Práctica 1: Introducción a Neatbeans y Java Swing

Asignatura: Sistemas Interactivos

Resumen

Esta práctica introducirá al alumno en un entorno de trabajo adecuado para la creación de entornos gráficos y dotarlos de funcionalidad. Para ello, deberán aprender las nociones básicas de Java y las peculiaridades de Java Swing y aplicarlo en una serie de ejercicios básicos vistos en clase de manera introductoria para dotar al alumno de las herramientas y los conocimientos iniciales. El alumno deberá entregar una aplicación Java que realice el factorial de un número y otra aplicación que aplique herencia.

1. Introducción.

Netbeans es un IDE (entorno de desarrollo) para el desarrollo de aplicaciones en Java, aunque es compatible también con otros lenguajes como C++ o PHP mediante packs. Los equipos informáticos de la UCO disponen de este IDE, por lo que no tendrán que realizar instalación previa en el sistema para su uso. Sin embargo, aquellos alumnos que utilicen sus ordenadores personales deberán descargar Neatbeans desde la página oficial[1]. También requerirán de Java Development Kit 7 o superior o descargar Netbeans con JDK[2]. Utilizamos este entorno de desarrollo ya que está muy enfocado al diseño de interfaces.

Los ejemplos explicados y desarrollados en clase son para la comprensión y aprendizaje del alumno, por lo tanto no entregables. Las dos aplicaciones, factorial y herencia, sí serán evaluables y entregables por el alumnado deberá cumplir unos mínimos de requisitos, a partir de los cuáles se recibirá una calificación acorde a la completitud e innovación de la aplicación.

Java aparte de ser un lenguaje multiplaforma, dispone de otras características. Las características del lenguaje Java que veremos a través de ejemplos serán los siguientes:

- Herencia.
- Encapsulación.
- Polimorfismo.
- Sobrecarga de operadores.
- Paquetes Java y convenciones de nombre de archivos y paquetes.

[1] Web de Netbeans https://netbeans.org/

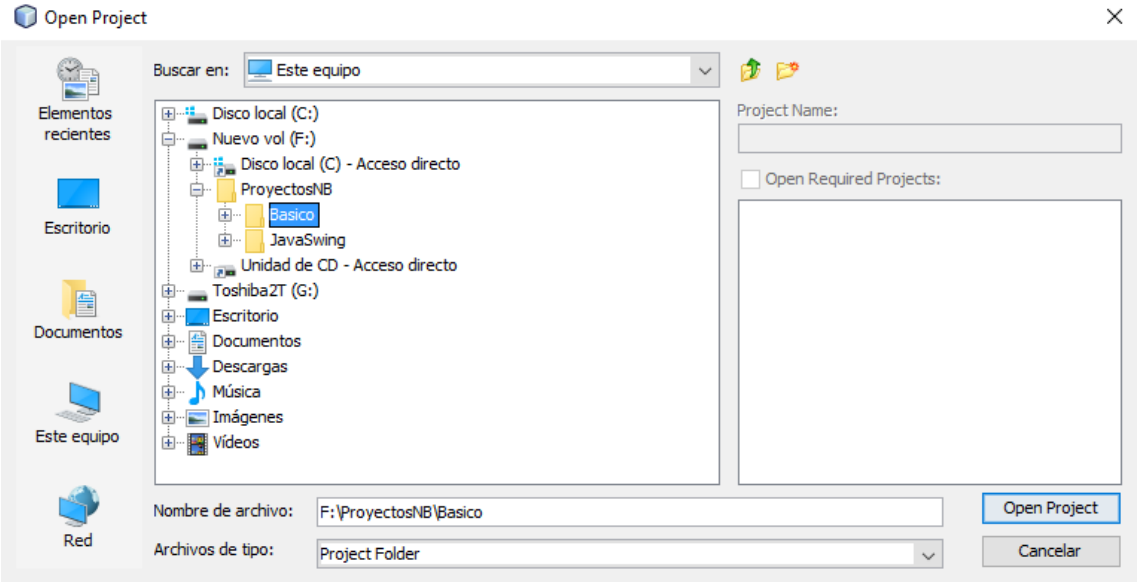
[2] Enlace a JDK http://www.oracle.com/technetwork/es/java/javase/downloads/index.html

2. Nociones básicas.

A continuación, a través de ejemplos, veremos lo básico para ir iniciándonos en este entorno y lenguaje. El material necesario se encuentra en la plataforma de la asignatura bajo el nombre Java Básico y Java Swing.

2.1. Importar un proyecto o crear uno nuevo

Nosotros tomaremos como base ambos proyectos que se han mencionado con anterioridad y que se encuentran en Moodle. Para ello tan sólo le damos a Open Project y seleccionamos la carpeta descomprimida del proyecto descargado. Automáticamente nos detectará todo lo que contiene el mismo y lo podremos importar.



A continuación, en nuestro IDE podemos ver, a la izquierda (por defecto), un listado de las partes del proyecto. En Source Packages podemos encontrar todos los .java del proyecto.

Nosotros trabajaremos con Java Básico para comenzar. El comprimido Java Swing contiene proyectos de aplicaciones a unos problemas e ideas sencillas que se verán en la siguiente unidad práctica para finalizar nuestra introducción a Java y Netbeans.

Para la creación de un nuevo proyecto, simplemente utilizaremos New Project en lugar de Open Project y seguiremos los pasos de Netbeans que nos facilita el proceso.

2.2. Archivo HolaMundo.java

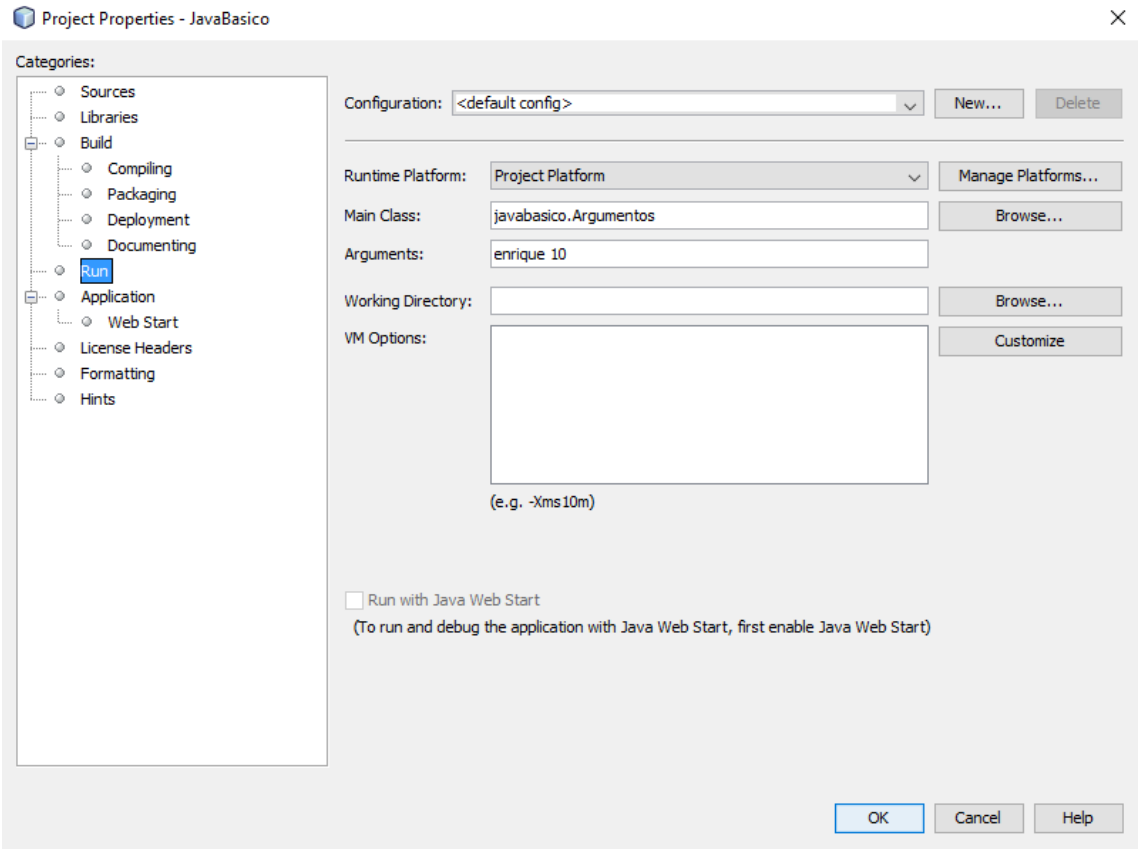
El ejercicio más básico que podemos encontrarnos en cualquier lenguaje de programación. Cabe destacar el uso de:

- Package, utilizado para relacionarlo con el proyecto al que pertenece.
- La clase contenedora, representa a la aplicación, y es la que contiene el main en ella.
- System, son funciones del sistema. Si usamos control y click sobre la misma, nos llevará a su código correspondiente en la librería, de igual manera podemos usarlo

sobre cualquier comando. Puede sernos útil para depurar en ocasiones. Utilizaremos System para la entrada/salida, reloj del sistema…etc.

2.3. Archivo Argumentos.java

En este ejemplo podremos aprender cómo manejamos los argumentos en Netbeans y también en Java. Para pasarle argumentos a la ejecución de nuestra aplicación, debemos ir a las propiedades del proyecto y en la categoría Run encontramos la opción para configurar nuestros argumentos. A la hora de tratar los argumentos, se utiliza el vector de cadenas args, de manera muy similiar a otros lenguajes como C, C++ o Python.



2.4. Archivo Matematicas.java

Esta aplicación hace uso de una serie de características que ya vistas en otros lenguajes de programación. Utiliza una librería o clase estática llamada Math, que contiene muchísimas utilidades matemáticas.

Las variables se deben declarar con su tipo y nombre, al igual que en C++, y los bucles tienen la misma estructura que dicho lenguaje de programación.

Lo más útil de este ejemplo es la sobrecarga de métodos. Tenemos dentro de la clase Suma, dos métodos distintos para sumar, dependiendo de si son doubles o enteros, para devolver el tipo correcto.

2.5. Archivo Cadenas.java

Las cadenas, o como las conocemos nosotros, string, tienen una serie de métodos ligeramente distintos en Java de lo que conocemos. En este ejemplo podemos ver los siguientes:

- toUpperCase y toLowerCase, para pasar a mayúsculas o minúsculas respectivamente.
- trim, elimina espacios al principio y el final de la cadena.
- replace, cambia los caracteres indicados por otros también pasados por función.
- indexOf, devuelve la primera posición del carácter indicado.
- toCharArray, pasa de String a una cadena de caracteres, lo que nos pide recorrerla.
- valueOf , pasa de String a entero, es un método de Integer.
- StringTokenizer, divide una cadena en tokens en función de un separador. En el caso de nuestro código distingue dos caracteres, uno separador **"="** y otro final **":"**.

2.6. Archivo Array.java

Java nos permite declarar arrays o vectores, y también arrays de arrays, formando matrices, de una manera muy parecida a C++ aunque con su propia sintaxis a la hora de realizar la reserva de memoria.

2.7. Archivo HashTable.java

Las hashtables pueden ser un concepto nuevo, pero muy útil. Es una clase que definimos según nuestras necesidades. En nuestro ejemplo, el primer valor corresponde a la clave, y está directamente asociado al segundo, por lo que esta clase dispone de métodos para obtener el segundo valor conociendo el primero. En temas posteriores veremos metodologías de internacionalización que usan técnicas parecidas.

2.8. Archivo Excepciones.java

Se utilizan códigos de excepción como ArrayIndexOutOfBoundsException, que nos permite identificar el problema que se ha dado en el programa con el uso de catch. Es un método muy útil para la depuración y supervisión del buen funcionamiento del programa.

2.9. Archivo LeeEscribeFichero.java

Para realizar la escritura y la lectura de ficheros, Java utiliza buffers que van recibiendo, procesando y escribiendo/leyendo lo que el programado quiera usando sus métodos. En este caso, hemos considerado métodos básicos como write o readLine. Pero es posible que dependiendo del fichero y del uso que queramos darle, debamos hacer uso de otros métodos.

A diferencia de otros lenguajes de programación, el final del fichero no viene representado por EOF, sino por null ya que el método deja de extraer cadenas.

2.10. Archivo claseBase.java y MiClaseBase.java

Con estos dos archivos podemos comprender el funcionamiento de herencia en Java. ClaseBase.java contiene una clase con una serie de métodos implementados. No es ejecutable pues no tiene main.

MiClaseBase.java hereda, con uso de extends, la clase y los métodos de ClaseBase. Como podemos observar en el método Suma, hace llamada a un método Suma superior a la clase actual, es decir, de la clase de la que hereda, utilizando super. getVar() al ser único entre clase y subclase, no es necesario hacer uso de super.

Si estos métodos no hubiesen sido públicos, no hubiésemos podido hacer uso de ellos en la clase hija. Ambas clases deben estar en el mismo package o ser importado, como veremos en el siguiente ejemplo.

2.11. Archivo Encapsulacion.java y otropaquete/Variables.java

En este caso otropaquete es otro package distinto a javabasico, por lo que veremos cómo hacer importaciones para el uso de métodos de javabasico en otropaquete. Siempre manteniéndose las características de privacidad y acceso de todas las variables y métodos.

En Encapsulacion.java podemos ver 4 maneras distintas de declarar las variables.

- Sin ninguna declaración de privacidad, lo que significa que sólo se podrá acceder a él desde ese paquete.
- Private es aún más restrictivo, pues sólo podrá ser accedido desde la misma clase.
- Protected permite el acceso a clases hijas o del mismo paquete. A este podría acceder otropaquete si heredara de Encapsulacion.java.
- Public, es el menos restrictivo, y siempre es accesible.

Una vez comprendido esto, podemos probar distintas situaciones en Variables.java. Para ello crearemos un main en Variables1 y otro en Variables, la primera de ellas hereda y la segunda no. Si intentamos acceder en el main de Variables1 a c, podremos hacer uso de ella, sin embargo de Variables no.

Por otro lado, no pasa lo mismo con la variable bo con a, que son innacesibles por cualquiera de las dos.

3. Entregable.

Con lo aprendido durante esta práctica se deberá desarrollar una aplicación básica, que muestre por consola el factorial de un número cualquiera. Como dificultad añadida, se ofrece al alumno extender el código para que el número sea leído a través de teclado.

Como segundo y último ejercicio entregable de esta práctica, se pide crear una clase base denominado Persona que tenga los siguientes atributos y métodos para devolverlos o establecerlos:

- name, surname, age
- getName(), getSurname(), getAge()
- setName(), setSurname(), setAge()

Para conseguir calificación extra y con objetivo de aumentar vuestra práctica, pueden añadirse contenidos que no sean contemplados en los objetivos en este último ejercicio, serán valorados positivamente. Por ejemplo, podrían añadirse más clases en las que existan restricciones en las que sea de necesidad aplicar encapsulación.

Los objetivos a superar por el alumno en esta práctica son los siguientes:

- Preparación de su equipo para el uso de Netbeans, si este usara ordenador personal.
- Aprender el entorno de desarrollo NetBeans y las peculiaridades del lenguaje de programación Java, en especial, obtener confianza en el uso de herencia y encapsulación.
- Comenzar a desarrollar, por parte del alumno, una actitud auto-informativa y auto-formativa en los aspectos que no se hayan visto en clase.

La entrega se realizará mediante una tarea en la plataforma Moodle de la asignatura como fecha límite

La entrega se realizará mediante una tarea en la plataforma Moodle de la asignatura como fecha límite XX-XX-2016. El nombre del alumno debe estar reflejado en el comprimido que contenga el Proyecto que recoja ambas actividades y también debe estar reflejado en el código (suele hacerlo automáticamente NetBeans). Se valorará negativamente copias y retrasos en la entrega.