Den Store AV-bog

Søren Pilgård - DIKUrevyen

15. oktober 2013

Indhold

1 Introduktion

Dette er en introduktion til DIKUrevyens AV-system. Systemet er udviklet af Søren Pilgård.

Systemet er på skrivende tidspunkt ikke færdigt men kan sagtens bruges til normalt AVbrug.

Hvad skal man kunne for at bruge DIKUrevyens AV-system?

- LaTeX
- Linux
- Emacs
- Emacs Lisp
- Python

LaTeX
Bruges til at skrive selve overteksterne. Der bruges Beamer til at lave et langt slideshow, overtex.sty definerer en række makroer der gør det let at lave en lang præsentation. De fleste på natfak burde kunne LaTeX, hvis ikke er den mængde der bruges til at lave normale overtekster ret lille og burde kunne mestres ved bare at kigge i nogle af de gamle filer.

Emacs
Emacs bliver brugt som kontrolcenter til det hele. Det er derfor væsentligt at have en hvis forståelse for hvordan Emacs virker. Emacs er et voldsomt konfigurerbart program til at arbejde med tekst. Jeg er selv stor Emacsbruger og har opbygget et helt unikt system. På sigt er det håbet at lave en standard konfiguration som folk der ikke er emacsbrugere kan udnytte. En sådan konfiguration ville kunne skjule at det overhovedet er emacs der kører bag det hele.

Emacs Lisp
Emacs Lisp er det primære scripting sprog der bliver brugt. Det bruges til at automatisere ting i Emacs, derudover kan der kaldes externe kommandoer og kommunikere med andre maskiner. Selve revysystemet bruger en stor del Emacs Lisp så hvis noget stopper med at virke er det godt at kende. Til almindeligt AV brug kan man dog nøjes med en stærkt begrænset del som denne guide nok skal introducere.

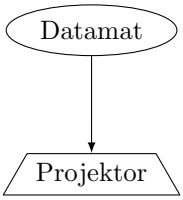
Linux
DIKUrevyens AV-system er bygget og kører på linux styresystemet. Det betydder at man for at bruge systemet effektivt er nød til at have en hvis forståelse for at bruge en linuxmaskine igennem en terminal. Et håb er en gang at have en form for standard opsætning af datamater hvor alt nødvendigt er installeret, som AVmand skal man så blot sætte udstyrret op og lave indholdet.

Python
Bliver brugt til de forskellige programmer og værktøjer der udgør resten af AV-systemet. Det burde kun være nødvendigt at kunne kode python for at udvikle/vedligeholde systemet.

2 Tanken bag

Den grundliggende filosofi bag systemet er at AV skal kunne komplementere en revy men aldrig komme i vejen. Der skal dermed kun vises det som publikum skal se/høre og intet andet. Det virker utroligt amatøragtigt lige så snart publikum ser en cursor, en film der maksimeres eller rammerne på et vindue.

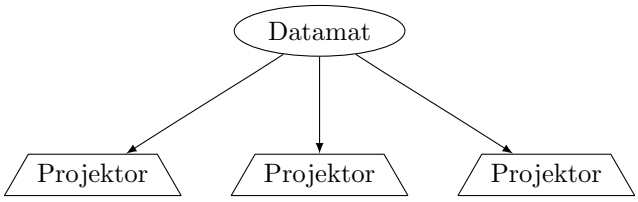
Det simpleste AVsystem man kan lave er at have en datamat tilkoblet en projektor. Datamaten kan derfra køre et presentationsprogram, f.eks. powerpoint.



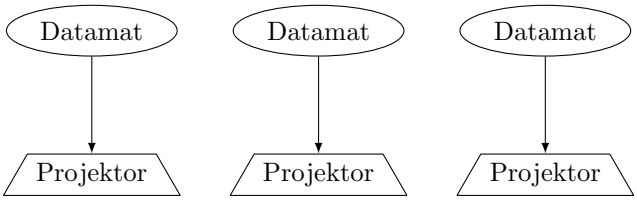
Figur 1: Simpel AV opsætning.

Dette er dog ret primitivt. Det er utroligt nemt at komme til at lave fejl så som at lige få skubbet cursoren hen på et andet desktop eller at filmen spiller på den forkerte skærm. Ting der er sket gang på gang til mange revyer, og som hurtigt får publikum til at råbe "TeXniken sejler!".

Derudover har man et problem hvis man skal bruge mere end 1 projektor.



Figur 2: En datamat, flere projektorer.

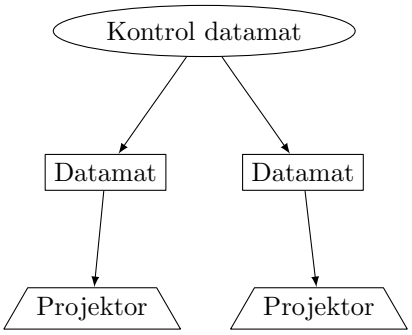


Figur 3: Flere datamater, flere projektorer.

Man kan enten koble flere projektorer på en datamat, hvilket kræver en datamat der er i stand til dette, hvilket ikke særligt mange er. Alternativt kan man have flere datamater koblet til hver sin projektor. Nu skal man så bare navigere rundt mellem en helt masse maskiner eller have en AV-mand pr. maskine hvilket heller ikke er særligt praktisk. Desuden har vi stadig problemet med at man let kommer til at dumme sig på samme måde som i den første opsætning.

Det vi i virkeligheden ønsker os er en abstraktion mellem det at styrre AV og det at vise AV. Hvis tingene kører adskilt og har klart definerede ansvars områder kommer man ikke lige så let til at lave fejl. Vi ønsker derfor at have en central maskine der styrer det hele, denne kommunikerer med andre maskiner der sørger for at vise ting via projektorer.

Det er dette princip DIKUrevyens AVsystem benytter sig af.



Figur 4: Central styring, flere arbejdere med hver deres projektor.

Det første system blev lavet af Troels Henriksen. Han lavede et programmeringssprog *Sindre* i Haskell til at lave grænseflader. I det sprog lavede han et `dmenu` lignende program kaldet *sinmenu* som han brugte som interface. Selve overteksterne bestod af en pdf som han lokalt oversatte til rå tekst som blev fodret ind i et script der brugte hans grænseflade. Dette script kommunikerede med en server koblet til en projektor over en sshforbindelse. `Xpdf` kan køres som en server der kan modtage kommandoer fra en kommandolinje, på denne måde kunne man derfor styre overtekster over ssh.

Desværre skalerede løsningen ikke særligt godt. Da der begyndte at komme mange AVeffekter kunne man ikke både køre overtekster og andet AV alene.

Dette blev (i hvertfald forsøgt) rettet op på da jeg (Søren Pilgård) efterfølgende begyndte at skrive et nyt AVsystem til DIKUrevyen 2012. At skrive et AVsystem er dog ikke en nem tjans og det endte med at foregå i en løbende process med forbedringer til hver revy.

Det nye AVsystem består af en centralt grænseflade der kan kommunikere med flere forskellige maskiner der kan vise AV materiale.

Selve grænsefladen er udviklet som en række udvidelser til Emacs. Grunden til dette er at da jeg startede indså jeg at systemet skulle kunne følgende ting:

- Kommunikere med en server der viser indholdet.
- Det skulle kunne vise ting overskueligt i en grafisk grænseflade.
- Der skulle være mulighed for at indlejre scripts så man kan køre ting automatisk på bestemte steder i forestillingen.
- Det ville være praktisk hvis man kunne rette fejl i texkoden mens man viste pdf'en da man ellers risikere at glemme dem.

Det gik hurtigt op for mig at det ville være fjollet at udvikle noget fra bunden da Emacs i forvejen kunne meget af dette. Desuden er jeg Emacsmand og så hurtigt hvordan en integration ville være nice. Den grundliggende formel for at vise overtekster er at man åbner et LaTeX dokument der bruger beamer pakken til at lave overteksterne. Så starter man `ubertex`minor modet, dette sørger for at pdfen bliver lagt op på serveren. Herefter skjules de fleste LaTeXkommandoer og et overlay lægges der viser hvad der bliver vist. Man kan så trykke "næste" hvilket rykker overlayet ned og synkroniserer serveren til at vise det tilsvarende slide i pdfen. Man kan også trykke vilkårlige steder i tex filen og rykke direkter hertil i overteksterne. Desuden kan der indsættes kode der eksekveres når man når til det pågældende slide.

Derudover findes minormodet `uberscript` der lader en afvikle scripts. Det kunne f.eks. være en sketch med en række lydeffekter eller et kald til en video. Et script kunne også være aktoversigten

hvorfra man ved at trykke "næste" kommer ind i det næste nummer, og når dette er færdigt kommer man så tilbage og er klar til næste.

Hvis alt går som det skal, skal man som AVmand kun trykke på en knap (næste) for at afvikle en revy. Dette må være essensen af et godt AV system, når man kun skal tage sig af timingen på skuespillerne/sangerne samt disses fejltagelser.

3 Sådan er man en god AV mand

TODO: Her kommer der til at være en række mere generele råd om hvordan man laver god AV

3.1 Om at lave gode overtekster

4 Opsætning af tekniken

4.1 Projektorer

Til DIKUrevyen bruges der typisk i hvertfald 2 projektorer. Der bruges 1 der peger op på lærredet over scenen, her vises der overtekster, almindelige av-ting til sketches samt små film.

Kantinen ejer en stor Epson som bruges til introen/av på scenen

4.2 Netværk

4.2.1 Statisk IP

Hvis dit netværks interface hedder enp0s25

```
sudo ip link set enp0s25 up
sudo ip addr add 192.168.0.XXX/24 dev enp0s25
```

alternativt

```
sudo ifconfig eth0 192.168.0.XXX
```

4.2.2 Hosts

Da det kan være svært at huske alle ip-addreserne kan man i stedet navngive maskinerne. Dette gøres lokalt i `/etc/hosts` et eksempel:

```
192.168.0.20    intro
192.168.0.30    brok
192.168.0.40    left
192.168.0.50    mid
192.168.0.60    right
```

Nu kan man ssh'e ind ved blot at skrive:

```
ssh brok
```

Det virker også med scp ol.

4.2.3 Forbindelse uden login

For at logge ind på systemerne over netværket bruges ssh. Da det bliver jævnt irriterende hele tiden at skulle taste løsener kan man lægge sin offentlige ssh nøgle ind på de forskellige maskiner

```
cat .ssh/authorized_keys | ssh revy@192.168.0.30 "cat >> ~/.ssh/authorized_keys"
```

det kan være du først skal oprette mappen `.ssh`.

alternativt kan du bruge

```
ssh-copy-id revy@192.168.0.30
```

Udskift `pilen` og ip'en med den relevante bruger og ip. Husk at du skal have en ssh nøgle først dannes med:

```
ssh-keygen
```

4.2.4 ssh mount

I stedet for at scp'e alt muligt crap frem og tibage kan man benytte sig af et ssh mount, som er et filsystem over ssh. Jeg anbefaler at man har filerne liggende lokalt og fra hver maskine der skal kommunikeres med køres `sshfs` som får filerne frem.

Installer sshfs F.eks.:

```
sudo pacman -S sshfs}
```

fuse `sshfs` benytter fuse

```
sudo modprobe fuse
```

4.2.5 Gateway

Hvis du er intereseret i at få internet på det lokale netværk kan man lave en gateway. Dette består i at en af datamaterne på det lokale netværk også er forbundet til et netværk med internet, f.eks. eduroam.

Der er en guide på `sigkill.dk/writings/guides/gateway.html`

På gateway datamaten sættes først den statiske ip (allerede gjordt i forrige trin) Her efter køres gateway.sh som root (hentes på siden)

På clienterne der vil udnytte gatewayen sørger man først for at der er en statisk ip.

herefter skrives

```
route add default gw 192.168.0.XXX
```

eller

```
ip route add default via 192.168.0.XXX
```

Hvor "192.168.0.XXX" er ip'en til gatewaymaskinen

herefter sættes en dns server, f.eks. google

```
echo nameserver 8.8.8.8 > /etc/resolv.conf
```

Hvis dette ikke virker kan man evt. kigge på `https://wiki.archlinux.org/index.php/Internet_Share` for en ip baseret løsning

4.3 Brok

Sæt ham til Projektor, tastatur, netværk og strøm.

Hvis der skal spilles lyd fra brok bør der være jord på da der ellers kan komme en brummen.

Brok har på skrivende tidspunkt ubunut 11.04 natty. Det betyder også at der er gnome, unity og ting og ækle sager på. Disse har en tendens til at gøre livet surt, da de overskriver xorg konfigurationer og gør mærkelige ting man ikke helt kan gennemskue.

Dette er IKKE optimalt! Support til 11.04 udløb oktober 2012. Der bør installeres et ordentligt styresystem. UDEN gui (installeres seperat).

1: Tænd brok
Brok logger automatisk ind med brugeren ***revy*** med løsnet ***hamster*** Herefter starter unity der har en 'kør' dialog ting.

2: Start en terminal
Skriv `terminal` og tryk enter. Der vil nu komme en terminal op i hjørnet.

3: Kom på netværket

```
sudo ifconfig eth0 192.168.0.30
```

Giv ham en passende ip du kan huske.

4: Opret SSH forbindelse Der kører allerede en ssh daemon. forbind til ***revy*** med løsnet ***hamster***. Du kan evt. uploade din offentlige ssh-nøgle så du slipper for at logge ind.

```
ssh revy@192.168.0.30
```

5: Stop gdm
Gnome Desktop Manager skal lukkes.

```
sudo service gdm stop
```

Dette lukker hele den grafiske grænseflade, inklusiv X.

6: Start screen

```
screen
```

screen køres så vi kan starte X i baggrunden. Når programmet startes vises der en menutekst, bare tryk enter. `screen` kører nu.

7: startx
Start X manuelt.

```
startx
```

Dette starter X op hvorefter .xinitrc eksekveres.

Som det er lige nu startes `xmonad` som windowmanager. Det betyder at skærmen pr. default er sort når x er startet

8: detach fra screen
Tryk `Ctrl-a d`. Dette lader alt der kører i screen fortsætte upåvirket mens du kommer tilbage til det forrige miljø. for at reattache skriv da `screen -r`.

9: test xmonad
Test om xmonad virker. På broks tastatur trykkes `Alt-Shift-Enter`. En terminal burde åbne sig. Luk igen med `Alt-Shift-c`.

Læs evt. afsnitet om xmonad.

Da terminalen er en default `xterm` med hvid baggrund er dette et snedigt trick til at se hele området projektoren kan lyse op. Jeg bruger dette ofte når jeg tweaker projektoren.

10: prøv ting af Med en ssh forbindelse åben, lad os prøve om vi kan få noget til at virke. Start med at vælge det 'display' der skal vises grafiske ting på

```
export DISPLAY=:0
```

Dette skal køres for hver gang du ssh'er ind.

start nu xpdf med en af de gamle overtekst filer der ligger et eller andet sted.

Se sektionen om kommandoer for nærmere detaljer

Hvis man gerne vil lave flere ting simultant på brok kan man sagtens åbne flere lokale terminaler og ssh'e ind parallelt.

4.3.1 Ting der køres på brok

For at hacke uden om alt muligt gøjl køres et par scripts gennem `.xinitrc`

swarp
`swarp` er et program der flytter musse-markøren (cursoren) til et givent koordinat. Swarp findes

på `tools.suckless.org/swarp`. `swarp` findes desuden i arch's repository, det kan være den også findes til debian baserede styresystemer, potientielt i en suckless pakke.

`swarp` køres på brok med argumenterne 20000 20000, hvilket flytter markøren ad helvede til.

istedet for `swarp` kan man bruge `unclutter` med kommandoen `unclutter -idle 0 -root &` Som i fjern cursoren efter 0 sekunders delay efter bevægelse inklusiv når cursoren er over rod baggrunden (altså ikke kun over vinduer).

fixsleep `fixsleep.sh` er et script der forsøger at forhindre X i at slukke skærm outputet. X bruger et system der hedder DPMS (Display Power Management Signaling) til automatisk at slukke skærm outputet efter en periode uden tastaturaktivitet.

fixsleep benytter følgende to kommandoer

```
xset dpms 0 30000 40000
xset s 30000
```

den første linje dækker over `xset dpms [standby [suspend [off]]]` den anden linje dækker over `xset s [timeout [cycle]]`

keepon

```
xset dpms force on
```

Denne kommando forcer dpms fra, (det svarer til at at trykke på en tast)

`keepon.sh` er et script der ligger i baggrunden og kører denne kommando whert 30 sekund.

fixsleep og keepon forsøger begge at holde dpms stangen ved at lave aktivitet. Man kan måske istedet bruge

```
xset -dpms; xset s off
```

Til at slå dpms fra. De to systemer kan dog ikke blandes.

Man kan se om dpms er slået til ved at kalde `xset -q`

4.4 Xmonad

Windos/super eller alt som modifier knap.

5 Komandoer

5.1 xpdf

Overtekster køres i xpdf der åbnes manuelt med:

```
xpdf -remote ubertex -fullscreen -mattecolor black -fg black
    -bg black -papercolor black filnavn
```

Da dette er meget langt kan man istedet bruge aliaset

```
p filnavn
```

5.2 mplayer

Til at vise videoer manuelt bruges mplayer:

```
mplayer -nolirc -msglevel all=-1 -msglevel statusline=5
    -vo gl2 -autosync 30 -cache 1048576
    -cache-min 99:100 -xy 500 -geometry 49%:40% filnavn
```

eller aliaset

```
m filnavn
```

5.2.1 Positionering af mplayer

TODO: Noget om positionering af mplayer her

6 Konvertering fra manuskrift til overtekster

7 ubertex

8 ubersicht

9 Hvordan virker koden

Der sørges automatisk for at det korrekte major mode køres på .tex og .el filer

10 FAQ

ubertex tager ikke højde for pauser i comments, latex gør.

11 Værktøjer

11.1 Schneider

Tysk: Skrædder

Dependencies:

- Python3
- ffmpeg

Schneider er et værktøj til at skære film og billeder op til at kunne blive vist over flere projektorer.

Schneider kan på skrivende stund kun dele medier op i flere snit ved siden af hinanden og ikke over under. Det burde dog være let at rette til.

11.2 Zeitherr

Tysk: Timelord Er en bastardiseret udgave af en NTP tidsserver til at holde de forskellige maskiner synkroniseret.

11.3 Zeiden

12 Problemer/løsninger

12.1 Der er lag/forsinkelser

Lag er irriterende, det opleves primært som en forsinkelse fra man har trykket på knappen til der sker noget på projektoren. Det er et problem når man skal lave overtekster.

Til DIKUs jubilæumsrevy var der ca. et sekunds forsinkelte fra jeg trykkede til at overteksterne blev vist. Jeg har ikke definitivt fundet fejlen endnu men der bliver arbejdet på det.

Her er først nogle metoder til at lokalisere hvor omtrentligt forsinkelsen opstår. Det er mere eller mindre umuligt at lave konkrete målinger så det er noget man må føle sig frem til (Super naturvidenskabelig metode!).
Følgende tager udgangspunkt i Xpdf, men gøres på samme måde med Zeiden

Prøv først at sætte systemet til at køre alt lokalt. Aka, kør både Emacs og Xpdf lokalt og der kommunikeres via ssh til `localhost`. Hvis forsinkelsen stadig er der, er fejlen enten i Emacs, i Xpdf, eller i den lokale hardware.

Mine erfaringer med Xpdf er dog at det ikke er her fejlen ligger. Prøv nu at starte Xpdf på serveren og ssh ind på denne, giv nu manuelt Xpdf ordre til at skifte slide.

Min erfaring er at det er meget tilfældigt hvad der præcist sker. Jeg oplevede at delayed forsvandt efter jeg gjorde ovenstående, også det mellem Emacs og Xpdf, men kun indtil Xpdf blev lukket.

Det virker også til at Xpdf bliver langsommer afhængigt af længden af overteksterne og ikke nødvendigvis størrelsen af pdfen.

Jeg ved ikke helt hvordan dette skal løses. Det er en af grundene til at vi vil lave vores egen fremviser (Zeiden).

13 APPENDIX A: En guide til Linux

cd ls cp mv

ip ssh nano

14 APPENDIX B: Installation af Arch Linux

```
loadkeys colemak
ip link show
ip link set enpXsX up
ip route add default via 192.168.0.XXX
echo nameserver 8.8.8.8 > /etc/resolv.conf
```

```
ping google.com
```

tryk Ctrl-c for at stoppe.

```
cgdisk /dev/sdX
```

Formententligt `sda`

Tryk enter.

`Delete` alle partitoner.

Lav plads til GPT partition *Dette giver plads ti en partions tabel til grub*

1. `New`
2. Tryk enter, first sector skal bare være default.
3. `1007KiB` enter
4. `ef02` enter
5. Tryk enter, der behøver ikke at være et navn

Lav swap *Dette er næppe nødvendigt, men jeg laver den af gammel vane*

1. Tryk ned.
2. `New`
3. Tryk enter, default er fint, formententligt 2048.
4. `3G` mindre swap kan også vælges (eller udelades).
5. `8200` swap Hex koden.
6. tryk enter.

Lav en partition

1. Tryk ned.
2. `New`

3. enter
4. enter
5. enter
6. enter

Vælg `Write`, skriv `yes`. Nu kan der stå at den gamle partitionstabel stadig er i brug. Afslut, genstart og udfør alle trinene inden `cgdisk` igen.

Når du er klar igen skal vi lave nogle filsystemer.

```
lsblk
```

For at få overblik

```
mkfs.ext4 /dev/sda3
mkswap /dev/sda2
swapon /dev/sda2
```

Ignorer sda1 indtil videre.

```
mount /dev/sda3 /mnt
```

Hvis du skulle have lavet en seperat partition til home så lav en mappe `mkdir /mnt/home` og mount home partitionen der `mount /dev/sdaX /mnt/home`.

Sørg for at være på nettet da vi nu skal hente pakker ned til styresystemet.

```
pacstrap -i /mnt base
```

Tryk enter til spørgsmål.

```
genfstab -U -p /mnt >> /mnt/etc/fstab
arch-chroot /mnt /bin/bash
nano /etc/locale.gen
```

Fjern kommenteringen til linjerne *#da_DK.UTF-8 UTF-8* og*#en_US.UTF-8 UTF-8*

```
locale-gen
echo LANG=en_US.UTF-8 > /etc/locale.conf
export LANG=en_US.UTF-8
ln -s /usr/share/zoneinfo/Europe/Copenhagen /etc/localtime
hwclock --systohc --utc
echo XXX > /etc/hostname
```

hvor XXX er navnet til maskinen

```
mkinitcpio -p linux
```

```
passwd
```

Skriv løsn til root.

```
pacman -S grub
grub-install --target=i386-pc --recheck --debug /dev/sda
grub-mkconfig -o /boot/grub/grub.cfg
exit
umount -R /mnt
shutdown -h now
```

Hiv usbstikket ud.

Tænd datamaten igen. Hvis du som mig til jubilæumsrevyen installerede systemet på en harddisk i en anden datamat en dens egen, kan det være at ramdisken fejler, i grub vælges da fallback løsningen og du kalder `mkinitcpio -p linux` for at skabe et nyt korrekt image.

```
ip link show
```

For at se netværks interfacet

```
useradd -m -s /bin/bash revy
```

```
passwd revy
```

Giv revy et løsn.

```
su revy
nano /home/revy/lan.sh
```

indtast

```
ip link set enpXsX up;
ip addr add 192.168.0.XXX/24 dev enpXsX;

ip route add default via 192.168.0.YYY;
echo nameserver 8.8.8.8 > /etc/resolv.conf;
```

Gem, luk nano og skriv `exit`.

```
pacman -S sudo
visudo
```

Tryk pil ned til du finder linjen

```
root ALL=(ALL) ALL
```

placer cursoren under denne og tryk `i` tast `revy ALL=(ALL) ALL` tryk enter, tryk escape tryk :wq enter.

revy kan nu sudo'e.

```
pacman -S openssh
```

```
systemctl enable sshd.service
systemctl start sshd
```

Nu kan vi ssh'e ind fra vores lokale maskine `ssh revy@192.168.0.XXX`. *Læs evt. afsnittet om ssh-nøgler.*

Nu kan vi vælge at køre ting via ssh forbindelse.

Nu skal vi installere pakker

```
pacman -S xorg-server xorg-server-utils xorg-xinit
```

Der skal formententligt installeres nogle grafik drivere, følgende er nogle udemærkede open source standard drivere

```
pacman -S mesa xf86-video-vesa
```

Der kan kun køre én instans af pacman af gangen. Så sæt ham til at arbejde mens vi i en anden terminal begynder at konfigurere.

```
pacman -S xmonad xmonad-cotrib alsa-utils rxvt-unicode feh mplayer ttf-dejavu
```

I en terminal anden åbner du `.xinitrc` og skriver aller nederst:

```
exec xmonad
```

Åben så `lan.sh` og i bunden tilføjer du

```
xset -dpms; xset s off
```

Åben `.bashrc` og tilføj i bunden:

```
if [[ -z $DISPLAY && $(tty) = /dev/tty1 ]]; then
    exec startx
fi
```

Du kan evt. også tilføje følgende alias:

```
alias d='export DISPLAY=:0'
```

15 APPENDIX C: Stripped Emacs

16 APPENDIX D: Emacs

17 APPENDIX E: mangler

revy-shell skal escape kommandoer Youtube Arduinoer Stripped Emacs

18 PDF/billedviser

Kan baseres på poppler. python-poppler-qt4?

mupdf er et mere minimalistisk/optimeret alternativ til xpdf's poppler. har mudraw der kan rendere til png.