

Clases, Objetos e Instancias

Abrimos el mes de julio con mucho trabajo y hemos estado lentos en la publicación por este vuestro humilde "web log" pero hicimos un paréntesis con la simple traducción de un artículo duro y contundente a propósito de un trabajo de investigación que venimos desarrollando. Estos conceptos -expuestos de una manera muy práctica y orientados sobre lenguaje Java- los consideramos fundamentales (a pesar que [algunos programadores han anunciado](#) este modelo como en decadencia) por ello los traducimos tal cual y además les agregamos unos enlaces para que veáis cuán profundo es el tema y como de maravilloso [lo simplificaron en el artículo que citamos](#) en idioma castellano a continuación.

Clase versus Objeto versus Instancia

En la Programación Orientada a Objeto frecuentemente escuchamos términos como "Clase", "Objeto" e "Instancia" pero ¿qué es una Clase / Objeto / Instancia en realidad?

En resumen, un objeto es un paquete de software de estado y comportamiento relacionados. Una **Clase** es un modelo o prototipo del que se crean los **Objetos**. Una **Instancia** es una sencilla y singular unidad de una clase.

Por ejemplo, tenemos un proyecto (**Clase**) que representa al "Estudiante" (**Objeto**) con campos como nombre, edad, curso (miembro de la **Clase**). Y tenemos dos estudiantes aquí, José y María. Así, José y María son dos **Instancias** diferentes de la clase "Estudiante" que representan el Objeto (personas Estudiantes). Veamos mayores detalles.

Objeto.

Los objetos del mundo real comparten 2 características principales, estado y comportamiento. Los humanos tienen estado (nombre, edad) y comportamiento (correr, dormir). Los objetos de software son conceptualmente similares a los objetos del mundo real: ellos también consisten en el estado y el comportamiento relacionado. Un **Objeto** almacena su estado en **campos** y expone su comportamiento a través de **métodos**.

Clase.

La clase es una "plantilla" o "modelo" que se utiliza para crear **objetos**. Básicamente, una **clase** consistirá en campo, campo estático, método, método estático y constructor. **Campo** se utiliza para mantener el estado de la clase (por ejemplo: nombre del objeto Estudiante). El **método** se utiliza para representar el comportamiento de la clase (por ejemplo: cómo un objeto Estudiante va a ponerse de pie). **Constructor** se utiliza para crear una nueva instancia de la clase.

Instancia.

Una **Instancia** es una copia única de una clase que representa un **Objeto**. Cuando se crea una nueva **Instancia** de una **Clase**, el ordenador asignará un espacio de memoria para esa **Instancia** de **Clase** (en nuestro ejemplo, José y María tendrán su espacio aparte e independientes en memoria).

Nota: aunque no está en el artículo original, quisieramos agregar algo adicional con respecto a las **Instancias**, y es que pudiera darse el caso que se pudieran compartir los valores de campo fácilmente entre dos o más Instancias y precisamente eso es lo que facilita la programación. Si la **Clase** Estudiante tiene un campo llamado "residencia" (la cual tiene subcampos que contienen la dirección de habitación - estado, ciudad, barrio, calle, casa, etc.- de José y María) y ellos son parejas y viven en la misma casa *pues con solo rellenar los campos de María al finalizar los "igualamos" -copiamos- en un solo paso a la Instancia Pedro ¡fácil!*.

Fuentes consultadas.

En idioma castellano.

- «[Programación orientada a objetos](#)» en Wikipedia.

En idioma inglés.

- «[Class vs Object vs Instance](#)» at Alfred's Blog.
- «[Object Oriented Programming](#)» at Wikibooks.
- «[Introduction to Object Oriented Programming Concepts \(OOP\) and More](#)» by Nirosh.
- «[Goodbye, Object Oriented Programming](#)» by Charles Scalfani.
- «[So You Want to be a Functional Programmer \(Part 1\)](#)» by Charles Scalfani.