

# UNIDAD 1: CONFECCIÓN DE INTERFACES DE USUARIO

DISEÑO DE INTERFACES - 2º CFGS  
Autoría: CARLOS SOLANO HIDALGO  
Curso: 2023/24

# Índice

- A. Introducción a las interfaces de usuario
- B. Bibliotecas de componentes disponibles para diferentes sistemas operativos y lenguajes de programación
- C. Herramientas propietarias y libres de edición de interfaces. Entornos y características
- D. Creación de proyectos básicos en diferentes entornos de desarrollo

# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

## ► ¿Qué es una interfaz de usuario?

Una **interfaz de usuario** es el conjunto de elementos hardware y software de una computadora que **presentan información al usuario** y le permiten **interactuar** con dicha información y con el ordenador. Lo ideal es que las interfaces de usuario sean fáciles de usar para que la interacción sea lo más **instintiva e intuitiva** posible. En el caso de los programas informáticos, esto se denomina interfaz gráfica de usuario (GUI).

- Estas interfaces gráficas están compuestas por componentes (botones, listas, cajas de edición de texto, etc.), con los que el usuario interacciona para llevar a cabo su tarea.
- Es **el usuario** el que, en cada momento, **decide** qué es lo siguiente que la aplicación va a hacer a través de su interacción.
- La aplicación debe responder a la interacción del usuario.

# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

## Principios de diseño y elementos

- ▶ Existen una serie principios generales que deben acompañar al diseño e implementación de las GUI:
  - ❑ *Sencilla*
  - ❑ *Intuitiva*
  - ❑ *Clara*
  - ❑ *Predecible*
  - ❑ *Flexible*
  - ❑ *Consistente*
- ▶ Los elementos básicos de una GUI son:
  - ❖ Componentes GUI (widgets)
    - ❑ Objetos visuales del interfaz
    - ❑ Conjunto de componentes anidados: ventanas, contenedores, menús, barras, botones, campos de texto, etc.

# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

## Principios de diseño y elementos

- ❖ Disposición (layout): cómo se colocan los componentes para lograr un GUI cómodo de utilizar
  - ❑ Layout managers: Gestionan la organización de los componentes gráficos de la interfaz
- ❖ Eventos: interactividad, respuesta a la entrada del usuario :
  - ❑ Desplazamiento del ratón, selección en un menú, botón pulsado, etc.
- ❖ Creación de gráficos y texto
- ▶ El escritorio es el contexto más global dentro de la interfaz gráfica de usuario ya que representa el espacio donde se mueve y administra la información. En base a este concepto se agrupan las carpeta, documentos y herramientas en general.

# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

## Desarrollo y tipos de interfaces

- Los antiguos ordenadores eran demasiado lentos para las interfaces gráficas de usuario. Al principio, la gente sólo tenía el CLI (Command-line interface). Con la interfaz de línea de comandos, los usuarios sólo podían emitir comandos en forma de líneas como símbolo del sistema. Esto evolucionó a la TUI (Text-based User Interface) que ahora se utiliza para la instalación de sistemas operativos. El hecho de que los ordenadores fueran cada vez más adoptados por la gente, y que el número de hogares con ordenadores siguiera aumentando, hizo necesario **desarrollar una interfaz fácil de usar para los ordenadores.**

# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

## Desarrollo y tipos de interfaces

- Esto llevó al desarrollo de la GUI (Graphical User Interface), que ahora se ha establecido permanentemente con un mejor rendimiento informático. Otros avances tecnológicos fueron el desarrollo de la VUI (Voice-User Interface), una interfaz que permite a los seres humanos interactuar con los ordenadores a través de una plataforma de voz. En varios juegos de ordenador, como Kinect, los jugadores son actualmente capaces de utilizar un NUI (Natural User Interface). Esta interfaz utiliza el movimiento natural de una persona para controlar el software del juego. Actualmente se está desarrollando BCI (Brain computer Interface), y tiene como objetivo controlar el software a través de los pensamientos de una persona. A continuación veremos más información sobre estas interfaces.

# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

## Desarrollo y tipos de interfaces





# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

## Interfaz de línea de comandos (CLI)

- ▶ Entre los lugares donde se utiliza la interfaz de línea de comandos (Command Line Interface en inglés), se encuentran los ordenadores DOS (Disk Operating System). El usuario ve una línea de comandos y un indicador que indica la posición actual.
- ▶ La interacción sólo es posible mediante la **introducción de comandos**. La máquina los procesa y, a continuación, muestra otra línea con una indicación de entrada. Este tipo de sistema operativo está obsoleto. Las CLI han sido reemplazadas en gran medida por las GUI.

# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

## Interfaz de usuario de texto (TUI)

- ▶ Una interfaz de usuario de texto (Text User Interface), está orientada a los caracteres. La ejecución se realiza en modo texto de hardware, pero la pantalla también se utiliza ampliamente.
- ▶ El programador sólo tiene 256 caracteres en una fuente. La navegación se realiza normalmente con **el teclado y no con el ratón**.
- ▶ Algunos ejemplos son el Norton Commander o Turbo Pascal a partir de la versión 5.0. Además, esta interfaz también se utiliza en todos los programas de configuración de la BIOS. La instalación de sistemas operativos también utiliza este tipo de interfaz.

# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

## Interfaz de gráfica de usuario (GUI)

- La interfaz gráfica de usuario (Graphical User Interface) es la interfaz más utilizada en la mayoría de las aplicaciones de software modernas. Se refiere a la ventana que contiene todos los elementos del software y que permite la interacción entre el usuario y el software a través del ratón y el teclado. También se pueden utilizar botones y menús en la ventana del software. Tales elementos también permiten un proceso algo similar a través de diferentes sistemas operativos para interacciones comunes. El diseño de una interfaz gráfica de usuario puede determinarse con la ayuda de un diseño de pantalla.

# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

## GUI- Modelo al mundo real

- ▶ Al desarrollar las primeras GUI, se utilizaron bits del mundo real como modelo para hacer más comprensible el funcionamiento del software. Esto se refleja principalmente en los símbolos utilizados: una papelera de reciclaje, una carpeta o un icono de disco para guardar. Actualmente, la mayoría de estas imágenes están anticuadas, pero siguen utilizándose. Se utilizan elementos comunes como "Escritorio" o "Carpeta".
- ▶ Con las nuevas imágenes también se debe crear siempre una referencia a las cosas que se conocen. Esto facilita la interacción con el usuario. El objetivo de la GUI es hacer posible que **la gente reconozca visualmente lo que hace un botón**. Como resultado, los usuarios no tienen que memorizar todos los comandos como en el caso de los CLIs.

# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

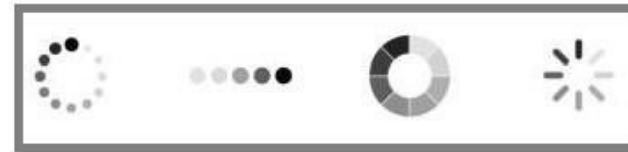
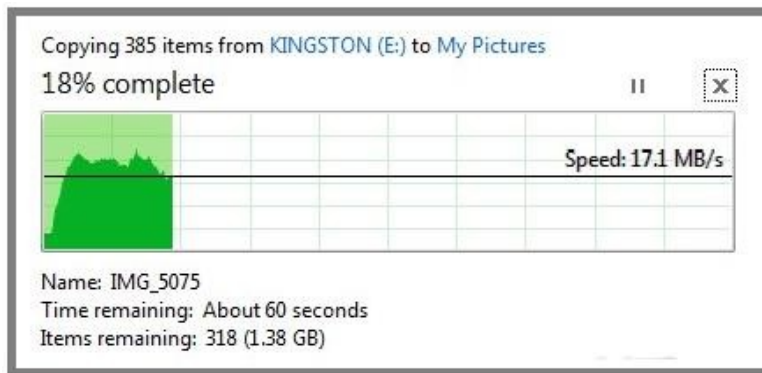
## GUI- Pautas

- ▶ Cuando se diseñan interfaces gráficas de usuario, existen pautas que ayudan a mejorar la facilidad de uso así como la estandarización. Ejemplos de ello son las ocho reglas de oro de Ben Shneiderman. A continuación se describen algunas de estas reglas:
  - ❑ **Consistencia:** Consistencia significa básicamente que una interacción debe ocurrir siempre de la misma manera. Aquí debe evitarse lo siguiente: colocar un menú entre botones con otras funcionalidades o el botón de apagar al principio. Estos ejemplos muestra inconsistencia y debe evitarse en una GUI.

# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

## GUI- Pautas

- ❑ **Feedback informativa:** Cada acción del usuario debe ir siempre seguida de una retroalimentación. Por ejemplo, si al hacer doble clic se abre un programa determinado, es posible que el usuario tenga que esperar unos segundos antes de poder utilizar el software. Para que el usuario sepa que el doble clic funcionó, es necesaria la retroalimentación. Esto puede ser en forma de un cambio en la forma del cursor. El ejemplo más antiguo es el reloj de arena de Windows.



- ❑ **No abrumes la memoria a corto plazo de los usuarios:** Los usuarios no pueden memorizarlo todo. Para diálogos largos que se extienden a lo largo de varias ventanas cambiantes, la información debe mostrarse siempre en la misma posición y ninguna información, que estaba disponible al principio, debe faltar al final. Ej: las ventanas que aparecen a lo largo de una instalación

# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

## Interfaz de usuario de voz (VUI)

- ▶ En una interfaz de usuario de voz (Voice User Interface), la interacción entre el usuario y la máquina se produce a través de la entrada y salida de voz. Por ejemplo, un usuario puede seleccionar verbalmente a una persona de una agenda telefónica guardada para llamar a la persona. Las aplicaciones de voz a texto o el software de reconocimiento de voz también utilizan la **interfaz controlada por voz**. La ventaja de esta forma de interacción es que los usuarios no necesitan nada más que la voz. La introducción de texto en dispositivos que tienen un teclado pequeño (smartphones) también se puede facilitar utilizando interfaces de usuario de voz.
- ▶ Algunos ejemplos son el asistente de Apple, Siri o S-Voice de Samsung.

# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

## Interfaz de usuario tangible (TUI)

- ▶ Con interfaces de usuario tangibles (Tangible User Interface), la interacción tiene lugar a través de dados, pelotas y otros objetos físicos. Los TUIs se encuentran raramente en la vida diaria, pero su desarrollo ha avanzado significativamente. La razón por la que se encuentran en contadas ocasiones es porque la interacción usando objetos físicos ya no funciona si los objetos no pueden ser localizados. Los museos y las exposiciones son buenos ejemplos de áreas en las que resulta útil disponer de interfaces de usuario tangibles. Los objetos físicos de una TUI fomentan la interacción.





# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

## Interfaz de usuario natural (NUI)

- La interfaz de usuario natural (Natural User Interface), debe permitir una interacción del usuario lo más natural e intuitiva posible. Al mismo tiempo, la interfaz real apenas es visible, por ejemplo, en una pantalla táctil. Con los NUIs, la entrada del usuario se hace usando gestos y toques. También es posible la combinación con la VUI. Gracias a la retroalimentación directa del dispositivo, la operación parece *más natural* que la entrada con un ratón y un teclado. Además del uso con pantallas táctiles, los NUIs también se utilizan en videojuegos.



- Por ejemplo, la Nintendo Wii permite acciones en la pantalla moviendo el mando con la mano. Otro ejemplo es una extensión de Xbox con Kinect que permite controlar un personaje en la pantalla a través del propio movimiento corporal.

# A.INTRODUCCIÓN A LAS INTERFACES DE USUARIO

## Interfaz de usuario perceptiva (PUI)

- ▶ Una interfaz de usuario perceptiva (Perceptual User Interface), es una interfaz de usuario orientada a la percepción que todavía está siendo explorada.
- ▶ Las PUI deben ser capaces de combinar los conceptos tanto de la interfaz gráfica de usuario como de la interfaz de usuario de voz e incorporar el reconocimiento electrónico de gestos para facilitar la interacción con el ordenador. La adición de los sentidos auditivo y visual de los gestos, esta interfaz tiene como **objetivo mejorar la percepción aún más.**

## Interfaz de cerebro-ordenador (BCI)

- ▶ La interfaz del cerebro-ordenador (Brain Computer Interface), usa **pensamientos humanos**. Hasta ahora, ha habido mucho éxito en este área y la investigación es muy prometedora. La investigación se basa en diferentes áreas de aplicación. Los electrodos se utilizan para medir las ondas cerebrales que a su vez se calculan utilizando varios algoritmos. Esto permite controlar los brazos robóticos, etc. Este tipo de interacción es un gran alivio para las personas con discapacidad en su vida diaria. Una implementación similar, que pretende controlar un vehículo a través del pensamiento humano, también se está trabajando en el campo del automóvil.



# Índice

B. Bibliotecas de componentes disponibles para diferentes sistemas operativos y lenguajes de programación

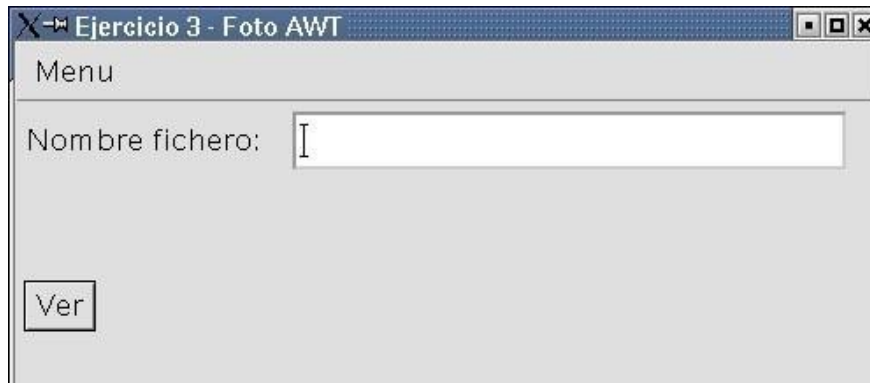
## B. BIBLIOTECAS DE COMPONENTES DISPONIBLES PARA DIFERENTES SISTEMAS OPERATIVOS Y LENGUAJES DE PROGRAMACIÓN

- ▶ Una **biblioteca** es un conjunto de subprogramas con una interfaz bien definida que son utilizados para desarrollar software. Las bibliotecas contienen código y datos, los cuales pueden modificarse de forma modular. Los ejecutables y las bibliotecas forman referencias (llamadas enlaces) entre sí.
- ▶ Existen diferentes bibliotecas para realizar el diseño de interfaces, unas de ellas ligadas al sistema operativo donde se mostrará dicha interfaz y otras propias del lenguaje de programación.
- ▶ Algunas bibliotecas significativas que veremos a continuación son: AWT, Swing, SWT, SwingX, JavaFX, Apache Pivot y Qt Gambi.

## B. BIBLIOTECAS DE COMPONENTES DISPONIBLES PARA DIFERENTES SISTEMAS OPERATIVOS Y LENGUAJES DE PROGRAMACIÓN

### AWT

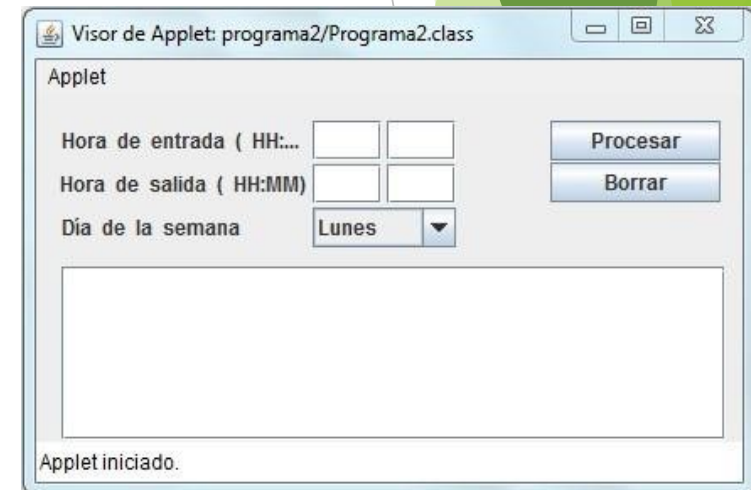
- ▶ AWT (Abstract Window Toolkit) es una biblioteca diseñada en Java. Sirve como base y referencia de una biblioteca más reciente y popular, Swing. AWT carece de controles avanzados por lo que es una biblioteca más enfocada a diseños prácticos y eficientes frente a diseños con grandes efectos visuales.
- ▶ Hoy en día es una biblioteca obsoleta



## B. BIBLIOTECAS DE COMPONENTES DISPONIBLES PARA DIFERENTES SISTEMAS OPERATIVOS Y LENGUAJES DE PROGRAMACIÓN

### Swing

- ▶ Biblioteca basada también en Java y creada a partir de la biblioteca anterior, AWT. La biblioteca Swing forma parte de la JFC (Java Foundation Classes), buscando resolver las deficiencias que de AWT.
- ▶ Algunas de las nuevas características son:
  - ❑ Aspecto modificable (look and feel): Se puede personalizar el aspecto de las interfaces
  - ❑ Contenedores anidados: Cualquier componente puede estar anidado en otro.
  - ❑ Gestión mejorada de la entrada del usuario
  - ❑ Diálogos personalizados
  - ❑ Amplia variedad de componentes



# B. BIBLIOTECAS DE COMPONENTES DISPONIBLES PARA DIFERENTES SISTEMAS OPERATIVOS Y LENGUAJES DE PROGRAMACIÓN

## SWT

- ▶ La biblioteca SWT (Standard Widget Toolkit) fue creada por la compañía IBM para eclipse. Se puede considerar como la tercera generación después de AWT y Swing.
- ▶ Es una biblioteca de bajo nivel y utiliza widgets nativos de la propia plataforma donde se ejecuta a través de JNI (Java Native Interface). Con el mismo código visualizaremos en cada sistema operativo nuestras ventanas como si hubieran sido creadas para ese SO en específico. Por otra parte, al ejecutarse a través de la JNI, hace que las interfaces basadas en SWT no se puedan ejecutar en todas las plataformas.
  - ▶ JNI. Java Native Interface clase interactúa directamente con el Sistema Operativo en el que estemos.
  - ▶ Display. Esta clase es la responsable de traducir o comunicar la Clase Shell con el JNI.
  - ▶ Shell. Clase de SWT, que representa una ventana y es la responsable de administrar los widgets (componentes) que tendremos en la misma. Tendremos una instancia de la clase Shell por cada ventana de nuestro proyecto. Para poder instanciar la clase Shell debemos hacer referencia a un Display anteriormente instanciado.
- ▶ Algunas de las aplicaciones creadas con SWT son Eclipse IDE y Azareus.

## B. BIBLIOTECAS DE COMPONENTES DISPONIBLES PARA DIFERENTES SISTEMAS OPERATIVOS Y LENGUAJES DE PROGRAMACIÓN

### SWING vs SWT

- ▶ Swing esta escrito totalmente en Java debido a que es una biblioteca grafica de Java, con lo cual es portable y puede correr en cualquier plataforma.
- ▶ **SWT es más rápida** ya que utiliza las bibliotecas nativas de la interfaz grafica de usuario del sistema operativo, permitiendo obtener el look-and-feel de dicha plataforma.
- ▶ El hecho de que **Swing** no utilice la GUI del sistema operativo, hace posible **mayor flexibilidad**, con lo cual nos brinda una apariencia que no tiene porque ser igual a la del SO.



## B. BIBLIOTECAS DE COMPONENTES DISPONIBLES PARA DIFERENTES SISTEMAS OPERATIVOS Y LENGUAJES DE PROGRAMACIÓN

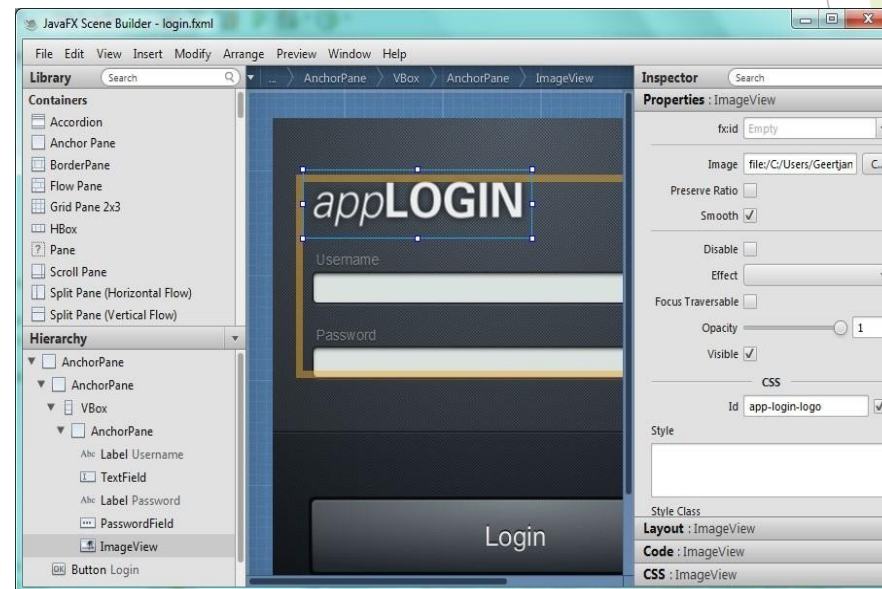
### SwingX

- ▶ Biblioteca basada en Swing que desarrolla gran parte de sus componentes sobre los ya existentes en swing.
- ▶ SwingX está enfocada al desarrollo de aplicaciones RIA. Las aplicaciones RIA (Rich Internet Application) buscan obtener el mismo funcionamiento de apariencia y funcionalidad que las aplicaciones tradicionales de escritorio buscando una mejor experiencia del usuario. Entre sus ventajas desatacamos que no necesitan mucho tiempo en cargar, la mejora audiovisual y con ello la facilidad de uso de la aplicación. Como ejemplos podemos encontrar aplicaciones como **Gmail, Twitter o Facebook**.

## B. BIBLIOTECAS DE COMPONENTES DISPONIBLES PARA DIFERENTES SISTEMAS OPERATIVOS Y LENGUAJES DE PROGRAMACIÓN

### JavaFX

- ▶ Es una plataforma de código abierto enfocada en el desarrollo de aplicaciones RIA (Aplicación de internet enriquecida), aplicaciones web que tienen las características y capacidades de aplicaciones de escritorio. Permite su uso en diferentes dispositivos y plataformas. Agrupa las tecnologías conocidas como JavaFX Script y JavaFX Mobile.
- ▶ Se eliminó del JDK de Java a partir de la versión 11.
- ▶ En la actualidad estamos en la versión 18.



## B. BIBLIOTECAS DE COMPONENTES DISPONIBLES PARA DIFERENTES SISTEMAS OPERATIVOS Y LENGUAJES DE PROGRAMACIÓN

### Apache Pivot

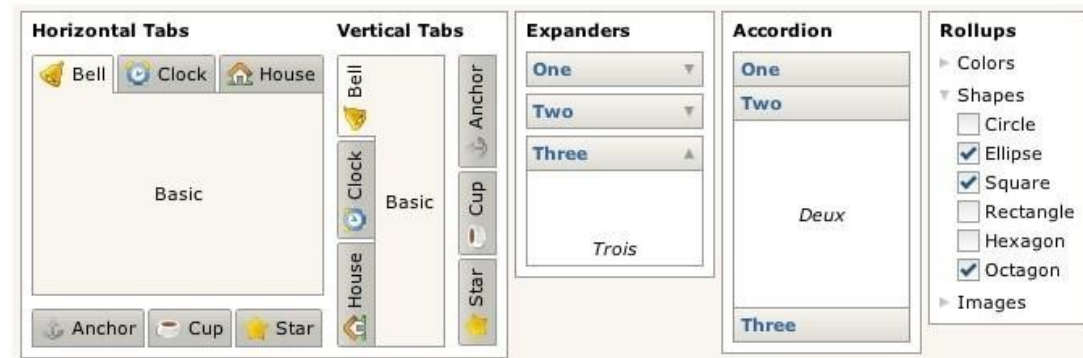
- Es una biblioteca de código abierto orientada al diseño de aplicaciones RIA basadas en Java o cualquier lenguaje apoyado sobre la máquina virtual de Java.
- Combina las características de usabilidad de una RIA moderna con la robustez de la plataforma Java. La robustez en un programa informático hace referencia a su capacidad para hacer frente a errores mientras se está ejecutando.

Las clases se agrupan en:

- Core - No UI clases
- WTK - Clases para el desarrollo de la interfaz de usuario que incluyen ventanas, diálogos, botones, listas etc.
- Web - Clases que posibilitan la implementación y comunicación con servicios de datos remotos.
- Charts - Clases para añadir gráficos interactivos.

No hay una versión estable nueva desde 2017

Diseño de interfaces



## B. BIBLIOTECAS DE COMPONENTES DISPONIBLES PARA DIFERENTES SISTEMAS OPERATIVOS Y LENGUAJES DE PROGRAMACIÓN

### Qt

- ▶ Es una biblioteca multiplataforma para desarrollar GUI, aunque también permite el desarrollo de programas sin interfaz gráfica.
- ▶ Era un software libre y de código abierto escrito en C++ de forma nativa, aunque se puede usar en otros lenguajes de programación. Ahora también está disponible bajo licencia comercial.
- ▶ El API de la biblioteca cuenta con accesos a bases de datos mediante SQL, uso de XML, soporte de red, manipulación de archivos, etc.
- ▶ Se usa en sistemas informáticos embebidos para automoción, aeronavegación y aparatos domésticos como frigoríficos.
- ▶ Usado por Google Earth, Photoshop, VLC...

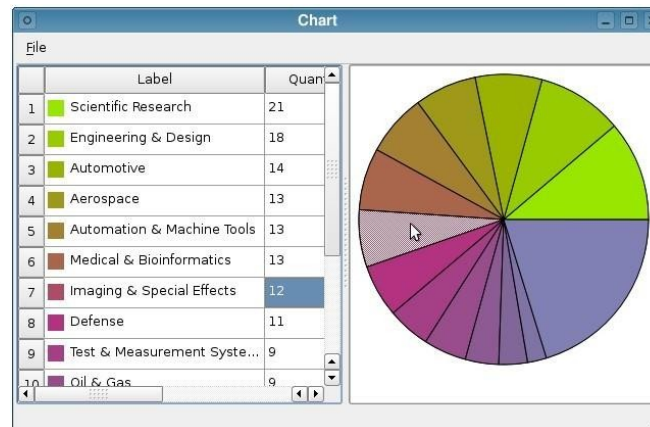
## B. BIBLIOTECAS DE COMPONENTES DISPONIBLES PARA DIFERENTES SISTEMAS OPERATIVOS Y LENGUAJES DE PROGRAMACIÓN

### Qt Jambi

- ▶ Está escrita en lenguaje Java. Usa bibliotecas nativas por lo que no es apta para todas las plataformas. No obstante es compatible con Linux, OS X y Windows, entre otros. Qt Jambi es la librería de Qt para Java.
- ▶ Actualmente es de código abierto.
- ▶ Es una biblioteca que tuvo gran aceptación debido a la gran cantidad de componentes prediseñados GUI que posee y cuenta con una API sencilla de usar pero no tiene actualizaciones desde 2015.

Documentación de referencia en:

[https://doc.qt.io/archives/qtjambi-4.5.2\\_01/com/trolltech/qt/qtjambi-index.html](https://doc.qt.io/archives/qtjambi-4.5.2_01/com/trolltech/qt/qtjambi-index.html)



## B. BIBLIOTECAS DE COMPONENTES DISPONIBLES PARA DIFERENTES SISTEMAS OPERATIVOS Y LENGUAJES DE PROGRAMACIÓN

### OpenGL

- ▶ OpenGL (Open Graphics Library) es una especificación estándar que define una API multilenguaje y multiplataforma para escribir aplicaciones que produzcan gráficos 2D y 3D. La interfaz consiste en más de 250 funciones diferentes que pueden usarse para dibujar escenas tridimensionales complejas a partir de primitivas geométricas simples. Una especificación es un documento que describe un conjunto de funciones y el comportamiento exacto que deben tener. Partiendo de ella, los fabricantes de hardware crean implementaciones, que son bibliotecas de funciones que se ajustan a los requisitos de la especificación.
- ▶ Se usa en CAD (Diseño Asistido por Ordenador), realidad virtual, diseño aeroespacial o en el desarrollo de videojuegos entre otras muchas aplicaciones.

## B. BIBLIOTECAS DE COMPONENTES DISPONIBLES PARA DIFERENTES SISTEMAS OPERATIVOS Y LENGUAJES DE PROGRAMACIÓN

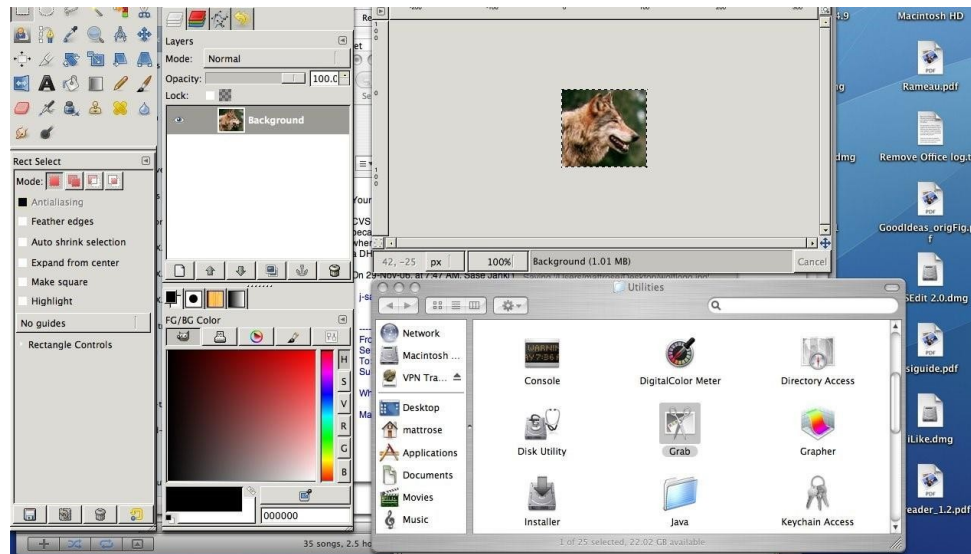
### Api DirectX

- ▶ DirectX es una colección de API desarrolladas para facilitar las complejas tareas relacionadas con multimedia, especialmente programación de videojuegos y vídeo en la plataforma Microsoft Windows.
- ▶ Las últimas versiones incluyen optimizaciones para aprovechar los procesadores con varios núcleos.

## B. BIBLIOTECAS DE COMPONENTES DISPONIBLES PARA DIFERENTES SISTEMAS OPERATIVOS Y LENGUAJES DE PROGRAMACIÓN

### Gtk (hasta febrero 2019 Gtk+)

- ▶ Es un conjunto de bibliotecas multiplataforma (Linux, Mac, Windows) para desarrollar interfaces gráficas de usuario. Está licenciado bajo los términos LGPL y permite la creación tanto de software libre como software propietario.
- ▶ El uso de GTK+ lo podemos encontrar en entornos como GNOME, algunas aplicaciones para desarrollar sus interfaces y gran variedad de lenguajes de programación, Java, C++ o Python entre otros.





# Índice

C. Herramientas propietarias y libres de edición de interfaces. Entornos y características

## C. HERRAMIENTAS PROPIETARIAS Y LIBRES DE EDICIÓN DE INTERFACES. ENTORNOS Y CARACTERÍSTICAS

### Eclipse

- ▶ **Eclipse** es uno de los entornos más conocidos y utilizados por los programadores, ya que se trata de un entorno de programación de código abierto y multiplataforma.
- ▶ Está soportado por una comunidad de usuarios lo que hace que tenga muchos plugins de modo que hacen que nos sirva para casi cualquier lenguaje, en este aspecto es de lo mejores. Sirve para Java, C++, PHP, Perl y un largo etcétera. También nos permite realizar aplicaciones de escritorio y aplicaciones web por lo que nos brinda una gran versatilidad.

# C. HERRAMIENTAS PROPIETARIAS Y LIBRES DE EDICIÓN DE INTERFACES. ENTORNOS Y CARACTERÍSTICAS

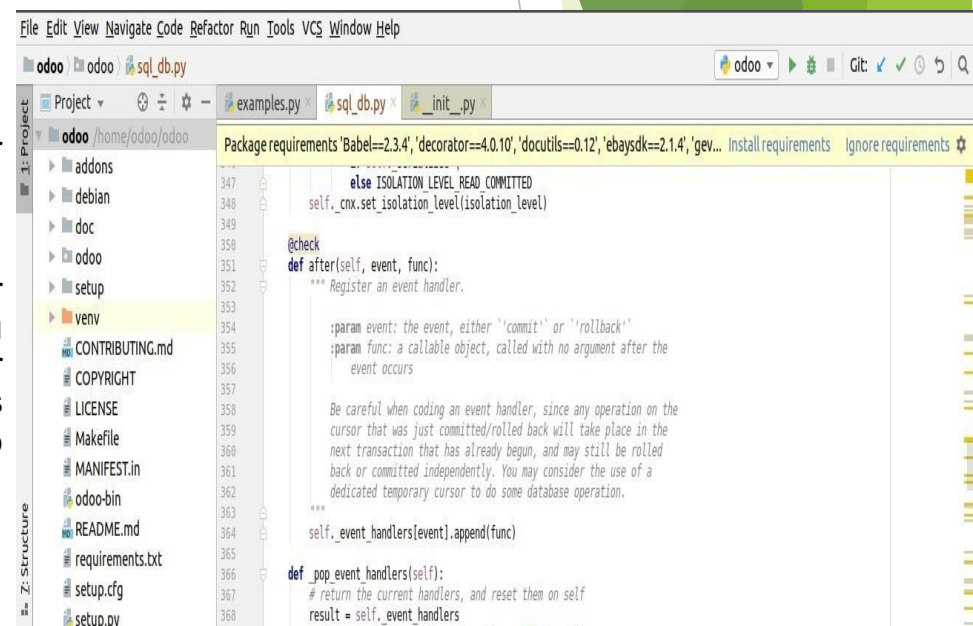
## Netbeans

- **Netbeans** también es un entorno de programación muy utilizado por los programadores. Se trata de otro entorno multilenguaje y multiplataforma en el cual podemos desarrollar software de calidad. Con él podemos crear aplicaciones web y de escritorio, además de contar con plugins para trabajar en Android.
- El lenguaje que mejor soporta es Java, ya que fue creado por Oracle. Aunque como hemos dicho, es multilenguaje debido a que soporta JavaScript, HTML5, PHP, C/C++ etc.

# C. HERRAMIENTAS PROPIETARIAS Y LIBRES DE EDICIÓN DE INTERFACES. ENTORNOS DE Y

## pyCharm

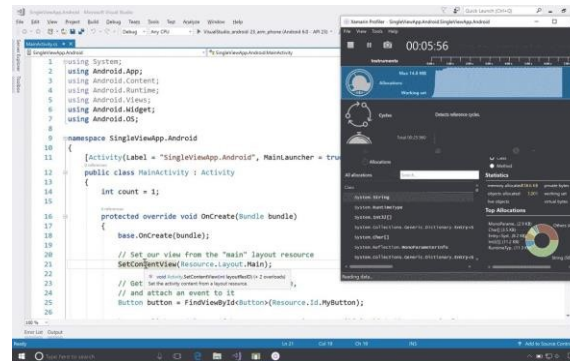
- ▶ Es uno de los mejores entornos de desarrollo integrados para Python.
- ▶ Tiene versiones para Windows y Gnu/Linux
- ▶ Tiene depurador, interprete y otras herramientas que nos ayudarán a crear y exportar los programas que creemos.
- ▶ Trabajar con *PyCharm* tiene ventajas básicas (similares a las ofrecidas por otros IDE) pero también algunas específicas a las cuales debe su popularidad. *PyCharm* tiene un **editor inteligente**, que permite completar código con algunos atajos de teclado. Asimismo, permite navegar a través de nuestro código, saltando entre las clases y métodos creados, haciendo el flujo de trabajo mucho más dinámico.
- ▶ Tiene la posibilidad que tiene de **refactorizar el código**, que en términos generales, significa modificar el código sin comprometer la ejecución del mismo.
- ▶ Los desarrolladores que trabajan con *PyCharm* han facilitado que tenga una **gran cantidad de temas y plugins** que se pueden usar para trabajar más cómodamente.



# C. HERRAMIENTAS PROPIETARIAS Y LIBRES DE EDICIÓN DE INTERFACES. ENTORNOS Y CARACTERÍSTICAS

## Visual Studio

- ▶ **Visual Studio** fue diseñado por Microsoft y es uno de los mejores entornos de programación que existe siempre y cuando utilices sus lenguajes. Antiguamente tenían una versión de pago que incluía todos los lenguajes, y versiones express que eran gratuitas para un lenguaje en concreto.
- ▶ Microsoft ha creado también el Visual Studio Community que es muy parecido al Visual Studio de pago,. Este entorno nos permite hacer aplicaciones web, de escritorio y ayuda sólo que este está soportado por la comunidad al programador. El inconveniente que tiene es que solo es válido para lenguajes de Microsoft.



# C. HERRAMIENTAS PROPIETARIAS Y LIBRES DE EDICIÓN DE INTERFACES. ENTORNOS Y CARACTERÍSTICAS

## Qt Creator

- **QtCreator** es un entorno de programación para C++ usan el framework de QT, es un entorno amigable. También es un entorno multiplataforma programado en C++, JavaScript y QML. Este IDE está diseñado específicamente para utilizar el framework de QT, que por otra parte es un muy interesante ya que nos permite hacer aplicaciones multiplataforma de una manera sencilla y rápida.

# C. HERRAMIENTAS PROPIETARIAS Y LIBRES DE EDICIÓN DE INTERFACES. ENTORNOS Y CARACTERÍSTICAS

## Android Studio

- **Android Studio** es un entorno de desarrollo integrado bajo la licencia Apache 2.0 para desarrollar principalmente aplicaciones móvil y de escritorio, basados en el lenguajes Java para el sistema operativo Android. El IDE puede ejecutarse para su uso en los sistemas operativos como: GNU/Linux, Windows, Mac.

## C. HERRAMIENTAS PROPIETARIAS Y LIBRES DE EDICIÓN DE INTERFACES. ENTORNOS DE Y CARACTERÍSTICAS

### CodeLite

- **CodeLite** es un IDE de código abierto y libre bajo la licencia GNU (General Public License) y para diversos sistemas operativos, el entorno de desarrollo integrado usa wxWidgets para su interfaz gráfica, ya que al cumplir con la filosofía de código abierto usa herramientas completamente libres. A día de hoy soporta los lenguajes C/C++, PHP y Node.js



# Índice

D. Creación de proyectos básicos en diferentes entornos de desarrollo

## D. CREACIÓN DE PROYECTOS BÁSICOS EN DIFERENTES ENTORNOS DE DESARROLLO.

- ▶ Apache NetBeans 11.1, julio 2019.
- ▶ JDK 8, 9, 10, 11, and 12
- ▶ Concepto de Maven: Cuando queremos crear un Proyecto, normalmente usaremos una librería o un conjunto de éstas. Las librerías a su vez pueden depender de otras librerías de las que debemos tener la información de version, requisitos, dependencias...Maven solventa esta problema a través del concepto de Artefacto. Un Artefacto puede verse como una librería con esteroides. Contiene las clases propias de la librería pero además incluye toda la información necesaria para su correcta gestión (grupo, versión, dependencias etc).
- ▶ Para definir un Artefacto necesitamos crear un fichero POM.xml (Project Object Model) que es el encargado de almacenar toda la información que hemos comentado anteriormente:

## D. CREACIÓN DE PROYECTOS BÁSICOS EN DIFERENTES ENTORNOS DE DESARROLLO.

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
```

```
<modelVersion>4.0.0</modelVersion>
```

```
<groupId>com.genbetadev.proyecto1</groupId>
```

```
<artifactId>proyecto1</artifactId>
```

```
<version>0.0.1-SNAPSHOT</version>
```

```
<packaging>jar</packaging>
```

```
<dependencies>
```

```
<dependency>
```

```
<groupId>log4j</groupId>
```

```
<artifactId>log4j</artifactId>
```

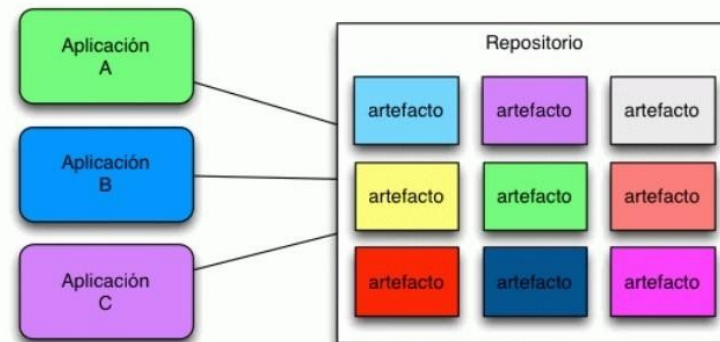
```
<version>1.2.17</version>
```

```
</dependency>
```

```
</dependencies>
```

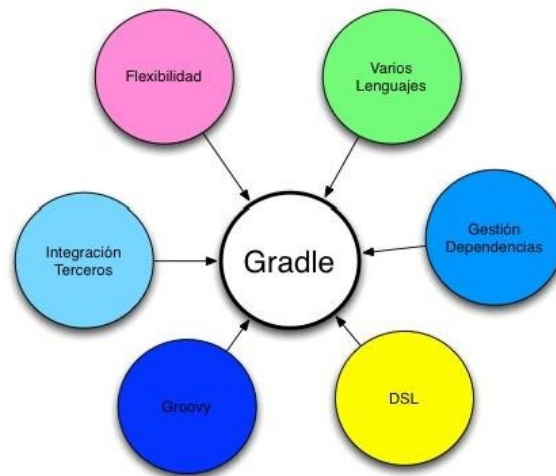
```
</project>
```

La estructura del fichero puede llegar a ser muy compleja y puede llegar a depender de otros POM. En este ejemplo se define el nombre del Artefacto (artifactID) el tipo de empaquetado (jar) y también las dependencias que tiene (log4j). De esta manera nuestra librería queda definida de una forma mucho más clara. Maven nos provee de un Repositorio donde alojar, mantener y distribuir estos.



## D. CREACIÓN DE PROYECTOS BÁSICOS EN DIFERENTES ENTORNOS DE DESARROLLO.

- ▶ Concepto de Gradle: es una herramienta para gestionar la automatización de los builds.
- ▶ Permite trabajar con ella utilizando otros lenguajes **y no solo Java**.



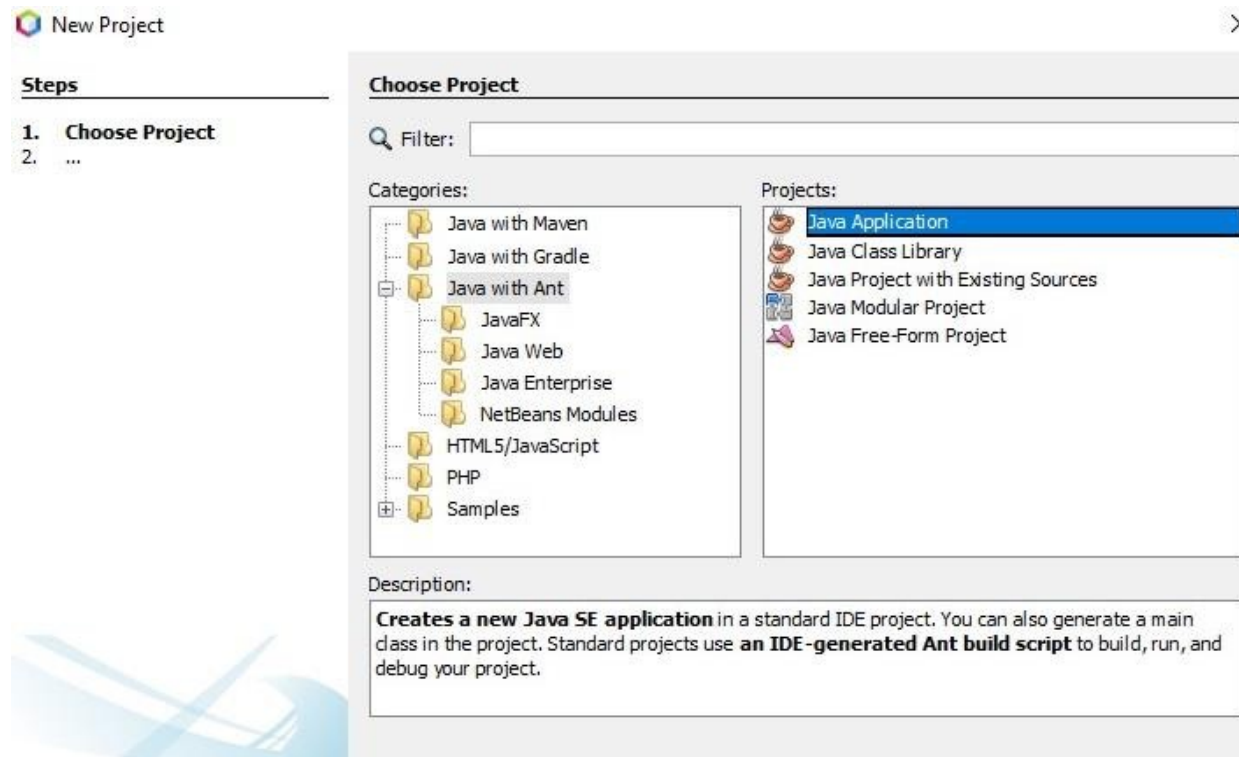
- ▶ Cuando usamos gradle, tenemos un fichero build.gradle que es el encargado de gestionar el build. El fichero se basa en un DSL (Domain Specified Language) que es bastante compacto.

## D. CREACIÓN DE PROYECTOS BÁSICOS EN DIFERENTES ENTORNOS DE DESARROLLO.

- ▶ Ant es una herramienta Open-Source utilizada en la compilación y creación de programas Java.
- ▶ Al construir cualquier programa ejecutable se debe *compilar* el código fuente de éste, generalmente este proceso de compilado implica otras tareas como: revisión de dependencias, creación del archivo ejecutable final y otros detalles.
- ▶ Ant resuelve varios problemas presentes en las herramientas de compilación como *make*, *gnumake* y *jam*. Ant esta escrito en XML y Java , lo que permite ofrecer una solución interoperable al nivel de sistema operativo (debido a Java) y configuraciones descriptivas (debido a XML).

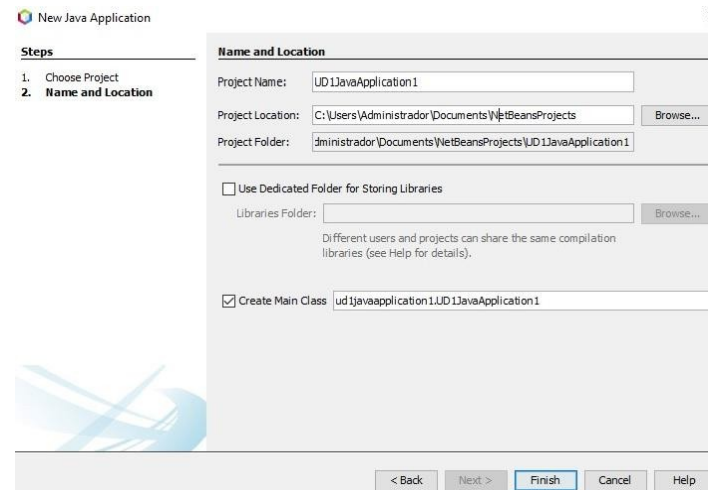
## D. CREACIÓN DE PROYECTOS BÁSICOS EN DIFERENTES ENTORNOS DE DESARROLLO.

- File → New Project → (seleccionar el tipo de proyecto)  
Java with Ant → Java Application

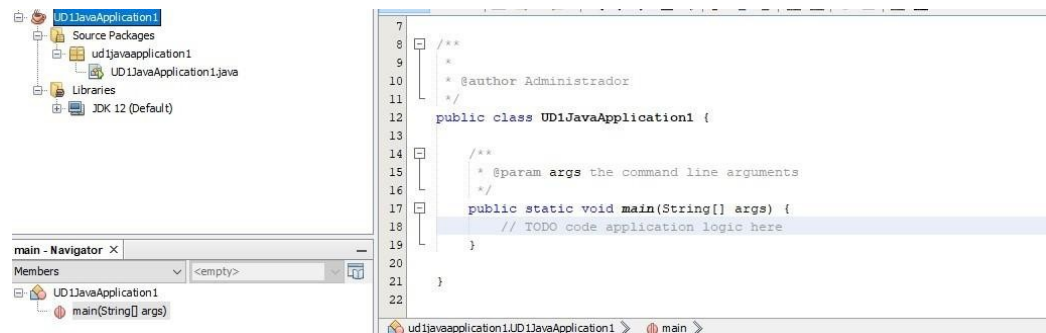


## D. CREACIÓN DE PROYECTOS BÁSICOS EN DIFERENTES ENTORNOS DE DESARROLLO.

- Escribimos el nombre que le queramos dar al proyecto y hacemos clic en finalizar



- Esto nos creará nuestro proyecto con la clase principal

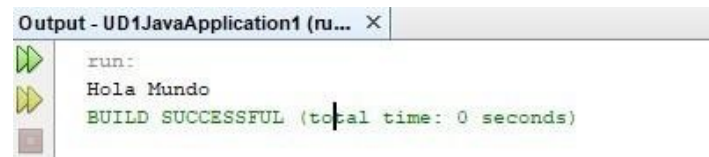



## D. CREACIÓN DE PROYECTOS BÁSICOS EN DIFERENTES ENTORNOS DE DESARROLLO.

- ▶ En la clase principal podemos hacer uso de `System.out` para verificar que efectivamente se ejecuta el proyecto recién creado correctamente. Si añadimos en la clase principal:

- ▶ 

```
public static void main(String[] args) {  
  
    System.out.println("Hola Mundo");  
  
}
```



Y ejecutamos con:  veremos que en la salida de consola aparece: