

7.D. Clases abstractas.

1. Clases abstractas.

1.1. Declaración de una clase abstracta.

Ya has visto que una **clase abstracta** es una clase que no se puede instanciar, es decir, que no se pueden crear objetos a partir de ella. La idea es permitir que otras clases deriven de ella, proporcionando un **modelo genérico** y algunos **métodos de utilidad general**.

Las **clases abstractas** se declaran mediante el **modificador abstract**:

```
[modificador_acceso] abstract class nombreClase [herencia] [interfaces] {  
    ...  
}
```

Una clase puede contener en su interior **métodos declarados como abstract** (métodos para los cuales sólo se indica la cabecera, pero no se proporciona su implementación). En tal caso, la clase tendrá que ser necesariamente también **abstract**. Esos métodos tendrán que ser posteriormente implementados en sus **clases derivadas**.

Por otro lado, una clase también puede contener **métodos totalmente implementados (no abstractos)**, los cuales serán heredados por sus **clases derivadas** y podrán ser utilizados sin necesidad de definirlos (pues ya están implementados).

Cuando trabajes con **clases abstractas** debes tener en cuenta:

- Una **clase abstracta** sólo puede usarse para crear nuevas clases derivadas. No se puede hacer un **new** de una **clase abstracta**. Se produciría un **error de compilación**.
- Una **clase abstracta** puede contener **métodos totalmente definidos (no abstractos)** y **métodos sin definir (métodos abstractos)**.

Autoevaluación

Puede llamarse al constructor de una clase abstracta mediante el operador **new**. ¿Verdadero o Falso?

- ☐ Verdadero
☒ Falso

Ejercicio resuelto

Basándote en la jerarquía de clases de ejemplo (**Persona**, **Alumno**, **Profesor**), que ya has utilizado en otras ocasiones, modifica lo que consideres oportuno para que **Persona** sea, a partir de ahora, una clase abstracta (no instanciable) y las otras dos clases sigan siendo clases derivadas de ella, pero sí instanciables.

Solución:

En este caso lo único que habría que hacer es añadir el modificador **abstract** a la clase **Persona**. El resto de la clase permanecería igual y las clases **Alumno** y **Profesor** no tendrían porqué sufrir ninguna modificación.

```
public abstract class Persona {
```

```
    protected String nombre;
```

```
    protected String apellidos;
```

```
    protected GregorianCalendar fechaNacim;
```

```
    ...
```

```
}
```

A partir de ahora no podrán existir objetos de la clase **Persona**. El compilador generaría un **error**.

Localiza en la API de Java algún ejemplo de clase abstracta.

Solución:

Existen una gran cantidad de **clases abstractas** en la API de Java. Aquí tienes un par de ejemplos:

- La clase **java.awt.Component**:

```
public abstract class Component extends Object  
  
implements ImageObserver, MenuContainer, Serializable
```

- La clase **javax.swing. AbstractButton**:

```
public abstract class AbstractButton extends JComponent  
  
implements ItemSelectable, SwingConstants
```