Organización General del Curso de Acceso a Datos (DAM ORD) Curso 2024-25

Comenzamos

En primer lugar, os quería desear mucho ánimo a todas/os, pues esta materia es la **continuación natural de la materia de Programación** que habéis cursado en primero, pero centrándonos en la **parte del modelo de acceso a los datos**. Es el **complemento principal de la materia de Desenvolvemento de Interfaces y Programación Multimedia y Dispositivos Móbiles, en la que se trabaja la parte de la vista y el controlador**.

Como habéis podido comprobar en la materia de primero, a programar se aprende **programando**, como con cualquier otra actividad que requiera destreza, como tocar un instrumento, cocinar, un deporte, etc. Seguro que más de un/a sabe a qué me refiero. Por ello intentaremos **minimizar la carga teórica para centrarnos en las actividades prácticas y las tareas**, eso no implica que no puedan caer cuestiones teóricas sobre cuestiones prácticas (patrones de diseño, etc.).

Realizaremos programas a diario y, si no podemos, organicemos nuestro tiempo para poder dedicarle a programar unas horas por semana. Como en primer curso (en el que también habéis trabajado con **Kodlin**) **trabajaremos principalmente con el lenguaje de programación Java**, para mí, el lenguaje más completo, flexible y útil desde un punto de vista didáctico, aunque veremos **también ejemplos y ejercicios con Kotlin, especialmente para acceso a datos desde Android (Room: https://developer.android.com/training/data-storage/room/ o Retrofit)**: https://square.github.io/retrofit/. En principio, nos centraremos en la parte de **acceso a ficheros/bases de datos**, proporcionando la parte de la vista, pero, a medida que avancéis en materias como diseño de interfaces, completaréis la parte de la vista para hacer programas completos.

A lo largo del curso veremos tecnologías de acceso a datos como:

- **JPA** (Jakarta Persistence): centrándonos en la versión Jakarta Persistence 3.2 con Hibertnate ORM, veremos la **versión 6.6.0**, pero si sale la versión 7, la veremos, para poder trabajar con JPA 3.2, en cuyo caso trabajaremos con la versión de JPA 3.1.
- **EclipseLink**.
- **Spring Data**: https://spring.io/projects/spring-data, sobre todo con Spring Data JPA.

- **Vaadin** (básico): https://vaadin.com/, para la parte de interfaz web.
- **MongoDB**: https://www.mongodb.com/es.
- **Bussiness As A Service (BaaS)** como: **Firebase** y, preferiblemente, **Supabase**.
- Si nos da tiempo, también veremos otras **bases de datos NoSQL** como la mencionada **Cassandra** y **bases de datos orientadas a objetos** como **BaseX** (sólo si llegamos a tiempo).

En definitiva, tecnologías actuales y demandadas en el mercado laboral para acceso a la información.

Consultad la información que aparece en el curso de tutoría de DAM:

https://mestre.iessanclemente.net/

Durante estos primeros días, **haremos un pequeño repaso de Java y empezaremos la primera unidad**, para que poco a poco os vayáis familiarizando con el entorno y la plataforma, es importante que actualicéis el perfil, y leáis esta guía completa (lo estáis haciendo), así como los documentos legales que aparecen en el curso de tutoría. Poned una foto real (perdón por no haberlo hecho, o casi).

Os adelanto, sin intención de dar miedo, al contrario, con ánimo de prevenir y que trabajéis desde ahora, porque, aunque con los conocimientos previos de programación os resultará sencillo, requiere práctica constante (sobre todo en clase) y enfrentarse a la materia con buen ánimo.

Contenidos

Este módulo profesional amplía la formación necesaria para desempeñar la función de desarrollador de aplicaciones multiplataforma, en la parte de **Acceso a Datos**. La función de desarrollador de aplicaciones multiplataforma incluye aspectos como:

- Desarrollo de aplicaciones de **gestión de ficheros y directorios**.
- Desarrollo de aplicaciones de **acceso a bases de datos relacionales**.
- Desarrollo de aplicaciones que hagan **uso de bases de datos orientadas a objetos**.
- Desarrollo de aplicaciones de acceso a **bases de datos XML y bases de datos NoSQL,** como MongoDB.
- Desarrollo de **componentes de acceso a datos y su integración en aplicaciones**.

Las actividades profesionales asociadas a esta función se aplican en el desarrollo de software de gestión multiplataforma con acceso a bases de datos utilizando lenguajes, bibliotecas y herramientas adecuadas las especificaciones.

Todas las empresas en las que los alumnos pueden trabajar al finalizar el ciclo utilizarán alguna **tecnología de persistencia de su información**. El alumno aprenderá a trabajar con las diferentes tecnologías de persistencia de los datos utilizadas más habitualmente, de forma que pueda adaptarse al entorno existente en el centro de trabajo una vez finalice el ciclo.

Unidades Didácticas

El curso de Acceso a Datos de DAM consta de **varias unidades didácticas**. Cuando se cuelgue la Programación del Curso podréis ver los detalles de cada una de ellas. **Por el momento se trata sólo de un primer borrador, que será completado al detalle en cada unidad didáctica**.

A modo de adelanto, se indica la temporización aproximada de dichas unidades, teniendo en cuenta que son **9 sesiones por semana**.

Primera evaluación

Se impartirán las dos primeras unidades y parte de la tercera unidad de herramientas ORM.

UD 1. ***Acceso a ficheros, flujos, serialización de objetos, ficheros XML y JSON****. (UD. 1)*

En esta unidad estudiaremos:

- ***Java I/O***
- ***Java NIO.2***
- ***JSON con Java (con la biblioteca GSON y una introducción de Moshi)***
- ***XML con Java: procesadores DOM y SAX, las clases específicas para el tratamiento de la información contenida en un fichero XML, las clases específicas para la vinculación de objetos, las bibliotecas para conversión de documentos XML a otros formatos****.*

Gestión de información almacenada en ***ficheros, flujos, haciendo especial hincapié en los formatos JSON y algo de XML*** *mediante aplicaciones informáticas escritas en Java.*

a) ***Gestión de flujos, ficheros secuenciales, Acceso Directo y Directorios****: desarrollo de aplicaciones que gestionan información almacenada en ficheros secuenciales, de acceso directo y en el sistema de directorios.* *En ella se aprenderá a identificar y utilizar las clases específicas para operar con cada tipo de fichero y con el sistema de directorios y a manejar las excepciones para el tratamiento de los posibles errores.*

b) ***Gestión de ficheros JSON y, en menor medida, XML**: desarrollo de aplicaciones que gestionan información almacenada en **ficheros JSON (con biblioteca Gson) y una introducción a Moshi**. También veremos algo de XML, y prenderemos a utilizar los **procesadores DOM y SAX, las clases específicas para el tratamiento de la información contenida en un fichero XML**, las clases específicas para la vinculación de objetos, las bibliotecas para conversión de documentos XML a otros formatos y a manejar las excepciones para el tratamiento de los posibles errores.*

*UD 2. **Acceso a BD remotas relacionales. Creación de una interfaz web sencilla (Vaadin y/o Thymeleaf)**.*

*Gestión de información almacenada en bases de datos relacionales por medio de aplicaciones informáticas escritas en **Java con JDBC y Spring Boot**.*

Para facilitar el trabajo de aplicaciones sencillas, existen muchos SGBD relacionales orientados a archivo (embebidos) opensource como H2, SQLite, HSQL, tinySQL, smallSQL o comerciales:

- ***SQLite**: Sitio Oficial*
- ***HSQLDB**: Sitio Oficial (HyperSQL database management system)*
- ***H2Database**: http://h2database.com/html/main.html*
- ***MariaDB**: https://mariadb.org/*
- ***PostgreSQL**: https://www.postgresql.org/*
- ***Derby**: https://db.apache.org/derby/*
- ***tinySQL**: Enlace*
- ***SmallSQL**: Enlace*
- ***Microsoft SQL Server***
- ***Oracle***

*Pondremos especial interés en **PostgreSQL, SQLite y H2, que son los más empleados en aplicaciones de escritorio y móviles**.*

*Veremos **patrones de diseño DAO y DTO**, y cómo realizar operaciones **CRUD** (Create, Read, Update, Delete) sobre la base de datos.*

Creación de una interfaz web sencilla, probablemente con el framework Vaadin https://vaadin.com/ o Thymeleaf https://www.thymeleaf.org/, que veremos también con Spring Boot.

*Aplicaciones que gestionan información **almacenada en bases de datos relacionales**. En ella se aprenderá a establecer conexiones con gestores de bases de datos relacionales embebidos e independientes utilizando conectores, a realizar operaciones de descripción, consulta y modificación de los datos contenidos en la base de datos, la extracción de los datos para realizar de forma adecuada las operaciones anteriores, a gestionar las transacciones y a manejar las excepciones para el tratamiento de los posibles errores.*

.

*UD 3. **Herramientas de mapeo objeto-relacional (ORM)**. Parte de esta unidad se realizará en la segunda evaluación.*

Parte de esta unidad se realizará en la segunda evaluación.

*Desarrollo de aplicaciones que gestionan información almacenada en base de datos relacionales utilizando herramientas mapeo objeto relacional (ORM), con **JPA sobre Hibernate/EclipseLink** y **Spring Data**, que traduzca la lógica de los objetos a la lógica relacional para su manipulación más sencilla.*

*En ella se aprenderá a **instalar y configurar la herramienta ORM**, a definir los ficheros de mapeo y las clases persistentes, a realizar el mapeo objeto relacional, a establecer sesiones, a cargar, almacenar y modificar los objetos persistentes, a realizar consultas en el lenguaje SuEL y en el lenguaje propio de la herramienta ORM.*

*También veremos **Spring Boot y Spring Data**, que facilitan el acceso a datos en aplicaciones Java, y que se integran perfectamente con **JPA y Hibernate**.*

Segunda evaluación

*En esta segunda evaluación se completará la tercera unidad y se impartirán las dos siguientes unidades. La unidad de **Herramientas de mapeo objeto-relacional (ORM)** se completará en esta evaluación, mientras que la unidad de creación de componentes se impartirá a lo largo del curso.*

*UD 4. **Bases de datos no SQL. MongoDB.***

*Características de las **Bases de datos NoSQL**. Manejo de la información en bases de datos NoSQL.*

Creación de aplicaciones informáticas que acceden a bases de datos NoSQL.

*En ella se aprenderá a manejar y establecer conexiones con el gestor de bases de datos NoSQL, a realizar consultas y modificaciones de los datos contenidos en la base de datos. Para ello emplearemos dos estrategias: **la API de bajo nivel y la API de alto nivel**:*

- *API nativa de MongoDB.*
- *API de alto nivel de Spring Data MongoDB.*

Ejemplos típicos de bases de datos NoSQL incluyen:

- ***MongoDB**, base de datos orientada a documentos.*
- ***Apache Cassandra**: https://cassandra.apache.org/ /index.html, base de datos distribuida.*
- *Redis: https://redis.io/, base de datos en memoria.*
- *Neo4j: https://neo4j.com/, base de datos de grafos.*
- *CouchDB: https://couchdb.apache.org/, base de datos orientada a documentos.*
- *BaseX: https://basex.org/, base de datos nativa XML.*
- *etc.*

UD 5. ***Bases de datos nativas XML. Bases de datos orientadas a objeto. BD objeto-relacionales****.*

Aplicaciones que gestionan la información almacenada en bases de datos nativas XML, como BaseX. En ella se aprenderá a manejar y establecer conexiones con el gestor de bases de datos nativas XML, a realizar consultas utilizado los lenguajes XPath y Xquery, o a modificar y eliminar los documentos XML.

Aplicaciones que gestionan información almacenada en bases de datos objeto-relacionales y orientadas a objetos. En ella se aprenderá a manejar gestores de base de datos que extienden las bases de datos relacionales añadiendo conceptos del modelo orientado a objetos y gestores de base de datos que almacenen los datos como objetos, estableciendo conexiones con estos tipos de gestores y realizando operaciones de almacenamiento, modificación y consulta de los objetos persistentes.

UD 6. ***Programación de componentes de acceso a datos.***

Programación de componentes de acceso a datos. En ella se aprenderá las bases de la programación orientada a componentes para las construcciones de aplicaciones basadas en el ensamblado de módulos reutilizables y se programarán componentes para el acceso los datos contenidos en diferentes sistemas de persistencia de datos utilizando herramientas de desarrollo de componentes.

Evaluación

Los detalles más precisos de la evaluación se extenderán en la Programación del Módulo de Acceso a Datos que se presentará a finales de este mes de septiembre. De momento, sólo presento las líneas generales de dicha programación.

Parciales y evaluación

En cada evaluación se realizará **un único examen parcial, aunque podría realizarse algún control después de cada unidad temática para realizar un seguimiento y una evaluación independiente de cada unidad. El examen parcial debe superarse para aprobar la evaluación correspondiente (nota mayor que 5).**

Las **pruebas o exámenes serán presenciales y en un único turno**, salvo que (algo que parece muy improbable) de decida lo contrario.

Cada unidad de evaluará de modo independiente, siendo **necesario superar todas las unidades para aprobar la materia**.

Se contempla la **realización de prácticas y trabajos**, al menos dos durante el curso, pero una por cada unidad, que se valorarán con un **máximo de 1,5 puntos sobre la nota final**.

Podrían plantearse trabajos o **prácticas no obligatorias**, que podrían emplearse para subir nota en caso de que fuese necesario y así lo considerase el profesor.

La **nota de la evaluación** está formada por el resultado del examen, las tareas obligatorias y un mínimo, referido a la participación y actitud durante el curso, con pesos de **85%, 10-15% y 0-5%**, respectivamente, Si durante la evaluación no se han realizado trabajos obligatorios el examen tendría un peso del porcentaje asociado a dicha nota. Una falta grave podría significar la suspensión de dicha evaluación.

El/los examen/es presencial de cada evaluación (2 en total) tendrá un valor de entre 8,5 y 10 puntos sobre la nota, dependiendo de si se ha realizado o no una tarea obligatoria. La puntuación restante se corresponde a la valoración de las tareas de la evaluación concreta.

Es imprescindible obtener un 5 sobre 10 en el examen para poder hacer media y aprobar. Aun así, es preciso que la **media total, contando las tareas, sea igual o superior a un 5**.

Aunque las notas tendrán decimales, en el boletín de la evaluación se redondearán a valores enteros más próximos a dicha nota, entre 1 y 10.

Los/as estudiantes que **hayan superado las evaluaciones parciales habrán aprobado la asignatura.**

La **nota final será la media de todas las evaluaciones, siempre que se superen las dos evaluaciones.**

Aunque la evaluación de cada unidad es independiente, **cada unidad podrá incluir conceptos básicos necesarios de unidades previas (flujos, ficheros, etc.)**

Examen final

Aquellas personas que **hayan suspendido alguna de las dos evaluaciones deberán realizar un examen final con las partes pendientes, en principio toda la evaluación suspensa, aunque podría realizarse de una unidad concreta si así se considera.**

El examen final constará de varios apartados, uno por cada unidad, **de carácter eminentemente práctico** que contenga la materia estudiada en cada una de las unidades. Si se supera, si se obtiene más de un 5, se aprobará la materia.

Se mantendrán las notas de **las prácticas obligatorias entregadas** a la hora de hacer la media del curso, que **sí contabilizan en la nota final** con el mismo peso que en las evaluaciones. Se calculará como media de las notas de cada evaluación con decimales, para ser redondeada a la hora de poner la **nota final, aproximada al entero más próximo a la nota media**.

Los **ejercicios de cada unidad que sean autoevaluables o boletines de clase no cuentan para la nota**, aun así, podrían ser tenidos en cuenta (no la nota, sí el hecho de participar), junto con la participación, actitud, etc. para decantar alguna nota que pueda no ajustarse a los baremos estrictamente legales y establecidos.

Como he dicho, ¡mucho ánimo!, estoy convencido de que los resultados van a ser muy positivos, sólo requiere constancia y trabajo constante. A programar se aprende programando, como con cualquier otra actividad que requiera destreza, como tocar un instrumento.

Referencias

- **Java I/O**:
  - https://docs.oracle.com/en/java/javase/22/docs/api/java.base/java/io/package-summary.html
- **Java NIO.2**:
  - https://docs.oracle.com/en/java/javase/22/docs/api/java.base/java/nio/package-summary.html
- GSON: https://github.com/google/gson
- **Moshi**: https://github.com/square/moshi)
- JPA (Jakarta Persistence): centrándonos en la versión Jakarta Persistence 3.2.
- Hibertnate ORM: https://hibernate.org/orm/, estudiaremos la versión 6.6.0, pero si sale la versión 7, la veremos, para poder trabajar con JPA 3.2.
- EclipseLink: https://eclipse.dev/eclipselink/
- Spring Data JPA: https://spring.io/projects/spring-data-jpa

- Spring Data: https://spring.io/projects/spring-data
- **Vaadin** (básico): https://vaadin.com/, para la parte de interfaz web.
- **MongoDB**: https://www.mongodb.com/es.
- **Firebase** (https://firebase.google.com/?hl=es-419).
- **Supabase**: https://supabase.com/.
- **Apache Cassandra**: https://cassandra.apache.org/_/index.html
- **BaseX**: https://basex.org/

Pepe Calo Domínguez, profesor de Acceso a Datos 2º DAM.