"Displays managers", Wayland i gestió de sessions

El "display manager" Gdm

Gdm és un “display manager”, és a dir, un programa encarregat de gestionar els inicis de sessió gràfics dels usuaris a un determinat escriptori. El Gdm forma part del projecte Gnome (de fet, cada escriptori sol tenir el seu propi display manager: KDE té el Kdm, LXDE té el Lxdm, etc) però això no impedeix fer servir Gdm per donar accés a altres escriptoris diferents.

El Gdm es configura amb el fitxer "**/etc/gdm/custom.conf**" (a Fedora) o "**/etc/gdm3/custom.conf**" (a Ubuntu). Aquest fitxer és te tipus "INI" i té vàries seccions, però, a la pràctica, l'única que ens interessarà conèixer és la secció *[daemon]*, sota la qual podem tenir les següents directives:

*Per entrar automàticament amb un usuari predefinit:

AutomaticLoginEnable=true : Activa l'inici de sessió automàtic
AutomaticLogin=pepito : Especifica l'usuari per defecte

o bé

*Per fer una entrada temporitzada:

TimedLoginEnable=true : Activa l'inici de sessió automàtic retardat
TimedLogin=pepito : Especifica l'usuari per defecte
TimedLoginDelay=10 : Especifica el nº de segons que s'esperarà abans d'entrar a l'usuari per defecte per poder escollir un altre

Després de cada canvi a l'arxiu anterior cal reiniciar el "display manager" executant *sudo systemctl restart gdm* (els "displays managers" funcionen com serveis permanentment en marxa...de fet, si s'aturen, s'acabarà la sessió dels usuaris que l'hagin iniciat mitjançant ell). Cal dir que també existeix un enllaç anomenat "/etc/systemd/system/display-manager.service" que apunta a l'arxiu "service" corresponent al "display manager" utilitzat en aquest moment; per tant, una comanda més genèrica, independentment del "display manager" concret que tinguem en marxa (ja que podríem tenir-ne varis instal·lats però només un pot estar en actiu) seria *sudo systemctl restart display-manager*

EXERCICIS:

1.-a) Fes que l'usuari "usuari" d'un sistema Ubuntu o Fedora amb entorn gràfic (corrent en una màquina virtual) tingui una entrada automàtica a l'escriptori temporitzada de 5 segons

b) Instal·la el paquet "sddm", corresponent al "display manager" del mateix nom. Per activar-lo, a Fedora has d'executar *sudo systemctl enable sddm* però veuràs que obtens un error dient que l'enllaç "display-manager" existeix i ja apunta a un altre "display manager" (el Gdm). Elimina aquest enllaç (directament amb *rm* o fent *sudo systemctl disable gdm*) i torna a intentar habilitar el servei *sddm* (veuràs que es regenerarà l'enllaç "display-manager" però ara apuntant *sddm*. Un cop aconseguit, reinicia la màquina i comprova què es veu en l'inici de sessió gràfic.

> NOTA: En el cas de fer servir Ubuntu, l'activació del nou "display manager" és més senzilla perquè es pot realitzar directament de forma interactiva durant el seu procés d'instal·lació gràcies a l'assistent "Debconf" que hi apareix, on es pregunta quin "display manager" dels actualment disponibles (incloent el recentment instal·lat) es vol triar com a "display manager" per defecte. Aquest assistent el pots tornar a executar sempre que vulguis amb la comanda *dpkg-reconfigure gdm*.

c) Ara deshabilita el "display manager" *sddm* però no n'habilitis cap altre d'alternatiu. Reinicia el sistema. ¿Què passa? A l'entorn que t'apareix, executa la comanda *systemctl get-default* i digues què mostra. Executa finalment *sudo systemctl start gdm*. ¿què passa? Finalment, habilita de nou el "display manager" Gdm.

d) Executa la comanda *systemctl cat gdm* i digues quin significat tenen els valors concrets indicats a les línies *Conflicts=* i *After=* , *Restart=* i *EnvironmentFile=* mostrades.

Altres subcarpetes interessants ubicades dins de la carpeta "**/etc/gdm**" són:

Init : conté els scripts que seran executats abans de presentar la pantalla de login. Útil per arrencar programes que s'hagin d'executar mentre es mostri aquesta pantalla de login, per exemple
PostLogin : conté els scripts que seran executats una vegada l'usuari s'ha autenticat correctament (però abans d'inicialitzar cap sessió). Útil per establir configuracions necessàries abans d'iniciar la sessió de l'usuari en qüestió
PreSession: conté els scripts que seran executats en el moment d'inicialitzar la sessió de l'usuari. Útil per tasques de gestió de sessió o "accounting", per exemple.
PostSession : conté els scripts que seran executats en el moment de tancar la sessió de l'usuari (un cop es torna a veure el login gràfic)

Dins de cadascuna de les subcarpetes anteriors, a les distribucions Fedora i Ubuntu ja hi ha un shell script amb permisos d'execució anomenat "Default" (aquest nom és obligatori que sigui així si es vol que Gdm el trobi i l'executi; un altre nom farà que l'script sigui ignorat). Tots aquests scripts s'executen amb el privilegi de "root" i retornen el valor 0 si ho fan correctament (i un valor diferent de zero si no, fet que provocaria que la sessió d'usuari avortés). També cal tenir en compte que aquests scripts no poden romandre executant-se infinitament perquè bloquejarien el procés de login: el Gdm és bloquejat mentre aquests scripts s'estan executant.

D'altra banda, una carpeta també interessant és **/usr/share/gdm/autostart/LoginWindow**, que conté els fitxers ".desktop" que s'executaran cada cop que s'inicïi sessió en qualsevol dels escriptoris als què el "display manager" hi doni accés.

2.-A continuació crearem "a mà" la configuració necessària al Gdm per tal d'habilitar l'accés al sistema a un usuari "convidat" (és a dir, un usuari que no guardi res del que hagi fet al sistema -documents, configuracions, instal·lacions,...- en tancar la seva sessió).

a) Crea l'usuari executant la comanda *sudo useradd -m -d /tmp/guest -p $(openssl passwd -6 "") guest* ¿Recordes per a què servia cadascun dels paràmetres indicats a la comanda anterior?

aII) Per a què no aparegui a cada inici de sessió de l'usuari convidat l'assistent de benvinguda del Gnome, elimina el paquet "gnome-tour"

b) Fes que l'script "/etc/gdm/PostLogin/Default" tingui el següent contingut (i fes-lo executable). ¿Què fa?:

```
#!/bin/sh
guestuser="guest"
if [ "$USER" = "$guestuser" ]; then
   mkdir /tmp/"$guestuser"
   cp /etc/skel/.* /tmp/"$guestuser"
   chown -R "$guestuser":"$guestuser" /tmp/"$guestuser"
fi
exit 0
```

c) Fes que l'script "/etc/gdm/PostSession/Default" tingui el següent contingut (i fes-lo executable). ¿Què fa?:

```
#!/bin/sh
guestuser="guest"
if [ "$USER" = "$guestuser" ]; then
   rm -rf /tmp/"$guestuser"
fi
exit 0
```

d) Cal fer que Gdm permeti els inicis de sessió sense contrasenya (com a mínim per l'usuari "guest"). Això es pot fer afegint la línia *auth sufficient pam_succeed_if.so user = guest* al principi de l'arxiu "/etc/pam.d/gdm-password".

e) Tanca sessió i hauràs de veure el nou usuari "convidat". Inicia sessió amb ell (sense contrasenya) i prova de fer algun canvi en el sistema (el fons de pantalla, o guardar algun document, etc). Tanca sessió i torna a iniciar-la amb el mateix usuari "convidat". Comprova que els canvis realitzats s'han perdut

f) D'altra banda, ¿per a què serveix l'arxiu "libcanberra-ready-sound.desktop" que hi ha dins de la carpeta "/usr/share/gdm/autostart/LoginWindow" (observa el seu contingut)? ¿Què fa, concretament, la línia *AutostartCondition=GSettings org.gnome.desktop.sound event-sounds* que apareix al seu interior?

Part del funcionament del Gdm també es pot configurar des del Dconf (sobretot és rellevant la clau *org.gnome.login-screen*). No obstant, no podem obrir el *dconf-editor* o executar la comanda *gsettings* directament perquè els canvis els estaríem fent sobre l'usuari actual (no sobre l'usuari "gdm"). Es podrien fer trucs com ara executar un terminal com l'usuari "gdm" amb la comanda avançada *machinectl shell gdm@ /bin/bash* i llavors fer servir la comanda *gsettings (*executar simplement *sudo -u gdm* no funcionaria perquè *gsettings* necessita establir un canal DBus coherent amb l'usuari utilitzat per treballar amb Dconf, cosa que *sudo* no fa) però, realment, el més senzill per a què els canvis desitjats afectin al Gdm en sí és simplement aprofitar la funcionalitat dels perfils vista al PDF d'apunts anterior. Concretament:

1.-Assegura't de què aparegui la línia *system-db:gdm* al final del fitxer "**/etc/dconf/profile/user**"

2.-Escriu la configuració volguda en un arxiu (anomenat com vulguem) ubicat obligatòriament dins de la carpeta "**/etc/dconf/db/gdm.d**" i seguidament executa la comanda *sudo dconf update*.

Un exemple de configuració possible escrita a l'arxiu anterior pot ser la següent (la qual serviria, respectivament, per canviar el logo mostrat a la zona inferior de la pantalla del Gdm, per activar i conseqüentment definir i mostrar un determinat missatge personalitzat, per ocultar el botó de reiniciar el sistema i per no mostrar la llista d'usuaris -sinó obligar a escriure el seu nom-):

```
[org/gnome/login-screen]
logo="/usr/share/pixmaps/kasumi.png"
banner-message-enable=true
banner-message-text="Hola guapos i guapes"
disable-restart-buttons=true
disable-user-list=true
```

3.-a) Després de llegir els paràgrafs blaus anteriors, configura el Gdm adientment per tal de què mostri un logo diferent a l'estàndard, mostri un missatge personalitzat i no mostri la llista d'usuaris

D'altra banda, l'aparença visual del Gdm ve definida per l'activació d'un determinat tema de Gnome Shell (és a dir, l'estètica del Gdm ve lligada a la del tema Gnome Shell que es trïi). No obstant, establir una determinada aparença (incloent la imatge de fons, etc) requereix de certes modificacions sobre el sistema que no són immediates de fer (això és així "per disseny": els desenvolupadors de Gnome no volen que sigui "fàcil" canviar l'estètica del Gdm per tal d'evitar així possibles problemes que podrien aparèixer en el funcionament i/o la seguretat d'una peça tan sensible de software). No obstant, existeixen diversos scripts/aplicacions no oficials que implementen tots els detalls necessaris relacionats amb aquestes modificacions per tal de simplificar el procés a l'usuari. Concretament, ens trobem amb les següents opcions (totes són similars):

* "Gdm-tools": https://github.com/realmazharhussain/gdm-tools
* "Gdm-settings": https://github.com/realmazharhussain/gdm-settings (com l'anterior però gràfic i més complet)
* "Change-gdm-background": https://github.com/thiggy01/change-gdm-background
* "Change-gdm-background" (un altre): https://c-nergy.be/blog/?p=18409
* "Ubuntu-gdm-set-background": https://github.com/PRATAP-KUMAR/ubuntu-gdm-set-background
* "Fedora-gdm-wallpaper": https://github.com/DimaZirix/fedora-gdm-wallpaper

b) Executa les següents comandes per instal·lar l'script "gdm-tools":

sudo apt install libglib2.0-dev dconf-cli #A Ubuntu
sudo dnf install glib2-devel dconf #A Fedora
git clone --depth=1 https://github.com/realmazharhussain/gdm-tools.git
gdm-tools/install.sh

bII) Instal·la un tema Gnome Shell per tots els usuaris. La manera concreta de fer-ho no importa gaire però, en tot cas, has d'assegurar-te que el tema estigui contingut dins d'una subcarpeta ubicada sota "/usr/share/gnome-shell/theme" (i no pas sota la coneguda "~/.themes", que és particular per un usuari concret!). Una manera senzilla a Fedora és instal·lant algun paquet anomenat "gnome-shell-theme-*xxxx*" i a Ubuntu és instal·lant algun paquet "*-gnome-shell-theme". Un cop instal·lat el tema, executa les següents comandes tal d'establir una imatge de fons personalitzada a la pantalla del Gdm:

set-gdm-theme list
set-gdm-theme set nomTema
set-gdm-theme set -b /ruta/una/fotoqualsevol.png

> NOTA: Es pot establir el tema i el fons de pantalla en una sola comanda,així: *set-gdm-theme set nomTema /ruta/imatge.png*
> NOTA: La comanda *set-gdm-theme* té altres possibilitats interessants, com ara fer una còpia de la configuració actual del Gdm (*set-gdm-theme backup update*) i restaurar-la, o bé reestablir els valors per defecte (*set-gdm-theme reset*),etc

Wayland

A la majoria de distribucions Linux actuals, l'entorn gràfic (i per extensió, qualsevol escriptori) se sustenta sobre un programa "invisible" (però fonamental) anomenat "compositor" Wayland, el qual permet configurar el mode de treball de l'ordinador per a què, bàsicament:

1) Reconegui píxels -en comptes de caràcters- a la/es seva/es sortida/es (pantalla/es)
2) Gestioni les entrades (ratolí, teclat, etc) adientment en aquest nou paradigma

Un cop hi hagi algun "compositor" Wayland en marxa, qualsevol programa gràfic s'hi podrà renderitzar "a sobre" (és a dir, mostrar-se visible a la pantalla, com si aquesta fos un "llenç", comunicant en tot moment al "compositor" qualsevol actualització en la seva aparença -moviment de cursor, pulsació de botó, redimensionament/tancament/desplaçament de finestra, etc- per tal de què aquest respongui amb el refresc de pantalla adient), a més de poder rebre atendre els events rebuts dels diferents dispositius d'entrada reconeguts pel "compositor" (el qual vehicularà aquests events al programa gràfic en qüestió); és per això que als tots els programes gràfics mostrats sobre un "compositor" Wayland s'anomenen genèricament "clients Wayland".

Wayland (https://wayland.freedesktop.org), en realitat, no és més que un protocol que defineix com s'ha de comunicar el "compositor" amb els "clients" que el vulguin fer servir (i viceversa), però més enllà d'això, cada "compositor" pot implementar la funcionalitat extra pròpia que vulgui (les anomenades "extensions"). En aquest sentit, ens trobem amb diversos "compositors" Wayland que ofereixen capacitats similars però no exactament iguals (caldrà, doncs, llegir la "lletra petita"): **Mutter** (el "compositor" sobre el qual es basa Gnome), **Kwin** (sobre el qual es basa KDE), **Weston** (la implementació oficial d'un "compositor" Wayland de referència, desenvolupat pel propi projecte Wayland), **Sway** ("compositor" de tipus "tiling", basat en la llibreria "wlroots"), **Wayfire** (un altre "compositor" basat en "wlroots" però no "tiling"), etc. Tots ells permeten obrir/tancar/minimitzar/maximitzar/moure /redimensonar finestres i (des)activar-lis el focus, gestionar "workspaces" i diverses configuracions del monitor, reconéixer combinacions de tecles, etc; entre les "extensions" més comunes, en canvi, podem trobar la capacitat de gravar vídeo de la pantalla i/o fer-ne captures, oferir la possibilitat d'emprar un "escriptori remot", etc.

> NOTA: Cal precisar que en el cas d'una aplicació client Wayland, la tasca de comunicar-se amb el "compositor" normalment recau en la llibreria gràfica amb la qual s'ha desenvolupat l'aplicació en qüestió (GTK+ en cas de programes Gnome, Qt en cas de programes KDE, etc)

NOTA: Teniu una llista més completa de diferents "compositors" a https://wiki.archlinux.org/title/Wayland#Compositors En general, tots ells tenen com a dependències la llibreria "libinput" (per gestionar els dispositius d'entrada) i KMS (mòdul del kernel que implementa a baix nivell el mode gràfic del sistema). D'altra banda, una llista molt més completa on no només apareixen "compositors" Wayland sinó tot un conjunt de llibreries i tota mena d'aplicacions compatibles amb aquest protocol es troba a https://github.com/natpen/awesome-wayland
NOTA: Cal saber que existeixen altres tipus de tecnologies que permeten implementar un entorn bàsic (com és el cas, per exemple dels anomenats "servidors X" o "X11). No obstant, cada cop s'utilitzen menys en favor de Wayland. En tot cas, si en algun moment volguéssim confirmar quina tecnologia estem fent servir, es pot consultar el valor de la variable **XDG_SESSION_TYPE**, el qual serà "wayland" si tenim funcionant un "compositor" Wayland
NOTA: Per saber més sobre el protocol Wayland, aconsello llegir https://wayland-book.com/introduction.html

EXERCICIS:

A continuació farem servir el compositor Weston per tal de desplegar un "display" (un "llenç") on s'hi podran mostrar programes gràfics. Tant se val si estàs en una distribució de tipus "Server" (terminal) o "Desktop" (gràfica); en el primer cas, Weston implementarà el "display" directament sobre el dispositiu-pantalla real (mitjançant el subsistema gràfic KMS/DRM del kernel i el mòdul "evdev" -a través de la llibreria/wrapper "libinput", responsables de gestionar els dispositius I/O) mentre que en el segon cas l'implementarà en forma de dispositiu-pantalla "fictíci" funcionant sobre de la pantalla real (on ja estarà funcionant, de fet, un altre compositor -que en el cas d'estar, per exemple, en un entorn Gnome, seria Mutter-), resultant així una espècie de cascada, doncs: *aplicació_gràfica --sobre--> weston --sobre--> mutter --sobre--> hardware* En qualsevol cas, els passos són pràcticament els mateixos:

1.- Arrenca una màquina virtual qualsevol (tant se val que sigui de tipus "Server" o "Workstation", "Fedora" o "Ubuntu") i instal·la-hi el paquet "weston". A partir d'aquí:

a) Executa en un terminal qualsevol la comanda següent: *weston &*

NOTA: En el cas de què el nostre sistema no tingui el dimoni "systemd-logind" funcionant, llavors caldria executar la comanda *weston-launch &*, la qual gestiona el correcte accés privilegiat als dispositius hardware com el monitor, la tarja gràfica i els dispositius d'entrada (en el cas de que estigui funcionat "systemd-logind", d'aquesta tasca Weston se'n podria deslliurar -si està compilat adientment, cal dir també- ja que li delegaria a "systemd-logind" a través de crides D-Bus)
NOTA: Les finestres Wayland no poden ser redimensionades: només estan disponibles en resolució 1024x768
NOTA: En el cas de treballar dins de contenidors, per posar en marxa Weston caldrà tenir-hi prèviament instal·lat el paquet "udev" (per a què Weston pugui reconèixer hardware d'entrada com el ratolí) i en el cas de Fedora, a més, el paquet "mesa-dri-drivers" (per a què Weston pugui reconèixer hardware de sortida com la tarjeta gràfica). Aquest paquets ja es troben instal·lats en qualsevol sistema amfitrió però dins dels contenidors no sol estar per defecte.

Veuràs que se't mostrarà a la pantalla un entorn gràfic format per un fons de color, el cursor del ratolí i una barra superior on apareixerà un accés directe a un terminal (tot això és gràcies, però, a incorporar la llibreria "weston-desktop-shell", que en principi és prescindible). A nivell intern, aquest entorn gràfic es renderitza sobre un "display" lògic creat en aquell mateix moment pel propi "compositor" Weston en arrencar. Depenent de si has executat Weston en un terminal virtual (és a dir, en un"mode text" pur on, per tant, Weston serà l'entorn gràfic primeri fonamental) o bé en un terminal visualitzat en forma de finestra dins d'un altre entorn gràfic "amfitrió" (és a dir, sobre un "compositor" Wayland ja existent, amb la qual cosa Weston crearà un nou "display" a sobre del que hi havia), Weston tria automàticament utilitzar de forma interna un "backend" diferent ("drm-backend" en el primer cas i "wayland-backend" en el segon).

NOTA: Tot i que Weston és prou intel.ligent per escollir "on-the-fly", segons l'entorn on s'executi, el millor tipus de "backend" sobre el qual es mostrarà, aquest es pot especificar un explícitament amb el paràmetre *-B*. Valors possibles per aquest paràmetre són, a més dels comentats "drm-backend" i "wayland-backend", "fbdev-backend" (alternativa al primer però menys versàtil i capaç), o "rdp-backend" (per quan volem que Weston no creï el "display" en la màquina local sinó en màquines remotes a través del protocol RDP; aquest backend necessita que s'especifiquin a més els paràmetres *--address* i *--port*), entre d'altres.

A la vegada que Weston crea un "display" (que en el cas d'utilitzar el "drm-backend" s'anomenarà normalment "wayland-0" i en el cas d'utilitzar el "wayland-backend" tindrà un nom com "wayland-nº", on "nº" serà un número a partir d'u en endavant), també assignarà automàticamnent aquest nom a una variable d'entorn anomenada **WAYLAND_DISPLAY.** Aquesta variable és molt important perquè el seu valor és el que (per defecte i a no ser que s'indiqui explícitament un altre valor) els clients Wayland consultaran sempre en iniciar la seva execució per saber en quin "display" s'hauran de renderitzar

NOTA: Es pot indicar un nom personalitzat al "display" creat per Weston en executar-se (en comptes del nom per defecte) simplement afegint el paràmetre *-SnomNouDisplay* (així, sense espai entremig): *weston -SnomNouDisplay &*
NOTA: En tot cas, tingui un nom personalitzat o no, el "display" "físicament" es troba en forma de "socket" local dins d'una carpeta la ruta de la qual està indicada a la variable **XDG_RUNTIME_DIR** (i que normalment sol ser "/run/user/nºUID")

b) Executa la comanda *echo $WAYLAND_DISPLAY* dins del terminal que ofereix el propi Weston a través d'un accés directe visible a la barra superior. ¿Quin valor veus? ¿I si executes la mateixa comanda en un terminal virtual de la màquina amfitriona (CTRL+Fx)? ¿I si executes la mateixa comanda dins d'un terminal "gnome-terminal" (suposant que tens iniciada una sessió dins de l'escriptori Gnome)? ¿Per què?

Com hauràs comprovat a l'apartat anterior, el més habitual és que si estem treballant dins del terminal obert dins de Weston el valor de WAYLAND_DISPLAY ja estigui assignat automàticament (essent o bé "wayland-0" en el cas de què Weston funcioni directament sobre el hardware gràfic o bé "wayland-nº" en el cas de funcionar sobre un altre "compositor" Wayland amfitrió) i, per tant, que els clients ja "tinguin clar" a quina pantalla hauràn de mostrar-se. No obstant, si estem treballant des d'un terminal fora de qualsevol sessió Wayland (o des de dins d'un contenidor, que també podria ser un altre cas similar), la variable WAYLAND_DISPLAY no estarà definida i per tant, caldria establir en aquest cas el seu valor "a mà" per tal de que a partir de llavors tots els clients (o un altre "compositor" Weston niat a sobre) que volguéssim executar des d'aquell terminal es poguessin renderitzar a la pantalla escollida. Això es pot aconseguir simplement executant la comanda *export WAYLAND_DISPLAY=nomDisplay* (on "nomDisplay" ha de ser el nom del "display" desitjat on es renderitzaran les aplicacions gràfiques). Òbviament, aquest "display" haurà d'estar ja disponible gràcies a l'execució permanent del corresponent "compositor" Wayland.

c) Executa una aplicació gràfica qualsevol (per exemple així: *gedit &*) des d'un terminal fora de Weston (en el cas d'un sistema "Server", això ho pots fer en un altre terminal virtual d'on Weston ha arrencat i veuràs que obtindràs un error perquè et trobes en un entorn de només text; en el cas d'un sistema "Desktop", ho pots fer des de qualsevol terminal pertanyent a l'escriptori amfitrió i veuràs que l'aplicació gràfica es visualitza precisament en aquest escriptori -perquè la variable WAYLAND_DISPLAY això ho indica per defecte-).

cII) Estableix el valor adient de la variable WAYLAND_DISPLAY per tal de què tornis a repetir l'apartat anterior però ara l'aplicació gràfica es visualitzi a l'interior de l'entorn gràfic (el "display") generat pel "compositor" Weston posat en marxa al primer apartat.

NOTA: Si volguéssim mostrar una determinada aplicació en una pantalla Wayland concreta, en comptes d'establir la variable WAYLAND_DISPLAY amb *export* (que afecta a totes les aplicacions executades a partir de la seva definició) podríem escriure directament *WAYLAND_DISPLAY=nomDisplay nomAplicacióGràfica &* (tot en una sola línia); d'aquesta manera només mostraríem l'aplicació gràfica concreta indicada (com ara "gedit") dins de la pantalla "nomDisplay" (generada per Weston, se suposa) però seguiríem mostrant la resta d'aplicacions al display "original" (normalment, *wayland-0*) Una altra manera alternativa seria fer ús del paràmetre *--display* que incorporen moltes aplicacions gràfiques (encara que no totes), així: *nomAplicacióGràfica --display=nomDisplay &*

En el cas d'haver arrencat Weston en un sistema "Server" (o bé, en un sistema "Desktop", després d'haver-lo triat de la llista visible a la cantonada inferior dreta del GDM com l'entorn gràfic principal on entrar en l'inici de sessió; -és a dir, en definitiva, en utilitzar el backend "drm" de Weston-) no hi ha una manera evident de "tancar-lo". En aquest cas, per tancar la sessió Weston (i amb ella, la d'usuari) es pot fer de vàries maneres: es pot executar en un terminal de Weston la comanda *loginctl kill-session nº* (on el número de la sessió d'usuari a indicar es pot consultar havent executat prèviament *loginctl list-sessions*), o bé la comanda *pkill -u usuari* (la qual mata tots els procesos pertanyents a l'usuari indicat) *kill -s 9 -1* (la qual mata tots els procesos excepte el nº1) o bé, el més adient, pulsar simplement la combinació de tecles CTRL+ALT+BACKSPACE

d) Després de llegir el paràgraf blau anterior, prova alguna de les alternatives que se t'ofereixen per tancar una sessió Weston iniciada sobre el backend "drm".

Alguns aspectes relacionats amb el comportament i l'estètica del compositor "Weston" es poden configurar a través d'un arxiu anomenat "**weston.ini**", el qual pot estar ubicat de forma general per tots els usuaris del sistema dins la carpeta "**/etc/xdg/weston**" o bé de forma particular per un usuari en concret dins la carpeta "**$HOME/.config**".

e) Crea un arxiu "weston.ini" particular pel teu usuari amb el següent contingut i torna a posar en marxa el "compositor" Weston (i, si vols, mostra també alguna aplicació gràfica qualsevol en ell), fent servir el backend que vulguis. ¿Què veus que tingui relació amb cadascuna de les línies de configuració següents?

```
[core]
idle-time=5
[libinput]
left-handed=false
[shell]
background-image=/usr/share/backgrounds/gnome/adwaita-day.png
background-type=centered
background-color=0xff00ff00
panel-position=bottom
panel-color=0xffff0000
locking=true
[launcher]
icon=/usr/share/weston/icon_flower.png
path=/usr/bin/gedit
[launcher]
icon=/usr/share/weston/icon_terminal.png
path=/usr/bin/weston-terminal
[keyboard]
keymap_layout=es
```

NOTA: Per saber més opcions d'aquest arxiu de configuració, consultar la pàgina del manual *man weston.ini*
NOTA: Existeixen més programes bàsics de prova per Weston (com l'editor de text *weston-editor*, el visor de fotos *weston-image*, etc) al paquet "weston-demo"
NOTA: Altres "compositors" minimalistes (però no tant com Weston) interessants són el mencionat a la teoria Sway (https://swaywm.org , de tipus "tiling") i el també mencionat Wayfire (https://wayfire.org) , així com Labwc (https://github.com/labwc/labwc , de tipus "stacking"), Waybox (https://github.com/wizbright/waybox, també de tipus "stacking") o Cage (https://github.com/Hjdskes/cage, especialitzat en ser la base d'entorns de tipus "kiosk")

Posada en marxa de diversos entorns gràfics

En el moment d'escriure la contrasenya a la pantalla del Gdm apareix un botó a la cantonada inferior dreta de la pantalla que, en clicar-lo, mostra un llistat dels escriptoris disponibles, a triar per iniciar sessió.

1.-a) En sistema Ubuntu o Fedora amb entorn gràfic (corrent en una màquina virtual) instal·la l'escriptori "LXQt" (llegeix la nota següents per saber com fer-ho) i tot seguit tanca sessió i entra a la sessió anomenada "LXQt Desktop". ¿Què veus?

NOTA: Per instal·lar un escriptori sencer (com per exemple el "LXQt" demanat), a Fedora ho pots fer executant la comanda *sudo dnf group install "Escriptori LXQt"* (on els grups de paquets disponibles -entre els quals es troben els grups corresponents a escriptoris complets- es poden llistar amb la comanda *dnf grouplist*); a Ubuntu només caldrà, en canvi, que facis això: *sudo apt install lxqt* (Ubuntu també ofereix "grups de paquets" però només per les seves variants oficials: "kubuntu-desktop", "xubuntu-desktop", "lubuntu-desktop", etc-)

El llistat en el botó de la cantonada inferior del Gdm (i, de fet, de qualsevol altre "display manager") s'omple a partir de l'existència de diferents arxius ".desktop" ubicats dins de la carpeta "/usr/share/xsessions" (per escriptoris que encara funcionin sobre X11) o "**/usr/share/wayland-sessions**" (per escriptoris que funcionin sobre Wayland), cadascun dels quals es correspon amb un d'aquests escriptoris disponibles (s'instal·len amb ells). Com a administradors podem afegir o editar aquests arxius ".desktop" segons el que ens convingui per tal de generar més sessions a escollir. El contingut d'aquests arxius és l'estàndar: línies *Name=*, *Type=*, *Exec=*... on, per exemple, en el cas de Gnome, a la línia Exec= se sol indicar la comanda *gnome-session*, programa que arrenca tot l'escriptori Gnome en conjunt (gestor de finestres, gestor de fitxers, icones, etc).

b) Raona la ubicació del fitxer "/usr/share/wayland-sessions/weston.desktop" i el significat del seu contingut

El darrer escriptori sel·leccionat d'entre els disponibles serà a partir de llavors l'escriptori per defecte on l'usuari hi entrarà totes les vegades, a no ser que el torni a canviar un altre cop explícitament a la llista mostrada a la pantalla del "display manager". Aquesta "memòria" és possible gràcies a que el nom de l'arxiu "desktop" (dins de "/usr/share/wayland-sessions") corresponent a l'escriptori per defecte en aquell moment per cada usuari en concret es guarda com a valor de la línia *Session=* de l'arxiu "**/var/lib/AccountsService/users/*nom_usuari***"

c) Fes que l'escriptori per defecte (fins nou canvi) en iniciar sessió el teu usuari sigui l'escriptori Gnome editant l'arxiu mencionat al final del paràgraf blau anterior.

Veuràs que en aquest arxiu particular per cada usuari hi han més característiques interessants, com ara la línia *Icon=* (per establir la ruta de la imatge que s'establirà com a icona de l'usuari, visible en el Gdm), la línia *Language=* (per establir la localització regional per defecte de l'usuari en iniciar sessió a l'escriptori triat) o *SystemAccount=* (per, si és *true*, fer desaparèixer aquest usuari de la llista d'usuaris mostrats al "display manager"), entre d'altres. Totes aquestes característiques, de fet, són llegides pel Gdm per completar convenientment la seva pantalla presentada als usuaris.

cII) Estableix la locale "en_US.UTF-8" i una icona qualsevol pel teu usuari editant manualment l'arxiu "/var/lib/AccountsService/users/*usuari*" corresponent i prova-ho reiniciant la màquina (o alternativament, el "display manager").

Els arxius ubicats sota la carpeta "/var/lib/AccountsService/users" (i "/var/lib/AccountsService/icons") són gestionats pel dimoni "accounts-daemon", que hauria d'estar encès permanentment. Aquest dimoni ofereix a més, tal com vam veure al seu moment, un servei D-Bus per poder interaccionar-hi a través d'aquest canal.

cIII) Executa la comanda *busctl get-property org.freedesktop.Accounts /org/freedesktop/Accounts/User1000 org.freedesktop.Accounts.User LoginFrequency* ¿Quina dada obtens?

> NOTA: Aquesta dada és justament la que fa servir Gdm per ordenar la llista d'usuaris mostrada: el primer és el que entra més freqüentment al sistema (si hi ha empat, llavors l'ordenació la fa alfabèticament)
> NOTA: Sabent el que ja sabem, ara podríem executar l'apartat c) d'aquest exercici executant la comanda: *busctl call org.freedesktop.Accounts /org/freedesktop/Accounts/User1000 org.freedesktop.Accounts.User SetSession s gnome*

cIV) ¿Quina funcionalitat introdueix el "commit" següent dins del servei "accounts-daemon": https://gitlab.freedesktop.org/accountsservice/accountsservice/-/merge_requests/77 ?

Imaginem ara que tenim un sistema de tipus Server però volem iniciar-hi un sistema gràfic mínim (concretament, Weston, possiblement personalitzat amb llençadors, però podria ser qualsevol altre) per tal de treballar una mica més còmodament. A més d'instal·lar el propi Weston, necessitaríem fer dues coses més:

> **1)** Instal·lar un "display manager" per tal de no haver d'executar Weston manualment. És a dir, sense "display manager" continuaríem tenint *login* com a gestor d'inici de sessions i això vol dir que seguiríem loguejant-no en un terminal virtual (i per tant, hauríem d'arrencar Weston executant manualment la comanda *weston* al terminal de treball)
>
> **2)** Fer que "graphical.target" sigui el "target" per defecte de la màquina. Això és perque encara que tinguem algun "display manager" instal·lat i el servei "display-manager.service" habilitat, aquest no es posarà en marxa si no s'arriba a aquest "target". Això es pot aconseguir de forma permanent fent *sudo systemctl set-default graphical.target* i reiniciant o bé, de forma temporal per només l'arranc actual, indicant el següent paràmetre del kernel a la configuració del gestor d'arranc: *systemd.unit=multi-user.target* (d'aquesta manera, aquesta modificació es fa "on the fly" i no es canvia el target per defecte del sistema)

2.-a) Arrenca una màquina virtual de tipus Server (Ubuntu o Fedora, és igual) i instal·la-hi, a més del paquet "weston", el paquet "emptty" (https://github.com/tvrzna/emptty), un "display manager" molt simple i minimalista, just el que necessitem per un servidor. En concret, executa el següent:

sudo dnf install golang make gcc dbus-x11 libX11-devel pam-devel #Dependències a Fedora
sudo apt install golang make gcc dbus-x11 libx11-dev libpam0g-dev #Dependències a Ubuntu
git clone https://github.com/tvrzna/emptty
cd emptty && make build
sudo make install #Instal·la el binari emptty
sudo make install-pam-debian #Instal·la la configuració "/etc/pam.d/emptty" per Ubuntu
sudo make install-pam-fedora #Instal·la la configuració "/etc/pam.d/emptty" per Fedora
sudo make install-config #Instal·la l'arxiu de configuració general d'Emptty
sudo make install-systemd #Instal·la el fitxer "service"

NOTA: Si no instal·léssim l'arxiu "service", només podríem executar Emptty un cop haguem iniciat sessió amb l*ogin*, cosa que faria que no tingués gaire gràcia tot plegat, tal com ja hem dit

b) Modifica les següents directives presents a l'arxiu de configuració general d'Emptty ("/etc/emptty/conf") per a què el seu valor sigui el següent (la resta de directives les pots deixar tal qual estan definides) i digues per a què serveixen cadascuna d'elles (pots trobar la resposta a la seva pròpia pàgina web, molt documentada):

TTY_NUMBER=2
SWITCH_TTY=true
PRINT_ISSUE=false
AUTOLOGIN=false
DEFAULT_USER=usuari #Només es té en compte si AUTOLOGIN=true
FG_COLOR=YELLOW
BG_COLOR=BLUE

NOTA: Per a què la directiva DEFAULT_USER funcioni, a més de què ha d'haver la línia *AUTOLOGIN=true* l'usuari en qüestió ha de pertànyer a un grup anomenat "nopasswdlogin", el qual s'haurà de crear manualment si no existeix. És a dir, cal procurar fer abans *sudo groupadd nopasswdlogin* i *sudo usermod -a -G nopasswdlogin usuari*
NOTA: Els colors possibles per les directives FG_COLOR i BG_COLOR són: BLACK, RED, GREEN, YELLOW, BLUE, PURPLE, CYAN, WHITE, LIGHT_BLACK, LIGHT_RED, LIGHT_GREEN, LIGHT_YELLOW, LIGHT_BLUE, LIGHT_PURPLE, LIGHT_CYAN, LIGHT_WHITE

bII) Crea l'arxiu "$HOME/.config/emptty" amb el següent contingut (llegeix les notes inferiors per saber per a què serveix):

Name=Hola
Exec=/usr/bin/weston
Environment=wayland
Lang=es_ES.UTF-8

NOTA: Aquest arxiu indica l'entorn gràfic preferit per l'usuari en qüestió. Si no existís, Emptty mostraria un menú per escollir l'entorn gràfic desitjat a cada inici de sessió, d'entre els existents sota la carpeta "/usr/share/wayland-sessions"
NOTA: La directiva *Environment* indica el sistema gràfic emprat: només pot valer "wayland" o l'alternativa obsoleta, "x11"

c) Habilita el "display manager" de la manera habitual (*sudo systemctl enable emptty*) i fer que el "target" per defecte sigui "graphical" (*sudo systemctl set-default graphical.target*). Tot seguit reinicia la màquina. ¿Què veus? ¿Què passa si inicies sessió? ¿I si tanques Weston amb CTRL+ALT+BACKSPACE?

d) Torna a fer que l'arranc acabi al target "multi-user" i reinicia la màquina de nou. ¿Què passa ara? Logueja't dins d'un terminal virtual i llavors executa la comanda *sudo systemctl isolate graphical.target*. ¿Què passa?

NOTA: Un altre "display manager" minimalista que funciona en mode text és "Ly" (https://github.com/fairyglade/ly)

La comanda *gnome-session* serveix per arrencar qualsevol tipus d'escriptori Gnome en general, però cal saber que Gnome Shell disposa dels anomenats "modes", els quals serveixen per definir els estils CSS a utilitzar, les extensions a activar, la ubicació espacial de la barra d'icones...i en general, la configuració que conforma l'entorn del Shell. D'aquesta manera, depenent del mode escollit, la mateixa comanda *gnome-session* pot arrencar un escriptori Gnome que no s'assembli res a un altre. Per saber quin mode ha de fer servir, en executar-se la comanda *gnome-session* interpreta el valor de la variable d'entorn GNOME_SHELL_SESSION_MODE, valor que haurà de correspondre's amb el nom d'un determinat mode, definit dins d'un arxiu anomenat "nomMode.json" i ubicat dins de la carpeta "**/usr/share/gnome-shell/modes**"

> NOTA: Per saber els modes de Gnome Shell existents al sistema, a més de poder fer simplement *ls /usr/share/gnome-shell/modes* es pot fer també *gnome-shell –list-modes*
> NOTA: Hem de saber que Gnome també disposa del que se'n diuen "modes", que no són més que conjunts personalitzats de components de baix nivell que s'hi volen implementar en la sessió (un "compositor", un gestor d'icones, una gestor de barra de tasques, un gestor de fitxers, etc). Per triar el mode concret que es desitja, la comanda *gnome-session* disposa del paràmetre *--session* al qual li hem d'assignar com a valor el nom d'un arxiu de tipus semblant als arxius ".desktop" però amb extensió ".session" (i ubicat a "**/usr/share/gnome-session/sessions**" o bé a "**~/.config/gnome-session/sessions**"), que contingui només la capçalera "[GNOME Session]" i dues directives: *Name=* (indicant el nom de la sessió, inventat) i *RequiredComponents=* (indicant la llista de components desitjats a incloure). No obstant, aquesta personalització ja és de nivell molt avançat i no hi entrarem.

3.-a) Arrenca la mateixa màquina virtual amb l'entorn Gnome amb què has treballat als exercicis anteriors i observa el contingut del seu arxiu "/usr/share/gnome-shell/modes/classic.json" i compara'l amb l'altre arxiu JSON que hauràs de trobar en la mateixa carpeta corresponent al tema Gnome Shell instal·lat en fer l'apartat bII) de l'exercici 3 de la secció sobre GDM. ¿Quines diferències hi veus?

b) Observa el valor de la línia *Exec=* present a l'arxiu "/usr/share/wayland-sessions/gnome-classic.desktop" i compara'l amb el valor de la línia *Exec=* present a l'arxiu "/usr/share/wayland-sessions/gnome.desktop". ¿Quina diferència hi veus?

> NOTA: A https://discourse.gnome.org/t/gnome-shell-single-application-mode-aka-kiosk-mode/3629 s'explica com crear un "mode" de Gnome Shell personalitzat propi. Es deixa com exercici extra.

La comanda *gnome-session-quit* , en canvi, tanca una sessió Gnome per comandes. En concret, té els paràmetres *--logout* (per tancar la sessió, preguntant primer), *--power-off* (per apagar la màquina, preguntant primer), *--reboot* (per reiniciar la màquina, preguntant primer), *--force* (per fer les accions anteriors encara que hi hagi algun programa en marxa) i *--no-prompt* (per no preguntar, només funciona amb *--logout*).

4.-a) Arrenca una màquina virtual que incorpori l'entorn Gnome i crea un accés directe al teu escriptori que tanqui la sessió actual sense preguntar

b) ¿Què passa si, mentre tens una sessió Gnome oberta, inicies sessió en paral·lel a la mateixa màquina amb el mateix usuari però mitjançant SSH i llavors executes la comanda *gnome-session-quit –no-prompt –logout*? ¿I si la connexió SSH la fas amb un altre usuari diferent (o bé, posteriorment, mitjançant *sudo -i* o similar), fins i tot, "root"?

Gestió dels sessions d'usuari

"Systemd-logind" és un dimoni del sistema que gestiona els logins dels usuaris. Concretament, és el responsable d'iniciar automàticament els programes *agetty->login* a partir de l'activació del terminal virtual *ttyX* respectiu (o de preparar l'execució del "display manager" per defecte, segons el "target" on s'hagi arribat en aquell terminal), així com de fer un seguiment continu de totes les sessions d'usuari iniciades (i tots els arxius temporals associats i tots els seus procesos fills respectius -continguts de sengles "control groups"-) incloent el control dels canvis entre sessions, de gestionar les eventuals configuracions "multi-seat" i el corresponent accés als dispositius per part dels usuaris,, d'implementar les inhibicions de suspensió/hibernació/apagada (a través de polítiques PolicyKit) així com el control dels botons hardware respectius, etc.

El seu fitxer de configuració és "**/etc/systemd/logind.conf**", on les directives més interessants són:

NAutoVTs=nº : Nombre de terminals virtuals disponibles. If you want all [Fx] keys to start a getty, increase the value of NAutoVTs to 12. Remember you can also pre-activate gettys which will be running from boot simply doing *systemctl enable getty@tty9.service*

StopIdleSessionSec=nº : Tanca una sessió sense activitat passat el nº de segons indicat

KillUserProcesses=yes : Configures whether the processes of a user should be killed when she or he completely logs out (i.e. after her/his last session ended). Defaults to "no" because of popular using of *tmux* and similar software. You can specify a list of usernames affected by this directive with *KillOnlyUsers=* directive (or, alternatively, not affected by it with *KillExcludeUsers=* **.** Former defaults to empty list and latter (which has precedence in case of conflict) defaults to "root"

IdleAction= { ignore | poweroff | reboot | suspend | hibernate | lock } : Configures the action to take when the system is idle. Defaults to "ignore". The system will execute the action after all user sessions report that they are idle, no idle inhibitor lock is active, and the time configured with *IdleActionSec=* has expired (this last directive configures the delay after which the action configured in *IdleAction=* is taken after the system is idle)

HandlePowerKey= Triggered when the power key/button is pressed. Defaults to "poweroff"
HandleSuspendKey= Triggered when the suspend key/button is pressed. Defaults to "suspend"
HandleHibernateKey= Triggered when the hibernate key/button is pressed. Defaults to "hibernate"
HandleLidSwitch= Triggered when the lid is closed. Defaults to "suspend"
The specified action for each event can be the same as in *IdleAction=* directive

Comandes interessants són *loginctl list-users* **,** *loginctl list-sessions* **,** *loginctl list-seats* **,** *loginctl show-user nomUsuari* o *loginctl show-session nº* (on el número corresponent a la sessió actual, si és gràfica, és el valor de la variable **XDG_SESSION_ID).** També, per sistemes "multiseat", poden ser útils les comandes *loginctl seat-status seatX* o *loginctl attach seatX /sys/devices/.../videoCard*

EXERCICIS:

1.-a) ¿Què mostra la comanda *loginctl list-sessions*? ¿Què mostra la comanda *loginctl show-session X -p Type* (on "X" és algun número de sessió mostrat a la primera comanda?

b) ¿Què fa la comanda *loginctl lock-session X*? (el verb contrari seria *unlock-session*)

c) ¿Què mostra la comanda *loginctl list-users*? ¿Què mostra la comanda *loginctl show-user usuari*?

d) ¿Què passaria si, en un sistema corrent sobre un portàtil, modifiquessis la línia *HandleLidSwitch=* present a l'arxiu /etc/systemd/logind.conf" per a què valgués "poweroff"?

La instància Systemd de cada usuari (*systemctl --user*) arrenca després del primer inici de sessió de cada usuari i s'atura després del darrer tancament de sessió de l'usuari en qüestió. Això vol dir que si iniciem algun "service" o "timer" com a usuari, aquest "service/timer" s'aturarà en sortir de la sessió. Sovint, però, ens interessarà mantenir el "service/timer" funcionant tot i haver tancat les sessions d'usuari (és a dir, mantenir la instància Systemd per aquell usuari encara que no tingui cap sessió oberta). Per aconseguir això, cal executar (un cop) la comanda *loginctl enable-linger nomUsuari* (per desfer-ho, seria *loginctl disable-linger nomUsuari*).

NOTA: En realitat, la comanda *enable-linger* el que fa és crear un fitxer buit amb el nom de l'usuari dins de la carpeta "/var/lib/systemd/linger"

e) Després de llegir el paràgraf blau anterior, digues un cas concret on la comanda *loginctl enable-linger* ... podria ser útil