



LABORATORIO DE SOFTWARE

Año 2012

Carrera/ Plan:

Licenciatura en Informática Plan 2003/07-2012
Licenciatura en Sistemas Plan 2003/07-2012

Año: 4°

Régimen de Cursada: *Semestral*

Carácter: Obligatoria (Lic. en Informática)
Optativa (Lic. en Sistemas)

Correlativas: Proyecto de Software

Profesor: *Claudia Queiruga*

Hs Semanales: 6

FUNDAMENTACIÓN

Laboratorio de Software es una asignatura obligatoria de cuarto año de la carrera Licenciatura en Informática y de carácter optativo para cuarto y quinto año de la carrera Licenciatura en Sistemas.

El alumno que cursa **Laboratorio de Software** cuenta con los conocimientos fundamentales de informática en diferentes áreas y ha comenzado a entrenarse en el uso de herramientas de desarrollo de software.

Laboratorio de Software provee a los estudiantes de los conocimientos específicos sobre la construcción de aplicaciones de Internet enriquecidas (en inglés RIA), orientadas a servicios, con acceso a bases de datos y aplicaciones nativas para dispositivos móviles inteligentes, utilizando tecnologías JAVA. El estudiante adquiere las habilidades necesarias para desarrollar un trabajo integrador que signifique la aplicación concreta de los conocimientos adquiridos hasta el momento en la carrera, integrando temas de lenguajes de programación, ingeniería de software y base de datos. Mediante este trabajo, el estudiante se enfrenta con problemas reales y con la utilización de tecnologías de desarrollo de software actuales.

Esta asignatura articula en forma vertical con **Algoritmos y Estructura de Datos**, asignatura de segundo año e introductoria sobre desarrollo en lenguaje JAVA y, con materias de quinto año relacionadas a tecnologías de desarrollo: **Java y Aplicaciones Avanzadas sobre Internet, Diseño Centrado en el Usuario, Interfaces Adaptadas para Dispositivos Móviles y SOA y su impacto en el Desarrollo**.

Laboratorio de Software consolida la formación experimental y profesional del estudiante, ubicándolo en un entorno de trabajo similar al real y cotidiano.



OBJETIVOS GENERALES

Trabajo integrador que signifique para el alumno una aplicación concreta de los conocimientos adquiridos hasta el momento (cuarto año) en particular como una evolución de lo que ya han visto en Proyecto de Software.

CONTENIDOS MINIMOS

Se realizará un desarrollo específico integrando conocimientos teóricos y herramientas conocidas por el alumno. El enfoque podrá ajustarse e incluso dividir la temática experimental de la asignatura según las pautas que la cátedra considere de interés anualmente.

PROGRAMA ANALÍTICO

Unidad 1: La Plataforma JAVA

Objetivos:

Profundizar en la universalidad de la plataforma JAVA y su impacto en el desarrollo de software.

Conocer la importancia de la comunidad de especificaciones de las tecnologías JAVA que transforma a JAVA en una tecnología estándar.

El desafío de la plataforma universal JAVA. La plataforma de desarrollo Java o J2SE (Java 2 Standar Edition). La plataforma de ejecución Java o JRE (Java Run-time Environment). IDEs (Integrated Development Environment) para desarrollo en JAVA. Software Libre y JAVA. La comunidad de especificaciones JAVA, el Java Community Process.

Unidad 2: Clases, Espacios de Nombres, Especificadores de Acceso, Interfaces y Tipos Enumerativos

Objetivos:

Profundizar en los conceptos de orientación a objetos del desarrollo de software en JAVA. Relacionar los especificadores de acceso y el control de visibilidad de los miembros de las clases con las características propias de los paradigmas de orientación a objetos como ocultamiento y encapsulamiento.

Definir tipos de datos usando interfaces y tipos enumerativos.



Usar el formato de empaquetado y compresión de archivos (.class y otros recursos) disponible para distribución de aplicaciones JAVA.

Usar la documentación en línea de la API JAVA.

Definición de clases. Miembros de una clase: atributos y métodos. Creación e inicialización de objetos. Constructores. La palabra clave this y this(). Especialización de clases.

Especificadores de acceso: private, protected, default y public.

Especificadores de calificación: static, final y abstract.

Interfaces y clases abstractas. Tipos Enumerativos. Paquetes, como espacio de nombres: la palabra clave package. El formato JAR (Java ARchive). La variable de entorno CLASSPATH

Documentación de la API de Java: ¿dónde está disponibles? y ¿cómo usarla?

Unidad 3: Clases Básicas y Arreglos

Objetivos:

Examinar la característica de autoboxing/unboxing de JAVA: usar tipos primitivos, transformarlos en objetos y viceversa, automáticamente.

Profundizar en el buen uso de Strings.

Las clases wrappers: Integer, Short, Long, Byte, Character, Boolean, Float, Double. Boxing, Unboxing. La clase String y StringBuffer. Arreglos en JAVA

Unidad 4: Herencia y Polimorfismo en JAVA

Objetivos:

Profundizar en el mecanismo de herencia simple implementado en JAVA.

Definir interfaces JAVA y su relación con la herencia múltiple.

Comprender la potencia del concepto de upcasting automático a clases e interfaces.

Usar buenas prácticas de programación para creación de objetos en JAVA.

Analizar cómo impacta en la herencia el control de acceso protected.

Creación de objetos. Encadenamiento de constructores. Bloques de inicialización. La palabra clave super y super()

La clase Object: los métodos toString(), equals() y hashCode()



La herencia y la accesibilidad de atributos y métodos. Relación entre el especificador de acceso protected y la herencia.

Sobreescritura de métodos y ocultamiento de atributos. Polimorfismo.

Upcasting

Comparación entre interfaces y clases abstractas. Relación entre interfaces y herencia múltiple. Interfaces y polimorfismo.

Unidad 5: Clases Anidadas y Clases Internas

Objetivos:

Agrupar clases relacionadas y controlar su visibilidad.

Relacionar la herencia múltiple y las clases internas.

Clases como miembros de otra clase. El acceso irrestricto a los miembros de la clase contenedora. Resolución de la ambigüedad de nombres de miembros. Creación de objetos de la clase anidada. Clases anidadas locales. Clases anónimas y el uso de bloques de inicialización. Clases internas estáticas.

Unidad 6: El framework de Colecciones y “Genéricos”

Objetivos:

Analizar las características de los tipos y métodos genéricos de JAVA y su impacto en la programación segura.

Analizar la arquitectura del framework de colecciones.

Promover el uso de colecciones genéricas.

Tipos Genéricos y Parametrizados. Comodines. Métodos Genéricos

Arquitectura del framework de Colecciones

Interfaces core: Collections, Set, List, SortedSet, Map, SortedMap

Interfaces para ordenación de objetos: Comparable y Comparator

Interfaces para iterar: Iterator, ListIterator.

Implementaciones: HashSet, HashMap, Hashtable, ArrayList, Vector, TreeSet, TreeMap, LinkedList.

Algoritmos polimórficos para ordenación, búsqueda, manipulación de datos



Unidad 7: Construcción de Interfaces de Usuario en JAVA y Applets

Objetivos:

Analizar la arquitectura MVC (Model View Controller) de las componentes de interfaces gráficas de usuario en JAVA. Sus beneficios.

Analizar el mecanismo de manejo eventos de las interfaces gráficas de usuario en JAVA.

Usar la librería de clases del JFC (Java Foundation Classes) y las librerías de clases relacionadas al desarrollo de interfaces de usuario gráficas.

Construir applets y aplicaciones de escritorio con interfaces gráficas de usuario.

Usar JNLP (Java Network Launch Protocol) como mecanismo unificado para el despliegue de aplicaciones enriquecidas (RIAs – applets y aplicaciones Java Web Start).

La JFC (Java Foundation Classes). AWT (Abstract Window Toolkit) es un mundo de componentes de interfaz de usuario. Componentes Swing. Componentes SWT. Manejo de eventos basado en delegación. Applets, su ciclo de vida y las restricciones de seguridad. La herramienta de deployment Java Web Start.

Unidad 8: Manejo de errores mediante excepciones

Objetivos:

Simplificar la creación de programas confiables mediante el tratamiento de excepciones.

Analizar el mecanismo de tratamiento de excepciones impuesto por el compilador.

Destacar el valor fundamental de las excepciones para “informar errores”.

Tipos de Excepciones: Chequeables y no-chequeables en compilación.

Los objetos Throwable: las clases java.lang.Exception y java.lang.RuntimeException.

Manejadores de excepciones: los bloques try, catch y finally.

Propagación de excepciones: las cláusulas throws y throw

Excepciones customizadas.

Sobreescritura de métodos que disparan excepciones

Unidad 9: Anotaciones

Objetivos:

Usar y definir anotaciones que agregan meta-información a los programas Java para usar en compilación o ejecución.

Construir procesadores de anotaciones.



Conceptos y terminología de Anotaciones. Anotaciones estándares. Definición de Anotaciones. Anotaciones y Reflection.

Unidad 10: Acceso a Bases de Datos

Objetivos:

Describir cómo se accede y consulta una base de datos desde un programa JAVA en forma independiente del motor de base de datos utilizado.

Escribir aplicaciones JAVA que accedan a bases de datos relacionales.

La API JDBC (Java DataBase Connectivity) para acceso universal a múltiples fuentes de datos. Tipos de Drivers JDBC. Establecimiento de una Conexión. Ejecución de Sentencias SQL. Las clases Statement, PreparedStatement y CallableStatement. DataSource

Unidad 11: Concurrencia: multithreading en Java

Objetivos:

Comprender los fundamentos de la programación concurrente en JAVA.

Escribir programas multithread de una complejidad razonable.

Analizar las mejoras introducidas en las versiones actuales de la plataforma JAVA para programación concurrente.

Creación y gerenciamiento de threads: la clase Thread y la interface Runnable. El ciclo de vida de un Thread

Métodos de la clase Thread: run(), sleep(), join(), interrupt() y yield().

Abstracción del gerenciamiento de threads: Ejecutores

Objetos compartidos y sincronización: bloque y métodos synchronized.

Los métodos wait(), notify() y notifyAll()

Unidad 12: Aplicaciones móviles

Objetivos:

Introducir a los estudiantes en el ciclo de vida del desarrollo de aplicaciones móviles usando tecnologías JAVA para dispositivos móviles.

Desarrollar aplicaciones móviles mashups novedosas, enriquecidas con la posición del usuario y aprovechen diferentes servicios disponibles en Internet.



La tecnología Android y la apertura del mundo móvil. La arquitectura del sistema operativo Android. Ciclo de vida y componentes de una aplicación Android: Activities, Intents, Services, Content Providers. El ambiente de desarrollo de aplicaciones Android. Modos de diseño de interfaces gráficas: declarativo y procedural. Geolocalización y Sensores. Mashups basados en mapas.

METODOLOGÍA DE ENSEÑANZA

Las clases están organizadas en actividades teóricas y prácticas de carácter presencial y se utiliza la plataforma de gestión de cursos en línea Moodle (<http://catedras.info.unlp.edu.ar>) como mecanismo de apoyo adicional. La asignatura es de tipo taller; teoría y práctica se encuentran estrechamente vinculadas.

Las estrategias empleadas para dictar la asignatura combinan la exposición oral para el desarrollo de la teoría y la resolución de trabajos prácticos para las clases prácticas.

En las clases teóricas se trabajan contenidos conceptuales que son vistos en forma aplicada durante las prácticas. Se preparan clases en las que se utiliza una PC y un cañón óptico, de manera tal de poder mostrar en el aula, ejemplos que pueden aplicarse a las clases prácticas. Estas explicaciones y guías se ponen a disposición de los estudiantes y docentes de la asignatura, mediante su publicación en la plataforma virtual de la asignatura.

Las clases prácticas se desarrollan en las salas de PC de la Facultad que cuentan con máquinas con doble *booteo* permitiendo trabajar tanto en sistemas Microsoft Windows como GNU/Linux.

Durante la cursada se realizan trabajos prácticos que abordan los distintos contenidos en complejidad creciente. En los trabajos prácticos, además, se trabaja sobre la resolución de los ejercicios que los estudiantes realizan supervisados por los docentes.

Para la implementación de los trabajos prácticos se utilizan herramientas de soporte y desarrollo típicas en la comunidad de software libre y en ambientes de desarrollo profesional como ser la utilización de un sistema de versionado de código, IDEs de desarrollo, virtualización, emuladores de dispositivos móviles, servidores web, etc. ubicando a los alumnos en un ambiente profesional actual. Esto permite consolidar la formación experimental del estudiante.



EVALUACIÓN

Inicialmente se aplica una prueba diagnóstica que permite obtener información acerca de la situación académica de los estudiantes y sus conocimientos previos en tecnologías JAVA. Esta encuesta es la primera actividad que los estudiantes realizan en la asignatura, es llevada a cabo mediante en la plataforma virtual de la asignatura, para ello los alumnos son citados en el horario establecido para los trabajos prácticos. Los resultados del procesamiento de la encuesta están disponibles para consulta de los estudiantes.

Para la aprobación de los trabajos prácticos se definieron dos modalidades, una incremental que consiste en evaluar el conocimiento adquirido por los alumnos en forma modular y la otra mediante una evaluación general. Ambas modalidades pueden combinarse.

Modalidad de evaluación incremental: durante la cursada se realizarán 4 (cuatro) instancias de evaluación práctica diferentes, que consistirán en la resolución de actividades prácticas individuales en la sala de PC que abarcan los temas estudiados en las clases prácticas hasta el momento. Para aprobar los trabajos prácticos de la asignatura, los alumnos deberán obtener al menos 6 puntos de promedio entre las cuatro instancias de evaluación parcial y en cada una de ellas deberán obtener al menos 4 puntos.

Modalidad de evaluación general: aquellos alumnos que no logren aprobar las instancias evaluatorias descriptas previamente o hayan optado por esta modalidad, podrán aprobar los trabajos prácticos, rindiendo una instancia de evaluación general que tendrá 2 instancias de recuperatorio. Esta instancia de evaluación y sus recuperatorios consisten en la resolución de actividades prácticas individuales en la sala de PC, que abarcan todos los temas estudiados en las clases prácticas de la cursada.

La aprobación final de la asignatura consiste en el desarrollo de un trabajo final integrador o en la realización de una prueba escrita.

BIBLIOGRAFÍA

Effective Java™ - Second Edition. Joshua Bloch. Addison Wesley

Thinking in JAVA - Fourth Edition. Bruce Eckel. Prentice Hall

API-Documentación:<http://download.oracle.com/javase/7/docs/api/>



FAQ sobre Tipos Genéricos:

<http://www.angelikalanger.com/GenericsFAQ/JavaGenericsFAQ.html>

Hello, Android (3rd edition): Introducing Google's Mobile Development Platform; Ed Burnette. Pragmatic Bookshelf

Unlocking Android; Frank Ableson, Charlie Collins, Robi Sen. Manning Publications

CRONOGRAMA DE CLASES Y EVALUACIONES

Clase	Actividades
Presentación de la Materia	Prueba diagnóstica
Herencia y Polimorfismo en JAVA	Clase Teórica Especificadores de Acceso Constructores. Cadena de herencia Polimorfismo. Clases abstractas. Clase Práctica Especificadores de acceso. Constructores de objetos. Clases abstractas.
Abstracción y Ocultamiento	Clase Teórica Interfaces Tipos Enumerativos Clase Práctica Interfaces Tipos Enumerativos Polimorfismo Importación de miembros estáticos Tipos Enumerativos



Ocultamiento y un abordaje de la herencia múltiple	Clase Teórica Clases Internas Clase Práctica Clases internas locales. Clases internas locales y anónimas. Clases internas estáticas.
Evaluación Incremental Parcial	Primera Instancia de Evaluación Práctica
Interfases Graficas de Usuario	Clase Teórica JFC Applets Swing Explicación de Acceso a Datos en Clientes Pesados (JDBC) y el protocolo jnlp Clase Práctica Definición, ciclo de vida. Applet AWT y Swing. Modelo de Delegación de Eventos. Acceso a Recursos (imágenes). Aplicaciones de escritorio con persistencia de datos. Aplicaciones JAVA Web Start
Colecciones Genéricas	Clase Teórica Tipos Genéricos y su aplicación a colecciones Clase Práctica Arreglos de primitivos y objetos. Clases de la API para manejar colecciones de objetos Genéricos
Evaluación Incremental Parcial	Segunda Instancia de Evaluación Práctica



Manejo de Errores en JAVA	Clase Teórica Manejo de Excepciones Explicación de comunicación de datos (java.net) Clase Práctica Conceptos y uso de excepciones Cláusulas: try/catch/finally Definición de excepciones Manejo de excepciones en aplicaciones TCP/UDP.
Definición y procesamiento de meta-datos	Clase Teórica Anotaciones Explicación de Flujos de Entrada/Salida (java.io) Clase Práctica Conceptos y uso de anotaciones Definición de anotaciones propias Definición y procesamiento de anotaciones para mapeo de objetos a archivos planos usando flujos de E/S
Mecanismos de concurrencia	Clase Teórica Multithreading Clase Práctica Threads. Ciclo de vida. Sincronización de threads. Ejecutores
Evaluación Incremental Parcial	Tercera Instancia de Evaluación Práctica



Framework Android	Clase Teórica El framework Android –primera parte- Clase Práctica Taller de Android –primera parte- Aplicación de diferentes funcionalidades, entre ellas geolocalización, sensores de movimiento, acceso al disco para recuperar contactos, etc.
Framework Android	Clase Teórica El framework Android – segunda parte- Clase Práctica Taller de Android –segunda parte- Aplicación de diferentes funcionalidades, entre ellas geolocalización, sensores de movimiento, acceso al disco para recuperar contactos, etc.
Evaluación Incremental Parcial	Cuarta Instancia de Evaluación Práctica
Evaluación General y Consulta	Clase de consulta y Primera Fecha de la evaluación general
Evaluación General y Consulta	Clase de consulta y Segunda Fecha de la evaluación general
Evaluación General y Consulta	Clase de consulta y Tercera Fecha de la evaluación general

Contacto de la cátedra (mail, página, plataforma virtual de gestión de cursos):

Prof. Claudia Queiruga: claudiaq@info.unlp.edu.ar

JTP Pablo Iuliano: piuliano@info.unlp.edu.ar

JTP Jorge Rosso: jrosso@info.unlp.edu.ar

Plataforma virtual: <https://catedras.info.unlp.edu.ar/> (sección “Categorías de Cursos”)

Firmas del/los profesores responsables: