적절하게 빌드 오류를 수정하여 패치도 할 수 있다.

$ make allyesconfig

이 옵션으로 빌드를 할 경우 디폴트 빌드와 달리 toys/pending에 있는 명령어까지 모두 포함해서 빌드가 가능하다. 만약 새로운 명령어를 넣어서 빌드를 할 경우 이 pending 디렉토리에 넣고 빌드해서 소스 코드를 넘기고 나중에 메인테이너가 완성도가 높다고 판단되면 적절한 디렉토리로 옮겨서 관리된다. [그림 5]는 위의 디폴트 빌드에 포함된 명령어뿐만 아니라 toys/example, toys/pending에 포함된 명령어까지 빌드된 결과이다.

```
lipi@ubuntu:~/src/toybox$ ./toybox
acpi addgroup adduser arp arping base64 basename blkid blockdev brctl
bunzip2 bzcat cal cat catv chattr chgrp chmod chown chroot chrt chvt
cksum clear cmp comm compress count cp cpio crond cut date dd deallocvt
delgroup deluser df dhcp dhcp6 dhcpd diff dirname dmesg dos2unix du
dumpleases echo egrep eject env expand expr factor fallocate false
fdisk fgrep file find flock fold free freeramdisk fsck fsfreeze fstype
fsync ftpget ftpput getfattr getty grep groupadd groupdel groups gunzip
gzip halt head hello help hexedit host hostid hostname hwclock iconv
id ifconfig init inotifyd insmod install ionice iorenice iotop ip
ipaddr ipcrm ipcs iplink iproute iprule iptunnel kill killall killall5
klogd last link ln logger login logname losetup ls lsattr lsmod lsof
lspci lsusb makedevs md5sum mdev mix mkdir mke2fs mkfifo mknod mkpasswd
mkswap mktemp modinfo modprobe more mount mountpoint mv nc netcat
netstat nice nl nohup nproc nsenter od oneit partprobe passwd paste
patch pgrep pidof ping pivot_root pkill pmap poweroff printenv printf
ps pwd pwdx readahead readlink realpath reboot renice reset rev rfkill
rm rmdir rmmod route sed seq setfattr setsid sh sha1sum sha224sum
sha256sum sha384sum sha512sum shred skeleton skeleton_alias sleep
sort split stat strings su sulogin swapoff swapon switch_root sync
sysctl syslogd tac tail tar taskset tee telnet telnetd test test_human_readable
test_many_options test_scankey tftp tftpd time timeout top touch toysh
tr traceroute traceroute6 true truncate tty tunctl ulimit umount uname
uniq unix2dos unlink unshare uptime useradd userdel usleep uudecode
uuencode vconfig vi vmstat w watch wc wget which who whoami xargs
xxd xzcat yes zcat
```

[그림 5] 전체 빌드 시 지원되는 Toybox 명령어

$ make distclean

빌드로 인해 생긴 오브젝트 파일들을 없애주는 옵션이다. 이전 빌드 시 생성된 파일들이 다음 빌드에 영향을 주는 경우가 많이 발생하니 빌드 전에 항상 클린 후에 빌드하는 것이 좋다.

8 패치 전달하기

만약 에러가 발견되거나 문서 작업 등으로 인해 수정 내역을 보내기 위해서는 Github의 Pull Request를 보내어 반영을 요청하여 반영될 수도 있으나, 공식적으로는 패치 파일을 메일로 보내는 것으로 명시하고 있다. 패치 파일을 만드는 방법은 아래와 같다.

$ git format-patch -1 $HASH

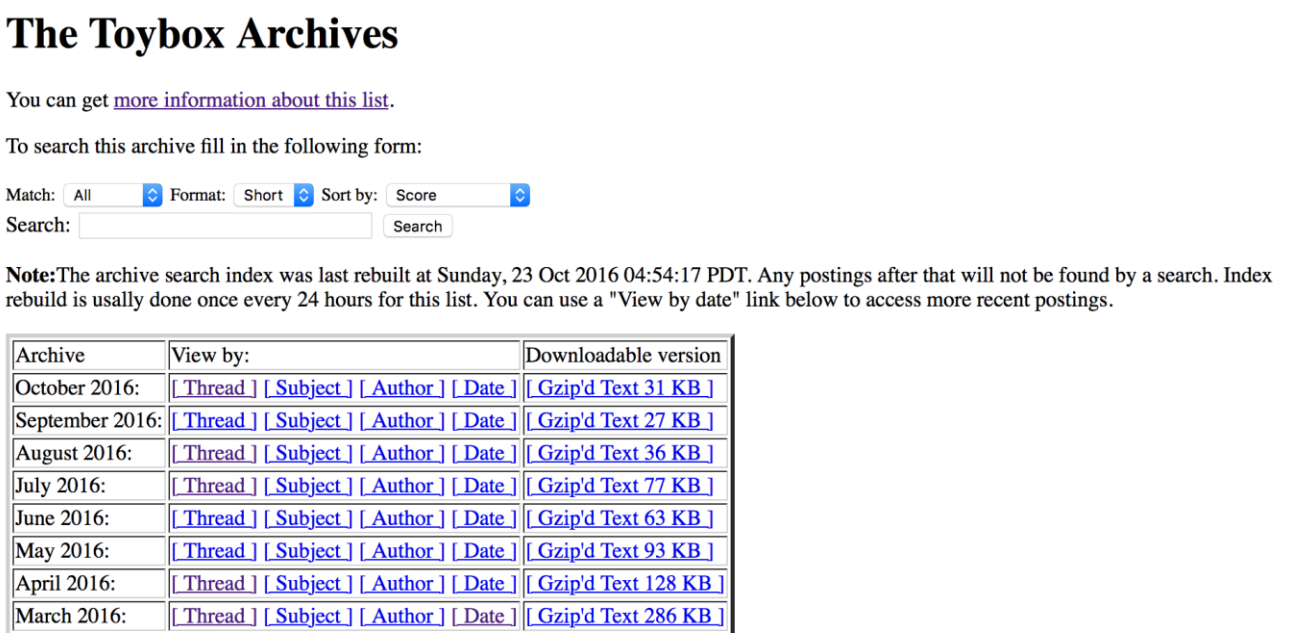
위 -1은 커밋 수를 지정하는 것인데 만약 2개 이상의 커밋을 보내야 할 경우라면 각각의 파일로 패치 파일을 만들어 보내는 것을 권장한다. 그리고 $HASH는 커밋 해쉬값을 넣어주면 된다.

9 외부 의존성

이 프로젝트는 다른 프로젝트의 의존성을 완전히 없애고자 노력을 하고 있지만, 일반적인 기능지원을 위해 외부 라이브러리를 쓰는 경우가 발생하고 있다. md5sum 명령이나 wget의 HTTPS 기능을

지원하기 위해 암호화 기능과 관련해서 openssl을 가져다 쓰고 있는데, 이것을 직접 프로젝트 내에서 구현하여 사용하려면 많은 시간과 노력이 들기 때문에 직접 구현하지 않고 사용하고 있다.

10 커뮤니티 토론 방식



The Toybox Archives

You can get more information about this list.

To search this archive fill in the following form:

Match: All Format: Short Sort by: Score
Search: Search

Note: The archive search index was last rebuilt at Sunday, 23 Oct 2016 04:54:17 PDT. Any postings after that will not be found by a search. Index rebuild is usally done once every 24 hours for this list. You can use a "View by date" link below to access more recent postings.

Archive	View by:	Downloadable version
October 2016:	[Thread] [Subject] [Author] [Date]	[Gzip'd Text 31 KB]
September 2016:	[Thread] [Subject] [Author] [Date]	[Gzip'd Text 27 KB]
August 2016:	[Thread] [Subject] [Author] [Date]	[Gzip'd Text 36 KB]
July 2016:	[Thread] [Subject] [Author] [Date]	[Gzip'd Text 77 KB]
June 2016:	[Thread] [Subject] [Author] [Date]	[Gzip'd Text 63 KB]
May 2016:	[Thread] [Subject] [Author] [Date]	[Gzip'd Text 93 KB]
April 2016:	[Thread] [Subject] [Author] [Date]	[Gzip'd Text 128 KB]
March 2016:	[Thread] [Subject] [Author] [Date]	[Gzip'd Text 286 KB]

[그림 6] 메일 저장소

아주 구시대적이기는 하나 메일링 리스트와 IRC를 통해 의견 교환과 커뮤니티 소통을 하고 있다. 사실 Github가 이런 기능들을 아주 손쉽게 제공해 주고 있으나 메인테이너가 잘 활용하지 않아 사용되지 않는다. [그림 6] Github 이슈에 글을 작성할 경우 메인테이터가 메일링 리스트로 공유하고 답변도 메일링 리스트와 Github를 통해 응답한다.