# CREACIÓN DE COMPONENTES VISUALES

Un componente software es una clase creada para ser reutilizada y que puede ser manipulada por una herramienta de desarrollo de aplicaciones visual. Se define por su estado que se almacena en un conjunto de propiedades, las cuales pueden ser modificadas para adaptar el componente al programa en el que se inserte. También tiene un comportamiento que se define por los eventos ante los que responde y los métodos que ejecuta ante dichos eventos.

Para que una clase sea considerada un componente debe cumplir ciertas normas:

-Debe poder modificarse para adaptarse a la aplicación en la que se integra.
-Debe tener persistencia, es decir, debe poder guardar el estado de sus propiedades cuando han sido modificadas.
-Debe tener introspección, es decir, debe permitir a un IDE que pueda reconocer ciertos elementos de diseño como los nombres de las funciones miembros o métodos y definiciones de las clases, y devolver esa información.
-Debe poder gestionar eventos.

El desarrollo basado en componentes tiene, además, las siguientes ventajas:

-Es mucho más sencillo y se realiza en menos tiempo y con un coste inferior.
-Se disminuyen los errores en el software ya que los componentes se deben someter a un riguroso control de calidad antes de ser utilizados.

Como en cualquier clase, un componente tendrá definido un estado a partir de un conjunto de atributos. Las propiedades son un tipo específico de atributos que representan características de un componente que afectan a su apariencia o a su comportamiento. Son accesibles desde fuera de la clase y forman parte de su interfaz. Suelen estar asociadas a un atributo interno.

Una de las principales características de un componente es que una vez instalado en un entorno de desarrollo, éste debe ser capaz de identificar sus propiedades simplemente detectando parejas de operaciones get/ set(Reflexión), mediante la capacidad denominada introspección. El entorno de desarrollo podrá editar automáticamente cualquier propiedad de los tipos básicos (también Color/Font). Un editor de propiedad es una herramienta para personalizar un tipo de propiedad en particular.

Se puede programar un componente para que reaccione ante determinadas acciones del usuario pero no solo eso, un componente puede también lanzar un evento cuando sea necesario y que su tratamiento se realice en otro objeto.

## INTROSPECCIÓN / REFLEXIÓN

Un componente, como cualquier otra clase dispone de una interfaz normalmente, la interfaz la forman los atributos y métodos públicos. La introspección es una característica que permite a las herramientas de programación visual arrastrar y soltar un componente en la zona de diseño de una aplicación y determinar dinámicamente qué métodos de interfaz, propiedades y eventos del componente están disponibles.

## PERSISTENCIA

A veces, necesitamos almacenar el estado de una clase para que perdure a través del tiempo. A esta característica se le llama persistencia. El mecanismo que implementa la persistencia se llama serialización. Al proceso de almacenar el estado de una clase en un archivo se le llama serializar. Al de recuperarlo después deserializar (Serialización Automatica - Programada).

Una propiedad simple representa un único valor, un número, verdadero o falso o un texto por ejemplo. Tiene asociados los métodos getter y setter para establecer y rescatar ese valor.

Las propiedades indexadas representan un conjunto de elementos, que suelen representarse mediante un vector y se identifica mediante los siguientes patrones de operaciones para leer o escribir elementos individuales del vector o el vector entero.

Los objetos de una clase que tiene una propiedad compartida o ligada notifican a otros objetos oyentes interesados, cuando el valor de dicha propiedad cambia, permitiendo a estos objetos realizar alguna acción. Cuando la propiedad cambia, se crea un objeto que contiene información acerca de la propiedad (su nombre, el valor previo y el nuevo valor), y lo pasa a los otros objetos oyentes interesados en el cambio.

---

HERRAMIENTAS PARA EL DESARROLLO DE COMPONENTES VISUALES

-BeanBox: Es la primera herramienta que se creó para trabajar con componentes de JavaBeans y se podría definir como un contenedor de Beans.
-Bean Builder: En una versión mejorada del BeanBox.
-NetBeans: Es la apuesta actual de Oracle para crear y utilizar Beans.

---

Una vez creado el componente, es necesario empaquetarlo para poder distribuirlo y utilizarlo después. Para ello necesitarás el paquete jar que contiene todas las clases que forman el componente:

-El propio componente.
-Objetos BeanInfo.
-Objetos Customizer.
-Clases de utilidad o recursos que requiera el componente, etc.
-Puedes incluir varios componentes en un mismo jar.

El paquete jar debe incluir un fichero de manifiesto (con extensión .mf) que describa su contenido.