Programación Modular y Orientación a Objetos - Laboratorio 0

Requisitos previos

Se asume que el alumnado está familiarizado con un lenguaje de programación, y que sabe construir programas utilizando estructuras condicionales.

Objetivos

Este laboratorio presenta y permite trabajar los conceptos básicos de la programación Orientada a Objetos en el entorno Alice, mediante la creación de clases sencillas. Se busca que el alumno entienda los conceptos de clase e instancia: clase como una abstracción que representa elementos reales con sus características (atributos) y sus comportamientos (métodos), y las instancias (u objetos) como entidades concretas pertenecientes a una clase y donde los atributos y métodos toman valores concretos.

Al finalizar este laboratorio, el alumnado deberá ser capaz de:

- Entender lo que es una clase y lo que es un objeto o instancia de una clase.
- Entender la diferencia entre clase y objeto.
- Crear mundos, clases y instancias muy sencillas, con atributos simples y los métodos básicos (métodos *get* y *set*). Los métodos constructores se nombrarán pero el entorno Alice es particular con respecto a ellos, así que no se profundizará en exceso.
- Familiarizarse con los conceptos y crear métodos sencillos que permitan a los objetos interactuar en el mundo de Alice.

Motivación

En la resolución de los problemas que se planteen se utilizará la Orientación a Objetos, por lo que es fundamental dominar los conceptos básicos presentados y que se irán complementando en sucesivos laboratorios. En éste se utilizará Alice, una herramienta muy amena y que rápidamente permite obtener resultados visualizables gráficamente y que son llamativos y divertidos.

 Tarea 1: el entorno Alice

En esta actividad solamente se pide lanzar Alice y visionar los tutoriales.

Si no aparecen por defecto al arrancar el entorno, se puede acceder a ellos a través de la pestaña Tutorial del menú File→Open World.

Una vez terminado el primer tutorial, habrá que visionar el segundo. Hasta no verse capacitado para seguir adelante no se recomienda dar por finalizada esta primera tarea y pasar a la siguiente.

 Tarea 2: las clases y objetos en Alice

Una vez realizados los tutoriales, se procederá a crear instancias de las clases **H**ombre y **M**ujer, y modelar algunas de sus características y comportamientos. También se generarán instancias de las clases **V**iejo y **J**oven y se buscará crear una interacción entre ellos.

Antes de comenzar, se pide visionar el vídeo-tutorial *"Husky en Alice"*, que se puede bajar de la siguiente dirección http://lsi.bp.ehu.es/asignaturas/PMOO/Alice

En el entorno Alice lo primero que se hace es crear un mundo, para lo que se tendrá que seleccionar de entre los mundo que se ofrecen (en caso de que al abrir Alice, no apareciese la pantalla de seleccionar mundo, habría que ir a File→New World). El mundo es importante porque el programa principal, que se llama "world.my first method", es un método perteneciente a él. Así pues, la ejecución comenzará por ese primer método.

Para llevar a cabo esta segunda tarea se tendrán que desarrollar métodos que permitan a los objetos de la clase **H**ombre y **M**ujer intercambiar información y comportarse de una forma u otra dependiendo de esa información. Se pide crear una instancia de la clase **H**ombre (hombre) que entre sus atributos tenga el nombre, apellido, edad y sexo (que por defecto será "varón"). Y crear otra instancia de la clase **M**ujer (mujer). Esta clase también tendrá los mismos atributos, la única diferencia es que el sexo tomará por defecto el valor "mujer".

Ayuda: En Alice la instancia *"se construye"* simplemente arrastrando la clase **H**ombre desde la galería de clases al mundo (pinchando sobre *addObject,* se abre la galería de clases posibles, en la que habrá que seleccionar People, y después Hombre). Lo mismo se hará para generar una instancia de la clase **M**ujer. En este caso bastará con arrastrar un hombre y una mujer.

IMPORTANTE: Es muy recomendable guardar el trabajo realizado con Alice cada muy poco tiempo.

El objetivo de esta tarea es implementar los métodos *setNombre*, *setApellido*, y *setEdad*, de manera que se les asignen valores a cada uno de esos atributos, tanto al hombre, como a la mujer que se han generado (en este punto se están introduciendo los conceptos de método *setter*, *getter* y *constructora*). Para hacerlo, habrá que situarse sobre el objeto que corresponda en el árbol que se encuentra arriba a la izquierda. El hombre tiene entre sus métodos uno denominado *setNombre*, que es el que habrá que arrastrar y situar dentro de *"world.my first method"*. Al hacerlo, se pedirá que se le dé un valor al atributo nombre. Lo mismo sucederá con el apellido y con la edad cuando se arrastren los métodos *setApellido* y *setEdad* del mismo modo. Una vez hecho esto con el hombre, se pasará a hacer lo mismo con la mujer. Ahora, solo quedará arrastrar el método *preguntarSiMayorDeEdad* del hombre

dentro de *"world.my first method"*, y luego el método *responderSiMayorDeEdad* de la mujer. Al pulsar el botón Play se desarrollará el diálogo. Puede comprobarse que si la mujer tiene más de 35 años no querrá decir su edad, mientras que si tiene menos no tendrá ningún inconveniente en señalarla.

Para finalizar la tarea, se pide hacer lo mismo, pero siendo la mujer la que le pregunta al hombre si es mayor de edad.

 Tarea 3: la programación de métodos en Java

En esta tarea se va a añadir más funcionalidad y nuevos atributos a la clase Hombre y a la clase Mujer utilizadas en la tarea anterior. En particular, se pide desarrollar:

- Dos nuevos atributos: idPersona (entero) y nacionalidad (String).
- Un método *tieneMismoID*, que toma un parámetro (pId) de tipo numérico y devuelve un booleano indicando si el identificador de la persona actual coincide con pId.
- Un método *puedeConducir*, que determine si la persona tiene la edad mínima permitida para obtener el carné de conducir. Por simplificar, se supondrá que esto ocurre a los 14 años en Etiopía, a los 16 años en Australia y Estados Unidos, a los 17 años en Reino Unido, y a los 18 años en el resto del mundo.

 Tarea 4: añadiendo interacción

En está tarea se van a crear dos nuevos objetos: uno perteneciente a la clase **V**iejo y otra perteneciente a la clase **J**oven. Además se creará una interacción entre el viejo y el joven que se regirá por el siguiente guión:

1) El viejo le pregunta al joven si le puede donar sangre a través del mensaje "Hola. Me puedes donar sangre?"
2) El joven le responde con una pregunta, "Qué tipo de sangre tienes?"
3) El viejo responde indicando su tipo de sangre.
4) Una vez que el joven tiene la información sobre el tipo de sangre del viejo, le responde bien afirmativamente "Ah!!Ah!! Te puedo donar sangre" o bien negativamente "Ah!!Ah!!Oh!! No puedo donarte sangre"

Para realizar esta interacción, la clase Viejo ya trae implementados los siguientes métodos:

getTipoDeSangre: devuelve el valor que posee el atributo tipoSangre del viejo.

pedirDonación: habla diciendo que necesita una donación de sangre y pidiéndola.

setTipodeSangre: recibe como parámetro un tipo de sangre y asigna al atributo tipoSangre del viejo el valor de ese parámetro.

responderConTipoDeSangre: habla indicando el tipo de sangre que posee.

A su vez, la clase Joven trae implementados los siguientes métodos:

getTipoDeSangre: devuelve el valor que posee el atributo tipoSangre del joven.

decirQueSí: habla diciendo que sí puede donar sangre.

decirQueNo: habla diciendo que no puede donar sangre.

decirTiposangre: habla el joven diciendo cuál es su tipo de sangre.

preguntarPorTipoSangre: habla preguntando cuál es el tipo de sangre que posee.

setTipodeSangre: recibe como parámetro un tipo de sangre y asigna al atributo tipoSangre del joven el valor de ese parámetro.

puedoDonar: el joven comprueba si su tipo de sangre es compatible con el tipo de sangre que recibe este método como parámetro. Si así lo fuese, dirá que sí (decirQueSí), y si no dirá que no (decirQueNo).

Para completar esta tarea se pide desarrollar el método *puedoDonar*, que recibe como parámetro un tipo de sangre (llámese pTipoSangre), y hace hablar al joven diciendo *"Ah!!Ah!! Te puedo donar sangre"* si el tipo de sangre que recibe como parámetro es compatible con el suyo propio, o *"Ah!!Ah!!Oh!! No te puedo donar sangre",* si no fuese el caso. Para ello, debe tenerse en cuenta qué tipos de sangre son compatibles, para lo que se proporciona la siguiente tabla:

		Donante							
		0+	0-	A+	A-	B+	B-	AB+	AB-
Receptor	0+	X	X						
	0-		X						
	A+	X	X	X	X				
	A-		X		X				
	B+	X	X			X	X		
	B-		X				X		
	AB+	X	X	X	X	X	X	X	X
	AB-		X		X		X		X