Práctica 1: Presentando a WALL·E

Fecha de entrega: 18 de Noviembre de 2012, 16:00

Material proporcionado:

Fichero	Explicación
`TestPr1.zip`	Proyecto de tests para la práctica.
`validador.zip`	Programa validador de la entrega.
`documentacion.zip`	Documentación de ficheros y clases necesarias para la implementación.

Objetivo: Iniciación a la orientación a objetos y a Java; uso de arrays y enumerados; manipulación de cadenas con la clase `String`; entrada y salida por consola.

1. Introducción

WALL·E[1] es una película de animación de géneros de comedia y ciencia ficción estrenada en 2008 y producida por los estudios Pixar. La trama sigue a un robot llamado WALL·E, diseñado para limpiar la basura que cubre la Tierra después de que fuese devastada y abandonada por el ser humano en un futuro lejano. Tras esto, se enamora de EVA, una robot tipo sonda que es enviada al planeta para investigar si existen indicios de vida, lo cual significaría que el lugar puede ser nuevamente habitado por la humanidad. Al encontrar una pequeña planta da su objetivo por cumplido y se dirige rápidamente a la nave de la que provino, Axioma, por lo que WALL·E la sigue al espacio exterior en una aventura que cambia el destino de ambos para salvar a la naturaleza y a la humanidad. Esta película fue ganadora del Oscar a la mejor película de animación en 2009

A lo largo del curso realizaremos distintas prácticas encaminadas a controlar el comportamiento del robot, usando inicialmente un conjunto de instrucciones que escribiremos en la consola, y más adelante utilizando interfaces gráficas de usuario.

[1] Para saber algo más: `http://www.imdb.com/title/tt0910970/`

Figura 1: Fotograma de WALL·E

2. Descripción de la práctica

En esta primera práctica vamos a crear una versión inicial muy sencilla de un simulador de control de nuestro robot. Concretamente WALL·E navegará por una ciudad compuesta de *calles* y *lugares*, hasta alcanzar su nave espacial, momento en el que WALL·E se desconectará y el simulador terminará su ejecución.

Cada lugar tiene un *nombre* y una *descripción* con información sobre lo que WALL·E ve en él. Cada vez que WALL·E llega a un lugar se mostrará por pantalla su descripción. El robot sólo puede estar en un lugar a la vez. Algunos lugares representan la nave del robot, que es su lugar de descanso. Una vez que WALL·E entra en uno de estos lugares se presenta su descripción y la simulación termina.

Los lugares de la ciudad se comunican con otros lugares a través de *calles*. Una calle comunica dos lugares A (origen) y B (destino) en una determinada dirección desde el origen. Las calles también pueden recorrerse en la dirección contraria para ir de B a A.

WALL·E siempre está mirando en una de las cuatro direcciones principales de una brújula –Norte, Sur, Este u Oeste– y se intentará mover en la dirección que mira desde el lugar en el que se encuentra, usando una calle (si la hay).

Podemos controlar a WALL·E mediante instrucciones en lenguaje natural. En esta primera versión tendremos las siguientes instrucciones:

- MOVE: Es la instrucción que ordena a WALL·E que avance en la dirección en la que está mirando. Si no existe una calle en esa dirección, se mostrará un mensaje que diga:

  WALL·E says: There is no street in direction «Dirección»

  En otro caso WALL·E avanzará por la calle hasta el siguiente lugar, presentando la descripción del mismo.

- TURN <LEFT | RIGHT>: Hace girar a WALL·E hacia la derecha o hacia la izquierda, con respecto a la dirección en la que está mirando. Por ejemplo, si está mirando hacia el norte, la instruccion TURN RIGHT pondrá el robot mirando hacia el este.
- HELP: Es una metainstrucción que sirve para mostrar información sobre las instrucciones que el robot es capaz de entender. En nuestro caso se presentará el siguiente mensaje:

  The valid instructions for WALL·E are:

  MOVE

  TURN <LEFT | RIGHT>

  HELP

  QUIT

- QUIT: Es una metainstrucción que nos permite abandonar la simulación presentando el siguiente mensaje:

  WALL·E says: I have communications problems. Bye bye

Si introducimos una instrucción que WALL·E no es capaz de interpretar, WALL·E contestará:

WALL·E says: I do not understand. Please repeat

3. Ejemplo de ejecución

A continuación mostramos un ejemplo de ejecución de nuestra simulación. Observa que las instrucciones *no* son sensibles a mayúsculas ni minúsculas (puedes escribirlas indistintamente).

```
PUERTA DEL SOL
You are at the PUERTA DEL SOL, the center of Madrid.
Ufff, there are a lot of streets, but all of them are full of garbage.
You should walk around this place to look for some interesting
objects to pick up.
Note that there is a big clock. Remember, leave the square before
night. It can be dangerous
WALL·E is looking at direction NORTH

WALL·E > turn left
WALL·E is looking at direction WEST

WALL·E > turn left
WALL·E is looking at direction SOUTH

WALL·E > help
The valid instructions for WALL·E are:
  MOVE
  TURN <LEFT | RIGHT>
  HELP
  QUIT

WALL·E > move
WALL·E says: Moving in direction SOUTH
```

```
JACINTO BENAVENTE
If you are cold you can start a fire with the wheels of those old buses
WALL·E is looking at direction SOUTH

WALL·E > turn left
WALL·E is looking at direction EAST

WALL·E > move
WALL·E says: Moving in direction EAST
NEPTUNO
Watch Wall-e! Another fountain! Perhaps this one has water for drinking
If you are cold, use that red and white scarf
WALL·E is looking at direction EAST

WALL·E > turn right
WALL·E is looking at direction SOUTH

WALL·E > moove
WALL·E says: I do not understand. Please repeat

WALL·E > move
WALL·E says: Moving in direction SOUTH
ATOCHA
You have a lot of old trains here, but they do not work
All trains were destroyed during the crisis of 2013
Take all the iron you find
WALL·E is looking at direction SOUTH

WALL·E > turn izda
WALL·E says: I do not understand. Please repeat

WALL·E > turn left
WALL·E is looking at direction EAST

WALL·E > turn left
WALL·E is looking at direction NORTH

WALL·E > move
WALL·E says: Moving in direction NORTH
NEPTUNO
Watch Wall-e! Another fountain! Perhaps this one has water for drinking
If you are cold, use that red and white scarf
WALL·E is looking at direction NORTH

WALL·E > move
WALL·E says: Moving in direction NORTH
CIBELES
Arggg, the fountain is ugly! The water is very dirty.
You should leave before the lions attack you
WALL·E is looking at direction NORTH

WALL·E > move
WALL·E says: Moving in direction NORTH
COLON
Sometime ago, Spanish people concentrates here to watch football
in a big screen.
```

```
Wall-e, did you know that in Spain there were very good football teams?.
Look for cans! People throw cans after watching football!
WALL·E is looking at direction NORTH
WALL·E says: I am at my spaceship. Shutting down... Bye bye
```

4. Clases que hay que implementar

Para crear esta primera aplicación en Java vamos a implementar las siguientes clases (tienes una información más detallada en el javadoc que te proporcionamos para hacer la práctica):

- Place: Representa un lugar de nuestra ciudad. Contiene un identificador del lugar (su nombre), su descripción e información sobre si representa o no la nave espacial de WALL·E.
- Street: Son las calles que comunican dos lugares. Una calle tiene información del lugar de origen A, el lugar de destino B y una dirección D que se interpreta como la dirección que hay que seguir para ir de A a B. La calle se puede recorrer también en sentido contrario, por lo que es posible ir de B a A siguiendo la dirección opuesta a D.
- Direction: Enumerado que representa las cuatro direcciones de una brújula, más un valor (UNKNOWN) que representa una dirección desconocida.
- Rotation: Enumerado que representa las dos direcciones en las que haremos girar al robot (LEFT y RIGHT), más un valor que representa una dirección desconocida (UNKNOWN).
- Action: Enumerado que representa las cuatro instrucciones (MOVE, TURN, HELP y QUIT) que se pueden ejecutar en la aventura conversacional, más un valor (UNKNOWN) que representa una acción desconocida.
- Instruction: Clase que encapsula la información de una instrucción, es decir, la acción que representa (del tipo Action) más una rotación (del tipo Rotation) para la instrucción TURN. Una instrucción es válida si la acción que representa es HELP, QUIT, MOVE o TURN, y para el caso de la última, la rotación es LEFT o RIGHT.
- Interpreter. Se encarga de convertir una cadena de caracteres (`String`) en un objeto de la clase Instruction. También se encarga de generar el mensaje de ayuda, ya que es el intérprete el que sabe las instrucciones que WALL·E es capaz de comprender.
- RobotEngine: Es la clase encargada de realizar la simulación.Almacena el *lugar* donde se encuentra WALL·E, la *dirección* a la que mira, y una *colección de calles* que representan implícitamente el mapa de la ciudad por la que se mueve. Esta clase se encargará de controlar la ejecución del simulador, hasta que éste termine. Concretamente, la simulación constará de los siguientes pasos:

  1. Mostrar la descripción del lugar en el que está WALL·E y la dirección en la que el robot mira. WALL·E siempre empieza mirando al norte
  2. Mientras que la simulación no termine, es decir, no se haya ejecutado el comando QUIT o WALL·E haya llegado a su nave espacial:
     a) Presentar el prompt (WALL·E>).

b) Leer de la consola lo que el usuario escribe.

c) Solicitar al intérprete que convierta en una instrucción lo que ha escrito el usuario.

d) Procesar la instrucción generada por el intérprete.

1) Si la instrucción no es válida entonces no hacer nada y mostrar el mensaje
`WALL·E says: I do not understand. Please repeat.`
2) Si es la instrucción HELP, mostrar la ayuda por pantalla.
3) Si es la instrucción QUIT terminar la simulación.
4) Si es la instrucción TURN entonces modificar la dirección actual del robot de acuerdo al giro.
5) Finalmente, si la instrucción es MOVE, el robot intentará cambiar de lugar de la siguiente forma:
   - Si existe una calle en la dirección actual del robot que comunica el lugar actual en el que ésta, con otro lugar de destino, entonces mover el robot (actualizar el lugar actual a lugar de destino) y mostrar por pantalla la descripción del lugar de destino.
   - En caso contrario presentar el mensaje que indica que no existe ninguna calle en esa dirección

3. Presentar el mensaje de fin de juego

- Main: Es la clase que contiene el método `main` de la práctica. Es la responsable de crear los lugares de la ciudad y de establecer las comunicaciones entre ellos a través de calles. Así mismo, se encarga de crear el robot, inicializarlo y hacer que comience la simulación.

5. Tests

Para esta práctica se proporciona el código fuente de los test de prueba de la práctica. Como en todas las prácticas del curso, es *obligatorio* que la práctica entregada supere esos tests. Se recuerda que los test anteriores *no* son infalibles, por lo que el funcionamiento completo de la práctica no está garantizado al pasarlos.

Como se puede ver, existe un fichero de test *por cada una de las clases principales de la práctica*. Eso permite pasar los test de manera incremental según se van implementando las distintas clases. Finalmente se ha de comprobar que se pasan *todos los test* ejecutando la suite `AllTests`.

6. Entrega de la práctica

La práctica debe entregarse utilizando el mecanismo de entregas del campus virtual, no más tarde de la fecha y hora indicada en la cabecera de la práctica.

Es indispensable que el código fuente supere los tests de unidad proporcionados y que el fichero enviado pase el programa validador publicado.

Sólo uno de los dos miembros del grupo debe hacerlo, subiendo al campus un fichero llamado `GrupoNN.zip`, donde `NN` representa el número de grupo con dos dígitos.

El fichero debe tener al menos el siguiente contenido[2]:

[2]Puedes incluir también opcionalmente los ficheros de información del proyecto de Eclipse

- Directorio `src` con el código de todas las clases de la práctica.
- Fichero `alumnos.txt` donde se indicará el nombre de los componentes del grupo.