CLASES ABSTRACTAS VS INTERFACES

[](http://univia.info/wp-content/uploads/2015/09/9.png)

Una de las preguntas frecuentes que surgen en la POO con Java es la siguiente ¿Cuál es la diferencia entre una clase abstracta y una interface en Java?

Una pregunta de Java sencilla pero no fácil de responder.

Caigo en cuenta que para dar una respuesta digna de mí necesito hablar de clases, instanciación de objetos, herencia, abstracción y polimorfismo.

Ahora bien, para poder mencionar las diferencias empiezo con la siguiente semejanza.

Una interfaz puede considerarse similar una clase completamente abstracta. En muchos casos una interfaz se utiliza de la misma manera que una clase abstracta. De una interfaz o de una clase abstracta no se pueden crear objetos.

**¿Y la diferencia?**

Como Java no permite herencia múltiple – una clase sólo puede extender una superclase – esto dificulta que una clase se adecue a más de un comportamiento. Una interfaz, por el contrario, permite que una clase implemente una o más interfaces para resolver el problema de mezclar diversos comportamientos en un mismo tipo de objeto.

Una clase abstracta ofrece comportamientos comunes a objetos del mismo tipo a través del mecanismo de la herencia. La implementación de una interfaz permite a un objeto comportamientos que no dependen de su jerarquía de clases.

Una interfaz se diferencia de una clase abstracta porque una interfaz sólo puede contener constantes y métodos abstractos. Una clase abstracta puede contener métodos concretos, una interfaz no.

En una interfaz todos los atributos son por defecto public final static (constantes) y todos los métodos son public abstract. Esto quiere decir que una clase abstracta puede contener atributos variables pero una interfaz no.

Para hacerme entender, necesito contar la historia desde sus inicios y por ello supongo que esto va a necesitar varias entradas para el blog. Acabo de decidir que no empiezo por el Génesis sino más bien por el evangelio de Juan.

Parafraseando a Juan antes ya existía Java. Java era un lenguaje de programación multipropósito y él era orientado a objetos. La POO estaba en Java desde el comienzo. Todo esto comenzó aquí y sin él no se habría hecho esta pregunta. El problema era encontrar la luz de la respuesta para que la oscuridad no pudiera contra ella.