The X Window User HOWTO
X 윈도 사용자 가이드

레이 브리글렙(Ray Brigleb) ray@croftj.net v1.0, 29 November 1998

심상현 artsilly@eeserver.korea.ac.kr 번역중, 1998년 12월 29일부터

이 문서는 X 윈도 시스템 환경 설정 및 여러 정보들을 담고 있다. 이 내용은 일반적인 리눅스 유저에게 뿐 아니라 다양하고 많은 옵션과 X 윈도 시스템의 자세한 구조에까지 관심이 있는 시스템 관리자에게도 유용하다. 이 글을 읽을 때는 X 윈도 시스템과 어느 정도의 프로그램이 깔려있음을 가정한다.

목 차

제 1 절 소개

*X 윈도 시스템*은 혁신적이고, 네트워크 투명성을 보장하며, 그래픽컬한 환경을 가진 윈도 시스템이다. 1984년 최초로 *MIT*(매사추세츠 공과 대학)에서 개발되었다. 이 문서에서는 독자의 컴퓨터에 X 시스템이 깔려 있고 사용할 수 있는 상태라고 가정할 것이다. 이 문서에서 다루는 범위는 어떻게 하면 X와 그 어플리케이션을 사용할 수 있을까 하는 것에 한정한다. X 시스템을 어떻게 컴파일하고 인스톨할 것인가에 대해서는 다루지 않도록 하겠다. 일반적으로 요즘의 리눅스 배포본에서는 설치과정중에 X 윈도 시스템을 컴파일/설치 하므로 사용하는데에 있어서는 큰 무리가 없을 것이다.

1.1 좀 더 정보를 얻으려면?

만일 당신이 지금 막 시작한 사람이라면 *XFree86 하우투* 문서와 *XFree86 비디오 타이밍 하우투(XFree86 Video Timing)* 문서를 이용하면 많은 도움이 될 것이다. 더 많은 정보를 원한다면 이 문서의 맨 뒤에 있는 '참고 및 자료들'을 이용하면 될 것이다. 그리고 **매뉴얼(man page)**를 읽어 봐야 한다는 것을 잊지 말기 바란다.:)

1.2 이 문서의 최신판을 읽고 싶다면?

이 문서의 최시 버전은 주기적으로 `comp.os.linux.help`에 포스팅 된다. 또한 다양한 WWW 사이트나 ftp를 이용하여 받을 수 있을 것이다. *리눅스 도큐먼트 프로젝트* `<http://sunsite.unc.edu/LDP>`를 확인하는 것도 잊지 말라. 또한 나의 홈페이지 `<http://www.croftj.net/~ray/howto>`에서도 쉽게 찾을 수 있을 것이다.

1.3 제안 및 수정

만일 이 문서에 대한 의문점이나, 하고 싶은 말이 있다면 내게 메일을 보내는 것에 대해 편하게 생각하고 즉시로 보내주기 바란다. `ray@croftj.net` 특히 이 문서의 잘못된 점을 발견했을 때에는 다음 버전의 문서에 적용할 수 있도록 꼭 알려주기 바란다. 또한 다음 버전 문서에 포함되기 원하는 것이 있거나, 이 문서의 배포를 원한다면 꼭 편지 주기 바란다.

1.4 감사의 글

리눅스 하우투 문서의 관리자인 팀 바이넘(Tim Bynum)의 도움과 XFree86을 제작하고 여러 문서들을 작성해준 XFree86그룹에 무척이나 감사드린다. 또한 이전부터 여러 문서와 참고 자료들을 써준 사람들에게도 감사의 말을 해야겠다. 이 문서를 작성하는데 이전 사람들의 자료를 꾸준히 모아온 것이 많은 도움이되었다.

윈도 매니저 가이드 홈페이지를 운여하고 있는 맷 채프먼(Matt Chapman), FVWM2 테마의 팁과 트릭을 쓴 스코트 스크리븐(Scott Scriven)으로부터 많은 영감과 힌트를 가져왔음을 밝혀야 겠다. 이 문서를 작성하고 진행하는데 인터넷 서비스로 엄청난 도움을 준 조 크로프트(Joe Croft)와 정신적인 도움과 다양한 팁을 안겨준 클라리어 갈페르(Claire Galper)의 이름도 밝혀둔다. 마지막으로 모든 GNU/Linux 공동체에게 많은 부분에서 감사를 드린다.

1.5 저작권

제 2 절 시작하기

2.1 X 윈도 시스템:역사와 아키텍춰(구조)

*X 윈도 시스템*은 DEC의 아테나(Athena)프로젝트의 일환으로 *MIT*의 컴퓨터학과의 연구실에서 1984년 최초로 그 모습을 보였다. 그 후 MIT에서는 1987년 가을 요즘 우리가 알고 있는 것과 같은 X11의 최초 버전을 내어놓았다. X 윈도의 많은 부분은 1970년대 후반에 제록스에서 나온 Parc과 Star같은 컴퓨터의 기반에서 시작되었다. 물론 그 컴퓨터들은 상용화되지 못했다. 하지만 시제품에서 데모용으로 돌아가는 스몰토크 80을 본 사람들은 그 모습에 매료되고 말았다. 단순히 WIMP 인터페이스(윈도, 아이콘, 메뉴, 포인터)를 보여준 것 뿐 아니라 컴퓨터 업계의 일대 혁명을 일으키고 말았다. 몇년 후 사용자들은 여러 종류의 윈도 시스템을 사용하게 되었고, 아마 옛날처럼 되돌아 가진 않을 것이다.

X 윈도는 X 콘소시엄이 주체가되어 개발하고, 배포한다. 하지만 자유로운 라이센스로 인하여 무료나, 저렴한 시스템도 사용가능하다. 리눅스에서 이런 종류의 X에는 XFree86이 있다. XFree86은 인텔x86계열의 유닉스계열 운영체계에서 동작하는 X 서버이다. XFree86은 X11R6가 발전하는데 많은 공헌을 한 X386으로부터 시작되었다. XFree86을 놓고 본다면 어떤 목적에 사용하든지간에 다른 X 서버를 구입할 필요없이 무료로 사용할 수 있는 리눅스의 X 시스템이라 할 수 있다.

X 윈도는 기초 X 툴킷(X Toolkit Intrinsics) 및 아데나 위젯을 비롯한 다양하고 많은 툴킷이나 라이브러리 위에서 작성되었다. 개중에 많은 프로그램들은 XView나 Mootif 툴킷을 쓴다. 또한 최근에 작성된 툴킷을 이용하거나, 윈도 시스템과 통신 시스템을 통합한 KDE나 GNOME같은 것도 있다. 따라서 많은 프로그램에서 컴파일이 안되거나, 혹은 이상하고 이해할 수 없는 에러가 나오는 것을 볼 수 있을 지도 모른다. 이럴 경우에는 X 윈도 시스템에서 필요로 하는 여러 라이브러리와 툴킷들이 올바르게 설치되었는지 확인해 보는 것이 좋을 것이다. 여러 라이브러리들은 거의 무료이므로 쉽게 구해 설치할 수 있을 것이다.

2.2 데스크탑 분석하기

좀더 편하게 X를 사용하기위해서 반드시 익숙해져야 할 기본과 개념에 대해 짧은 설명을 하도록 하겠다. 이런 것들은 도움이 필요할 때마다 매뉴얼(man page)이나 도움말에서 자꾸 반복해서 나올 것이다.

스크린은 사용자 전체의 "데스크탑(화면)"을 말한다. 이 두가지의 말은 자주 섞여서 나오곤 한다. 기술적으로 본다면 스크린이란 당신이 사용하는 X가 보여주는 가장 자주쓰는 디스플레이 화면이다. 자주 쓰는 이라는 말에는 X가 1개 이상의 화면을 가질 수 있다는 의미이다. 사실 하나의 X 서버를 통해 여러 컴퓨터를 사용할 수 있다. 하지만 여기서 더 깊이 들어가는 것은 이 문서의 범위를 넘어설 것이다. 다른 문서를 통해 자세히 알 수 있을 것이다.

루트윈도란 현재 사용중인 스크린의 바탕화면이다. 이름에서 풍기는 느낌처럼 다른 윈도와는 다른 기능과 모습을 가지고 있다. 하지만 평범한 윈도처럼 그림을 보여준다든가, 색깔을 입힌다든가 혹은 루트윈도 상에서 프로그램을 실행시킨다든지 하는 기능은 다를바 없다.

윈도매니저는 X 윈도 시스템과 사용자 사이를 이어주는 인터페이스이다. 윈도매니저가 없었다면 X는 사용하기에 무척 힘든 시스템이 되었을 것이며 지금처럼 경쟁력있게 발전하지 못하였을 것이다. 윈도매니저에서는 윈도의 둘레(border)나 메뉴, 아이콘, 가상화면, 버튼바, 툴바등을 제공하여 좀 더 편리하게 사용할 수 있게 한다. 또한 X의 윈도매니저는 사용자의 설정 사항을 다양하게 구비하여, 자신의 입맛에 맞는 윈도매니저를 쓸 수 있도록 하고 있다.

포인터는 화면상에 마우스나 포인팅 장치의 위치를 알려주는 화살표 - 종종 다른 모양을 띄고 있을 수도 있다 - 표시이다. 또한 포인터는 모양 변화를 통해 마우스 포인터가 있는 위치에서 무슨 일이 일어날지, 혹은 무슨 일이 일어나고 있는 지에 대한 정보를 알려주기도 한다.

윈도란 윈도매니저에 의해 관리되며 주어진 어플리케이션이 실행되는 하나의 프레임으로 둘러싸인 공간이다. 위에서 언급한 루트윈도와는 좀 다른 모습일 것이다. 윈도의 모습은 타이틀, 둘레, 버튼들등을 가지고 있기도 하며 혹 이런 것들이 하나도 없을 수도 있다. 윈도는 윈도 매니저에 의해 관리된다.

항상 그런 것은 아니지만 전통적으로 마우스 포인터가 가리키고 있는 윈도는 사용자의 키보드 입력을 받을 수 있으며 흔히 활성 윈도 - 액티브 윈도라고 부른다. 이런 이유로 활성 윈도를 '포커스 되었다(focused)'고도 한다. 한 윈도가 포커스 되면 다른 윈도는 '포커스 되지 않은(unfocused)' 비활성 윈도가 되어 사용자의 입력을 받아들일 수 없다. (요즘은 포커스 하기 위해 마우스 버튼을 누르기도 한다)

메뉴, 아이콘은 다른 윈도 시스템에서와 같은 전형적인 동작을 보여준다. 문자 위주로 이루어진 콘솔 같이 보이는 텍스트 기반의 윈도를 **터미널 에뮬레이터**라고 한다. hanterm, xterm등이 그것이다. 터미널 에뮬레이터는 기본적으로 콘솔의 디스플레이를 에뮬레이팅 한다. X를 쓰는 이점중 하나로 동시에 여러개의 터미널 에뮬레이터를 띄워놓고 여러개를 동시에 감시(?)할 수 있다는 것이다. 또한 터미널 에뮬레이터를 사용함으로 X에서 제공하는 여러 다른 이점을 이용할 수 있을 것이다. 이런 것들에 대해서는 추후 이야기 하도록 하겠다.

2.3 X 윈도 시스템 불러내기

X 윈도 시스템을 시작하는데에는 여러가지 방법이 있다. 만일 컴퓨터를 켜자마자 X가 뜬다면 이부분을 건너뛰고 넘어가도 좋다. 하지만 대부분의 리눅스 시스템에서는 택스트 기반의 로긴 화면을 거쳐 로긴하고, 명령행으로 X를 실행하도록 세팅되어 있다. X를 실행하는 것은 전적으로 사용자 마음이다.

가장 간단한 방법으로는 **xinit**를 이용하는 것이다. xinit를 실행하면 아마 아무 것도 놓여지지 않은 데스크탑을 볼 수 있을 것이다. 아무런 설정없이 xinit를 수행하면 윈도 매니저조차 뜨지 않는다.(배포본을 사용할 경우 xinit로 X를 수행해도 윈도 매니저나, 터미널 에뮬레이터등이 뜰 수 있게 이미 세팅해 놓은 경우도 있다.) 이 경우 다음과 같이 콘솔에서 타이핑 함으로 X 시스템에 터미널 에뮬레이터가 뜨는 것을 볼 수 있을 것이다.(콘솔 가상 화면으로 가는 핫키는 alt-f1 f6이고, X가 떠있는 상태에서 X는 alt-f7/f8을 이용한다. X에서 콘솔로 가는 핫키를 누를때는 ctrl 키를 함께 눌러 ctrl-alt-f1 f6이 된다. 역주)

```
xterm -geometry +1 +1 -n login -display :0
```

매번 이런 명령행을 이용해 터미널 에뮬레이터를 띄우고 싶지 않다면 *.xinitrc*를 시작 쉘 스크립트로 쓸 수 있다. 하지만 너무나 불편한 방법이다. 배포본이라면 .xinitrc 파일에 어떤 내용이 있을 지도 모른다.

좀 더 보편적인 방법으로는 **startx**를 사용하는 것이다. 좀 더 많은 설정화일들을 부를 수 있으며, 좀 더 강력하다. 아마 startx 명령을 통해서는 좀더 화려한 화면을 볼 수 있을 지도 모른다.(많은 배포본에서 startx를 통

해 수행되는 X 시스템에서 윈도 매니저를 비롯한 여러 세팅을 기본적으로 제공하고 있다.) 이를 통해 좀더 쉽게 X를 사용할 수 있다. 여전히 많은 사용자들이 이용하지만 텍스트 기반의 쉘로 로긴한 후 명령행을 통해 X를 수행시킨다는 것은 단점이자 장점으로 남는다. 장점으로는 개개의 사용자가 자신이 원하는 방식으로 윈도를 세팅하여 띄울 수 있다는 것이다.

```
startx -- -bpp 8    #start x in 256 color mode
startx -- -bpp 32   #start x in true color mode
```

위 명령에서 '–' 의 표시는 명령행의 아규먼트(옵션)을 xinit로 넘겨주는 역할을 한다. 위 명령을 통해 X의 해상도를 자신이 원하는 목적에 맞게 바꾸어 실행시킬 수 있다. 물론 설정화일을 읽어들이는 것은 여전하다. 설정에 대해서는 나중에 다루도록 하겠다.

2.4 X 디스플레이 매니저(xdm) 사용하기

xdm은 텍스트 터미널의 **getty**와 **login** 명령이 하는 일과 비슷한 일을 한다. 즉 사용자가 사용할 수 있는 디스플레이를 열고 로긴을 통해 사용자를 인증한다. xdm을 이용하며 무엇을 해야 할지 직관적으로 느껴지는 친숙한 로긴화면에 사용자 이름과 패스워드를 넣는 것으로 모든 일이 끝난다! 적합한 사용자라면 바로 X 시스템의 그래피컬한 환경을 쓸 수 있다. 아마 인터넷 카페나, 회사, 학교 컴퓨터실등 유닉스에 친숙하지 않은 사용자를 위한 공간에서 자주 보았을 것이다. xdm은 다른 추가장비 없이 좀 더 쉽게 접근하도록 해준다.(외국에서는 X를 인터넷 카페나 학교에서 쓰나보죠? 좋겠다.^^)

xdm의 설정 파일은 */usr/X11R6/lib/X11/xdm*에 놓여지고 매우 편리하게 관리할 수있게 해 준다. *xdm-config* 파일을 확인해 보기 바란다. 또한 매뉴얼(man page)을 참고 하는 것을 잊지 말기 바란다.

제 3 절 윈도 매니저 고르기

이 섹션에서는 좀 더 편리한 윈도 매니저를 고르기 위해 여러 윈도 매니저를 비교하고 특징에 대해 이야기 하도록 하겠다. 자칫 하나하나 확인해 보다보면 엄청난 갯수의 윈도 매니저에 눌려 더 이상 이 문서를 진행해 나갈 수 없으므로(어차피 불가능 하다는 것을 알고있다.) 인기있는 몇가지를 선택하는 것으로 한정짓도록 하겠다. 비록 그리 많은 갯수의 윈도 매니저를 다루지는 않지만 이 것들을 통해 우리가 알하야할 개념과 목적을 어느정도는 충족시킬 수 있을 것이라 생각한다. 일단 기본적인 개념에 대해 이해를 한다면 더 많은 것들을 쉽게 접근할 수 있을 것이다.

윈도 매니저를 고르는 것은 얼마나 컴퓨팅 환경을 편리하게 할 수 있는 지를 결정하게 되는 중대한 결정이 될 수도 있다. 아무도 X 시스템에서 실제 자신의 일을 하기보다 윈도를 정리하고, 조절하고, 다루는데 많은 시간을 보내고 싶진 않을 것이다.

어떤 윈도 매니저의 경우에는 사용자의 취향에 맞출 수 있도록 준비 되어 있다. 잘 바뀌어진 윈도 매니저의 경우 상당히 멋지고 때론 새로운 운영체계를 쓰는 듯한 느낌을 주기도 한다. 아이콘이 필요 없다면 제거 할 수도 있고, 툴바가 필요하다면 새롭게 제작할 수 도 있다. 게다가 이런 일은 그리 복잡하지도 않고, 돈이드는 그런 일도 아니다.

3.1 FVWM과 역사

가장 일반적인 윈도 매니저는 로버트 내이션(Robert Nation)이 제작한 **FVWM** 윈도 매니저일 것이다. 여기서 FVWM이라 하면 FVWM, FVWM2, FVWM95를 비롯한 여러 클론을 포함한다. FVWM은 TWM으로부터 파생되었다. TWM은 X 윈도 시스템과 함께 딸려오는 기본 윈도 매니저 프로그램이다(이 문서에서 다루지는 않겠다). TWM의 설정들 및 내용들이 FVWM과 비슷하기 때문에 곳 쉽게 쓸 수 있을 것이다. TWM은 실제로 더 많은 메모리를 사용하고 별 모양 없으며 사용하기 쉽지 않기 때문에 별 인기가 없다.

FVWM은 최초의 버전이며 약간 구식이 되어버렸지만 상당히 인기를 끌던 리눅스 윈도 매니저이다. 지금은 2.0버전까지 나왔고 상당히 안전적이며 훨신 일반화된 사용법을 제시한다. 이 버전에서는 좀 더 직관적이고 간

단하며 유연한 설정방식을 지원한다. 그러나 많은 사람들은 여전히 최초버전의 FVWM을 선호하는 것 같다. 오랜 시간동안 인기를 끌었으며 표준화 되었고, 많은 설정 파일들을 직접 사용할 수 있다는 점이 아직도 이 구식을 돋보이게 하는 것 같다. 당연한 것일런지도 모른다.

신기한 것은 어느 누구도 FVWM에서 'F'가 무엇을 의미하는 것인가에 대해 잘 모른다는 것이다. (아마 제작자는 알고 있을런지도 모른다.) 하지만 VWM 만 놓고 본다면 분명히 가상 윈도 매니저(Virtual Window Manager)일 것이다. 실제로 FVWM 윈도 매니저 시리즈는 가상 데스크탑을 지원하고 이것이 장점이 되고 있다. 아마 마우스나 적절히 설정한 핫키를 이용해서 가상 데스크탑을 오갈 수 있을 것이다.

FVWM2는 많은 발전과 새로운 특징을 갖춘 새로운 버전의 표준이다. 최근의 윈도 매니저의 경향과는 달리 단지 8비트/256색의 디스플레이에서도 깔끔한 화면을 보이도록 설계되었다. 사실 이점으로 인해 난 아직도 FVWM2를 사용하고 있다. 최근에는 여러 아이콘이 추가되고 그레디언트한 화면을 보여주는 기능을 추가해 사용자가 원하든 원하지 않던 간에 윈98의 인터페이스와 비슷해 져 가는 것 같기도 하다. 이 윈도 매니저는 또한 일반적인 관점에서 볼때 무척 확장성이 있다. 테마를 쓸 수 있게 해 주기도 하고 동적 메뉴를 지원하기도 한다. 기술적인 면에서 볼때 FVWM2는 아직도 beta 버전이다. 하지만 잘 동작하며 아직까지 하나의 문제점도 발견하지 못했다.

Resources:

공식적인 FVWM과 FVWM2의 사이트는 `<http://www.hpc.uh.edu/fvwm/>`이다.

또한 가장 최신 버전의 프로그램은 `<ftp://ftp.hpc.uh.edu/pub/fvwm/>`에서 구할 수 있다.

3.2 다른 종류의 윈도 시스템 에뮬레이션

다른 OS의 다양한 윈도 환경에 익숙해져 있는 사람을 위해 이런 윈도 환경을 에뮬레이션 하는 여러 윈도 매니저들이 있다.

icewm `<http://berta.fri.uni-lj.si/~markom/icewm>` 윈도 매니저는 좀더 일관된 겉모습과 느낌을 제공해 준다. icewm을 이용하여 여러 다른 윈도 시스템의 핫키라든지, 모습이라든지등을 에뮬레이션 할 수 있다. OS/2 에뮬레이션은 그럭저럭 쓸만하다.

AmiWM `<http://www.lysator.liu.se/marcus/amiwm.html>`는 아미가 워크밴치 타입의 윈도 매니저고 *mlvwm* `<http://www.bioele.nuee.nagoya-u.ac.jp/member/tak/mlvwm.html>`은 맥OS를 에뮬레이션한다. *dfm* `<http://www-c.informatik.uni-hannover.de/~kaiser/dfm/dfm.html>`은 OS/2 워크플레이스 쉘을 데스크탑부터 파일매니저까지 꽤나 정교하게 표현해 냈다.

Resources:

윈도 매니저를 모아놓은 웹사이트 `<http://www.PLiG.org/xwinman/>`를 통해 여러 다양한 윈도매니저들을 구할 수 있을 것이다. 또한 위에 언급한 다른 윈도 시스템을 에뮬레이션 하는 윈도매니저도 `<http://www.PLiG.org/xwinman/others.html>`에서 구할 수 있다. 그 방대하고 거의 완벽하다 싶은 리스트로 보아 위 홈페이지의 제작자가 무척 수고한듯 싶다. 리눅스 용 윈도 매니저만을 보고 고르고 싶다면 *sunsite* `<ftp://sunsite.unc.edu/pub/Linux/X11/window-managers>`를 통해 구하는 것도 좋은 방법이다.

3.3 X의 그래픽 인터페이스

오픈룩(Open Look)과 모티프(Motif)는 초기부터 X 윈도의 표준화를 주도해 왔고 그 결과로 다양한 종류의 워크스테이션에서 중심적인 환경으로 사용되어왔다. 둘 다 어느 면에서는(겉모습과 느낌에서) 표준화를 이루었고 따라서 현대의 리눅스 시스템에서도 사용되고 있다. 또한 각각의 환경에서 거의 비슷한 겉모습과 느낌을 가질 수 있도록 컴파일되어 제공되는 라이브러리를 필요로 한다.

X11R5와 함께 사용가능한 선 마이크로 시스템의 오픈 윈도 그래픽 인터페이스를 보자. 여기에는 두가지의 오픈룩 윈도 매니저가 있다. olwm과 olvwm이다. olvwm은 이름에서 느껴지는 것처럼 가상 데스크탑을 지원한다.

이것은 선의 윈도 시스템의 표준으로 계획하여 개발한 환경이고 솔라리스 시스템에서 적어도 몇년간은 표준이었다. 만일 당신의 배포본에 olwm이나 olvwm이 없고 새로 인스톨 하려한다면 XView 라이브러리가 있어야 한다는 것을 잊지 말자. 또한 PATH에 인스톨된 디렉토리(/usr/openlook)를 넣도록 해야한다.

만일 OSF/MOTIF의 툴킷을 리눅스에서 사용하고자 한다면 돈을 내고 구입해야한다. 이 툴킷은 무료가 아니다. 그러나 모티프 툴킷을 가지고 있어야 컴파일이 되는 모티프를 사용하는 프로그램들을 컴파일 할 수 있게 해주는 '가난한 프로그래머'가 제작한 레스티프(Lesstif)가 있다. 즉 레스티프는 모티프의 클론이라고 할 수 있다. 레스티프는 많은 부분의 API에서 부분적으로나마 모티프의 자리를 대신해 나가기 시작했다. 많은 프로그램들 그리고 윈도 매니저까지도 레스티프를 가지고 컴파일 할 수 있다. 사실 많은 부분의 FVWM의 코드는 모티프로부터 온 것이다. 결국 가장 중요한 레스티프의 장점은 모티프가 있어야 컴파일이 되는 프로그램들을 모티프 없이 컴파일 할 수 있다는 것이다. 윈도 매니저의 경우에도 특별한 것이 없다. 원래의 모티프로부터 레스티프로 옮겨가도 아무 것도 달라지지 않는다. 모든 설정파일을 그대로 쓸 수 있다. 그러나 사실 사용 목적이나 의도 면에서 본다면야 FVWM이 더 쓸만하고 더 많은 기능들을 제공해 준다. 그러나 FVWM 이 컴파일 될때 모티프 툴킷이 필요하다는 사실을 모르더라도, 생긴 모양이나 느낌에서 어느정도 비슷하다는 것을 느낄 수 있을 것이다.

Resources:

가난한 프로그래머를 위한 레스티프 `<http://www.lesstif.org/>`

3.4 X 데스크탑 환경

제 2 세대 윈도 매니저의 개념은 KDE로부터 시작되었다. 그리고 곧 그놈(GNOME)으로 연결되게 될 것이다. 이 두가지는 현저한 공통점과 엄청난 차이를 가지고 있으며 이 절에서는 그것들에 대해 이야기 하도록 하겠다. 하지만 이 두가지는 아직도 완저하지 않다는 점을 기억해야 한다. 오랜 시간 동안의 개발해 왔지만 여전히 그리 안정적이지는 않다. 만일 중요한 작업을 하고 있다면 KDE나 그놈을 쓰는 것에 대해서는 한번 더 생각해 봐야 할 것이다.

KDE 데스크탑 환경 - KDE 홈페이지로부터 인용한 것

"KDE는 여러 유닉스 워크스테이션의 거대한 프로그램들을 하나로 통합한 완전히 새로운 데스크탑이다. KDE는 윈도매니저와 파일매니저, 패널, 제어판등 요즘 시대에 유행하는 데스크탑에서 찾을 수있는 모든 것을 제공해 준다. 이 특별한 데스크탑 환경의 가장 큰 장점이라면 여러 컴퍼넌트들을 통합적으로 사용가능하게 놓여졌다는 것이다."

KDE는 단순한 윈도 매니저가 아닌, 데스크탑 환경을 구현하기 위한 하나의 시도이다. KDE의 여러 기능들은 꽤나 유기적으로 잘 작동한다. 자칫 잘못 생각하게 되어 KDE를 아예 완전한 하나의 운영체계 보는 오류를 범하기도 한다. 윈도 시스템을 위한 여러 기능들이 포함되어있고, 여러 프로그램들이 KDE 환경으로 포팅되어지고 있다. KDE는 개발 수준에서 이미 놀라울 정도까지 이르렀다. 그러나 KDE를 개발하는데 기초로 쓰여진 QT 툴킷의 라이센스 규정은 많은 사람들이 KDE 설치를 주저하는 원인이 되고 있다. 이 라이센스는 최근에 들어 어느 정도 변하기는 했지만 오픈 소스라 제한된 이 라이센스는 GNU 소프트웨어의 규정과는 같지 않다.

그놈(GNOME) - GNU 객체 모델 환경 - 그놈 홈페이지로부터 인용한 것

"그놈(GNOME)이란 GNU 네트워크 객체 모델 환경(GNU Network Object Model Envirionment)를 의미한다. 완전히 무료이며 사용자에게 좀 더 친숙한 완벽한 데스크탑을 구현하겠다는 목적을 지니고 있다. 그놈은 GNU 프로젝트의 일환이며 오픈소스(OpenSource(tm)) 운동의 일부이다. 그놈의 데스크탑은 몇몇 유틸리티와 일관되게 유지되는 겉모습과 느낌을 지닐 수 있도록 계획된 큰 어플리케이션들을 포함한다. 그놈은 그래픽컬한 사용자 인터페이스를 구현하기 위해 데스크탑과 그에 동반되는 어플리케이션에 GTK+를 사용한다."

사실 그놈의 시도는 KDE와 많은 면에서 같다고 할 수있다. 그러나 몇몇 부분에 있어서는 다르다. 우선 그놈은 하나의 윈도 매니저에 구속되지 않는다. 또한 그놈은 어플리케이션 사이의 상호작용뿐 아니라 코바(CORBA, Common Object Resource Broker Architecture)를 이용하여 컴퓨터와 플랫폼 사이에서도 상호운영된다. 또한 그놈은 무료이고 오픈소스인 GTK+ 툴킷에 기초하고 있다. 이는 KDE의 기초가 되는 툴킷과는 정 다른 개념이자 리눅스 자체의 철학과도 일치하는 바이다.

Resources:

공식 KDE 홈페이지 <http://www.kde.org/>와 공식 그놈 홈페이지 <http://www.gnome.org/>를 통해 많은 정보를 얻을 수 있다. 그놈을 둘러싼 여러 이슈들은 <http://www.gnome.org/gnomefaq/FAQ.txt>에서 얻을 수 있다. 또한 코바와 GTK+ 홈페이지는 각각 <http://www.corba.org/>와 <http://www.gtk.org/> 이다.

3.5 아주 괜찮은 윈도 매니저

최근 윈도 매니저의 추세는 아주아주 커여워져 가고 있다. 사람들이 생각해 볼 수 있는 여러 편리한 기능들을 자랑하고, 지금까지 사용해 봤던 아주 고급스러운 워크스테이션등에 쓰이던 윈도 매니저등을 에뮬레이션 하기도 한다. 물론 이런 윈도 매니저들은 좀 더 빠른 CPU와 좀 더 많은 RAM을 요구한다.

윈도 메이커(Window Maker) - 윈도 메이커 홈페이지로부터 인용한 것

"윈도 메이커는 GNUstep 어플리케이션의 추가적인 통합 지원을 위하여 디자인 되었다. 이것은 NEXTSTEP(tm)의 품위있는 겉모습과 느낌을 그대로 에뮬레이션 하고 있다. 꽤나 빠르고 풍부한 특성들이 있으며 설정하고 사용하기에 아주 쉽다."

이 윈도 매니저, 윈도 메이커의 가장 큰 장점으로는 GNU 데스크탑을 지원한다는 것이다. 즉 이것은 GNOME 프로젝트에 아주 적합하고, 보기에 좋은 프런트 엔드가 된다. 또한 위에 말한 바처럼 윈도 메이커는 설정하기에 아주 쉬운 윈도 매니저중 하나이다. 그래피컬한 인터페이스로부터 설정이 가능하고 OffiX의 드래그 앤드 드롭을 사용할 수 도 있다. 데스크탑 테마로부터 환경을 쉽게 변화 시킬 수도 있다. 레드햇에서는 기본적으로 설치되어 제공되고 있다. 게다가 기분이 내킬 때는 아주 쉽고 다른 수고 없이 FVWM으로 변환도 가능하다.

애프터스텝(Afterstep) - 애프터스텝 홈페이지로부터 인용한 것

"NEXTSTEP의 겉모습과 느낌을 에뮬레이팅 하는데서 시작했다. 그러나 다양한 사용자의 요구사항을 수용하기 시작하면서 약간은 다른 모습을 가지게되었다. 많은 전문가들이 이야기하길 넥스트 스텝은 보기에만 아주 좋은 인터페이스일뿐 아니라 아주 기능적이고 직관적인 점을 많이 가지고 있다 한다. 애프터스텝은 이런 NEXTSTEP 인터페이스의 장점을 통합하고 좀더 유용한 기능들을 추가하는데 그 초점을 맞추고 있다. 또한 애프터스텝의 개발자들은 안정성과 호환성에도 신경을 쓰고 있다."

엔라이튼먼트(Enlightenment)

엔라이튼먼트는 일반 윈도 매니저에 비하면 그 이상이라 할 수 있다. 엔라이튼먼트는 아주 극단적이고, 세밀하며 여러 설정이 가능한 환경을 제공해 준다. 또한 윈도 모양에 대한 평범하지 않은 개인적인 설정까지도 완벽하게 지원해 준다. 또한 디자인면에 있어서 열려 있어 무작정 어떤 규정을 따르기 보다는 사용자가 마련한 고유의 규정에 따라 기능적인 면에서 겉모습까지 아주 세밀하고 작은 부분까지 설정을 해 줄 수 있다.

엔라이튼먼트는 여전히 개발중에 있다. 따라서 쉽게 시스템을 다운시키거나 불안정하게 만들 수 있다. 그러나 엔라이튼먼트의 저자인 래스터맨(Rasterman)이 현재 그놈 프로젝트와 함께 일하고 있으므로, 그놈 프로젝트가 완성되어 가며 엔라이튼먼트에 대한 지원 또한 완성될 것으로 보인다. 이 윈도매니저에 관심을 가져주기 바란다.

Resources:

공식 윈도 매이커 홈페이지 <http://www.windowmaker.org> 공식 애프터스텝 홈페이지 <http://www.afterstep.org> 이젠 좀 익숙해 졌을 지도 모르겠다. 엔라이튼먼트는 당연히 <http://www.enlightenment.org>에서 찾을 수 있다.

제 4 절 X에서 작업하기

이 섹션에서는 X에서 운용되는 클라이언트를 사용하는데 좀 더 가깝게 접근하게 될 것이다. 또한 X 시스템의 기본적인 운용 절차에 대해 이야기 한다. X는 윈도 매니저 없이는 그 자체로 어떤 직관적인 인터페이스도 없

다. 그러나 명령행으로부터 프로그램을 시작하는 것은 매우 쉽다. 이런 방법으로 동시에 여러개의 X 터미널을 띄우는 것은 가장 일반적인 방법이다. 즉 그래피컬한 수단을 최대한 이용한다기 보다는 유용한 시스템적 측면을 사용하는 것이다. 실재로 명령행의 유연성이나 유용함은 무척 쓸만하다. 윈도 매니저 자체가 거의 필요없을 정도이다. 사실 윈도매니저는 매우 유용하고, 얼마 지나지 않아 윈도 매니저를 원하게 되겠지만 마우스가 없이 키보드만으로도 모든 작업을 빠르고 쉽게 수행할 수 있다는 점은 명령행 방식의 우수한 점이다.

아직은 윈도매니저와 클라이언트의 구분이 명확하지 않겠지만 그 차이점은 당신에게 새로운 국면을 맞이하게 할 것이다.한 가지 유용한 키바인딩으로 Ctrl-Alt-Backspace 가 있다. 이것은 X를 쓰던 중 곤란한 상황에 처했을 때 X에서 바로 빠져나오게 해 줄 것이다.

4.1 명령행 옵션

대부분의 X 프로그램은 항상 같은 기본적인 명령행 옵션을 제공하려 한다. 'MIT X 기본 툴킷(MIT X Toolkit Intrinsics)'을 사용하여 제작된 모든 어플리케이션은 자동으로 아래와 같은 옵션을 사용할 수 있다.

-display 디스플레이

이 옵션은 어떤 X 서버를 쓸 것인가를 선택하게 해 준다. 다음 절에 나오는 '디스플레이 이름'이라는 절을 참고하기 바란다.

-geometry 위치

초기의 윈도의 위치와 사이즈를 세팅한다. 여기에는 *윈도의 폭x윈도의 높이+수평위치+수직위치* 또는 *+수평위치-수직위치*와 같은 형식을 이용한다. 만일 음수 값으로 수평, 수직위치를 넣는다면 상대적으로 거꾸로 움직여진 위치에 놓여지게 될 것이다.

-font 폰트

생성될 윈도의 텍스트를 위해 쓰여질 폰트

-bg 바탕색

윈도의 바탕색

-fg 전경색

윈도의 전경색 또는 폰트의 색

-name 리소스 이름

어플리케이션에서 사용가능 한 리소스의 이름을 설정해 주는 것은 상당히 유용하다. 예를 들어 같은 어플리케이션을 두개 실행시키고 그 두개를 꼭 구분하기 원하는 경우 각기 다른 리소스 데이터 베이스에 기초한 다른 리소스를 가짐으로 좀더 쉽게 구분할 수 있게 해 준다.

-title 타이틀 문자열

일반적으로 윈도 매니저를 통해 각 윈도의 상단에 표시하게끔 해주는 윈도의 이름이다. `-name` 옵션과 구분하길 바란다.

-iconic

어플리케이션을 실행할 때 아이콘화 되어 실행시킨다.

-xrm 리소스 문자열

이 옵션은 이미 정해진 리소스 이름에 따르는 값을 새롭게 설정해줄 수 있게 해 준다. 또한 매번 특정 어플리케이션을 실행할 때 마다 지금 설명하고 있는 명령행 옵션을 꼭 세팅할 필요업이 X 시스템을 사용할 수 있도록 해 준다. 즉 다음과 같이 `xterm -xrm "xterm*background:  blue"` 리소스 값을 설정해서 xterm을 수행하는 것은 내부적으로 `xterm -bg blue`와 똑같다. 그러나 리소스 값은 디폴트로 정해 줄 수 있으며 디폴트로 정해진 값은 특정 옵션이 없을 경우 기본 옵션이 된다. 이것에 대해서는 추후 설명하도록 하겠다.

4.2 디스플레이 이름

모든 X 서버는 다음과 같은 형태의 디스플레이 이름을 갖는다.

디스플레이 이름:디스플레이 번호.화면 번호

호스트 이름은 물리적으로 디스플레이에 연결된 컴퓨터의 이름이다. 따라서 서버에서 사용할 때 디스플레이 이름을 생략할 경우에는 자동으로 현재 사용중인 그 서버의 이름이 들어가게 된다. 따라서 단 한대의 컴퓨터에 연결되어 있고 단 하나의 X를 사용한다면 특별히 디스플레이 이름을 지정할 필요는 없다. 이 경우 해당 디스플레이 번호는 0 번이 된다. 디스플레이 번호는 하나 이상의 키보드와 모니터를 가진, X 터미널과 같은 기기에서 쓰인다. 화면 번호는 다중 모니터 세팅을 한 경우(한대의 모니터에서 여러개의 화면을 쓰는 경우를 말함. 화면은 동시에 보이는 것이 아니라 스위칭 됨) 특정 화면을 지정하기 위해 쓰인다. 만일 지금 이 글을 읽고 있는 로컬 머신을 지정하고 싶다면 `-display :0.0` 처럼 해 주면 된다. 호스트 이름이 생략되었으니 지금 당신의 앞에 보이는 컴퓨터의 이름으로 알아들을 것이다.

리눅스에서는 환경변수 DISPLAY 값을 이용해 디스플레이 이름을 지정할 수 있게 해 준다. 지금 내 시스템에서 DISPLAY는 :0.0 으로 설정되어 있으며 이것이 대부분의 시스템에서도 찾아볼 수 있는 값일 것이다. 다른 목적으로 이 기본 디스플레이를 바꾸고 싶다면 터미널에서 환경변수를 바꿔주거나(이것은 각 쉘에 따라 또한 다르다. 여기에서는 설명하지 않겠다.) X가 시작할때의 스크립트를 이용해 다르게 지정할 수 있다. 물론 매번 `-display` 옵션을 사용할 수도 있다.

혹 다음과 같은 경우를 만나게 될 지도 모른다. X에 들어 가있고 xterm이나 rxvt를 사용하는 도중에 관리자의 일을 수행하기 위해 su 명령을 이용해 루트로 들어갔을 경우 X 어플리케이션을 수행할 X의 DISPLAY 가 설정되어 있지 않아 어플리케이션을 띄우지 못할 지도 모른다.(일반적인 su 명령으로 root가 되었을 경우에는 root의 환경변수를 읽지 못한다.) 이 경우 -display 옵션을 이용해 지금 바로 사용중인 디스플레이에 root의 어플리케이션을 띄울 수 있다.(보안 문제로 현재 디스플레이의 원래 사용자가 아닌 다른 사용자가 이 디스플레이를 사용하는 것을 막아 놓았을 수도 있다.)

제 5 절 X의 시작

앞으로의 과정에서 우리는 충분히 안정이 입증된 FVWM2를 윈도 매니저로 쓴다고 가정한 후에 여러 설정에 대한 예를 볼 것이다. 꼭 특정 윈도 매니저를 지지 하는 것은 아니고 단지 좀더 쉽게 설명하기 위한 것이다. 하지만 이 절의 내용을 꼼꼼히 살펴본다면 다른 윈도 매니저를 이용하는 것도 어렵지 않을 것이ㅏㄷ.

5.1 간단한 시작과정 설정

가장 쉽고도, 가장 기초적인 우리가 지금 해야할 일을 X가 시작될 때 사용하는 초기화 파일을 설정하는 것이다. 사실 이 파일은 사용자 모두가 사용하는 파일이며 `/var/X11R6/lib/xinit/xinitrc` 등의 디렉토리에서 찾을 수 있다. 개개의 사용자마다 다르게 설정하고 싶다면 각 사용자의 홈 디렉토리에 `.xinitrc`라는 파일을 만들어 주면 위의 파일 대신에 이 파일을 사용하게 된다. 일반적으로 사용자 모두가 사용하는 파일은 기본적이며 아주 기초적인 내용들을 담고 있으며 보안을 이유로 모두가 이 파일을 사용하는 것은 나쁘지 않다. 그러나 자신의 설정을 하고 싶다면 사용해도 좋다.

우선은 홈 디렉토리에 `.xinitrc`를 만들고, 자신의 손에 익은 텍스트 에디터를 이용하여 다음과 같이 만들어 보길 바란다. 혹 이미 파일이 있다면 적절히 수정해도 좋을 것이다.

```
#!/bin/sh

# if your backspace and delete are reversed, try this:
xmodmap -e "keysym BackSpace=Delete" -e "keysym Delete-BackSpace"
```

```
xsetroot -solid darkslateblue

# start some basic applications
xclock -geometry 96x96+2+2 -bg grey40 -fg black -hl white &
xload -geometry 120x96+2+147 -bg grey40 -fg white -hl darkred -update 4 &
xterm -sb -ls -geom 80x25-2+2 -title "shell" &
xterm -sb -ls -geom 80x25-2-2 &

# start the window manager
fvwm2
```

기본적으로 위의 예제에서 많은 것을 배울 수 있다. 첫줄에서 눈치 챘을 지도 모르지만 우선 이 파일은 쉘 스크립트 이다. 두번째 줄의 xsetroot에서는 바탕화면을 푸른색으로 바꿔준다. 매일 푸른 화면과 함께 시작하게 되는 것이다. 셋째, 넷째줄에서는 시작과 동시에 실행시키고 싶은 프로그램을 적어준다. 색이나 위치(geometry)등 몇몇 옵션줄은 좀 더 편하게 사용할 수 있는 환경을 제공해 준다. 옵션 설정에 대한 개선된 사항은 추후 논의하게 될 것이다. 다섯째, 여섯째 줄은 시작과 동시에 곧 쓰게 될 것이라 생각되는 xterm 두개를 띄워준다. 물론 지금까지 이야기에서 #으로 표시된 주석라인은 특별히 세지 않았다.

마지막 줄은 특별히 중요하다. 이 마지막 줄이 바로 윈도 매니저를 띄우는 기능을 맡는다. 이 줄과 xsetroot, xmodemap등을 실행하는 명령은 &를 쓰지 않아 백그라운드로 실행이 되지 않는다는 것을 주의하기 바란다. 그러나 나머지 다른 프로그램들은 백그라운드로 실행이 된다. 마지막 줄에 포그라운드로 수행되는 프로그램이 종료되면 X 윈도 세션이 통째로 종료되어 버린다. 의심적다면 한번 해보기 바란다. 윈도 매니저가 종료되면 X에서 빠져나오게 될 것이다.

5.2 좀 더 개선된 시작

처음으로 제작한 스타트업 파일에 몇가지 유용한 것들을 포함시키도록 하겠다. 예를 들어 root로 X를 시작했을 경우 실수를 방지하기 위해 주의를 주고자 특정한 모습으로 보이게 할 수 있다. 레드햇 사용자라면 이런 상황을 경험했을지도 모르겠다. 레드햇에서 제공되는 많은 GUI 설정툴은 root로 로긴해야 하며 일반적으로 관리자는 아예 root로 X를 수행시키길 원한다. 혹은 일반 사용자 계정으로 로긴하여 X 프로그램을 사용할때 -display:0.0 옵션을 이용할 수도 있다.(위에 한번 언급한 것처럼 보안문제로 막아놓았을 지도 모른다. 이때는 X를 시작한 계정의 사용자로 'xhost +' 명령을 줌으로 누구나 이 디스플레이에 출력을 가능하게 해 주어야 한다.) 루트로 X를 시작할 때 아래와 같은 예를 통해 좀 다른 모습을 보이게 된다.

```
# change background color for root
if [ "$USER" = "root" ];
then
     xsetroot -solid darkred
else xsetroot -solid darkslateblue
fi
```

이 예제에서는 사용자가 root 인지 확인하여 root라면 붉은 바탕화면을 쓰게 설정되어 있다. 일반 사용자라면 위에서와 마찬가지로 푸른색일 것이다.(좀 다른 푸른색이다. 이 색은 내가 선호하는 색이다. :) 마지막 예제는 .xinitrc 파일에서 일반적인 시스템 설정을 불러 함께 쓸 수 있게 설정하는 예이다. 포괄적인 시스템 설정과 개인적인 시스템 설정을 합쳐주는 역할을 한다. 일반적인 시스템이라면 예제에 있는 디렉토리에서 그 특정파일을 찾을 수 있겠지만 우선은 파일이 존재하는 지 확인해 보는 것도 나쁜 생각은 아니다.

```
 userresources=$HOME/.Xresources
 usermodmap=$HOME/.Xmodmap
 sysresources=/usr/X11R6/lib/X11/xinit/.Xresources
```

```
sysmodmap=/usr/X11R6/lib/X11/xinit/.Xmodmap
# Merge in defaults and keymaps
if [ -f $sysresources ]; then
   xrdb -merge $sysresources; fi
if [ -f $sysmodmap ]; then
   xmodmap $sysmodmap; fi
if [ -f $userresources ]; then
   xrdb -merge $userresources; fi
if [ -f $usermodmap ]; then
   xmodmap $usermodmap; fi
```

5.3 윈도의 위치를 알아내기

-geometry 옵션을 이용해 스크린의 어떤 위치에나 특정 어플리케이션을 위치시킬 수 있다. 적절한 설정을 통해 윈도의 위치를 정한다면 사용을 더 편하게 할 수 있겠지만, 처음에 원하는 윈도의 위치를 알아내는 것은 많은 시행착오를 요구한다.(물론 대충 설정한다면 그냥 만족하고도 살아갈 수 있다.) 이런 이유에서 FVWM에서는 좀 더 편리한 기능을 제공해 준다. 기본적으로 FVWM에서는 윈도의 이동이나 크기 조정시 위치표시기가 나타나 윈도의 위치를 보여준다. 한번 해 보기 바란다. 또한 윈도의 크기를 조정해 보기 바란다. 하지만 이렇게 윈도의 위치를 알아내는 것은 이런 기능을 지원해 주는 윈도 매니저에서만 가능하다. 일반적인 방법으로 쓰기에는 약간 까다롭지만 좀 더 자세히 윈도의 상황을 보여주는 프로그램을 소개하고자 한다. 항상 그렇듯이 xterm에서 xwininfo라 타이핑함으로 프로그램을 실행시켜보자. xwininfo는 특정 윈도를 클릭하라는 요청을 하게 되고 클릭한 후 그 윈도에 대한 위치 및 그밖의 정보를 자세히 출력하게 된다. X 윈도와 동시에 실행되는 프로그램의 위치를 세팅하고자 한다면 이와 같은 여러가지 방법으로 위치를 알아낸 후 스타트업 파일(.xinitrc)에 이 정보를 적절한 옵션(-geometry등)으로 세팅하면 앞으로는 항상 지정된 위치에서 실행시킬 수 있을 것이다.

제 6 절 윈도 매니저 설정하기

이제는 윈도 매니저 설정에 대한 방법에 대해 이야기 하도록 하자. 물론 윈도 매니저에 따라 매우 다양한 세팅방법이 있겠지만, 여기서는 FVWM2를 대상으로 사용자가 변경 가능한 선에서 알아보도록 하겠다. .fvwm2rc를 편집해 보도록 하자.

6.1 기본적인 FVWM2 설정

초기의 FVWM의 설정파일은 무척 특이하고, 복잡하였다. 그러나 최근의 버전에서는 복잡하지 않게 많이 개선되었으므로 그리 긴자하지 말기 바란다. 내가 아는 바로는 메뉴 설정하는 부분만 좀 복잡하다 생각될 뿐이다. 여기에 ㅐ해서는 추후 논의하도록 하겠다. FVWM2를 설치하면 기본적인 예제 설정파일이 있을 것이다. /var/X11R6/lib/fvwm2/system.fvwm2rc 정도에서 찾을 수 있다. 이 파일은 시스템의 기본 설정이다. 이 파일을 홈 디렉토리에 .fvwm2rc로 복사해 놓고 시작하는 것은 무척 괜찮은 생가기다. 즉 FVWM2가 시작될 때 홈 디렉토리를 살펴 .fvwm2rc가 있다면 이 파일을 읽고 이 파일이 없다면 시스템의 기본 설정파일을 읽는 방식이다. 자 그렇다면 지금 홈디렉토리에 .fvwm2rc가 있거나 복사했다는 가정하에 이 파일을 편집하도록 하자. 만일 X 상에서 이 작업을 한다면 설정 파일과 눈에 보이는 것과의 관계를 쉽게 알 수 있을 것이다. 몇몇 설정을 바꾸어 좀 더 좋은 모습을 보이도록 하자. 아마 첫 설정사항(#으로 묶여서 주석처리된 라인 이외의 첫 엔트리)은 윈도 폰트 설정일 것이다. 폰트 설정에 대한 사항은 X 폰트와 색 섹션을 참고 하기 바란다.

6.2 개선된 FVWM2설정

사실 FVWM2의 설정은 미묘한 부분까지 포함하고 있기 때문에 종종 매우 복잡해 지곤한다. 이런 문제의 해결책으로 모듈화 방법이 쓰이고 있다. 아래의 예는 fvwm2gnome을 위한 .fvwm2rc의 설정파일이다.

```
# import config files

Echo -Styles-
Read .fvwm2gnome/config/styles/app.styles
Read .fvwm2gnome/config/styles/window.styles

Echo -Buttons & Keys-
Read .fvwm2gnome/config/buttons.config
```

이 예제에서 보면 다른 설정파일을 읽어오는 명령이 있다. 이런 방법을 이용하여 자칫 설정파일이 엄청나게 커져버릴지 모르는 상황을 적절히 모듈화 시켜 관리할 수 있게 해 준다. 모듈화 된 설정은 바꾸거나 디버깅하는데 용이하며 테마 작성을 쉽게 해 준다. 테마에 대한 것은 곧 설명하게 될 것이다.

6.3 FVWM2 설정을 더 쉽게 하기

사실 개개의 설정 사항을 여기에 설명하는 것은 무리이다.(프로그램과 함께 제공되는 문서파일을 참고하기 바란다.) 게다가 설정에 익숙해 있는 사람일지라도 자칫 실수하거나 혼돈을 이르키기도 한다. 따라서 FVWM2를 비롯한 FVWM, bash등 다양한 설정파일을 작성하는데 잇어서 도트파일 제네레이터(Dotfile Generator)를 사용하는 것은 많은 작업을 쉽고 간단하게 해준다. 제스퍼 K. 페테르슨(Jesper K. Pedersen)이 Tcl/Tk를 이용하여 제작하였으며 `<http://www.imada.ou.dk/blackie/dotfile/>` 에서 구할 수 있다. (물론 지금 쓰려는 컴퓨터에 Tcl/Tk가 올바르게 설치되어 있어야 한다. 아마 대부분의 리눅스 배포본이라면 설치되어 있을 것이다.) 이 프로그램은 기존의 설정파일을 부르거나 기본 초기값을 가지고 사용자가 원하는 여러 옵션이나 메뉴를 설정해 준다. Tcl/Tk 프로그램이기 때문에 어느정도 느린 것은 감수해야 한다. 그러나 직접 아무것도 없는 설정파일에서부터 메뉴를 설정할 수 있기 바란다. 이것이 흔히 말하는 (Do it yourself)라는 Unix의 방식일 뿐 아니라 세밀하고 사소한 부분까지도 스스로 알고 제어할 수 있게 해 준다. 어떤 방법을 선호할 지는 모르겠지만 기존의 설정파일을 가지고 시작하는 사람도 많이 있다.

6.4 FVWM2 테마

FVWM2의 새로운 기능 중 하나가 흔히 테마라 불리우는 것이다. 테마라 함은 아주 간다하게 한번에 모든 윈도의 기본적인 모습을 바꿀 수 있다는 것이다. 이미 윈도 매니저에서 제공하는 기능과 애플리케이션에서 제공하는 기능을 구분해 낼 수 있다면 윈도 매니저의 테마라는 것은 개개 어플리케이션의 겉모습과 내부적인 행동방식에는 변화를 주지 않는 것을 알 수 있을 것이다. 그러나 통합환경을 제공하는 KDE나 GNOME은 전반적인 어플리케이션에 까지 일정한 겉모습과 행동양식을 부여해 줄 수 있다. GNOME과 KDE는 이런 면에서 매우 유용하다 할 수 있다. 테마를 만들기 위해서는 역시 .fvwm2rc를 편집해야 한다. 아래 예제는 나의 .fvwm2rc 파일의 Style 섹션 이후에 추가한 테마 설정이다.

```
# Blue Theme
DestroyDecor Blue
AddToDecor Blue
+ WindowFont -b&h-lucida-bold-r-*-*-*-140-*-*-*-*-*-*
+ TitleStyle ActiveDown (Solid DarkSteelBlue) ActiveUp (Solid
SteelBlue) Inactive (Solid Grey)
+ HilightColor white blue
+ ButtonStyle 1 -- UseBorderStyle
Style Blue UseDecor Blue, BorderWidth 5, HandleWidth 5, MWMborder,
MWMbuttons
```

```
# Function to change all windows to a new style.
DestroyFunc ChangeStyle
AddToFunc ChangeStyle
+ "I" Style $0 $1
+ "I" Recapture
```

이것은 상당히 많은 내용을 포함하고 있다. 기본적으로 맨 위에 우리는 "Blue"라 불리우는 테마를 선언하고 있다. 이것을 기본으로 삼아 다른 테마 작성에 유용하게 이용할 수 있다. 물론 실제 테마에서는 이것보다 더 많은 내용을 포함할 수 있으며 충분히 다른 테마와 구분될 수 있는 많은 차이점을 둘 수 있다. 이런 많은 기능을 이용해 다른 컴퓨터에서 한번도 보지 못한 새로운 윈도 스타일을 만들 수도 있다.

위의 예제에 보면 새로운 테마를 생성하기 전에 제거하는 것을 주의하여 보기 바란다. 이것은 FVWM2를 여러 다른 스타일로 재시작할 때 좀더 부드러운 시작을 할 수 있게 해 준다. 우리가 제작한 테마를바꾸기 위해서는 일반적인 인터페이스를 통해 이루어 질 수 있다. 메뉴를 통해 바꾸는 것은무척 쉽고, 편리하다. 다음의 예제를 통해 우리가 제작한 테마나 이미 제작된 테마로의 변환을 쉽게 해 주는 메뉴를 만들 수 있다.

```
DestroyMenu "Themes"
AddToMenu "Themes"
+ "Choose a theme..." Title
+ "" Nop
+ "Blue" ChangeStyle "*" "UseStyle Blue"
+ "Mwm" ChangeStyle "*" "UseStyle Mwm"
+ "Flat" ChangeStyle "*" "UseStyle Flat"
```

좀 더 다양한 메뉴설정을 위해서는 프로그램과 함께 제공되는 예제 설정 파일이나 시스템에서 이미 쓰이고 있는 설정파일을 참고하면 될 것이다. 위의 예에서는 ChangeStyle 이라는 펑션을 이용하였다. 이 펑션을 이용하여 사용중인 스타일에서 다른 스타일로의 변경을 하고 있다. 아까 우리가 집어 넣었던 '생성 이전에 제거'라는 기능을 이용한 것이 무척 유용했다는 것을 확인해보기 바란다. 이제 변경한 메뉴를 적용하기 위해 FVWM2를 재시자하면 테마를 변경할 수 있는 메뉴항목이 나오는 것을 볼 수 있다. 이미 제작된 FVWM2 테마를 얻고 싶다면 다음 사이트에서 구할 수 있다. `<http://www.vis.colostate.edu/scriven/Linux/fvwm/index.html>`

제 7 절 폰트와 색

X에서 폰트와 색을 설정하는 데에는 많은 기법이 있다. 하지만 다른 시스템(Windows, MAC)과 비교해 봤을 때 그리 간단하지 만은 않다. 폰트는 간단한 단 하나의 이름이나 속성으로 표시되지 않고 여러 속성이 포함된다. 처음에는 이런 방법이 복잡하고 혼란을 일으키지만 곧 그리 복잡하지 않다는 것을 깨닫게 될 것이다.

7.1 폰트의 설정사항

X의 논리 폰트 속성(XLFD - X logical font description) XLFD는 다음의 속성을 가지게 된다.

- `fndry` - 폰트 제작자, 회사나 개인등 폰트 제작자를 나타낸다.
- `fmly` - 폰트 패밀리, 가장 익숙한 폰트 종류의 이름이다.
- `wght` - 폰트 두께(bold-굵게, medium-보통 등)
- `slant` - 폰트 장식(italics-이탤릭, oblique-기울어진, roman-로만등
- `sWdth` - 폰트 폭(normal-보통, condensed-좁은, extended-넓게)

- `adstyl` - 부가적인 장식 (sans serif, serif 등)
- `pxlsz` - 픽셀 크기, 문자의 세로 길이
- `ptSz` - 포인트로 나타낸 크기, pxlsz와 비슷하다.
- `resx` - 가로 해상도, dpi로 표시된다.
- `resy` - 세로 해상도, dpi로 표시된다.
- `spc` - 간격 Schumacher 폰트에서나 유용한 것 같다.
- `avgWidth` - 폰트의 평균 폭
- `rgstry` - 폰트 목록에서 찾을 수 있는 레지스트리
- `encdng` - 국가별 인코딩

지금으로선 위의 속성들이 쉽게 이해가 가지 않을 것이다. X 윈도의 폰트선택 프로그램인 xfontsel 을 이용하면 각 속성에 따라 어떻게 변하는지 알 수 있을 것이다. 지금 수행해 보기 바란다. 사실 xfontsel의 프로그램은 또 하나의 혼란을 야기할 수도 있다.(좀 직관적이지 못한 인터페이스를 가지고 있다.) 하지만 해 보라. 곧 쉽게 익숙해 질 것이다. 우선 fndry 버튼을 누르고 그중 하나를 선택해 보기 바란다. 지금 시스템에 폰트가 올바로 설정되었다면 adobe나 b&h bitstream 과 같은 몇몇 선택 가능한 항목을 볼 수 있을 것이다. 만일 b&h등 하나를 선택한다면 버튼 아래의 윈도창의 폰트가 바뀌는 것을 볼 수 있다. 왼쪽으로부터 시작해서 오른쪽으로 선택해 나가는 것이 일반적인 폰트선택 방법이다. 오른쪽에 있는 항목은 왼쪽 항목에 종속적이며 몇몇 폰트에 한정된 특정한 옵션일 수 있기에 폰트 선택의 폭이 좁아진다. 즉 지금 fndry에서 b&h등을 선택한 후 fmly의 메뉴를 보면 몇몇 메뉴만이 시행 가능하게 되어있을 것이다. 이 fndry에서 제작된 fmly만 보이는 것이다. 그러나 Courier 폰트처럼 여러 제작사에서 제작한 폰트도 있다. 이제 이런 규칙에 따라 wght나 나머지 속성을 정할 수 있다. 만일 맘에드는 폰트가 골라졌다면 더 이상 선택하지 않아도 된다. 사실 시스템에 폰트가 무제한으로 많은 것이 아니기 때문에 선택하지 않아 와일드카드 옵션인 * 로 표시된 부분이 특정 옵션을 대표하는 것이 되기 때문이다. 자 이제 폰트선택에 익숙해 졌다면 이 폰트를 특정 어플리케이션에 적용해 보도록 하자.