Distribución de aplicaciones.

Caso práctico

Ministerio de Educación y Formación Profesional (Elaboración propia)

BK programación ha terminado de desarrollar la aplicación de Gestión Hotelera y necesitan gestionar la forma de empaquetar y distribuir la aplicación, para poder ser entregada e instalada por los clientes.

El equipo de BK programación no tiene claro cómo distribuir la aplicación, si crear un paquete ejecutable o utilizar algún software de libre distribución, para crear un instalador que ayude en la instalación en la máquina cliente.

Ada quiere tantear todas las posibilidades de distribución de la aplicación, así que encarga a Juan que cree un paquete con la aplicación y a **María** le pide que pruebe a crear instaladores de la aplicación, usando las herramientas de software libre que hay en el mercado.

Como la aplicación puede tener éxito comercial, **Ada** propone a **Carlos** y a **Ana**, que investiguen las diferentes posibilidades que hay para poder distribuir la aplicación a través de Internet.

Por último, **Ada** quiere que las aplicaciones desarrolladas por BK Programación sean de calidad, por lo que se propone que la aplicación, para poder ser distribuida, sea firmada digitalmente.

1.- Definición y composición de una distribución. Sistema de gestión de paquetes.

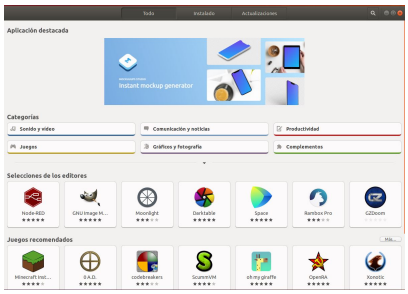
Caso práctico

El equipo de desarrollo de BK Programación se reúne para decidir la forma de distribución de la aplicación de Gestión Hotelera. ¿Cómo hacerlo? ¿Qué patrones hay que seguir?

Ministerio de Educación y Formación Profesional. (Elaboración propia)

Una distribución de software es un conjunto de programas específicos que se presenta compilado y configurado.

De igual forma, las **distribuciones** también hacen referencia a la colección de un conjunto variado de aplicaciones y paquetes de software junto con el sistema operativo, siendo muy común en las distribuciones de Linux, como Debian, Ubuntu, Red Hat, etc.



Montaña Martín Vergel (Elaboración propia)

Cuando nos encontramos con una distribución, normalmente tiene asociada una licencia, siendo la más habitual la licencia **GPL** (General Public License (Licencia Pública General). Es una licencia creada para proteger la libre distribución, modificación y uso de software) u Open source. Otro tipo de distribución presente en el mercado, es la distribución binaria, donde nos encontramos con un instalador (fichero .exe o .msi en sistemas Microsoft Windows), que puede ser descargado desde Internet. Las distribuciones pueden ser oficiales si provienen de los autores o autoras originales o distribuciones 3rd party (software desarrollado por tercera personas o empresas).

Un **sistema de gestión de paquetes** es una colección de herramientas que sirven para automatizar el proceso de instalación, actualización, configuración y eliminación de paquetes de software. Normalmente, este término se usa para referirse a los gestores de paquete en sistema Linux, que los usan como base principal de gestión de paquetes. Como

ejemplo, en Ubuntu nos encontramos el Centro del software o en versiones antiguas el Gestor de Paquetes Synaptic.

En estos sistemas, el software se distribuye en forma de paquetes, que se encapsulan en un único fichero. Dentro del paquete nos encontramos: el software propiamente dicho, el nombre completo del paquete, una descripción de su funcionalidad, el número de versión, el distribuidor del software, la suma de verificación y una lista de otros paquetes requeridos para el correcto funcionamiento del software. Esta información se suele introducir normalmente en una base de datos de paquetes local.

Para saber más

En el siguiente enlace se amplía la información sobre Distribución de Software y Gestores de Paquetes.

Distribución de software y Gestores de Paquetes.

2.- Instaladores. Pasos en la instalación. Asistente de instalación.

Caso práctico

Juan y **María** se van a encargar de preparar la aplicación para poder distribuirla. ¿Qué es necesario en una instalación? ¿Cómo puedo crear instaladores? ¿Qué puede modificar el usuario o usuaria final?

Ministerio de Educación y Formación Profesional
(Elaboración propia)

Un instalador es un programa especial que realiza las tareas de instalación de software de forma automática.

Ministerio de Educación y Formación Profesional
(Elaboración propia)

En la mayoría de los casos, un programa está formado por un conjunto de archivos. Normalmente, esos archivos necesitan ser copiados en determinadas carpetas o directorios, y en muchos casos, deben registrarse en el registro de Windows, si utilizamos ese sistema operativo.

Los instaladores van a realizar todas las operaciones anteriores de forma transparente al usuario. El uso del instalador suele ser muy sencillo. En la actualidad, los instaladores presentan al usuario una serie de formularios donde le van mostrando las indicaciones pertinentes, limitando al usuario a pequeñas modificaciones o directamente a pulsar el botón siguiente. El instalador copiará los archivos de la aplicación a instalar en los directorios adecuados, registrará la aplicación, creará los menús y los accesos directos en el Escritorio.

Ejemplos de instaladores son: InstallAnywhere, InstallBuilder, Windows Installer, InstallShield, InstallAware, Emco Package Builder, Visual Studio 2019, NSIS, Null soft Scriptable Install System, Iexpress, IzPack, etc.

Los pasos en la instalación son los siguientes:

1. **Verificación de la compatibilidad:** se debe comprobar que se cumplen los requisitos para la instalación, tanto hardware como software.
2. **Verificación de la integridad:** se verifica que el paquete de software es el original.
3. **Creación de los directorios requeridos.**
4. **Creación de los usuarios requeridos:** cada grupo de usuarios puede usar un determinado software.
5. **Copia, desempaque y descompresión de los archivos desde el paquete de software.**
6. **Compilación y enlace con las bibliotecas requeridas.**
7. **Configuración.**
8. **Definición de las variables de entorno requeridas.**
9. **Registro de la aplicación** ante el autor o autora de la aplicación.

Los asistentes de instalación son aplicaciones que ayudan al usuario a personalizar la instalación de software. Cuando utilizamos un instalador, normalmente tiene asociado un asistente de instalación. Cada paso de la instalación se muestra mediante un formulario con la información del paso se está dando. Los asistentes nos permiten elegir los directorios donde queremos que se instale la aplicación, el grupo de programas donde se integra la aplicación en el menú del escritorio, información sobre la licencia, registro de la aplicación, etc.

Autoevaluación

¿Qué paso no se realiza en la instalación de un programa?

- ○ Creación de directorios requeridos.
- ○ Verificación de la compatibilidad.
- ○ Compilar el programa.

Incorrecta. Se deben crear los directorios de la instalación.

No es correcta. El software debe cumplir los requisitos software y hardware.

Muy bien. En la instalación, disponemos ya de un ejecutable.

Solución

1. Incorrecto
2. Incorrecto

3. Opción correcta

3.- Paquetes autoinstalables.

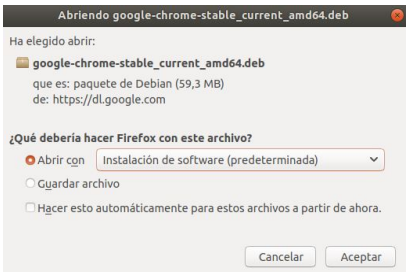
Caso práctico

Juan está probando la creación de un paquete autoinstalable para la aplicación. Él se propone crear un paquete para distribuir la aplicación en sistemas Linux tipo Ubuntu. Desea empaquetar la aplicación en un paquete Debian.

Ministerio de Educación y Formación Profesional
(Elaboración propia)

Cuando se está finalizando el ciclo de desarrollo de una aplicación llega el momento de decidir la mejor manera de distribuirla. No sólo tenemos que decidir la mejor manera de distribuirla, sino que también hay que tener en cuenta añadir características adicionales, parches y revisiones de la aplicación.

Cuando se decide distribuir una aplicación usando un paquete autoinstalable, lo que se está haciendo es empaquetar la aplicación en un único archivo, que contendrá todos los archivos y directorios que forman la aplicación.



Montaña Martín Vergel (Elaboración propia)

Ese fichero será un fichero de ejecutable en Windows (extensión exe), un paquete Debian (fichero con extensión deb) en distribuciones Linux basadas en Debian (Debian, Ubuntu, etc.), un paquete rpm en distribuciones estilo Red Hat (Red Hat, Suse, etc).

Si nos encontramos en Windows, el paquete autoinstalable será una aplicación que, una vez lanzada por el usuario, realizará la descompresión de todos los archivos de la aplicación, creará las carpetas que la aplicación necesita, copiará los archivos a sus directorios de destino, añadirá y/o modificará entradas en el Registro de Windows, añadirá las entradas en el menú de aplicaciones y mostrará accesos directos en el Escritorio.

Durante todo este proceso, el usuario puede interaccionar eligiendo componentes a instalar, modificando los directorios de instalación, si se crean o no accesos directos, o puede optar por las opciones por defecto.

Si estamos distribuyendo una aplicación para Ubuntu, por ejemplo, lo que se crea es un paquete deb. Este tipo de archivo contiene todos los archivos y directorios de la aplicación. Cuando el usuario quiere realizar la instalación del paquete, ejecuta el "Software de Ubuntu", que nos irá mostrando las ventanas de instalación de la aplicación.

4.- Herramientas para crear paquetes de instalación. Repositorios.

Caso práctico

Juan ha creado el paquete deb con la aplicación de Gestión Hotelera, su objetivo ahora es poder añadirlo a un repositorio. Mientras, **María** está analizando el software más conocido para la creación de paquetes de instalación.

Ministerio de Educación y Formación Profesional
(Elaboración propia)

En la actualidad existe un amplio abanico de herramientas de creación de paquetes de instalación. Las más extendidas para crear instaladores son las siguientes: InstallAnywhere, InstallBuilder, Windows Installer, InstallShield, InstallAware, Emco Package Builder, Visual Studio 2019, NSIS, Null soft Scriptable Install System, Iexpress, IzPack, etc.

La instalación de una aplicación es el primer contacto que va a tener un usuario con la aplicación.

Si el proceso de instalación de software es lento o no termina de forma correcta, supone uno de los problemas más irritantes en un ordenador. Un instalador rápido y amigable debe ser parte esencial en cualquier producto software.

Las herramientas de instalación señaladas con anterioridad permiten a los programadores crear instaladores para Windows y algunas también para Linux. Si tomamos como ejemplo NSIS(Nullsoft Scriptable Install System) nos encontramos con una herramienta open-source (software desarrollado y distribuido libremente) que permite crear instaladores para nuestras aplicaciones en Windows. **NSIS** crea instaladores capaces de instalar, desinstalar, configurar el sistema, extraer archivos, etc. Este sistema está basado en scripts, con lo que el programador va a tener el control total en cualquier parte del instalador. El lenguaje script soporta variables, funciones, manipulación de cadenas, como un lenguaje de programación normal, pero diseñado para la creación de instalables.

Si trabajamos en Linux, necesitamos crear un paquete de instalación acorde con la distribución Linux donde queramos instalarlo. Si nos encontramos en Linux Ubuntu, el tipo de paquete será debian (.deb). El paquete de instalación deb, va a empaquetar todos los ficheros que necesita nuestra aplicación y debe de estar configurado para indicar al "Software de Ubuntu", donde se deben copiar esos archivos y que opciones de configuración se deben modificar.

En los sistemas Windows o Mac OS, los programas que queremos instalar se suelen buscar en Internet y se encuentran, en su mayoría, en forma de instaladores ejecutables. También es muy común la distribución de software en Cds y DVDs. En sistemas open source como Ubuntu GNU/Linux nos encontramos esta forma de distribución de software, pero la mayoría del software se encuentra empaquetado en ficheros .deb, (.rpm en Red Hat), que contienen programas y las bibliotecas que necesitan. Los **repositorios** son servidores que contienen conjuntos de paquetes. A esto servidores se accede con herramientas como **Centro de Software**

Los **repositorios** van a centralizar todos los paquetes que se pueden instalar, ofreciendo un amplio abanico de software. En Ubuntu, generalmente, se querrá tener al menos los repositorios oficiales de Ubuntu (que pueden incluir el CD de instalación) pero es bastante común tener otros repositorios (de otros empaquetadores) activados.

5.- Personalización de la instalación.

Caso práctico

María quiere crear un instalador para la aplicación lo más amigable e intuitivo posible. Ella cree necesario que la aplicación pueda ser instalada en sistemas Windows y Linux, por lo que está investigando el software disponible para la creación de instalaciones en Windows.

Ministerio de Educación y Formación Profesional
(Elaboración propia)

Una vez que un programador ha finalizado el desarrollo de una aplicación, debe decidir el mecanismo que va a utilizar para su distribución.

Si el producto desarrollado se va a instalar en sistemas Windows deberá crear instaladores para Windows (ficheros ejecutables .exe). Si lo va a distribuir en Linux lo normal es que cree un paquete de instalación (paquete .deb en Ubuntu).

Ministerio de Educación y Formación Profesional
(Elaboración propia)

En ambos casos, el programador utilizará herramientas de generación de instaladores que le permitan personalizar su instalación.

Generalmente, cuando se crea un instalador para una aplicación, este instalador mostrará el logotipo de la aplicación, o de la empresa de desarrollo, tendrá un icono propio, unos colores y formato de ventana propios, formularios en los que se muestren los acuerdos de licencia, donde se pueden seleccionar el idioma de instalación, los directorios donde se desean copiar los archivos de la aplicación, etc.

Todos estos parámetros permiten crear un instalador propio para cada aplicación.

Para saber más

En el siguiente enlace se muestra un ejemplo de creación de paquete debian.

Creación de paquete deb en Ubuntu.

El enlace siguiente deriva en la página oficial de NullSoft.

Crear instalables para Windows con NSIS.

Autoevaluación

¿Qué herramienta no crea programas de instalación?

- ○ IzPack.
- ○ NSIS.
- ○ Install Shield.
- ○ Centro de software

Incorrecta.

No es correcta.

No es la opción correcta.

Correcta. Es una herramienta de Ubuntu para instalar paquetes.

Solución

1. Incorrecto
2. Incorrecto
3. Incorrecto

4. Opción correcta

5.1.- Logotipos.

Caso práctico

Ada propone que el programa de Gestión Hotelera tenga un logotipo que lo identifique como un programa de gestión de hoteles, que sea legible y fácilmente recordable. Este logotipo debe de estar presente en la instalación de la aplicación.

Ministerio de Educación y Formación Profesional (Elaboración propia)

Dentro de los programas de instalación de software, un componente muy importante va a ser el logotipo.

El logotipo es un elemento gráfico que va a identificar a la empresa que ha desarrollado el programa o al propio programa.

Alf (CC BY-NC-SA)

Normalmente, los logotipos suelen incluir símbolos que se asocian claramente con quienes representan.

El logotipo es el activo más importante de un servicio y producto y se utiliza como sello distintivo. Es por ello que, a la hora de diseñar el logotipo de una aplicación, debemos tener presente algunos conceptos. Para que un logotipo resulte exitoso se debe seguir el principio fundamental de diseño de "menos es más", la simplicidad va a permitir que el logotipo sea:

- Legible.
- Escalable.
- Reproducible.
- Distinguible.
- Memorable.

En un instalador se puede insertar el logotipo arriba, abajo, a la derecha o la izquierda del instalador. El tamaño del logotipo vendrá en función del ancho/alto especificado, del ancho y alto del instalador y de la fuente utilizada en el instalador. Normalmente el logotipo será una imagen que insertaremos en el instalador, usando la herramienta de creación de instaladores.

Si tomamos como ejemplo NSIS, para añadir el logotipo en el instalador, se hace con el atributo **AddBrandingImage**. Este atributo presenta la siguiente sintaxis:

(left|right|top|bottom) (width | height) [padding]

5.2.- Fondos.

Caso práctico

A la hora de crear el programa instalador de la aplicación de Gestión Hotelera, **María** va a utilizar como fondo del programa, imágenes y colores acordes con la imagen y colores corporativos de la empresa para la que diseñan la aplicación.

Free-photos (Dominio público)

Cuando se diseña un instalador para una aplicación de interfaz gráfica, la interfaz del instalador deber adquirir y presentar la información de forma consecuente a la interfaz de la aplicación a la que sirve. En el caso de los fondos del instalador, estarán organizados en función del diseño estándar que se mantiene en todas las ventanas de la aplicación.

El fondo del instalador, por tanto, deberá implementar las mismas reglas de diseño para mantener la interacción en toda la aplicación.

Reflexiona

Está comprobado que aquellos contenidos con color de fondo son interpretados por los usuarios como contenido poco importante; normalmente, se trata de contenido publicitario.

Cuando el color de fondo es blanco (o, sencillamente, no hay color de fondo) el usuario interpreta esa información como relevante y su nivel de atención aumenta considerablemente.

Autoevaluación

¿Qué tipo de archivo es el que se utiliza en Ubuntu para distribuir aplicaciones?

- ○ Ficheros ejecutables de extensión exe.
- ○ Fichero ejecutable **jar.**
- ○ Script ejecutables sh.
- ○ Paquetes deb.

Incorrecta. Este tipo de ficheros son para sistemas Windows.

No es correcta. Aunque pueden ejecutarse en Ubuntu, no son la forma habitual de distribuir paquetes.

No es la opción correcta. Se pueden instalar aplicaciones con script ejecutables, peor no es la forma más habitual en Ubuntu.

Correcta. Los paquetes deb (Debian) son la forma más común de distribuir aplicaciones específicas para distribuciones Ubuntu.

Solución

1. Incorrecto
2. Incorrecto
3. Incorrecto
4. Opción correcta

5.3.- Botones.

Caso práctico

María, en su afán por personalizar el programa de instalación, también está considerando la posibilidad de cambiar el aspecto estándar de los botones del programa de instalación, para que sean más amigables y fáciles de utilizar.

Ministerio de Educación y Formación Profesional
(Elaboración propia)

Cuando se hace un instalador, los botones que se implementan son los de aceptar o rechazar el acuerdo de licencia. Los botones normalmente que aparecen son: siguiente, anterior, instalar y finalizar.

En el proceso de instalación gráfica de una aplicación, lo que se nos presentan son un conjunto de ventanas, en las que el usuario toma algunas decisiones. Los botones de cada una de las ventanas, son de aceptar o cancelar los valores ofrecidos por el instalador, y siguiente o anterior, para avanzar o retroceder en las ventanas de instalación.

Tenemos que ser consistentes con el diseño gráfico de los botones del instalador y del diseño gráfico de la aplicación a la que sirve.

Siempre hay que ser consistente con todos los elementos que forman parte de una aplicación, incluido su instalador.

Mantener la consistencia implica que el formato de las ventanas, los colores, iconos, imágenes, botones y demás componentes gráficos guarden el mismo formato, tanto en colores, fuentes y tamaños.

La forma más habitual de los botones gráficos es la rectangular, aunque a veces pueden tener las esquinas redondeadas. Otras formas cada vez más comunes son la circular, elíptica, trapezoidal…

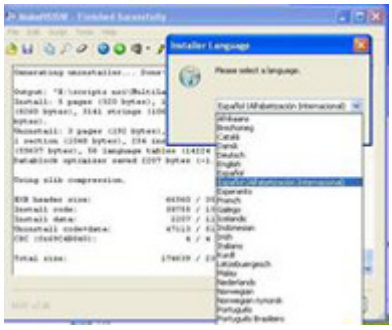
5.4.- Idioma.

Caso práctico

Ada propone que la instalación se pueda realizar en varios idiomas, proponiendo que la instalación se pueda realizar en castellano, en los demás idiomas cooficiales de España y en inglés. Cree que con estas adaptaciones el programa se podrá instalar en todos los hoteles de la cadena, independientemente de donde se encuentren.

Ministerio de Educación y Formación Profesional (Elaboración propia)

La mayoría de las aplicaciones que se distribuyen están a disposición de cualquier usuario en alguna página web, o páginas de descargas. Esta disposición global de las aplicaciones, implica que cualquier usuario de cualquier parte del mundo, pueda tener acceso a la aplicación y pueda descargarla e instalarla.



Verónica Cabrerizo (Elaboración propia)

Cualquier aplicación que se desee distribuir a cualquier lugar, debe tener la interfaz implementada en inglés, pero es razonable traducir la interfaz a varios idiomas, y que el usuario final elija durante el proceso de instalación el idioma en el que desea que se instalada.

El programador deberá tener en cuenta es su diseño, la posibilidad de que su aplicación se pueda distribuir en varios idiomas. Si esta posibilidad está presente, el programa instalador también deberá presentarse en diferentes idiomas y en él se podrá elegir el idioma de instalación final.

Autoevaluación

Para crear un instalador personalizado en Windows deberemos:

- ○ Crear un paquete de instalación deb.
- ○ Crear un paquete ejecutable jar.
- ○ Utilizar algún software específico como NSIS.

Incorrecta. Esa extensión es propia del sistema operativo Linux.

No es correcta.

Correcta. Muy bien.

Solución

1. Incorrecto
2. Incorrecto
3. Opción correcta

6.- Generación de paquetes de instalación.

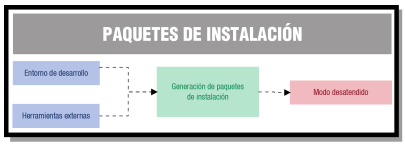
Caso práctico

Con la aplicación totalmente finalizada y probada, **Juan** se dispone a enseñar a **Antonio** a empaquetar la aplicación en paquetes JAR, para poder distribuirla y probarla en los ordenadores de los clientes.

Ministerio de Educación y Formación Profesional
(Elaboración propia)

Para generar paquetes instalación de una aplicación, disponemos de varias alternativas:

- Utilizar entornos de desarrollo.
- Hacer uso de herramientas externas.
- Instalar en modo desatendido.



Ministerio de Educación y Formación Profesional
(Elaboración propia)

La primera alternativa es la utilización de las herramientas de generación de paquetes incorporadas en el Entorno de Desarrollo. Los entornos de desarrollo, suelen disponer de herramientas de generación de paquetes, pero no suelen incorporar herramientas que creen instaladores "amigables". En el caso del entorno de desarrollo NetBeans, podemos crear un paquete de instalación JAR de nuestra aplicación Java, sin embargo, la instalación y la ejecución del paquete, deberá realizarla el usuario.

Existen en el mercado herramientas que complementan a los entornos de desarrollo para la creación de paquetes de instalación. Estas herramientas, en el caso de aplicaciones Java, pueden crear el paquete JAR, o directamente partiendo del paquete JAR generado a partir del entorno de desarrollo, crear un nuevo proyecto de instalación, donde se cree un instalable o un autoinstalable, para la aplicación desarrollada.

En este punto, vamos a profundizar en la creación de paquetes de instalación para aplicaciones Java.

Autoevaluación

Los ficheros JAR:

- ○ Son paquetes ejecutables que contienen clases Java y otros recursos.
- ○ Son ficheros con código fuente Java.
- ○ Son ficheros que deben compilarse para poder ser ejecutados.

Correcta. Los ficheros JAR contienen los ficheros class, resultado de la compilación, y los recursos necesarios para que una aplicación pueda ejecutarse.

No es correcta.

No es la opción correcta.

Solución

1. Opción correcta
2. Incorrecto
3. Incorrecto

6.1.- Entorno de desarrollo.

Caso práctico

Dado que, para el desarrollo de la aplicación de Gestión Hotelera, el equipo de desarrollo de BK Programación ha utilizado NetBeans, **Juan** va a empaquetar la aplicación en un fichero JAR utilizando las herramientas y opciones, suministradas por el propio entorno de desarrollo.

Para crear un paquete de instalación desde un entorno de desarrollo, lo primero a realizar, es la creación de la aplicación, la depuración de la aplicación y en su caso la realización de pruebas. Una vez que hemos finalizado todo el ciclo de vida de desarrollo de la aplicación, y si éste ha tenido éxito, procedemos a la fase de distribución de la aplicación.

En el caso del desarrollo de aplicaciones Java, el objetivo para distribuir una aplicación, es generar el fichero de empaqueta Jar.

Ministerio de Educación y Formación Profesional (Elaboración propia)

En un entorno de desarrollo para Java como NetBeans, una vez depurado y probado el proyecto, vamos a proceder a construirlo y crear el fichero JAR. Cuando se realiza este paso, en el **explorador del proyecto** aparece una nueva carpeta de nombre **dist**. En esta carpeta se va a crear el fichero JAR. Si la aplicación requiere **bibliotecas adicionales** para poder ejecutarse, algo habitual en aplicaciones con interfaz gráfico, dentro de dist se crea una subcarpeta de nombre **lib**, que contiene las librerías necesarias para la ejecución (las que no estén incluidas en el JDK).

La aplicación así generada, se puede ejecutar dentro del IDE y fuera de él. Para ejecutarla dentro de NetBeans, se selecciona el proyecto con el ratón en el explorador de proyectos, y se selecciona la opción **ejecutar** del menú contextual. Si la generación realizada en el paso anterior fue correcta, nos mostrará la ejecución de la aplicación desarrollada. Si queremos ejecutar la aplicación implementada fuera del IDE, navegamos por el explorador de nuestro sistema operativo (Windows, o Linux) y hacemos doble clic sobre el **fichero Jar** que se encuentra en la carpeta dist de nuestro proyecto.

Para distribuir la aplicación a otros usuarios, en primer lugar, debemos comprobar que la aplicación funciona fuera del IDE. Para distribuir la aplicación, primero creamos un fichero

zip que contenga el fichero de aplicación Jar y el directorio lib conteniendo las librerías que necesita la aplicación.

Una vez que se ha distribuido el fichero zip, la ejecución de la aplicación fuera del IDE se puede realizar de varias formas. La manera más rápida de ejecutar la aplicación, es descomprimir el fichero zip en cualquier carpeta seleccionada por el usuario final, y hacer doble clic sobre el fichero JAR.

Si la aplicación no se ejecuta, puede que tengamos problemas con la asociación de fichero JAR con aplicaciones, en nuestro sistema operativo. Esto puede deberse a dos razones:

- Que el tipo de fichero JAR no esté asociado a Java Runtime Environment.
- Que el tipo de fichero JAR estén asociados a JRE (Acrónimo de Java Runtime Environment (Entorno de Ejecución para Java). Es un conjunto de utilidades que permite la ejecución de programas Java), pero la opción -jar no está incluida en el comando que se pasa a JRE al hacer doble clic en el icono.

A continuación, veremos todos los pasos necesarios para realizar el empaquetado y distribución de una aplicación java.

Para saber más

Para ampliar la información sobre empaquetado distribución de aplicaciones en NetBeans, podemos visitar la siguiente página.

Empaquetado y distribución en NetBeans.

6.1.1.- Empaquetado y distribución de aplicaciones Java de escritorio.

Una vez que un programador o programadora ha desarrollado una aplicación, surge la necesidad de poder ejecutar esa aplicación fuera del entorno de desarrollo y la de poder distribuir la aplicación, de forma que funcione como si tuviera a su alcance todos los recursos de IDE.

Utilizando entornos de desarrollo para Java como NetBeans, es posible preparar la aplicación que se está desarrollando, para poder ejecutarla fuera del entorno de desarrollo y también, poder distribuirla. Hay que tener en cuenta, que para poder distribuir una aplicación Java, el sistema donde se quiera utilizar, tiene que estar preparado para la ejecución de la aplicación.

Existen diferentes alternativas para que una usuario o usuaria, pueda ejecutar una aplicación Java:

- Haciendo doble clic sobre el Archivo Java de Aplicación (fichero JAR).
- Invocando a la aplicación desde la línea de comandos.
- Llamando a la aplicación desde un fichero de script.

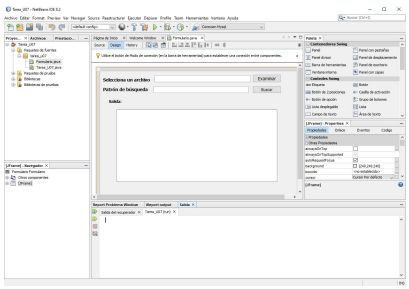
Empaquetado y distribución de Aplicaciones Java de Escritorio

Lo más común para distribuir aplicaciones Java, es crear un archivo JAR desde el entorno de desarrollo, que permita la ejecución de la aplicación fuera del IDE. La aplicación se empaqueta en forma de un archivo JAR ejecutable. Un fichero JAR es un archivo que contiene múltiples ficheros y carpetas. Los ficheros JAR son similares a los ficheros comprimidos zip, pero los ficheros JAR pueden tener atributos adicionales que los hacen muy útiles para la distribución de aplicaciones Java.

Para ilustrar el proceso, vamos empaquetar una aplicación de escritorio, que ya tenemos desarrollada en NetBeans. Partimos de un proyecto que contiene, al menos, un fichero con código fuente Java. Compilaremos las clases y construiremos un fichero JAR ejecutable. Una vez realizado el proceso, veremos cómo ejecutar el fichero JAR fuera del IDE.

Configuración del Proyecto

Una vez que hemos diseñado la aplicación que queremos distribuir, nos aparecerá una ventana en el IDE similar a la imagen siguiente:

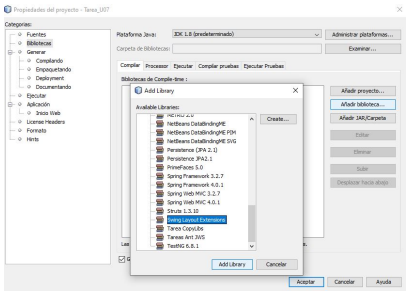


Montaña Martín Vergel (Elaboración propia)

A pesar de disponer de todo el código fuente creado, y que el proyecto funciona, es necesario realizar algunas configuraciones adicionales. Es necesario:

- Para poder crear el fichero ejecutable JAR, es necesario definir la clase principal del proyecto.

Puesto que hemos creado una aplicación de escritorio, en el proyecto de ejemplo, es necesario añadir la biblioteca “Swing Layout Extension”, ya que se utiliza el layout “GroupLayout”. Este paso es necesario con todas las librerías (bibliotecas), que utilicen componentes que no estén incluidos en el JDK



Montaña Martín Vergel (Elaboración propia)

Configuración de la clase principal

Para que un usuario o usuaria pueda ejecutar el fichero JAR (haciendo doble clic en el fichero JAR o escribiendo **java -jar aplicación.jar** en la línea de comandos), la clase principal debe especificarse en el interior del fichero *manifest* del fichero JAR. (Manifest es una parte estándar del fichero JAR que contiene información acerca del fichero JAR que se necesaria para el lanzador java cuando queremos ejecutar la aplicación).

Cuando construimos un proyecto, el IDE construye el fichero JAR e incluye un manifest. Cuando configuramos la clase principal, debemos asegurarnos que la clase principal será designada en el manifiest cuando se construya el proyecto.

En el siguiente apartado veremos cómo se configura la clase principal.

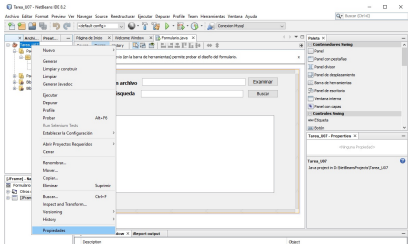
6.1.2.- Configuración de la clase principal.

Para que un usuario o usuaria pueda ejecutar el fichero JAR (haciendo doble clic en el fichero JAR o escribiendo **java -jar aplicación.jar** en la línea de comandos), la clase principal debe especificare en el interior del fichero *manifest* del fichero JAR. (Manifest es una parte estándar del fichero JAR que contiene información acerca del fichero JAR que se necesita para el lanzador java cuando queremos ejecutar la aplicación).

Cuando construimos un proyecto, el IDE construye el fichero JAR e incluye un manifest. Cuando configuramos la clase principal, debemos asegurarnos que la clase principal será designada en el manifiest cuando se construya el proyecto.

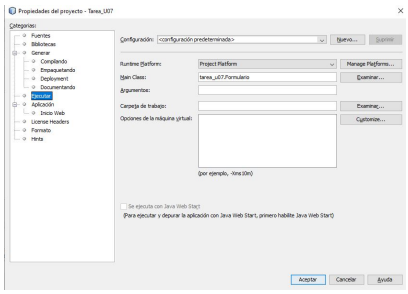
Para establecer la clase principal, procederemos de la siguiente forma:

1-. Seleccionaremos la raíz o el nombre de nuestro proyecto y accediendo al menú contextual, seleccionaremos la opción Propiedades.



Montaña Martín Vergel (Elaboración propia)

2-. Seleccionamos Ejecutar y entramos en la clase que queremos establecer como clase principal en nuestro proyecto. En el apartado Main class, pulsamos Examinar y seleccionamos la clase principal del proyecto:



Montaña Martín Vergel (Elaboración propia)

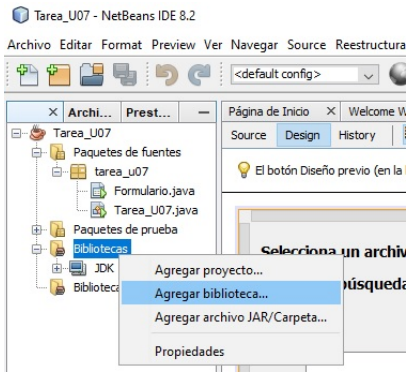
3-. Pulsamos el botón Aceptar para cerrar el cuadro de diálogo de Propiedades de Proyecto. Cuando posteriormente se construya en proyecto, el manifest habrá generado e incluido la siguiente entrada:

```
Main-Class: nombreProyecto.clasePrincipal
```

6.1.3.- Añadir las librerías necesarias.

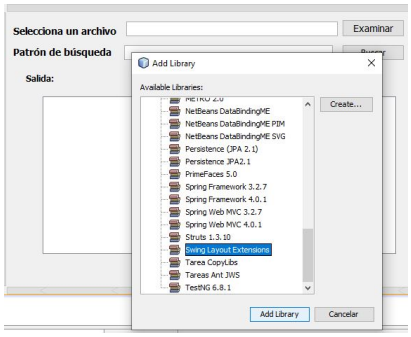
Si el proyecto que estamos desarrollando utiliza código de otras librerías, es necesario agregar esas librerías a nuestro proyecto. Esto es necesario para que nuestro proyecto pueda compilarse y la aplicación sea fácil de distribuir.

Podemos añadir las librerías a un proyecto, a través del nodo Bibliotecas de la ventana del proyecto.



Montaña Martín Vergel (Elaboración propia)

En el ejemplo que estamos siguiendo, es necesario añadir el control GroupLayout, que no se encuentra incluido en el JDK. Las clases del GroupLayout se encuentran disponibles en la biblioteca “Swing Layout Extensions”. El IDE incluye la biblioteca *Swing Layout Extension*s en el Gestor de Bibliotecas, así que es fácil añadirla a nuestro proyecto. Para añadir la biblioteca Swing Layout Extensions, la buscamos en la lista de bibliotecas disponibles y pulsamos el botón Agregar Librería (Add Library)



Montaña Martín Vergel (Elaboración propia)

Cuando se construya posteriormente el proyecto, se habrá agregado la siguiente entrada en el manifest del fichero JAR:

Class-Path: lib/swing-layout-1.0.jar

6.1.4.- Construcción del proyecto y creación del fichero jar.

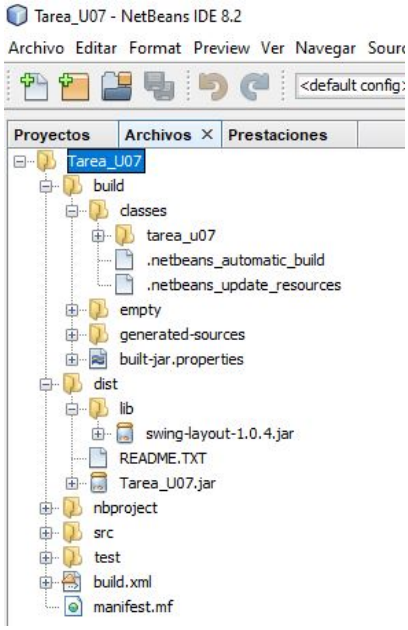
Una vez que tenemos preparado el código fuente y el proyecto está configurado, procedemos a construir el proyecto.

Para construir el proyecto:

- Elegimos Ejecutar → Limpiar y Generar Project

Cuando construimos el proyecto:

- Se agregan las carpetas **build** y **dist** a la carpeta del proyecto.
- Todos los ficheros fuentes son compilados a ficheros .class, que se encuentran en la carpeta PROJECT_HOME/build. (En este caso PROJECT_HOME es igual a Tarea_U07)
- Se ha creado un fichero JAR que contiene nuestro proyecto en la carpeta PROJECT_HOME/dist.
- Se han especificado bibliotecas para el proyecto (aparte de las del JDK), se crea una carpeta **lib,** en la carpeta **dist**. Las bibliotecas se copian en **dist/lib.**
- El manifest del fichero JAR es actualizado para incluir entradas designadas por la clase principal y bibliotecas que nos están en el classpath del proyecto.

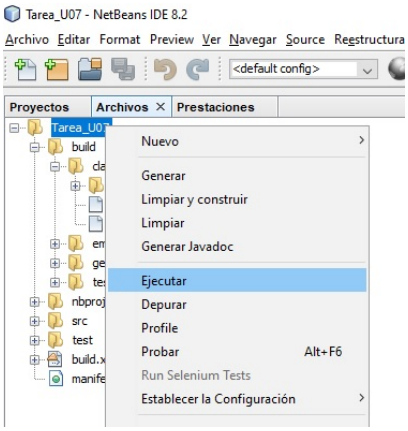


Montaña Martín Vergel (Elaboración propia)

6.1.5.- Ejecución de la aplicación dentro del IDE.

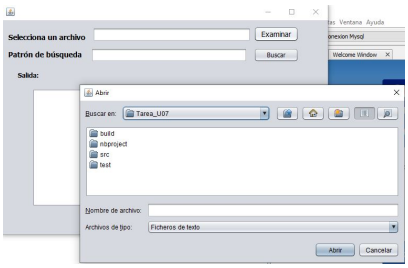
Cuando se desarrollan aplicaciones en el IDE, normalmente necesitaremos probarlas y redefinirlas antes de distribuirlas. Podemos fácilmente probar la aplicación en el que estamos trabajando y ejecutarla desde el IDE.

Para ejecutar una aplicación en el IDE, hacemos doble clic sobre el nodo del proyecto, en la ventana de Proyectos del IDE y elegimos ejecutar.



Montaña Martín Vergel (Elaboración propia)

La aplicación se abrirá y podremos utilizarla:



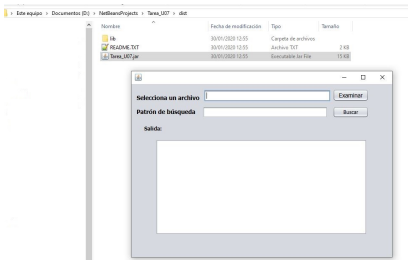
Montaña Martín Vergel (Elaboración propia)

6.1.6.- Ejecución de la aplicación fuera del IDE.

Una vez que hemos finalizado el desarrollo de la aplicación, y antes de distribuirla, hay que asegurarse de que la aplicación también funciona fuera del IDE.

Podemos navegar por el Explorar del sistema hasta la carpeta *PROJECT_HOME*/dist y hacer doble clic sobre el fichero .jar

La carpeta PROJECT_HOME equivale a la ruta de donde se encuentra nuestro proyecto almacenado en el disco. Esta ruta dependerá de donde cada usuario está guardando los proyectos que cree con NetBeans. En el ejemplo, los proyectos creados con NetBeans se están guardando dentro de la ruta D:\NetBeansProyects. Por lo tanto, la variable *PROJECT_HOME es igual a la ruta D:\NetBeansProyects*



Montaña Martín Vergel (Elaboración propia)

Deberemos de acceder a la carpeta o directorio en donde tengamos nuestro proyecto, acceder a la carpeta dist (para que exista no debemos de olvidar haber generado el proyecto), y ejecutar el fichero jar para comprobar que la aplicación funciona correctamente.

Importante: para distribuir la aplicación no solo debemos distribuir el fichero jar sino todo el contenido de la carpeta dist.

6.1.7.- Distribución de la aplicación.

Una vez que hemos verificado que la aplicación funciona fuera del IDE, podemos distribuirla.

Podemos distribuir la aplicación siguiendo los siguientes pasos:

- Creando un fichero zip que contenga el fichero JAR de la aplicación y que vaya acompañado de la carpeta **lib** que contiene swing-layout-1.0.4.jar
- Se puede distribuir el fichero, teniendo como única consideración que, para poder trabajar con la aplicación, tanto el fichero **JAR** con la carpeta **lib**, deben de encontrarse en la misma carpeta.

Los usuarios y usuarias finales pueden ejecutar la aplicación haciendo doble clic sobre el fichero JAR. Si no llegara a ejecutarse, puede deberse a que no se tenga instalada una máquina virtual Java, o existan problemas con la asociación de ficheros con aplicaciones.

6.1.8.- Posibles problemas.

En la mayoría de los sistemas, podemos ejecutar un fichero JAR haciendo doble clic con el botón izquierdo del ratón. Si al hacerlo, no pasa nada, se puede deber las siguientes razones:

- El tipo de fichero JAR probablemente no esté asociado con el Entorno de Ejecución Java (Java Runtime Environment) JRE) en ese sistema. Si el fichero JAR está asociado con una JRE, el icono que lo representa es el logo Java.
- El fichero JAR es asociado al JRE, pero la opción -jar no está incluida en el comando que se le pasa al JRE al hacer doble clic.

Para añadir la asociación del fichero JAR en los sistemas Microsoft Windows:

1. Hay que asegurarse que tenemos instalada una versión de JRE en nuestro sistema.
2. Si no la tenemos instalada, podemos descargarla desde el sitio web de Java. http://www.oracle.com/technetwork/java/javase/downloads/index.html
3. Si tenemos instalada JRE en nuestro sistema, lo que debemos es asociar los ficheros JAR con el JRE. Para ello, en la opción “Abrir con” , hay que asociar los ficheros JAR con “Java Platform SE Binary”.

En los sistemas UNIX y Linux, el procedimiento para cambiar la asociación de ficheros, depende del tipo de entorno de escritorio (GNOME o KDE) que se esté usando. En las preferencias de escritorio o en la documentación.

6.2.- Herramientas externas.

Caso práctico

Como la idea de BK Programación es que la aplicación de Gestión Hotelera sea multiplataforma y pueda ser instalada en diferentes sistemas, **María** está utilizando diferentes programas para crear instaladores que se encuentran en el mercado. Como parte de un paquete JAR, se decanta por probar NSIS, que es una herramienta que permite crear instaladores para aplicaciones Java y es libre.

Ministerio de Educación y Formación Profesional
(Elaboración propia)

Hay muchas herramientas externas a los entornos de desarrollo que nos permiten crear paquetes de instalación para aplicaciones. Dentro de las aplicaciones Java, hay varias herramientas que nos permiten crear un instalador a partir del paquete JAR. Entre esas herramientas nos encontramos con IzPack y NSIS.

Izpack es una herramienta que nos permite crear instaladores a partir de aplicaciones Java. Funciona en cualquier sistema operativo que tenga instalada una Máquina Virtual Java (JVM). Con esta herramienta podemos personalizar el instalador de una aplicación Java. La aplicación se puede descargar para cualquier sistema operativo que tenga instalada un JRE o JDK de Java. Una vez instalada la herramienta, ya podemos implementar instaladores para aplicaciones Java que hayamos desarrollado.

Para crear instaladores con IzPack, debemos crear un documento XML, donde cada tag (etiqueta) tiene un significado para IzPack. Definidos todos los atributos que queramos en el fichero xml, ya podemos generar nuestro programa de instalación; para ello, sólo debemos ejecutar el siguiente comando:

```
"ruta de instalación IzPack"/bin/compile instalacion.xml -o install.jar
```

Donde el fichero instalacion.xml contiene las características de la instalación e install.jar en nuestra aplicación Java ya empaquetada.

A diferencia de IzPack, NSIS es una herramienta específica para entornos Windows, pero también es libre. NSIS nos proporciona un completo entorno para crea instaladores, tanto

para aplicaciones Java, como aplicaciones desarrolladas en otros lenguajes.

Nullsoft Scriptable Install System funciona a través de un lenguaje propio de scripts. Para implementar un instalador, el programador o programadora escribe el script correspondiente, que una vez finalizado, será compilado por NSIS, creando un ejecutable como instalador.

NSIS proporciona un amplio abanico de script de ejemplo, que combinándolos y adaptándolos a nuestras necesidades, nos permiten crear instaladores completos, incluidos los desinstaladores.

En el siguiente apartado, vamos a ver, mediante un ejemplo como se utiliza NSIS.

6.2.1.- Creación de un instalador utilizando NSIS.

La mayoría de los paquetes de software que se usan en la actualidad, vienen con un instalador. El instalador se va a encargar de copiar, actualizar, eliminar ficheros, escribir claves en el registro, generar una configuración personalizada, crear accesos directos, etc. Todo este proceso se realiza de a través de un asistente que el usuario o usuaria ejecuta, y que se va presentando las diferentes opciones para que la persona que realiza la instalación, la personalice. Cuando el instalador finaliza su función, el usuario o usuaria puede ejecutar la aplicación.

Para poder realizar instalaciones personalizadas de nuestras aplicaciones, podemos seleccionar una gran cantidad de herramientas, que nos permiten crear instaladores. En nuestro caso, vamos a trabajar con la herramienta NSIS que, es una herramienta Open Source que, permite a los programadores y programadoras, mediante un lenguaje de script, realizar instaladores de las aplicaciones desarrolladas, de forma que, al usuario o usuaria final, se le presentan una serie de pantallas con diferentes opciones, así como otras tareas como generar clave en el registro o generar accesos directos a la aplicación.

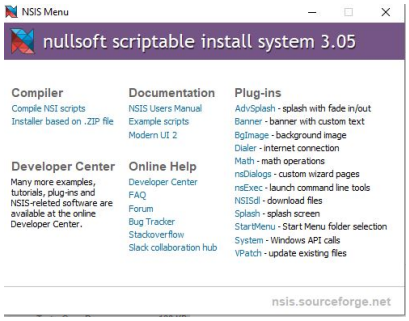
Una vez que se implementan los scripts, NSIS los compila dentro de un fichero ejecutable, de forma que la aplicación que hemos desarrollado, se puede distribuir fácilmente.

Debes conocer

Para instalar NullSoft Scriptable Install System, lo debemos descargar la página oficial, utilizando el siguiente enlace: Descarga de NSIS .

Montaña Martín Vergel (Elaboración propia)

Procederemos a instalarlo en nuestro equipo y una vez instalado, disponemos del compilador que nos va a permitir generar instaladores a partir de los ficheros de script .nsi que desarrollemos.



Montaña Martín Vergel (Elaboración propia)

6.2.2.-. Creación de scripts mediante NSIS.

Para crear un instalador con NSIS, debemos escribir un script NSIS. Un script NSIS es un fichero de texto plano con una sintaxis especial. Son scripts en los que cada línea es tratada como un comando. Dentro de la instalación de la aplicación, nos encontramos con una gran cantidad de scripts desarrollados como ejemplo. En función de nuestras necesidades, podemos editarlos y combinarlos.

Lenguaje de Script

La extensión por defecto de los scripts es .nsi. También existen ficheros header (al estilo de los .h de C/C++) que tienen la extensión .nsh.

Un script NSIS puede contener atributos del instalador, páginas, secciones y funciones.

Atributos del Instalador

Los atributos del instalador determinan el comportamiento y el aspecto del instalador. Estos atributos determinan los diferentes mensajes que se irán mostrando durante el proceso de instalación. En cada proceso de instalación, el instalador mostrará una ventana o una página.

Por ejemplo, **Name** es el atributo correspondiente al nombre de nuestra aplicación, **InstallDir** será el directorio elegido para instalar la aplicación etc.

Páginas:

Un instalador puede mostrar diferentes páginas al usuario, por ejemplo, la página bienvenida, la de aceptación de licencia, la de selección del directorio de instalación, etc:

- **Page license**
- **Page components**
- **Page directory**
- **Page instfiles**
- **UninstPage uninstConfirm**
- **UninstPage instfiles**

En caso de usar el UI moderno al incluir: !include "MUI.nsh" usaremos sus macros:

Mostramos la página de bienvenida: !insertmacro MUI_PAGE_WELCOME

- Página donde mostramos el contrato de licencia: `!insertmacro MUI_PAGE_LICENSE "licencia.txt"`
- Página donde se muestran las distintas secciones definidas: `!insertmacro MUI_PAGE_COMPONENTS`
- Página donde se selecciona el directorio donde instalar nuestra aplicación: `!insertmacro MUI_PAGE_DIRECTORY`
- Página de instalación de ficheros: `!insertmacro MUI_PAGE_INSTFILES`
- Página final: `!insertmacro MUI_PAGE_FINISH`

Secciones:

En un instalador pueden hacerse categorías de instalación. Y así separar la instalación en varios componentes, dando a elegir al usuario cuales instalar y cuáles no.

```
Section "My Program"

  SetOutPath $INSTDIR

  File "My Program.exe"

  File "Readme.txt"

SectionEnd
```

Dentro de cada sección usamos instrucciones que son ejecutadas en tiempo de ejecución. Estas instrucciones, leen y escriben en el registro, crean, borran y copian ficheros y directorios, crean accesos directos etc.

Los desinstaladores también pueden tener varias secciones teniendo como prefijo "un.":

```
Section "Installer Section"

SectionEnd



Section "un.Uninstaller Section"

SectionEnd
```

Funciones

Las Funciones contienen código semejante a las secciones, pero se diferencian de éstas en el modo en que se llaman. Hay dos tipos de funciones, las definidas por el usuario, que se llaman con la instrucción Call y las que se activan cuando ocurren determinados eventos en la instalación:

```
Function .onInit

     MessageBox MB_YESNO "Esto instalará mi programa ¿Quiere continuar?" IDYES gogogo

    Abort

  gogogo:

FunctionEnd
```

Abort es una función especial que hace que el instalador termine inmediatamente.

Variables:

En este lenguaje se declaran las variables mediante Var :

```
Var VARIABLE ;Declaramos la variable

Section variable

  StrCpy $VARIABLE "valor" ;Ahora la variable VARIABLE vale  o contiene "valor"
```

```
SectionEnd
```

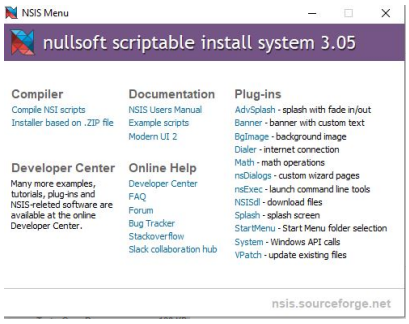
Una vez que hemos implementado nuestro script de instalación, la herramienta NSIS lo compila, creando un ejecutable que será nuestro instalador.

Estos son a grandes rasgos los componentes del lenguaje de script de NSIS, con el paquete se incluye un completo sistema de ayuda acerca del lenguaje, así como diferentes ejemplos.

6.2.3.- Entorno de NSIS.

Una vez que hemos diseñado nuestro script de instalación, nos queda el paso final, que es compilarlo. Para generar el fichero ejecutable del instalador nos bastará con pulsar el botón derecho sobre el script en el explorador de ficheros y seleccionar "Compile NSIS Script".

El compilador va a ir interpretando una a una todas las líneas del script, de forma que, si no se produce ningún error, se genera un archivo ejecutable, que será nuestro instalador.



Montaña Martín Vergel (Elaboración propia)

Al acceder a la aplicación, se nos muestran mediante enlaces, las posibles opciones de NSIS. Por ejemplo, Compile NSIS scripts (para comenzar a definir un nuevo script) Installer based on .Zip file (para instalaciones basadas en ficheros zip).

También nos muestra el acceso al manual de la aplicación (NSIS Users Manual) como ejemplos de scripts (Example scripts).

6.2.4.- Ejemplo de uso de NSIS.

Vamos a ver mediante un ejemplo, cómo podemos crear un instalador mediante NSIS. Para ello, comenzaremos por tener situado en una carpeta o directorio todos los ficheros que queremos que pueda utilizar el instalador.

Si construimos la aplicación con NetBeans, vimos en apartados anteriores que nos creó una carpeta dentro de nuestro proyecto con el nombre Dist en donde incluía el fichero jar generado junto con todas las librerías que necesitábamos para poder ejecutar la aplicación, partiremos del contenido de la carpeta Dist.

Vamos a empezar a utilizar NSIS creando un script y guardándolo en la carpeta Dist. Debemos de crear un fichero de texto nuevo con la extensión .nsi. Por ejemplo: Ejemplo.nsi

En su interior comenzaremos por definir la interfaz que vamos a utilizar. Si no definimos ninguna, nos selecciona la interfaz más básica y nuestro instalador tendrá un estilo de Windows XP. Para nuestro ejemplo, hemos utilizado la versión más reciente. Por ello, comenzamos nuestro script escribiendo la línea: !include "MUI2.nsh"

A continuación, definimos los parámetros generales de la instalación. Son:

1-. El título que aparecerá en el instalador. Para ello, incluiremos la línea Name "Ejemplo de creación de instalador"

2-.Indicaremos el fichero de salida que queremos generar. En nuestro caso será el ejecutable que servirá de instalador. Para ello, incluiremos la línea: OutFile "Ejemplo.exe"

3-. Directorio en donde se instalará la aplicación. Podemos reutilizar variables de entorno del sistema operativo escribiéndolas en mayúsculas y precedidas del símbolo $. Escribiremos la línea InstallDir "carpeta donde se desea instalar la aplicación"

4-. Definiremos las claves en el registro de Windows. Escribiremos la línea: InstallDirRegKey HKCU "Software\ejemplo" ""

5-. Estableceremos los permisos que necesitará el instalador. Admite los valores admin (para permisos de Administrador) o user.

6-. Debemos de permitir cancelar la instalación. Incluiremos la línea: !define MUI_ABORTWARNING

7-. A continuación, incluiremos las páginas o ventanas que deseamos mostrar. Estos valores dependerán de la versión del instalador que estamos utilizando, en nuestro ejemplo, MUI2.nsh. Vamos a incluir las páginas siguientes:

```
!insertmacro MUI_PAGE_COMPONENTS
!insertmacro MUI_PAGE_DIRECTORY
!insertmacro MUI_PAGE_INSTFILES
!insertmacro MUI_UNPAGE_CONFIRM
!insertmacro MUI_UNPAGE_INSTFILES
```

8-. Podemos establecer el idioma del instalador escribiendo la orden: !insertmacro MUI_LANGUAGE "Spanish". Especificaremos el idioma. Podemos consultar la ayuda de NSIS para ver los idiomas soportados.

9-. Definimos los componentes que podemos marcar durante el proceso de instalación. Deberemos de incluir todos los ficheros que se tienen que instalar. En nuestro caso el

fichero jar y las librerías necesarias. Si necesitamos crear alguna carpeta o directorio dentro del directorio en donde se debe de instalar la aplicación se lo deberemos de indicar utilizando la instrucción SetOutPath . En la imagen ser puede apreciar cómo aparece la sección "Fichero jar" en donde se indican los ficheros que se tienen que copiar y la carpeta que se tienen que crear. También se define la entrada en el registro de Windows en donde se encuentra instalada la aplicación

10-. Por cada componente que deseamos mostrar para que el usuario pueda indicar si desea instalarlo o no, podemos mostrar una definición de para qué sirve cuando se selecciona. Esto lo podemos conseguir empleando las líneas:

```
;Descripciones que aparecerán junto al componente cuando se seleccione. Aparece junto al componente
LangString DESC_SecDummy ${LANG_SPANISH} "Instalación del fichero Tarea_U07.jar"
!insertmacro MUI_FUNCTION_DESCRIPTION_BEGIN
!insertmacro MUI_DESCRIPTION_TEXT ${SecDummy} $(DESC_SecDummy)
!insertmacro MUI_FUNCTION_DESCRIPTION_END
```

11-. Para finalizar definimos un componente para desinstalar la aplicación. Para ello, incluiremos la siguiente sección en donde se especifica que elimine los ficheros junto las carpetas o directorios creados y elimine las claves del registro definidas:

```
Section "Desinstalar"
  Delete "$INSTDIR\Uninstall.exe"
  RMDir "$INSTDIR"
  DeleteRegKey /ifempty HKCU "Software\ejemplo"
SectionEnd
```

Montaña Martín Vergel (Elaboración propia)

Una vez que tengamos creado el fichero, lo guardaremos en la misma carpeta que contenga los ficheros que deseamos incluir en el instalador, y arrancaremos la aplicación NSIS. Seleccionamos la opción Compile NSI script y abriremos nuestro script. Una vez abierto, pulsamos la opción Script -> recompile y se nos intentará compilar el script. Al compilarse obtendremos el fichero ejecutable .exe que nos aparecerá en la misma carpeta en donde hemos guardado el script.

A partir de este momento ya tendremos generado el instalador.

Si al compilar apareciera algún error, nos indicaría a partir de que línea se encuentra el error y una posible causa.

6.3.- Modo desatendido.

Caso práctico

Juan se centra en crear la instalación, de forma que no necesite la interacción del usuario, es decir, la aplicación se va a instalar utilizando una serie de parámetros que no van a ser modificados por el usuario final.

Ministerio de Educación y Formación Profesional
(Elaboración propia)

Una instalación en modo desatendido, es aquélla en la que el usuario no tiene que decidir el lugar de instalación de la aplicación, ni tampoco características tales como el idioma, variable de entorno, etc.

Ministerio de Educación y Formación Profesional
(Elaboración propia)

El uso de instalaciones desatendidas es útil cuando se deben instalar programas en gran cantidad de ordenadores, cuando hay que instalar sistemas operativos.

Para automatizar el proceso de instalación, podemos crear un script que nos guarde las pulsaciones del ratón y del teclado y los datos que introducimos. De esta forma, al ejecutarse el instalador, reproduciría el patrón que nosotros hemos grabado previamente en el instalador.

Otra forma de realizar instalaciones desatendidas, es la utilización de los parámetros que los instaladores ponen a nuestra disposición. Por ejemplo, con el instalador NSIS podemos utilizar el parámetro /S para ejecutarlo en modo silencioso. INNO Setup es otro instalador para aplicaciones en Windows, que permite la instalación desatendida. Cuando arrancamos un programa que ha sido empaquetado con este instalador, y le hacemos clic en el pequeño icono, podemos ver que debajo de la opción cerrar aparece la opción "About Inno Setup". Este es el claro ejemplo de un instalador del tipo INNO. Para poder instalar en modo silencioso un programa que sido empaquetado con el instalador de INNO, el parámetro que tenemos que usar es **/SP- /VERYSILENT /NORESTART**.

Los paquetes MSI, o también conocidos como Windows Installer suelen estar en combinación con InstallShield. En el caso del segundo lo que tenemos que intentar ver, es si podemos extraer de alguna manera los ficheros de ahí dentro, bien sea con un

descompresor como Winrar, o bien haciendo una instalación administrativa con el parámetro /a. Esto último, lo que nos hará será extraernos todos los ficheros que están empaquetados en el instalador hacia una carpeta determinada. Una vez tengamos sus archivos fuera seguramente encontraremos un archivo MSI, que es el instalador de Windows. Este tipo de instalador lo podemos poner en modo silencioso con los comandos **/qn (o/qb o /quiet) /norestart**. También puede ser que directamente nos saque el contenido del programa. En este caso, lo podemos re-empaquetar con otro instalador que si admita switches silenciosos o con un compresor con el modo silencioso marcado (como por ejemplo WinRar).

Autoevaluación

En una instalación desatendida:

- ○ El usuario decide la carpeta de instalación de la aplicación y todas las opciones de instalación.
- ○ El instalador interactúa continuamente con el usuario final.
- ○ La aplicación se instala de forma transparente al usuario.

Incorrecta. Serían para los comentarios en línea.

No es correcta.

Muy bien. Correcta.

Solución

1. Incorrecto
2. Incorrecto
3. Opción correcta

7.- Parámetros de la instalación.

Caso práctico

María quiere que el programa de instalación que está desarrollando, permita al usuario interactuar con él. Para ello, el instalador va a presentar al usuario todas las opciones de la instalación para que pueda personalizarlas.

Ministerio de Educación y Formación Profesional
(Elaboración propia)

Los parámetros de la instalación van a personalizar la aplicación, dando la opción al usuario que realiza la instalación de elegir entre diferentes alternativas y opciones, para ajustar la instalación de la aplicación a sus necesidades.

Dado que la mayoría de las aplicaciones actuales se distribuyen a través de repositorios que se encuentran alojados en sitios web, las aplicaciones son accesibles para cualquier usuario de cualquier parte del mundo.

Las aplicaciones que se desean que tenga mayor difusión, requieren estar disponibles en diferentes idiomas, de forma que el usuario que desea utilizar la aplicación pueda instalarlo en su propia lengua. Es por ello, que el primer paso de la instalación en esas aplicaciones es la selección del idioma.

Todas las aplicaciones se distribuyen bajo diferentes acuerdos de licencia, por tanto, el usuario debe configurar es la aceptación o rechazo del acuerdo de licencia, si el usuario no acepta los términos de la licencia, la instalación será abortada.

En las instalaciones se pueden incluir aplicaciones o barras de herramientas para exploradores web adicionales, estas instalaciones son opcionales, eligiendo el usuario final su instalación o no.

Una vez que el usuario acepta los términos de la licencia, el siguiente parámetro a configurar, suele ser la ruta de instalación de los archivos de la aplicación, el instalador ofrece una ruta por defecto, que el usuario puede modificar si lo desea.

Elegido las carpetas donde se van a copiar los archivos necesarios de la aplicación, el siguiente paso que se parametriza es la creación de accesos directos en los menús de

inicio del sistema operativo y de un acceso directo en el escritorio, estos parámetros deben ser configurables. En Windows se suele dar la opción al usuario de crear o no un acceso directo en el escritorio y un acceso directo en el Inicio Rápido.

El último parámetro que suele ser accesible por el usuario es la opción de ejecutar la aplicación una vez instalada.

8.- Interacción con el usuario.

Caso práctico

En el instalador que **María** está creando, el usuario va a poder establecer las carpetas donde se instala el programa, la instalación o no de paquetes adicionales, la configuración del idioma, etc. La interfaz será amigable, para facilitar el proceso de instalación por parte del usuario.

Ministerio de Educación y Formación Profesional
(Elaboración propia)

Cuando se implementa una aplicación para ejecutar en interfaz gráfica (GUI), se basa en requerimientos no funcionales de usabilidad. En el caso de los instaladores de interfaz gráfica, también nos encontramos con una interfaz amigable.

En todo proceso de instalación de software en interfaces gráficas, se siguen una serie de pasos aceptados y estandarizados, que son los siguientes:

1. Ventana de selección de idioma.
   Lo primero que decide el usuario es el idioma en el que se le mostrarán los mensajes del proceso de instalación del programa.

2. Ventana de bienvenida.
   Información de la versión de la aplicación, recomendaciones, etc.

3. Acuerdo de licencia. Aceptación de los términos de uso.
   Se tienen que aceptar los términos de licencia del software que se quiere instalar. Si no es así, la instalación será abortada en este momento.

4. Aceptación o no aceptación de herramientas opcionales a instalar.
   Se trata normalmente de complementos web para nuestro navegador. Podemos seleccionarlos o no, y esa elección no tendrá efecto sobre nuestra instalación.

5. Selección de la ubicación donde se guardan los archivos.
   Normalmente, el programa nos propone por defecto la ruta `C:\programas` pero el usuario debe poder cambiar esta ubicación y elegir otra distinta.

6. Selección de accesos directos.
   Opcional en todo caso.

7. Proceso de instalación.

En este punto, comienzan a copiarse los archivos en el disco duro del ordenador del usuario, en la ubicación que él haya elegido.

8. Finalización.
   El proceso de instalación termina en este punto y debe avisar al usuario de su terminación.

9.- Ficheros firmados digitalmente.

Caso práctico

Para garantizar la autenticidad de la aplicación de Gestión Hotelera, **Ada** propone que los ficheros de instalación, ya sean paquetes JAR o instalables exe, vayan firmados digitalmente.

Ministerio de Educación y Formación Profesional
(Elaboración propia)

Una firma digital o esquema de firma digital, es un esquema matemático que permite demostrar la autenticidad de un documento digital, en nuestro caso un fichero.

Una firma digital válida nos garantiza que el fichero ha sido creado por una empresa o autor conocido y que no ha sido modificado durante su transferencia.

INTEF (CC BY-NC-SA)

En el caso de la distribución de software, sobre todo a través de descargas de Internet, nos garantiza la autenticidad del fichero, el autor y que durante la descarga no ha sufrido ningún tipo de alteración.

La plataforma Java nos permita firmar digitalmente ficheros Jar. Cuando se firma un fichero Jar, los que se está asegurando a los usuarios que ejecuten la aplicación es la autoría de ese software. Cuando digitalizamos el fichero, cualquiera puede reconocer la **firma digital** del autor. El proceso de reconocimiento de la firma digital se denomina **verificación**.

La firma y verificación de ficheros es una parte importante de la arquitectura de seguridad de la plataforma Java. Podemos configurar la política de seguridad para permitir a un applet (Componente de una aplicación que se ejecuta en el contexto de otro programa, por ejemplo, un navegador) y a una aplicación puedan realizar operaciones normalmente prohibidas como pueden ser la lectura y escritura de ficheros locales, ejecución programas, etc.

La plataforma Java permite firmar y verificar usando números especiales denominados **claves públicas y privadas**. Las claves públicas y privadas vienen en parejas, y tienen roles complementarios. La clave privada es el "lápiz" electrónico que nos permite firmar un fichero. La **clave privada** sólo es conocida por el **autor de la firma**. Un fichero firmado con llave privada, solo puede ser verificado con su correspondiente llave pública. Existe un elemento adicional necesario para la firma y verificación. Este elemento es el **certificado** que el firmador incluye en un fichero JAR firmado. El certificado es una declaración digital firmada reconocida por una autoridad de certificación que indica quién es el dueño de una clave pública determinada. Resumiendo, la firma digital:

- El desarrollador firma el fichero JAR usando una llave privada.
- La correspondiente llave pública se coloca en el fichero JAR, junto con el certificado, que estará disponible para que cualquiera pueda verificar la firma.

Autoevaluación

El reconocimiento de la firma digital de un archivo JAR se conoce como:

- ○ Clave privada.
- ○ Verificación.

No es correcta. La clave privada es la que permite firmar digitalmente un archivo.

Muy bien. Correcta.

Solución

1. Incorrecto
2. Opción correcta

9.1.- Firma digital en fichero JAR.

En este documento, vamos a resumir cómo utilizar las herramientas proporcionadas en el Kit de Desarrollo Java ™ para firmar y verificar los archivos JAR:

INTEF (CC BY-NC-SA)

La plataforma Java ™ le permite firmar digitalmente los archivos JAR. Firmamos digitalmente un archivo por la misma razón que podemos firmar un documento en papel con pluma y tinta, para que los lectores sepan que escribió el documento, o al menos que el documento tiene nuestra aprobación.

Cuando usted firma una carta, por ejemplo, todos los que reconocen su firma pueden confirmar que escribió la carta. Del mismo modo cuando se firma digitalmente un archivo, cualquier persona que "reconoce" su firma digital sabe que el archivo procede de usted. El proceso de "reconocimiento" de las firmas electrónicas se llama verificación.

La posibilidad de firmar y verificar archivos es una parte importante de la arquitectura de seguridad de la plataforma Java. La seguridad es controlada por la política de seguridad que esté en vigor en tiempo de ejecución. Puede configurar la directiva para conceder privilegios de seguridad para los applets y aplicaciones. Por ejemplo, puede conceder permiso a un applet para realizar operaciones normalmente prohibidas, tales como leer y escribir archivos locales o ejecutar programas locales ejecutables. Si ha descargado un código firmado por una entidad de confianza, puede utilizar este hecho como un criterio para decidir cuál de los permisos de seguridad asignará al código.

Una vez que usted (o su navegador) han comprobado que un applet es de una fuente confiable, usted puede relajar las restricciones de seguridad para que el applet realice operaciones que normalmente estarían prohibidas. Un applet de confianza puede tener libertades como se especifica en el archivo de política en vigor.

La plataforma Java permite a la firma y la verificación mediante el uso de números especiales llamados claves públicas y privadas. Las claves públicas y claves privadas vienen en pares, y desempeñan papeles complementarios.

La clave privada es la "pluma" electrónica con la que se puede firmar un archivo. Como su nombre indica, la clave privada es conocida sólo por usted, para que nadie más puede "falsificar" su firma.

Un archivo firmado con su clave privada sólo puede ser comprobada por la correspondiente clave pública.

warszawianka (Dominio público)

Las claves pública y privada por sí solas, sin embargo, no son suficientes para verificar realmente la firma. Incluso si usted ha verificado que un archivo firmado contiene un par de claves correspondientes entre sí, aún necesita alguna manera de confirmar que la clave pública en realidad proviene de la persona de la que la firma pretende provenir.

Se requiere un elemento más, por lo tanto, para hacer la firma y verificación. Este elemento adicional es el **certificado** de que el

firmante incluye en un fichero JAR firmado. Un certificado es una declaración firmada digitalmente de una autoridad de certificación reconocida que indica que es dueño de una clave pública determinada. Una autoridad de certificación son las entidades (generalmente empresas especializadas en seguridad digital) que son de confianza en toda la industria para firmar y emitir certificados para las teclas y sus propietarios.

En el caso de los archivos JAR firmados, el certificado indica que es dueño de la clave pública contenida en el archivo JAR.

Al firmar un fichero JAR su clave pública se coloca dentro del archivo, junto con un certificado asociado para que sea de fácil acceso para su uso por cualquier persona que desee verificar su firma.

Para resumir la firma digital:

- El firmante firma el archivo JAR utilizando una clave privada.
- La clave pública correspondiente se coloca en el archivo JAR, junto con su certificado, por lo que está disponible para su uso por cualquier persona que quiera verificar la firma.

Resúmenes y el fichero de firmas.

Al firmar un fichero JAR, cada fichero del archivo tiene una entrada de resumen en el archivo de manifiesto. He aquí un ejemplo de lo que tal entrada podría ser:

```
Name: test/classes/ClassOne.class
            SHA1-Digest: TD1GZt8G11dXY2p4olSZPc5Rj64=
```

Los valores son representaciones hash o codificadas de los contenidos de los archivos tal como estaban en el momento de la firma. El resumen de un archivo cambiará sí y sólo si el propio fichero cambia.

Cuando se firma un archivo JAR se genera automáticamente un archivo de firma y se coloca en el directorio META-INF del archivo JAR, el mismo directorio que contiene el archivo de manifiesto. Los archivos de firmas tienen nombres de archivo con una extensión. SF. He aquí un ejemplo del contenido de un archivo de firma:

```
Signature-Version: 1.0
            SHA1-Digest-Manifest: h1yS+K9T7DyHtZrtI+LxvgqaMYM=
            Created-By: 1.6.0 (Sun Microsystems Inc.)
            Name: test/classes/ClassOne.class
            SHA1-Digest: fcav7ShIG6i86xPepmitOVo4vWY=
            Name: test/classes/ClassTwo.class
            SHA1-Digest: xrQem9snnPhLySDiZyclMlsFdtM=
            Name: test/images/ImageOne.gif
            SHA1-Digest: kdHbE7kL9ZHLgK7akHttYV4XIa0=
            Name: test/images/ImageTwo.gif
            SHA1-Digest: mF0D5zpk68R4oaxEqoS9Q7nhm60=
```

Como puede ver, el archivo de firma contiene entradas de resumen para los archivos del archivo comprimido que se parecen a las entradas de valor de síntesis, en el manifiesto. Sin embargo, mientras que los valores de resumen en el manifiesto se calculan a partir de los propios archivos, los valores de resumen en el archivo de firma se calculan a partir de las

entradas correspondientes en el manifiesto. Los archivos de firma también contienen un valor de resumen para el manifiesto completo (ver la cabecera **SHA1-Digest-Manifest** en el ejemplo anterior).

Cuando un fichero JAR firmado está siendo verificado, los resúmenes de cada uno de los archivos se vuelven a calcular y comparar con los resúmenes, registrada en el manifiesto para asegurar que el contenido del archivo JAR no han cambiado desde que se firmó. Como una comprobación adicional, los valores de resumen para el archivo de manifiesto en sí se vuelven a calcular y se compara con los valores registrados en el archivo de firma.

Puede obtener información adicional sobre los archivos de firma en la página Formato de manifiesto de la documentación del JDK ™.

El archivo de bloque de firma.

Además de los archivos de firmas, se coloca automáticamente un archivo de bloque de firma en el directorio **META-INF** cuando se firma un archivo JAR. A diferencia del archivo de manifiesto o el archivo de firma, los archivos de bloques de firmas no son legibles.

El archivo de bloque de firma contiene dos elementos esenciales para la verificación: La firma digital para el archivo JAR que se generó con la clave privada del firmante El certificado que contiene la clave pública del firmante, para ser utilizado por cualquier persona que desea verificar el archivo JAR firmado.

Los nombres de los archivos de bloques de firmas suelen tener una extensión. DSA lo que indica que fueron creados por el algoritmo predeterminado de firma digital. Otras extensiones de nombre de archivo son posibles si las claves asociadas con algún algoritmo estándar de otros se utilizan para la firma.

Firmar archivos JAR.

warszawianka (Dominio público)

Se utiliza la firma de JAR y una herramienta de verificación para firmar ficheros JAR. La invocación de la firma JAR y herramienta de verificación mediante el comando **jarsigner**, por lo que nos referiremos a ella como "Jarsigner".

Para firmar un fichero JAR, primero debe tener una clave privada. Las claves privadas y sus correspondientes claves públicas se almacenan en bases de datos protegidas llamadas Keystore. Un keystore puede contener las claves de muchos firmantes potenciales. Cada clave del almacén de claves puede ser identificado por un alias que suele ser el nombre del firmante y que posee la clave. La clave que pertenecen a María Jiménez, podría tener el alias de "Maria", por ejemplo.

La forma básica del comando para la firma de un archivo JAR es:

```
jarsigner jar-file alias
```

En este comando:**jar-file** es la ruta del archivo JAR a firmar, alias es el alias que identifica la clave privada que se utiliza para firmar el archivo JAR, y el certificado asociado.

La herramienta Jarsigner le pedirá la contraseña para el almacén de claves y alias.

En esta forma básica del comando se supone que el almacén de claves a utilizar está en un archivo llamado keystore en su directorio personal. Se va a crear la firma y los archivos de bloques de firmas con nombres x.SF y x.DSA respectivamente, donde x son las primeras ocho letras del alias, todas en mayúsculas. Este comando básico sobrescribe el archivo JAR original con el fichero JAR firmado.

En la práctica, es posible que desee utilizar este comando junto con uno o más de estas opciones, que debe preceder a la ruta jar-file.

10.- Instalación de aplicaciones desde un servidor.

Caso práctico

Carlos está comprobando los requisitos que solicitan los servidores de Internet que distribuyen aplicaciones. También baraja la posibilidad de crear un servidor propio de BK Programación, donde alojen y distribuyan las aplicaciones desarrolladas por la empresa.

Ministerio de Educación y Formación Profesional
(Elaboración propia)

Cuando se implementa un paquete software para su distribución, existe la posibilidad de alojarlo en un servidor web para que sea accesible a un conjunto de usuarios. Estos usuarios pueden instalar directamente el paquete en su ordenador, simplemente haciendo clic en un hipervínculo. Esta forma de distribuir software es muy común en distribuciones Linux como Ubuntu.

Las principales ventajas de los servidores de aplicaciones es la centralización y la disminución de la complejidad del desarrollo de aplicaciones, ya que éstas no necesitan ser programadas. Basta con ensamblarlas desde bloques y ponerlas en el servidor a disposición de quien las necesite.

Para poder instalar aplicaciones desde un servidor, lo único que el usuario realiza es la pulsación sobre el hipervínculo donde aparece el paquete que quiere instalar.

Para poder realizar esta instalación, existe apturl. AptUrl es un miniprograma gráfico para instalar programas desde el repositorio donde se encuentran. Para poder distribuir aplicaciones utilizando esta herramienta, se edita una página web, donde se suele dar una descripción del programa a instalar, y se añade el enlace siguiente:

```
<a href=”apt:paquete”>Nombre del paquete a instalar</a>
```

Si queremos distribuir múltiples paquetes en el mismo enlace, la sintaxis sería la siguiente:

```
<a href=”apt:paquete1, paquete2, paquet3”> Nombre de la aplicación </a>
```

Un ejemplo de instalación de esta forma, sería:

```
apturlapt:pidgin,pidgin-plugin-pack
```

Donde se instalaría la aplicación Pidgin y Pidgin Plugin Pack .

Para saber más

Instalación de aplicaciones desde servidores en Linux. A partir de un gestor de paquetes, desde servidores de internet y desde la terminal.

Instalar aplicaciones en Linux

11.- Descarga y ejecución de aplicaciones ubicadas en servidores web.

Caso práctico

Ana está investigando las características que debieran tener las aplicaciones desarrolladas por BK Programación, para poder ser ubicadas en servidores de Internet, y desde allí, que los posibles clientes las puedan descargar e instalar.

Ministerio de Educación y Formación Profesional
(Elaboración propia)

Si el fichero descargado es un ejecutable, bastará con la ejecución de ese fichero para que se produzca su instalación. Este sería el caso de la descarga de un paquete autoinstalable .exe en Windows o un sh en Linux.

Se puede presentar otra posibilidad, y es que la distribución se produzca a través de un fichero que tenga empaquetada y comprimida la aplicación. Se pueda dar el caso de la distribución a través de fichero iso, los cuales nos van a obligar a crear un CD o DVD con el software de instalación de la aplicación, o bien utilizar programas como Daemon's tool para montar la imagen ISO y poder instalar el software.

Otro tipo de distribución, muy común en Linux, es la posibilidad de descargar desde Internet paquetes deb o rpm. Esto paquetes contienen aplicaciones instalables en nuestra distribución, deb si estamos en Ubuntu. Para poder instalar estos paquetes se puede usar el gestor de paquetes de Ubuntu, bien entorno gráfico, o bien en entorno de consola.

Otra forma de distribuir software a través de web, sería el empaquetado en paquete jar o archivos comprimidos zip o rar. En el caso de la primera forma de distribución, lo único que necesitamos para instalar el paquete, sería disponer de una máquina virtual Java. En el caso de distribuir los paquetes comprimidos, necesitamos en el ordenador cliente un programa para descomprimir compatible. En todos los casos, hay que tener en cuenta que el software que se descarga y se quiere instalar, esté diseñado para nuestro sistema operativo y sea compatible con nuestra arquitectura.

Autoevaluación

En una instalación desde un servidor web:

- ○ El usuario siempre instala directamente la aplicación sin tener que guardarla antes.
- ○ La instalación siempre sigue el mismo procedimiento, independientemente del tipo de fichero a descargar.
- ○ La aplicación se instala automáticamente sólo si se trata de un archivo ejecutable.

Incorrecta. Tiene que guardarla antes en su equipo para poder ser ejecutada, a no ser que se trate de un archivo ejecutable.

No es correcta. El proceso depende del tipo de fichero.

Muy bien. Correcta.

Solución

1. Incorrecto
2. Incorrecto
3. Opción correcta