Linux uses the X Window System for its GUI

- de facto GUI standard Unix-wide
- original project at MIT and DEC
- Version 11 Release 6 (X11R6) released 1994
- current oversight and maintenance
  - x.org, an industry consortium
    - http://www.x.org/
    - http://www.freedesktop.org/Software/xorg
**X Window System topics**

- Composition and architecture
- Ways to launch X
- X client-server model and networks

**X Window System’s pieces**

- an X server
- X clients
  - Graphical apps
    - eg, xclock, xeyes
  - Window managers
    - eg, mwm, twm, fvwm, wmaker, sawfish, metacity
    - and more: http://xwinman.org
  - Desktop environments
    - eg, gnome, kde
Server’s job

- track input from input devices
  - convey to relevant client apps, if any
- track output from client apps, if any
  - convey to display device
- absent: the concept of windows

Clients’ function

- accept input from server
  - originated from input devices there
- send output to server
  - to be forwarded onto display there
- output is a response to input, as client desires
  - input – server to client:
    “I want to let you know there was a click, at (x,y)”
  - client figures out:
    hey, this click is strategic! it was on “minimize”
  - output – client to server:
    “in that case please minimize that window for me”
Client example: xeyes

- X tells xeyes where cursor is, whenever it moves
- xeyes calculates corresponding pupil positions
- xeyes tells X where to put pupils
- X does so

X client-server model

- Client and server loosely coupled
- Just need to communicate events (server-to-client) and requests (client-to-server)
- Can be on
  - the same machine
  - different, if transport available for events/requests
server <-> client communication

User input passes from server to client in the form of events. An event is a packet of information that tells the client something it needs to act on, such as keyboard or mouse input. When a client program receives some kind of event, it responds with some sort of action affecting the display. For instance, it may request that a window be resized to particular dimensions. The server responds to requests by updating the appropriate window on the physical display.

X Window System User’s Guide, O’Reilly, pp 21, 312

close paraphrase

xev to show local events

local log, local event tester display
xev to show remote events

command was: xev –display 192.168.3.2:0

on 192.168.3.4:
local log.
event tester display on 192.168.3.2 (server)
remote activities:
left mouse button pushed, then released

Note: modern distributions usually disable the X server’s port 6000 listening feature. To demonstrate remote display you can terminate the X server and start:

```
# xinit /usr/bin/xterm -- /usr/bin/X -listen tcp
```

transport from server to client

event transport concurrent to button push

X11 has its own protocol – port 6000

expresses the particular event(s) that happened
Window managers

- X clients that control other X clients
- In terms of
  - framing
  - position
  - size
  - movement
  - focus

Desktop environments

- orchestrate inter-client communication
- integrate clients/apps into common environment with consistent capabilities
  - drag and drop
  - common clipboard
  - common menus
  - uniform iconic interface to filesystem
### To launch X from command line

- technical but un-useful
  - X
- more useful
  - xinit
- normal
  - startx, frontend to
  - xinit, frontend to
  - xinitrc, frontend to
  - $HOME/.Xclients, $HOME/.Xclients-default

### Using “X” to launch

- runs the X server but no client to go with it
  - X tracks input on behalf of clients but there are none
  - X sends to display any output from clients but there is none
- serves no purpose

- emergency exit: ctrl-alt-backspace key
Clientless (useless) X

Using “xinit” to launch

- launches X server, then a client
  xinit <client> -- <server>
- certain clients can launch further clients
  - xterms
  - window managers
- runs xterm if no <client> specified
- X server torn down when xinit’s <client> terminates
Launching “xinit”, xeyes as client

1. xinit launched server
2. Then xinit launched client

Launching “xinit”, xterm as client

1. xinit launched server
2. Then xinit launched client
…then launch xeyes from xterm

Launching “xinit”, fvwm as client
Using “startx” to launch

- starts xinit with reasonable options
- xinit after starting X, delegates client launching to a script, if any, named .xinitrc in user’s home directory

.xinitrc is your list of clients to launch

```
#!/bin/bash
# sample/simple .xinitrc shell script
#put up a clock
xclock &
#start a window manager
exec fvwm
```

or calls it

```
#!/bin/bash
# sample .Xclients or .Xclients-default file
#start a window manager
exec fvwm
```
By default, with Fedora 10

- window managers*
  - mwm /usr/bin/mwm
  - twm /usr/bin/twm
- desktop environments
  - gnome /usr/bin/gnome-session

* Neither by default with Fedora 17
app, manager interfaces separate

- windows themselves, controlled by managers, differ
- window content, controlled by app, does not

- xclock with mwm window manager
- xclock with twm window manager
- xclock with window maker window manager

X Windows client/server model

- X client/application
- X server
- The rendered service

The service is the rendered display. The client is the application which has output to be displayed.
X’s client-server model is counter-intuitive

- X is a display server, not application server
- application servers
  - application is served to the user
  - server is where the application is, client is where user sees it
- display servers:
  - screen-rendering is served to the application
  - client is where the application is, server is where user sees it
Apps on a machine, screen output elsewhere

Screen output on a machine, apps elsewhere

© David Morgan 2003-17
Operation

Issued in the client, where xapp resides:

$ xapp -display <server>:0

Prerequisite server permission, in server:

$ xhost +

Special-guest client appearance!

- any client can display to a server elsewhere
- window managers are clients
- so a window manager can display to a server elsewhere (thus create and manage its windows)
- subject to the not-more-than-one window manager limitation on any server
  - launch X on server using xinit not startx (to avoid a wm)
  - then apply/display a wm as client from another machine (e.g., run “fvwm –display <this box>:0)” on that box)