Técnico Universitario en Sistemas Informáticos

Laboratorio V

Trabajo Práctico Nº 1

# Integrantes:

* Cristian Benitez
* Pablo Maciel
* German Medina
* Gustavo Pavichevich
* Leonardo Yermoli

**2021**

# Introducción

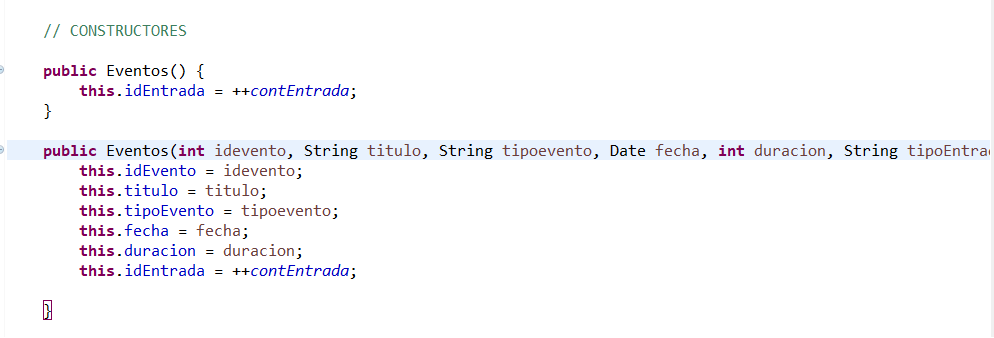
Para el desarrollo de este trabajo práctico nos enfocamos en hacer que el código sea organizado y escalable. Para eso realizamos dos reuniones grupales vía ZOOM y algunas más individuales.

Para poder avanzar y facilitar la integración de los desarrollos individuales, usamos el versionador GIT ([Link al repositorio](https://github.com/gustavopavichevich/TP1_GRUPO_9.git)) tanto desde su integración en eclipse, como por la consola propia del versionador.

# Conceptos

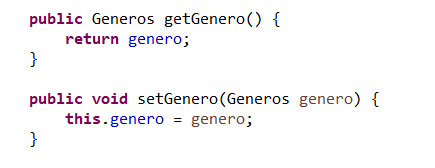
1. Constructores

Son métodos de las clases que se utilizan para inicializar las instancias de las clases. Deben tener el mismo nombre que la Clase. Pueden o no asignarse valores a las propiedades. En este caso tenemos dos constructores, uno que recibe parámetros y otro que no.



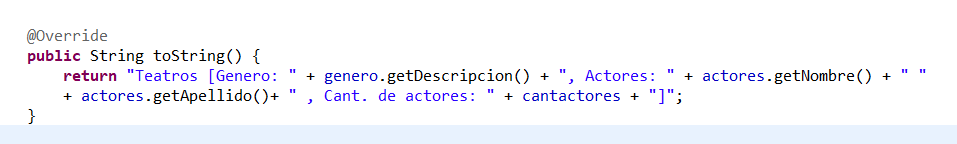
1. Encapsulamiento

Lo aplicamos para no permitir que, en cualquier parte de la ejecución del programa, se pueda acceder a las propiedades de una clase si no es a través de los métodos setters y getters.



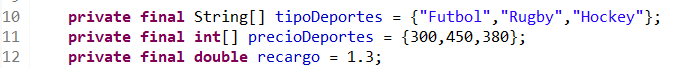
1. Método toString()

Este método se utiliza en las clases para ver información de la misma devolviendo un string con todos los datos que necesitemos.



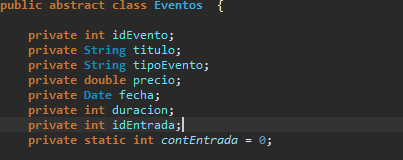
1. Variables constantes

Son variables cuyo valor no debe cambiar durante la ejecución del programa. Se utilizan para que los valores estén harcodeados a lo largo del código y sea más fácil su búsqueda en caso de que haya que modificarlos y recompilar. Se utiliza la palabra reservada “*final*” al definirla.



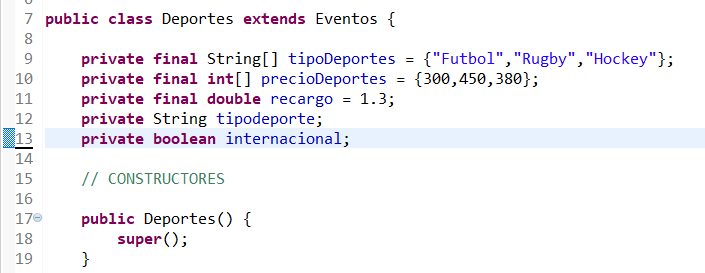
1. Variables estáticas

Se utiliza la palabra “*static”* cuando se quiere que un atributo sea de la clase y no de cada objeto que se instancia de la misma.



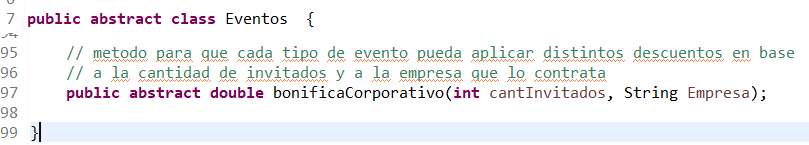
1. Herencia

Utilizamos las herencias cuando hay clases que comparten atributos y métodos, pero a la vez algunas tienen características particulares. Para esto decimos que la clase con características particulares (hija) “hereda” de la clase con características comunes (padre). En este caso, en el constructor usamos el método “*super*” para instanciar los atributos de la clase padre.



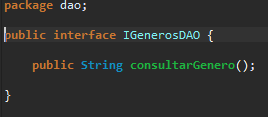
1. Clases y Métodos Abstractos

En el caso de las clases abstractas las definimos así cuando no queremos permitir que se instancien objetos de esa clase, solo que se hereden. Para los métodos, es cuando necesitamos que cada clase que herede, implemente el método según las características propias de esa clase. Para ambos se usa el modificador “*abstract*”.



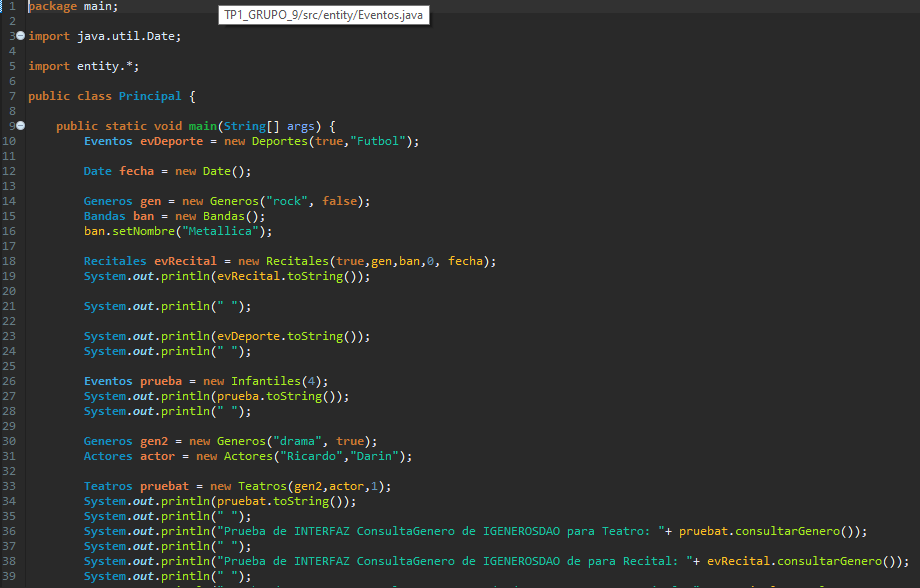
1. Creación de Interfaz

Una interfaz en Java es una colección de métodos abstractos y propiedades constantes. En las interfaces se especifica qué se debe hacer pero no su implementación. Serán las clases que implementen estas interfaces las que describen la lógica del comportamiento de los métodos.



1. Este es el main en donde se crean las entradas a los eventos

El método main en java es un estándar utilizado por la JVM para iniciar la ejecución de cualquier programa Java. Dicho método se conoce como punto de entrada de la aplicación java



1. Polimorfismo

Es la capacidad que tienen los objetos de una clase en ofrecer respuesta distinta e independiente en función de los parámetros (diferentes implementaciones) utilizados durante su invocación.

