

TICS200: App #2

Universidad Adolfo Ibañez

Felipe Aguilera
felipe.aguilera@edu.uai.cl

Danilo Bórquez Paredes
danilo.borquez.p@uai.cl

14 de Abril de 2021

Objetivos

- Aprender a estructurar un programa en varios archivos fuente de manera ordenada, lógica y modular utilizando Orientación a Objetos.
- Utilizar herramientas clásicas de apoyo a la programación como GIT, make, debuggers, etc.

1. Biblioteca

La biblioteca de la universidad, en su afán de ser vanguardista y estar constantemente actualizada, ha decidido dar de baja el actual software que administra el inventario de libros (sí, aquel que usted con tanto esfuerzo hizo y que cariñosamente le llamó “app 1”).

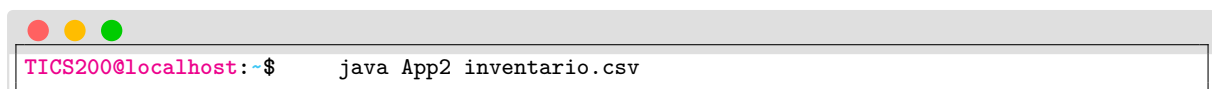
La administración argumenta que el software está pasado de moda, y que el boom de la orientación a objetos aun se encuentra vigente (sí, como no... lleva 25 años vigente...).

Por otro lado, con su antiguo programa usted ha generado una fama creciente dentro de la institución, destacando por sus grandes habilidades en la programación que lo llevan a ser el candidato ideal en esta nueva etapa del software, dada su experiencia previa con el antiguo software. Es por esto último que la biblioteca lo ha contactado a usted (nuevamente) para desarrollar su nuevo sistema de inventario de libros.

Los requisitos de este nuevo sistema serán detallados en la siguiente subsección.

1.1. Requerimientos funcionales del sistema

Su programa consistirá en un conjunto de archivos **.java**, donde su clase principal se llamará **App2.java**. Este programa será ejecutado desde una consola (terminal), con un nombre de archivo como parámetro. Este archivo representará la persistencia de su programa. Su contenido serán los datos que corresponden a los libros del inventario. La línea de código que ejecutará su programa se muestra a continuación, donde *inventario.csv* es el archivo en formato *comma separated values*:



```
TICS200@localhost:~$ java App2 inventario.csv
```

El formato del archivo será de la siguiente manera:

inventario.csv

```

titulo,autor,anio,estante_numero,estante_seccion,piso,
    ↳ edificio,sede
"Structure and Interpretation of Computer Programs","Abelson,
    ↳ Sussman, and Sussman",1996,4,"Lenguajes de
    ↳ Programacion",2,"B","Vina del Mar"
"Introduction to Algorithms","Thomas H. Cormen, Charles E.
    ↳ Leiserson, Ronald L. Rivest, and Clifford Stein
    ↳ ",2009,4,"Estructura de Datos y Algoritmos",1,"A","
    ↳ Santiago"
"Programming Language Pragmatics","Michael L. Scott, Morgan
    ↳ Kaufman",2015,1,"Estructura de Datos y Algoritmos",1,"
    ↳ B","Vina del Mar"

```

Su programa debe cumplir con lo siguiente

1. Al ejecutarse, debe contener un menú (y posibles sub-menús) que permita acceder a las diferentes funcionalidades. Este menú debe ser mostrado por consola.
2. El usuario debe poder agregar un libro nuevo.
3. El usuario debe poder quitar un libro
4. El usuario debe poder agregar una sede
5. El usuario debe poder quitar una sede **sólo si no tiene libros asociados**
6. El usuario debe poder editar un libro
7. El usuario debe poder cambiar un libro de sede
8. El usuario debe poder cambiar un libro de sección
9. El usuario debe poder cambiar un libro de piso
10. El usuario debe poder agregar secciones
11. El usuario debe poder eliminar secciones **sólo si no tiene libros asociados**
12. El usuario debe poder agregar pisos
13. El usuario debe poder quitar pisos **sólo si no tiene libros asociados**
14. El usuario debe poder buscar un libro, y el sistema le entrega la información del mismo junto a su ubicación.
15. Cuando el sistema se cierre (seleccionando la opción salir del menú), se debe guardar toda la información en el archivo de inventario con que se abrió el programa.

1.2. Supuestos

1. Sólo existe una copia de cada libro **en todas las sedes**. Si el libro está en Santiago, entonces no puede estar en ninguna otra sede.
2. El usuario siempre ingresará información válida. No es necesario que el programa valide cada texto ingresado por el usuario.
3. No habrán campos vacíos en el archivo de inventario.

2. Sobre la entrega

- La tarea debe ser hecha en lenguaje de Programación Java.
- Cada grupo puede ser de 4 o 5 personas.
- La tarea se debe entregar el día **Miércoles 12 de Mayo** a las 23:59.
- Por cada día de atraso se descontará 1 punto, comenzando a las 00:00 horas del siguiente día. Por ejemplo si entrega la tarea a las 00:00 del siguiente día, la nota máxima que puede obtener es un 6.0
- Para la corrección se utilizará un compilador java oracle 16 o superior
- La entrega se realiza por la plataforma Webcursos¹
- El archivo a entregar debe ser un zip que contenga una carpeta en su interior (y sólo una carpeta) con el nombre **tarea2**. Dentro de esa carpeta debe haber un README con la información del grupo y sus archivos .java (puede -y se recomienda- utilizar paquetes si lo desea). Además el nombre de su zip debe ser grupoX-app2.zip, donde X es el número de su grupo.

¹<http://webcursos.uai.cl>