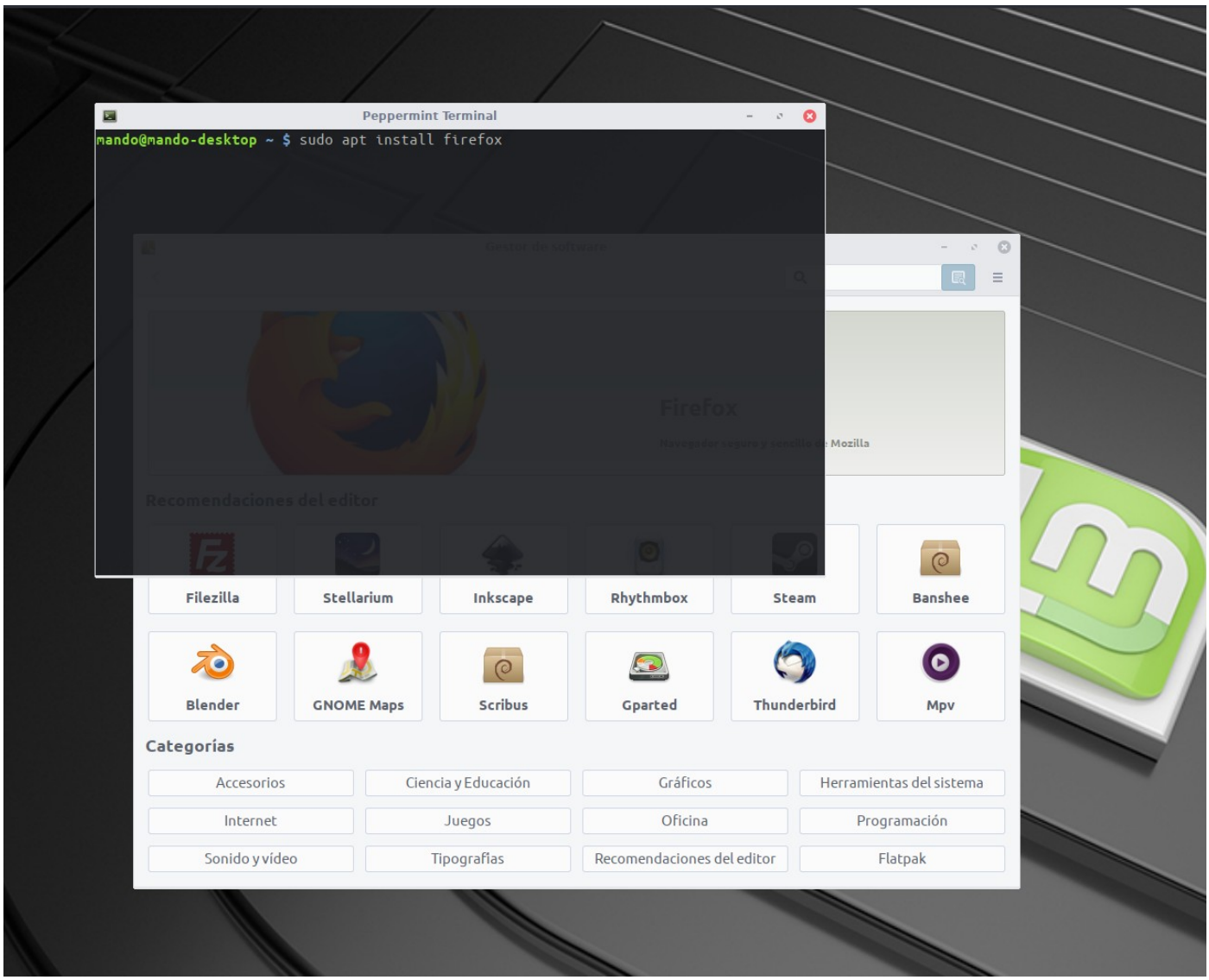
Instalar software en Linux usando la Terminal y otros métodos

Instalar software en Linux usando la Terminal es más sencillo de lo que te imaginas. Sin embargo, los comandos utilizados pueden variar un poco dependiendo del Gestor de Paquetes en tu sistema operativo.

En cualquier caso, a continuación te mostraremos las diferentes formas de instalar software en Linux, usando la Terminal y otras herramientas. Antes de eso, hablemos primero de los gestores de paquetes.

¿Qué es un Sistema de Gestión de Paquetes?

Básicamente es un conjunto de herramientas y formatos de archivos, que se utiliza para instalar, actualizar y desinstalar software en Linux. Actualmente los dos sistemas de gestión de paquetes más comunes son Red Hat y Debian.



En cuanto a Fedora, Mandriva, CentOS y otras distribuciones, utilizan el sistema rpm de Red Hat, (archivos .rpm). Por su parte, Ubuntu, Linux Mint, Peppermint, entre otros, utilizan el sistema dpkg de Debian, (archivos .deb).

Sin embargo, la principal diferencia entre estos dos gestores de paquetes es la forma en que se instala y se mantiene el software.

Instalar software en Linux Mint, Ubuntu, Peppermint y otros (Debian)

Realmente hay dos **formas de instalar programas en Linux desde la Terminal** usando Debian. Es decir, puedes usar el programa apt para instalar el software desde un repositorio.

También tienes la opción de utilizar el programa dpkg para instalar aplicaciones desde archivos .deb. Es decir, puedes usar cualquiera de las dos según lo requieras.

Instalar un programa en Linux usando apt:

```
sudo apt install nombre_del_software
```

Desinstalar un programa en Linux:

```
sudo apt remove nombre_del_software
```

Y si solo quieres actualizar el software instalado en Linux, primero tendrás que actualizar el repositorio de aplicaciones. Para ello se utiliza el siguiente comando:

```
sudo apt update
```

Después que se actualice el repositorio, puedes actualizar cualquier programa que lo requiera con este comando:

```
sudo apt upgrade
```

Si únicamente deseas actualizar una aplicación, entonces utilizas esté comando:

```
sudo apt update nombre_del_software
```

También puedes instalar un programa en Linux que hayas descargado en formato .deb. Para ello solo utiliza:

```
sudo dpkg -i nombre_del_software.deb
```

Instalar programas en Linux usando Red Hat

Hay dos comandos que se pueden utilizar en Red Hat para instalar software en Linux desde la Terminal: yum y dnf. Lo puedes hacer así:

```
sudo yum install nombre_del_software

sudo dnf install nombre_del_software
```

Además, el software en formato .rpm también se puede instalar usando el comando rpm:

```
sudo rpm -i nombre_de_software
```

Si deseas eliminar programas no deseados o que no uses:

```
sudo yum remove nombre_del_software

sudo dnf remove nombre_del_software
```

Y para actualizar los programas de Linux ya instalados se utilizan los siguientes comandos:

```
yum update

sudo dnf upgrade --refresch
```

¿Hay otra forma de instalar software en Linux sin usar la Terminal?

Ciertamente la hay, y de hecho es el método para instalar y eliminar programas en Linux preferido por muchos. Principalmente aquellos usuarios que recién comienzan su experiencia en este sistema operativo.

El Gestor de Software o Administrador de Software, es la aplicación desarrollada internamente en Linux Mint. Evidentemente su objetivo es ofrecer un entorno simple e intuitivo para gestionar todos los paquetes de software disponibles.

La ventaja principal del Gestor de Software es su comodidad y facilidad de uso. Es decir, aquí encontrarás todos los programas para Linux, desglosados en categorías, de acuerdo a su uso previsto.

No solo eso, también puedes acceder a descripciones, reseñas y calificaciones para cada uno de los programas. Además, usualmente hay una sección donde aparecen las aplicaciones más destacadas, no solo las recomendaciones del editor.

Por lo tanto, si quieres instalar aplicaciones en Linux desde el Gestor de Software, solo debes hacer lo siguiente:

- Lo primero es navegar hasta el programa que deseas instalar.
- Puedes encontrar la aplicación accediendo a la categoría correspondiente, o también puedes buscar la función de búsqueda.
- Cuando encuentres la aplicación, simplemente haz clic en el botón “Instalar”.
- Después el Gestor de Software se encargará de todo lo demás.

Después, solo debes esperar unos pocos segundos y podrás acceder al programa desde el menú de Linux. Y si lo que deseas es desinstalar una aplicación, el procedimiento es el mismo. Sin embargo, esta vez se mostrará el botón “Eliminar”, ya que se trata de un programa ya instalado.

Gestor de Paquetes Synaptic

Esta es otra forma de instalar software en Linux, aunque no es del todo recomendable para usuarios principiantes. Además, la herramienta permite instalar, actualizar o desinstalar paquetes de software en el sistema.

Por lo general lo encuentras en la sección Sistema, del Menú de Linux. Y ya sea que desees instalar, actualizar o eliminar un programa, tendrás que marcar los paquetes indicados. Después de esto solo tienes que hacer clic en el botón “Aplicar”.

Instalador de paquetes GDebi

Finalmente también es posible instalar software en Linux a través de la herramienta GDebi. Es muy fácil de usar, aunque solo funciona con los paquetes de software en formato .deb. Además y a diferencia de las otras opciones, GDebi no requiere de repositorios, ni tampoco de una conexión a internet.

En otras palabras, la instalación del software se realiza de forma local. En consecuencia solo debes hacer doble clic sobre el paquete de software descargado. Después de esto se te mostrará una pequeña ventana con la descripción del programa y otros detalles.

Para instalar el programa simplemente debes hacer clic en “Instalar paquete”.

equivalencias apt-get, pacman y zypper (Debian, Arch, OpenSuse)

Todo linuxero que se precie instala con frecuencia paquetes en modo consola/terminal, para eso existen los potentes y rápidos programas de gestión de paquetes en modo texto, tanto en Debian (apt-get), Arch (pacman) u OpenSuse (zypper).

Veamos sus equivalencias principales:
APT-GET – DEBIAN / UBUNTU y derivadas:

sudo apt-get update (actualizar la base de datos de los repositorios)

sudo apt-get upgrade (actualizar el sistema)

sudo apt-get install paquete *(instala el paquete)*
sudo apt-get remove paquete *(desinstala el paquete)*
sudo ***apt-cache search paquete*** *(buscar un paquete)*

sudo apt-cache show paquete *(muestra información del paquete)*

PACMAN – ARCH / MANJARO y derivadas:

sudo pacman -Syu (actualizar la base de datos de los repositorios y actualizar el sistema)

sudo pacman -S paquete *(instala el paquete)*

sudo pacman -R paquete *(desinstala el paquete)*

sudo pacman -Rs paquete *(desinstala el paquete y sus dependencias no útiles para el sistema)*

sudo pacman -Ss paquete (busca un paquete específico)

sudo pacman -Sw paquete (descarga el paquete pero no lo instala)

sudo pacman -U /ruta/paquete.pkg.tar.gz (instala un paquete desde una carpeta local)

sudo pacman -Q (muestra la lista de todos los paquetes instalados en el sistema)

sudo pacman -Scc (borra todos los paquetes guardados en la cache de pacman en: /var/cache/pacman/pkg)

ZYPPER – OPENSUSE y derivadas:

sudo zypper update (actualizar la base de datos de los repositorios y actualizar el sistema)

sudo zypper in paquete *(instala el paquete)*

sudo zypper rm paquete *(desinstala el paquete)*

para más información **man apt-get** , **man pacman** , **man zypper**

comandos emerge de Portage en Gentoo, explicados

Portage es otro de los grandes gestores de paquetes de por ejemplo Gentoo. Presenta similitudes con los Ports de BSD y es compatible con POSIX y el entorno python. Además también lo emplea FreeBSD. Para instalar un paquete con él:

emerge es el comando para controlar **Portage**, el ya legendario sistema de administración de paquetes de **Gentoo**. Por eso en Gentoo hablamos de "emerger" un paquete de Portage, que significa: descargar sus fuentes, compilarlo e instalarlo en el sistema.

He aquí los principales comandos de emerge seguidos de una breve explicación.
El directorio principal de configuración de Portage está en: **/etc/portage/**
donde están sus archivos de configuración:
make.conf (archivo de configuración principal de Portage)
package.use (donde se pueden definir las "USE flags" para paquetes individuales)
package.mask (donde se pueden "enmascarar" paquetes para que no se instalen o actualicen)
package.unmask (donde de pueden desenmascarar paquetes enmascarados en Portage para permitir que se instalen)
package.accept keywords (donde autorizar paquetes inestables)
****

sincronizar Portage:

`sudo emerge --sync` actualiza el árbol de Portage que está en: /usr/portage/
`sudo emerge-webrsync` actualiza el árbol de Portage desde la última instantánea de la web de Gentoo
****

buscar en Portage:

`emerge -s paquete` busca el paquete `--search`
`emerge -S palabra` busca también en las descripciones `--searchdesc`

****

instalar paquetes:

`emerge -p paquete` muestra las dependencias del paquete sin instalarlo `--pretend`
`sudo emerge -a paquete` instala el paquete, (-a pide confirmación antes de hacerlo `--ask`)
`sudo emerge -f paquete` descarga el paquete de fuentes pero no lo instala `--fetchonly`

Portage guarda las fuentes en: **/usr/portage/distfiles/**
****

desinstalar paquetes:

`sudo emerge -Ca paquete` desinstala el paquete y sus dependencias (Portage no mira si las dependencias las necesita otro paquete, tampoco desinstala los archivos de configuración (`--unmerge --ask`)
****

actualización básica del sistema:

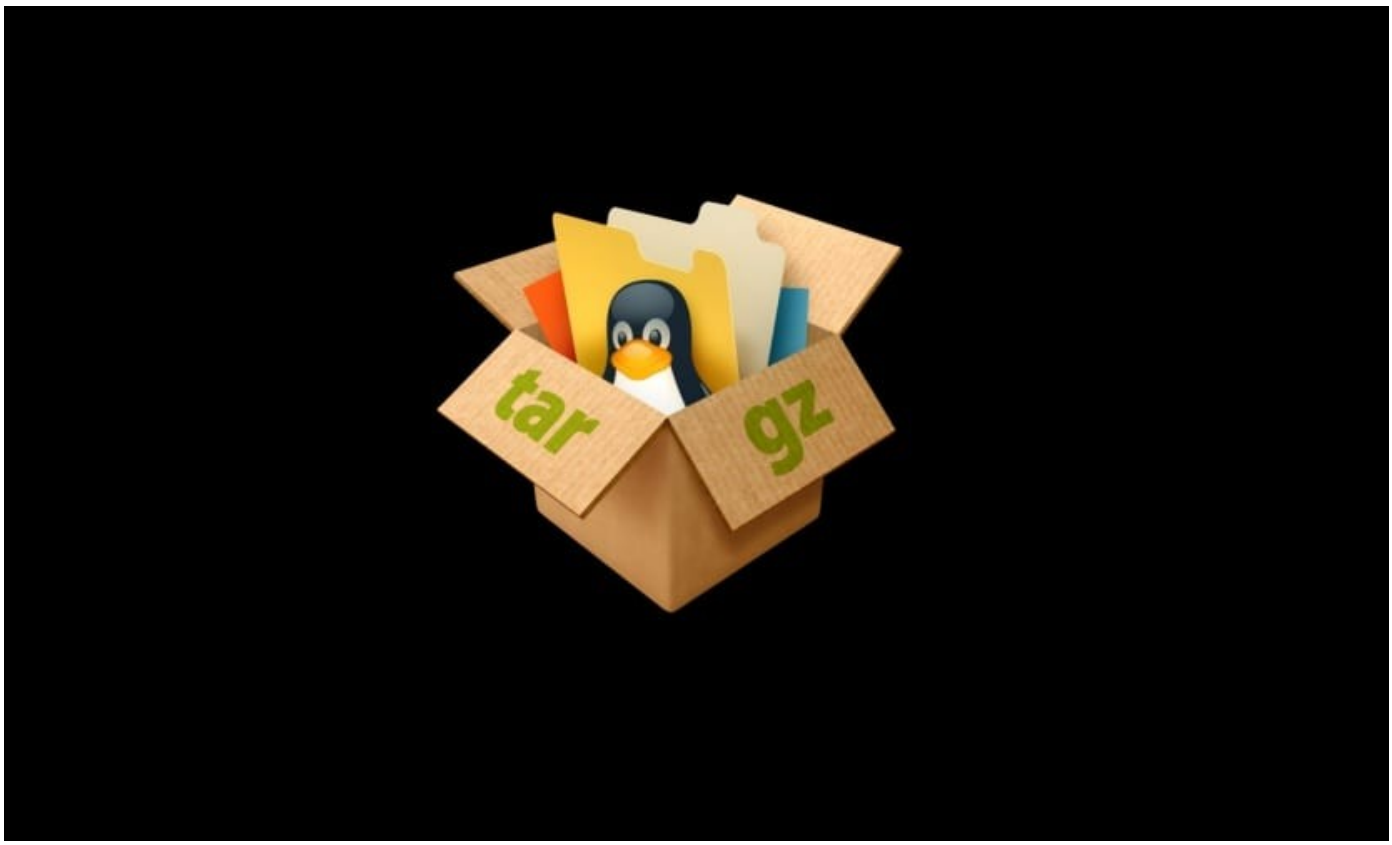
`sudo emerge -ua world` actualiza el sistema (no necesariamente las dependencias (`--update --ask @world`)
`sudo emerge -uaD world` actualiza el sistema incluidas todas las dependencias (`--update -ask --deep @worl`)
****

actualización avanzada del sistema:

`sudo emerge -uaD --with-bdeps=y world` actualiza el sistema incluidas todas las dependencias + ("build dependencies") (`--update --ask --deep --with-bdeps=y @world`)
`sudo emerge -uaDN --with-bdeps=y world` todo lo anterior + revisa por si hay cambios USE.
(`--update --ask --deep --newuse --with-bdeps=y @world`)
****

desinstalar dependencias huérfanas en 3 pasos:

`sudo emerge -uaDN world` (`--update --aks --deep --newuse @world`)
`sudo emerge --depclean`
`sudo revdep-rebuild`
****



Cómo instalar tarballs:

Los paquetes que se instalan directamente desde la fuente se empaquetan con la primitiva, pero aun útil y eficiente, herramienta Tar (de ahí el nombre **tarball**) y luego se comprimen empleando algún tipo de formato comprimido.

Algunos paquetes de este tipo vienen con ficheros en su interior tipo .jar, .bin, .rpm,..., en ese caso tan solo hay que desempaquetar y emplear el procedimiento correcto para el binario que alberga. Pero por lo general es **código fuente** que hay que compilar e instalar. Veamos como. Lo primero, cuando trabajamos **desde la consola**, es situarnos en el directorio donde se encuentra el paquete con el que queremos trabajar. Para ello utilizamos la herramienta "*cd*". Por ejemplo, si te has descargado un paquete y lo tienes en la carpeta Descargas, teclea en el terminal:

```
1cd Descargas
```

Y el **prompt** cambiará con esa ruta para indicarte que estás dentro de este directorio del sistema. También debes recordar que necesitas privilegios para ejecutar ciertas acciones como ./configure, make, o make install… que veremos a continuación.

Instalar tar.gz o tgz:

Estos tipos de tarball es muy empleado en **Slackware y derivados**, aunque se ha extendido para empaquetar código para el resto de distribuciones. Instalar tar.gz es así (recuerda ejecutar ./configure, make y make install con privilegios, ya sabes, como root o anteponiendo sudo al comando…):

```
cd directorio_donde_se_encuentra_el_tarball
tar –zxvf nombre_paquete.tar.gz (o nombre_paquete.tgz, en caso de ser un .tgz)
cd nombre_paquete_desempaquetado
./configure
make
make install
```

Si esto no funcionara para instalar tar.gz, puedes acceder al directorio desempaquetado para comprobar si existe algún fichero de texto con las instrucciones para instalarlo. A veces, cuando no siguen este procedimiento estándar, los desarrolladores incluyen este tipo de ficheros para explicarte las particularidades, dependencias, etc.

Tar.bz2 o .tbz2:

Se trata de un paquete muy empleado **en BSD** y que también se ha extendido a Linux y otros *nix. Es un empaquetado con tar y comprimido utilizando BSD Zip 2. El procedimiento para instalar este tipo de programas es:

```
cd directorio_donde_se_encuentra_el_paquete
tar –jxvf nombre_paquete.tar.bz2 (o nombre_paquete.tbz2, e incluso nombre_paquete.tbz)
cd nombre_directorio_desempaquetdo
./configure
make
```

```
make install
```

Así debería bastar para instalar programas en Linux. Asegúrate que usas **privilegios** para los últimos comandos.

Otros Tape Archive:

En ocasiones se emplea un tape archive o **fichero tar sin compresión**. Este tipo de paquetes mantiene la información necesaria para restaurar totalmente los ficheros que contiene y para desempaquetarlo tan solo hay que hacer esto:

```
tar xvf nombre_paquete.tar
```

Luego busca un fichero con nombre **README.txt** (o similar) dentro del directorio desempaquetado y busca las instrucciones de instalación. Normalmente se trata de hacer un procedimiento similar a los anteriores...

Tar.xz o .xz o .txz:

Ultimamente estoy viendo más de este tipo. Para operar con este tipo de paquetes hay que tener la herramienta **xz-utils** instalada. Para desempaquetarlos e instalarlos se emplea:

```
tar Jxvf nombre_paquete.tar.xz
```

o

```
Xz –d nombre_paquete.tar.xz
Tar –xf nombre_paquete.tar
```

o

```
Unxz nombre_paquete.xz
```

Y una vez descomprimido se busca un fichero **README.txt o INSTALL.txt** para ver los detalles de la instalación, que por lo general es la típica ./configure, make y make install. Aunque a veces puede emplearse cmake.

.gz o .gzip o .bzip2:

Con **GNU Zip** se pueden comprimir paquetes de tipo .gz o .gzip. Estos son tratados de forma similar a los paquetes comprimidos con BSD Zip 2 con extensión .bzip2. Para tratar este tipo de paquetes debemos tener disponibles las herramientas unzip y bunzip2 en nuestro sistema:

```
1gunzip –c nombre_paquete.gz
2bunzip2 nombre_papuete.bz2
```

El resto es **similar a los pasos vistos** con los tarballs anteriores... Asegurate de ver los ficheros README o INSTALL presentes.

.tar.lzma, .tlz:

Tanto si aparece por su nombre largo, .tar.lzma, como si aparece por su nombre corto .tlz, estos paquetes utilizan el algoritmo de compresión Lempel-Ziv-Markov y para extraerlos e instalarlos, debes teclear en la consola (previamente se necesita tener instalado el paquete lzma):

```
1unlzma nombre_fichero.lzma
```

o

```
1lzma -d file.lzma
```

o

```
1tar --lzma -xvf file.tlz
```

o

```
1tar --lzma -xvf file.tar.lzma
```

Dependiendo del formato en el que se nos presente el paquete. Luego puedes mirar algún fichero de texto en su interior con instrucciones o seguir los pasos que hemos descrito para instalar los otros tarballs (./config, make, make install). Otra buena práctica es mirar en la **web**

del desarrollador, donde aparecen tutoriales de como se instalan los paquetes o existen sitios Wiki con multitud de información.

*Nota: también puedes instalar ciertos paquetes empaquetados con una herramienta denominada **installpkg.***

Cómo instalar paquetes binarios:

.jar:

Para instalar **paquetes java** es bastante sencillo. Los requisitos son evidentes, tener instalada la máquina virtual Java de Oracle (ya sea la JRE o JDK). Para instalarlo debemos hacer clic con el botón derecho del ratón sobre él y seleccionar "*Abrir con otra aplicación*" en el menú desplegable. Aparecerá una ventana con una lista de aplicaciones de nuestro sistema y una línea de formulario abajo para escribir una. Pues en dicho espacio escribes "*java –jar* " sin comillas, incluido el espacio tras jar que he dejado yo. Luego presionas sobre el botón "*Abrir*" y se debería ejecutar sin problema. Como puedes comprobar no es necesario instalarlo.

.bin:

Los podemos ejecutar haciendo doble clic sobre ellos para abrirlos, si previamente le hemos dado **permisos de ejecución**. Para ello haz clic con el botón derecho del ratón sobre el fichero y luego ve a "*Propiedades*" para asignarle permisos de ejecución en la pestaña «*Permisos*». También se puede instalar desde la consola haciendo lo siguiente:

```
1cd directorio_donde_está_el_binario
2./nombre_binario.bin
```

.run:

Para los **.run** procederemos de forma similar a los .bin. Este formato es muy utilizado para drivers, como por ejemplo los AMD Catalyst Center. Para instalarlo se puede usar la consola:

```
cd directorio_donde_está_el_paquete
sh ./nombre_paquete.run
```

Recuerda asignarle previamente permisos de ejecución. Además, algunos necesitan ejecutarse con privilegios, en tal caso hazlo como root o con sudo.

Si quieres instalar el .run **en modo gráfico**, puedes hacer clic con el botón derecho del ratón sobre él y seleccionas “**Propiedades**”, luego en la pestaña “**Permisos**” marcas “**Permitir ejecutar el archivo como un programa**” y aceptas para cerrar. Ahora al hacer doble clic sobre el .run verás que se abre un instalador muy similar a los de Windows (tipo Siguiente, Siguiente, Aceptar…).

Cómo instalar scripts:

.sh:

En Linux también podemos encontrar **scripts con extensiones .sh o los .py**. Para instalar este tipo de scripts nos dirigiremos al directorio donde se cuentra el script con el comando “cd” como hemos visto anteriormente. ¡Ojo! Si el script estuviese empaquetdo, primero desempaquetalo o descomprímelo. Luego, puedes darle permisos de ejecución como ya sabes (puedes hacerlo en modo gráfico o desde el terminal con el comando “*chmod +x nombre_script*» sin comillas). Una vez tengan permisos de ejecución, desde el terminal:

```
1sh nombre_script.sh
```

o

```
1./nombre_script.sh
```

.py:

Para los ficheros con **extensión .py** se debe llamar al interprete del lenguaje de programación Python. Para ello, teclea en la consola esto:

```
1python nombre_script.py install
```

Otros:

Existen otros tipos de ficheros y paquetes para instalar programas en Linux. Ciertos paquetes de BSD, Solaris, Mac OS X, y otros *nix se pueden instalar en Linux. Un ejemplo de ello son los **.pkg de Solaris**. Para instalar los .pkg se puede hacer clic sobre ellos con el botón derecho del ratón, ir a “*Propiedades*” y “*Permisos*” y asignarle permisos de ejecución. Luego haces doble clic sobre ellos para instalarlos.

También existen herramientas como **Alien** para convertir de un formato a otro, por ejemplo de rpm a deb, etc. Esto no es muy

recomendable y en ocasiones puede generar problemas. Así que no te lo recomiendo.

Continuando con el galimatías de los paquetes en Linux, decir que existen más de los vistos aquí, pero son más raros e inusuales. Un ejemplo de rareza es el **.slp** que usan desde el proyecto Stampede Linux. Para transformar .slp en otros formatos más cotidianos puedes emplear Alien (previamente instalado Alien) así:

```
1sudo alien nombre_paquete.slp nombre_paquete.extensión_nueva generated
```

Por ejemplo, para transformar de .slp a rpm:

```
1sudo alien miprograma.slp miprograma.rpm generated
```

Pueden dejar sus comentarios con peticiones, **dudas o comentarios**. Si tienen algún problema siguiendo los pasos de este tutorial, Estaré encantado de ayudaros.

OTRAS FORMAS DE INSTALAR SOFTWARE

PPA: Es un Archivo de Empaquetado Personal. Se utilizan a menudo para instalar versiones más avanzadas de los programas que cuenta en tu sistema y además para instalar nuevos programas que no están en tu paquetería. Los PPA son compatibles con Linux Mint y Ubuntu. La mayoría de los PPA están en Lauchpad, que es una web que plataforma de colaboración de software.

Los PPA se instalan desde la terminal o desde interfaz gráfica. En Linux Mint para añadir un PPA abría que abrir Orígenes de Software desde el Menú u clic sobre el botón PPA.

También, y lo más habitual, es instalar los PPA desde la terminal. Por ejemplo, si quisiéramos instalar Shutter desde un PPA, abriríamos la terminal y añadiríamos la siguiente secuencia de comandos.

SUDO ADD-APT-REPOSITORY PPA:SHUTTER/PPA #AÑADIMOS EL PPA
SUDO APT-GET UPDATE #ACTUALIZAMOS EL SISTEMA PARA QUE RECONOZCA LOS CAMBIOS
SUDO APT-GET INSTALL SHUTTER -Y #INSTALAMOS, LA OPCIÓN -Y EVITA PEDIR CONFIRMACIÓN AL USUARIO

Paquetes .deb: Los paquetes deb son lo más parecido a la forma de instalar programas en Windows. La idea es la misma: buscar, descargar y ejecutar para abrir. En Ubuntu se abrirá el Centro de Software o si tienes instalado otro gestor, ese concreto gestor. En Linux Mint viene por defecto Gdebi, que es un programa que instala los paquetes .deb muy fácilmente. Está disponible también en Ubuntu, de hecho es una de los primeros programas que instalo en Ubuntu. Nota: Para instalarlo en Ubuntu sudo apt-get install gdebi -y desde una terminal.

A veces queremos instalar un programa pero viene en muchos archivos .deb y no sabemos el orden correcto. Como en OpenOffice. Para estos casos es mejor recurrir a la terminal. Nos situamos en el directorio donde están los archivos .deb y ejecutamos el comando sudo dpkg -i *.deb

Código fuente

Muchos desarrolladores disponen sus programas en este formato nativo. Lo que obtenemos, en la mayoría de los casos en un archivo comprimido con extensión tar.gz o similar, al descomprimir tendremos unos archivos con un README, que hay que leer muchas veces ya que contienen las instrucciones para el compilado y otro llamado INSTALL.

Por orden había que hacer:

Descomprimir el código fuente. Desde la terminal o de manera gráfica.
Resolver las dependencias. Usando Synaptic por ejemplo o la terminal.
A continuación deberemos configurar los archivos que nos permitirán compilar el programa.

./CONFIGURE

Luego escribimos:

MAKE

Y finalmente, instalamos con:

MAKE INSTALL

Este proceso tardará dependiendo de la cantidad de elementos a instalar, velocidad del procesador, memoria, uso del sistema, etc.

Archivos sh y .bin

Lo más fácil es abrir una terminal (Ctrl+ Alt+ T) y ubicarnos el al archivo. Una vez que estés en la ruta por ejemplo el archivo está en tu Escritorio.

CD ESCRITORIO

Habrá que otorgarle permisos de ejecución:

Si tenemos un archivo .sh

SUDO CHMOD A+X NOMBRE_ARCHIVO.SH

Si tenemos un archivo .bin

SUDO CHMOD A+X NOMBRE_ARCHIVO.BIN

WINE (Wine Is Not an Emulator)

Si tienes las librerías de Wine en tu sistema podrás ejecutar programas que son nativos de Windows que usan la extensión .exe o .msi. No es objeto de este pequeño tutorial informar de cómo utilizar Wine.

PlayOnLinux

PlayOnLinux es un programa que os permite instalar y usar fácilmente numerosos juegos y programas previstos para correr exclusivamente en Windows® de Microsoft®. No es objeto de este pequeño tutorial informar de cómo utilizarlo.

AppImage

AppImage no es, técnicamente hablando, un administrador de paquetes; tampoco una tienda de aplicaciones. Quizá entendamos mejor lo que ofrece AppImage si recordamos que, **entre 2011 y 2013 se le conoció con el**

nombre de "PortableLinuxApps" (antes de eso, desde su lanzamiento en 2004, se le denominó "Klik").

En resumen, **se trata de una imagen de la aplicación (similar a un archivo .iso)** que no requiere de instalación ni -lo que es más importante- de permisos de administrador para funcionar: ofrece aplicaciones portables, empaquetadas en un único archivo con todas sus dependencias, al margen de la distribución utilizada.

Cuando usemos un archivo .appimage, **sólo deberemos perder tiempo otorgándole permisos de ejecución**, automáticamente se montará en el sistema de archivos del espacio de usuario (lo que lo convierte en compatible con sistemas de archivo inmutables, como los basados en OSTree).

Algunas aplicaciones **ofrecerán la opción al iniciarlas de 'instalar un archivo de escritorio'**, lo que lo integrará en los menús de aplicaciones de nuestra distribución. Sólo en estos casos su "desinstalación" requerirá de algo más que mandar a la Papelera el archivo ejecutable.

Puedes descargar imágenes AppImage desde AppImageHub, pero no existe ningún repositorio centralizado.

Snap

Snap es una iniciativa impulsada por Canonical para su uso en su propia distribución Linux para móviles, la desaparecida Ubuntu Touch, en 2014. Si bien desde entonces se ha exportado a múltiples

distribuciones relevantes (aunque **su excesiva dependencia de Canonical** aún lastra su adopción).

Su objetivo era lograr un formato único de distribución de software que pueda usarse por igual en PCs de escritorio y en dispositivos IoT. Como en el caso de AppImage, **cada Snap empaqueta dentro del mismo todas las dependencias del software en cuestión**, facilitando que funcione en cualquier equipo.

Al contrario que AppImage, los snaps están diseñados para admitir actualizaciones, sin obligarnos a borrar y sustituir el paquete por uno más nuevo. Más aún, esta actualización se lleva a cabo en segundo plano incluso si estamos usando la aplicación.

Es un sistema seguro, porque **los paquetes vienen firmados** y, una vez se empieza a usar la aplicación, ésta lo hace en un entorno aislado, con acceso limitado a los recursos del equipo.

El programa que permite gestionar los snaps es Snapd, y su tienda de aplicaciones, Snapcraft.

Flatpak

El creador de FlatPak, Alexander Larsson, se inspiró en el primer antecesor de AppImage, Klik, para crear un sistema de paquetes centrado en la ejecución del

software dentro de entornos aislados, donde pudieran funcionar sin privilegios de root. Larsson lanzó en 2015 un sistema llamado *xdg-app* que **se terminó convirtiendo en Flatpak un año más tarde, con el apoyo de Red Hat**.

Su principal diferencia con respecto a los otros dos sistemas de gestión de paquetes es que **no empaqueta el software que nos interesa junto a todas sus dependencias**: eso puede obligarnos a instalar numerosos paquetes, pero también **reduce sensiblemente el tamaño de los mismos**.

Otra diferencia es **su alto nivel de integración con los entornos de escritorio KDE y Gnome y con el estándar FreeDesktop**, lo que facilita la integración de las aplicaciones instaladas mediante este sistema con las herramientas gráficas de los principales escritorios de Linux. El apoyo de FreeDesktop, que incluso aloja la web del proyecto Flatpak, le ha granjeado muchos apoyos.

El principal repositorio de paquetes Flatpak es Flathub.org (aunque no depende de ninguna 'tienda de aplicaciones' en concreto).