NOMBRE Y APELLIDOS:

FECHA: **CURSO**: 2º DAM

MÓDULO: Acceso a datos (AAD)

CALIFICACIÓN

Examen 1ª Evaluación (90 min)

Instrucciones

No se permite el uso de ningún material de consulta ni el acceso a Internet.

Antes de comenzar el examen, el alumno debe:

1. Leer el enunciado del examen completo.
2. Descargar los recursos del AV necesarios para la realización del examen.
3. Desconectar el latiguillo de red de su PC.
4. Cerrar todos los proyectos abiertos del Eclipse (desde la sección "package explorer" -> botón derecho sobre cada proyecto -> close Project)
5. Una vez cerrados todos los proyectos, se deberá aplicar un filtro para ocultar los proyectos cerrados (desde la sección "package explorer" -> View menu -> Filters -> seleccionar "closed projects")

Ejercicios

1. Crea un proyecto llamado AAD_EX01E01_NomApe1 (NomApe1 es tu nombre y primer apellido) donde desarrolles un programa en JAVA que parsee el fichero "Sega_Dreamcast.xml" y que, a partir de él, cree un nuevo fichero de texto plano llamado juegos_NomApe1.txt (NomApe1 es tu nombre y primer apellido) con la siguiente información:

   nombreJuego | desarrollador | año

   Nota: El nombreJuego debe ser el **atributo "name"** de los elementos "game". El desarrollador debe ser el contenido del **elemento**

“manufacturer” y el año debe ser el contenido del **elemento “year”**, ambos dentro de cada elemento “game”

```
nombreJuego|desarrollador|año
102 Dalmatians - Puppies to the Rescue (USA)|Eidos Interactive|2000
18 Wheeler - American Pro Trucker (USA)|Sega|2001
4 Wheel Thunder (USA)|Midway|2000
4x4 Evolution (USA)|Gathering|2000
90 Minutes - Sega Championship Football (Europe)|Sega|2001
Advanced Daisenryaku 2001 (Japan)|Sega|2001
AeroWings (USA)|Crave|1999
AeroWings 2 - Air Strike (USA)|Crave|2000
AirForce Delta (USA)|Konami|1999
Alien Front Online (USA)|Sega|2001
Alone in the Dark - The New Nightmare (USA) (Disc 1)|Infogrames|2001
Alone in the Dark - The New Nightmare (USA) (Disc 2)|Infogrames|2001
```

Para ello usa una clase **Programa** con el main, **JuegoDefaultHandler** como manejador de SAX y una clase adicional llamada **Juego** cuyos atributos sean *titulo* de tipo String, *desarrollador* de tipo String y *anyo* de tipo int.

Cuando *parsees* el fichero XML, ve almacenando el listado de juegos en la estructura de tipo *collection* que más te guste y una vez completes la lectura completa del fichero XML, crea después el nuevo fichero de texto usando como entrada la colección previamente creada.

El parseo o lectura del documento original xml debe hacerse mediante **la API SAX**.

2. Crea un proyecto llamado AAD_EX01E02_NomApe1 (NomApe1 es tu nombre y primer apellido) donde desarrolles un programa en JAVA que parsee el fichero “Sega_Dreamcast.json” y que, a partir de él, almacene en una BDOO llamada **catalogo_NomApe1.db4o** (NomApe1 es tu nombre y primer apellido), objetos de la clase **Genero** (nueva clase)

   Nota: La clase Genero dispondrá de tres atributos: *nombre* de tipo String, *total* de tipo int y *mediaPuntuacion* de tipo float
   - El nombre será la información del elemento “genero” del json.
   - El total será el número de juegos de ese género.
   - La mediaPuntuacion será la media aritmética de elemento “puntuacion”.

   Finalmente, se ofrecerá al usuario la búsqueda de un determinado genero por teclado para visualizar la puntuación media y el número de juegos que tengan ese género en el catálogo.

Criterios de evaluación

El examen puntúa sobre 10 y ambos ejercicios puntúan lo mismo.

Para obtener la máxima puntuación en cada ejercicio, se tendrá en cuenta:

- No hay errores de compilación.
- Cumple con los requisitos funcionales de la práctica.
- Se adelanta y gestiona posibles errores provocados por el usuario controlando número de parámetros, tipos, etc.
- Se entrega en el Aula virtual toda la información requerida, en el plazo establecido y con la nomenclatura y formato exigido.
- Se hace uso de las **buenas prácticas de programación** (evitando hardcodes, parametrizando la aplicación, optimizando el uso de recursos, modularizando funcionalidades en métodos, reutilizando los recursos siempre que sea posible, organizando en clases la información y funcionalidades…)
- Se gestionan correctamente las posibles **excepciones** que se puedan disparar en el programa evitando su propagación desde el método main.
- Se documenta el código haciendo uso de comentarios **JavaDoc** que describen la funcionalidad de las clases y métodos utilizados para los que no sea inmediato conocer su lógica.
- Que el alumno añade **trazas** (con nivel **debug** o **trace**) a sus programas para facilitar futuras correcciones o cambios de alcance en la funcionalidad de los mismos.

La calificación de dicha práctica se tendrá en consideración para la evaluación de los siguientes **Resultados de Aprendizaje** y **Criterios de Evaluación**:

RA1. Desarrolla aplicaciones que gestionan información almacenada en ficheros identificando el campo de aplicación de los mismos y utilizando clases específicas.

c) Se han utilizado clases para recuperar información almacenada en ficheros.
d) Se han utilizado clases para almacenar información en ficheros.
e) Se han utilizado clases para realizar conversiones entre diferentes formatos de ficheros.

f) Se han previsto y gestionado las excepciones.
g) Se han probado y documentado las aplicaciones desarrolladas.

RA4. Desarrolla aplicaciones que gestionan la información almacenada en bases de datos objeto relacionales y orientadas a objetos valorando sus características y utilizando los mecanismos de acceso incorporados.

a) Se han identificado las ventajas e inconvenientes de las bases de datos que almacenan objetos.
b) Se han establecido y cerrado conexiones.
c) Se ha gestionado la persistencia de objetos simples.
e) Se han desarrollado aplicaciones que realizan consultas.
g) Se han gestionado las transacciones.
h) Se han probado y documentado las aplicaciones desarrolladas.

Entrega

Cada alumno/a entregará en tarea habilitada para el examen en el Aula Virtual del módulo dos ficheros por cada ejercicio: un fichero comprimido **en formato jar** donde se incluya el proyecto de eclipse con el paquete y las clases JAVA utilizadas, así como el código fuente. Un segundo fichero en formato "**runnable JAR file**" que incluirá todas las librerías necesarias para la ejecución correcta de la aplicación.

- El primer fichero deberá nombrarse con el nombre del proyecto de cada ejercicio con extensión **jar**.
- El segundo fichero se nombrará igual que el primero, pero añadiendo "runnable" para diferenciarlo del primero. Con extensión **jar**.

Ejemplo:

- Si el usuario **jose.sala** es quien realiza la entrega, los ficheros deberán llamarse de la siguiente forma:
    - AAD_EX01E01_jose_sala.jar
    - AAD_EX01E01_runnable_jose_sala.jar
    - AAD_EX01E02_jose_sala.jar
    - AAD_EX01E02_runnable_jose_sala.jar

Tips & Tricks

Puedes apoyarte en la siguiente sección de código para parsear el fichero XML mediate API SAX:

```
SAXParserFactory saxParserFactory = SAXParserFactory.newInstance();
saxParserFactory.setNamespaceAware(true);
JuegoDefaultHandler defaultHandler = new JuegoDefaultHandler();
SAXParser saxParser = saxParserFactory.newSAXParser();
saxParser.parse(<variable File>,defaultHandler);
```

Puedes apoyarte en el siguiente código para la creación del LOGGER:

```
private static final Logger LOGGER = LoggerFactory.getLogger(Principal.class);
```

Puedes apoyarte en la siguiente sección de código para escribir el fichero de texto:

```
FileWriter ficheroEscribible = new FileWriter(<variable File>);
PrintWriter pt = new PrintWriter(ficheroEscribible);
pt.println(<texto a escribir>);
```

Puedes apoyarte en el siguiente código para la creación/apertura de la BDOO:

```
ObjectContainer db = Db4oEmbedded.openFile(Db4oEmbedded.newConfiguration(),<variable nombre de la Base De Datos>);
```