

7.A. Relación entre clases.

1. Relaciones entre clases.

1.2. Herencia.

El mecanismo que permite crear clases basándose en otras que ya existen es conocido como **herencia**. Como ya has visto en unidades anteriores, Java implementa la herencia mediante la utilización de la palabra reservada **extends**.



El concepto de **herencia** es algo bastante simple y sin embargo muy potente: cuando se desea definir una nueva clase y ya existen clases que, de alguna manera, implementan parte de la funcionalidad que se necesita, es posible crear una nueva **clase derivada** de la que ya tienes. Al hacer esto se posibilita la reutilización de todos los atributos y métodos de la clase que se ha utilizado como **base (clase padre o superclase)**, sin la necesidad de tener que escribirlos de nuevo.

Una **subclase** hereda todos los miembros de su **clase padre** (atributos, métodos y clases internas). Los **constructores** no se heredan, aunque se pueden invocar desde la **subclase**.

Algunos ejemplos de herencia podrían ser:

- Un **coche** es un **vehículo** (heredará atributos como la **velocidad máxima** o métodos como **parar** y **arrancar**).
- Un **empleado** es una **persona** (heredará atributos como el **nombre** o la **fecha de nacimiento**).
- Un **rectángulo** es una **figura geométrica** en el plano (heredará métodos como el cálculo de la **superficie** o de su **perímetro**).
- Un **cocodrilo** es un **reptil** (heredará atributos como por ejemplo el **número de dientes**).

En este caso la **expresión idiomática** que puedes usar para plantearte si el tipo de relación entre dos clases A y B es de herencia podría ser "**es un**": "**la clase A es un tipo específico de la clase B**" (**especialización**), o visto de otro modo: "**la clase B es un caso general de la clase A**" (**generalización**).

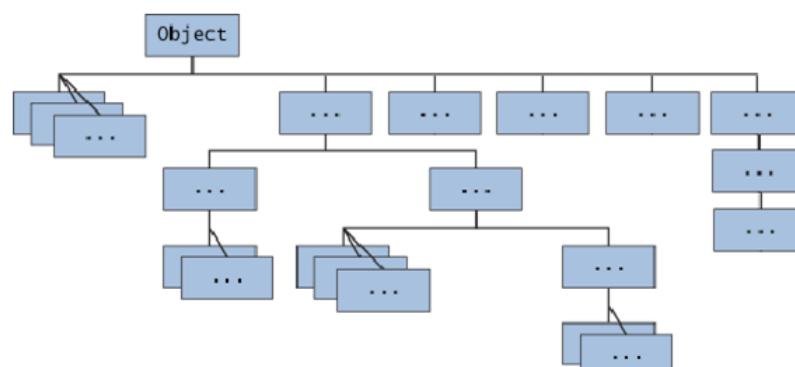
Recuperando algunos ejemplos de clases que ya has utilizado en otras unidades:

• Una **ventana** en una **aplicación gráfica** puede ser una clase que herede de **JFrame** (componente **Swing**: **javax.swing.JFrame**), de esta manera esa clase será un marco que dispondrá de todos los métodos y atributos de **JFrame** mas aquellos que tú decidas incorporarle al rellenarlo de componentes gráficos.

- Una **caja de diálogo** puede ser un tipo de **JDialog** (otro componente **Swing**: **javax.swing.JDialog**).

En Java, la **clase Object** (dentro del paquete **java.lang**) define e implementa el comportamiento común a todas las clases (incluidas aquellas que tú escribas). Como recordarás, ya se dijo que en Java cualquier clase deriva en última instancia de la **clase Object**.

Todas las clases tienen una **clase padre** que a su vez también posee una **superclase**, y así sucesivamente hasta llegar a la **clase Object**. De esta manera, se construye lo que habitualmente se conoce como una **jerarquía de clases**, que en el caso de Java tendría a la **clase Object** en la raíz.



Ejercicio resuelto

Cuando escribas una clase en Java, puedes hacer que herede de una determinada clase padre (mediante el uso de **extends**) o bien no indicar ninguna herencia. En tal caso tu clase no heredar  de ninguna otra clase Java.  Verdadero o Falso?

Soluci n:

No es cierto. Aunque no indiques **expl citamente ning n tipo de herencia**, **el compilador asumir  entonces** de manera **impl cita** que tu clase hereda de la clase **Object**, que define **e implementa el comportamiento com n a todas las clases**.