14: Acceso a base de datos

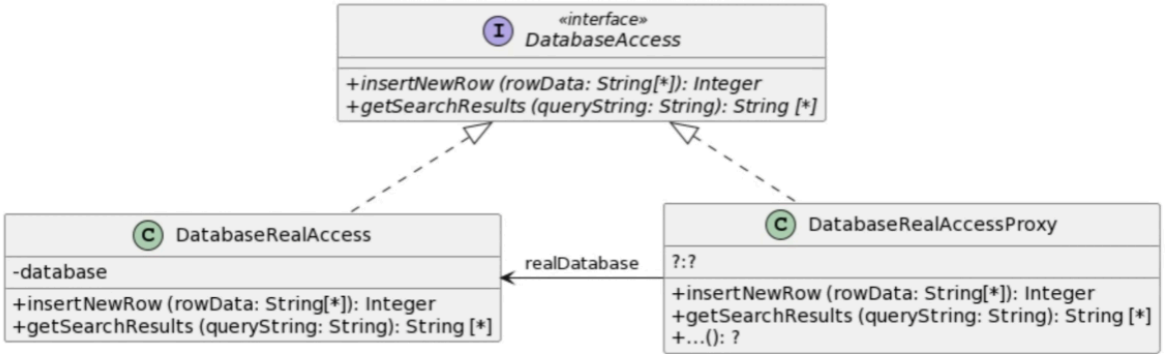
Ejercicio 14 - Acceso a la base de datos

Comportamiento esperado:

```
// Una contraseña muy difícil.
   String password = "p455w0rD"

// Instancia una base de datos que posee dos filas, y se accede con contraseña
   database = new YourClass(... some mysterious parameters ..., password);

// No retorna nada.
   database.getSearchResults("select * from comics where id=1");

// Pero… si previamente realizó el login con la contraseña correcta…
   database.login(password)
// Retorna el siguiente arreglo: ['Spiderman' 'Marvel'].
   database.getSearchResults("select * from comics where id=1");
```

- Enfocarse en la protección de la base de datos
- La “implementación prototípica” sirve para poder “probar”
- Test provistos por la cátedra

Creamos una instancia de una base de datos que no sabemos cual es (parte del ejercicio), le mandamos los parámetros, que no sabemos cuales, entre ellos la **password**.
Después de este punto tenemos una DB con una contra donde podemos preguntarle lo mismo que a la anterior mostrada, si es que hicimos login previamente.
Claramente la idea es implementar una protección a la base de datos usando el patrón **PROXY**. En la página 15 de patrones de diseño.



Podemos implementar usuarios, y otras cosas para solucionarlo, o hacerla simple.
Centrarse en la comunicación entre el proxy y el original.

15: File manager

Ejercicio 15 - File Manager

En un **File Manager** se muestran los archivos. De los archivos se conoce:

- Nombre
- Extensión
- Tamaño
- Fecha de creación
- Fecha de modificación
- Permisos

Implemente la clase **FileOO2**, con las correspondientes variables de instancia y *accessors*.

En el File Manager el usuario debe poder elegir cómo se muestra un archivo (instancia de la clase FileOO2), es decir, cuáles de los aspectos mencionados anteriormente se muestran, y en qué orden. Esto quiere decir que un usuario podría querer ver los archivos de muchas maneras. Algunas de ellas son:

- `nombre - extensión`
- `nombre - extensión - fecha de creación`
- `permisos - nombre - extensión - tamaño`

Son sólo *algunos* ejemplos

Ejercicio 15 - File Manager

Para esto, el objeto o los objetos que representen a los archivos en el FileManager debe(n) entender el mensaje `prettyPrint()`.

Es decir, un objeto cliente (digamos el FileManager) le enviará al objeto que Ud. determine el mensaje `prettyPrint()`. **De acuerdo a cómo el usuario lo haya configurado se deberá retornar un String con los aspectos seleccionados por el usuario en el orden especificado por éste**. Considere que un mismo archivo podría verse de formas diferentes desde distintos puntos del sistema, y que el usuario podría cambiar la configuración del sistema (qué y en qué orden quiere ver) en runtime.

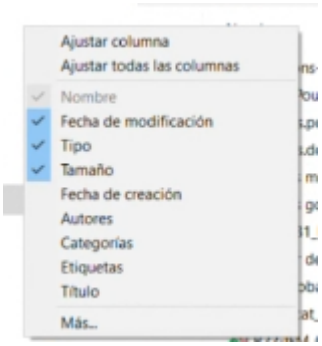
- `nombre - extensión`
- `nombre - extensión - fecha de creación`
- `permisos - nombre - extensión - tamaño`

Tareas

1) Discuta los requerimientos y diseñe una solución. Si aplica un patrón de diseño, indique cuál es y justifique su aplicabilidad.
2) Implemente en Java.
3) Instancie un objeto para cada uno de los ejemplos citados anteriormente y verifique escribiendo tests de unidad.

Las configuraciones de como se ve tiene que poder cambiarse en runtime.

Usamos **decorator**. Página 23 de patrones de diseño.



Los decoradores serían esas cosas que vamos agregando, y su orden. No hay que checar que se ponga 2 veces nombre. (En la vida real si).

Hacer hasta el 16