Laberint amb Entrada/Sortida

Aquesta darrera sessió parteix de la solució de la sessió anterior "Laberint amb excepcions", on havíeu utilitzat el mecanisme d'excepcions de Java per a la gestió d'aquelles situacions anòmales que poden sorgir durant l'execució del codi del laberint.

En aquesta sessió, utilitzarem les funcionalitats d'algunes de les classes incloses en les jerarquies de classes d'entrada/sortida de Java (*package* java.io) per tal de: 1) guardar el resultat de la partida en un fitxer de text que emmagatzemarà els resultats de les diferents partides jugades i 2) llegir d'un fitxer de text l'estructura del laberint, amb les seves sales, ítems i elements secretes.

Objectius

- Comprendre la funcionalitat de la classe **FileWriter** i **PrintWriter** del package java.io i ser capaç d'utilitzar instàncies d'aquestes per a l'escriptura de text en un fitxer de text.
- Comprendre la funcionalitat de les classes **FileInputStream** i **Scanner** del *package* java.io i ser capaç d'utilitzar instàncies d'aquestes per a la lectura d'un fitxer de text.
- Comprendre i gestionar de forma adequada les excepcions generades pels mètodes de les classes esmentades en els dos punts anteriors

Material que s'adjunta

El material que s'adjunta per aquesta sessió és el següent:

- Aquest enunciat.
- Un fitxer "laberint.txt" amb la informació necessària per crear un laberint d'exemple. Cal copiar-lo al directori arrel del projecte NetBeans, el mateix directori on hi ha les carpetes build, src, dist, etc.
- Un projecte NetBeans "ExempleES" amb exemples de com fer l'escriptura i la lectura d'un fitxer, amb les classes recomanades.
- El javadoc complet del projecte.

Tasques a realitzar

Aquesta sessió es divideix en dos apartats, on s'introduiran de forma progressiva els diferents conceptes que haureu d'aplicar.

Apartat 1: Escriptura dels resultats de la partida

En aquest apartat volem que, al final de la partida, la nostra aplicació guardi els resultats en un fitxer de text anomenat `resultats.txt`. Aquest fitxer, que dipositarem també al directori arrel de l'actual projecte Netbeans, contindrà informació seguint el següent format d'exemple:

```
nom: Joan; valor total ítems: 20; temps total joc: 50 seg
nom: Marta; valor total ítems: 120; temps total joc: 150 seg
nom: Josep; valor total ítems: 35; temps total joc: 20 seg
```

on cada línia conté el resultat d'una partida finalitzada, amb el nom de l'aventurer, la suma dels valors de tots els ítems que havia aconseguit recollir en finalitzar la partida i el temps total de joc (en segons) des de l'inici. En aquest exemple, fixeu-vos que s'haurien jugat i finalitzat 3 partides.

Per guardar els resultats de la partida actual definirem un nou mètode de la classe **Controller**, que invocarem al servir la comanda "Quit":

```
private void saveResults() throws ResultWritingException;
```

Com hem vist en altres casos, aquest mètode invocarà el corresponent mètode "saveResults()" de la classe **UserInteface**, encarregat de fer pròpiament la gestió d'entrada sortida.

Per a l'escriptura de text en el fitxer `resultats.txt` heu d'utilitzar un objecte de la classe **PrintWriter**. Un dels mètodes constructors de **PrintWriter** pren com a paràmetre un objecte de tipus **Writer**. Tenint en compte que volem escriure en un fitxer, podem crear prèviament un objecte de tipus **FileWriter** (subclasse de **Writer**,) i passar-lo com a paràmetre al constructor de **PrintWriter**. Podeu consultar documentació d'Oracle d'ambdues classes disponible a Internet si ho creieu necessari, o consultar l'exemple que se us proporciona (el projecte NetBeans "ExempleES").

En cas que succeeixi qualsevol situació inesperada a l'escriure els resultats de la partida, el mètode capturarà les **IOException** corresponents, i llançarà una excepció de tipus **ResultWritingException** que capturarem des de "UserInterface.start()".

Per a la gestió del temps total de joc, heu de definir una nova classe **TimeCounter** i implementar-ne els seu mètodes i atributs (veure javadoc). El còmput del temps es basa en el mètode "System.currentTimeMillis()" que retorna el temps transcorregut, en ms, des de que s'ha arrencat el sistema.

Apartat 2: Lectura del laberint des d'un fitxer de text

Prenent la vostra solució del laberint, dediqueu uns minuts a identificar i comprendre el codi encarregat de la configuració del laberint "Controller.createDungeon()": objectiu de la partida, creació de les sales del laberint, creació dels ítems, connexions entre sales, creació de contenidors i portes secretes, i sala inicial.

L'objectiu d'aquest apartat és crear un mètode a la classe **Controller**

```
public Room loadDungeon();
```

que s'invocarà al principi de "UserInterface.start()". Com abans, aquest mètode invocarà "UserInterface.loadDungeon()", que serà l'encarregat de llegir la informació del laberint del fitxer de text `laberint.txt`.

Amb la documentació que s'adjunta teniu un exemple del fitxer d'entrada, `laberint.txt`.

Fixeu-vos que cada línia de text està formada per una sèrie d'arguments separats pel caràcter ';'. Les línies en blanc no conten, i les linies que comencen amb // són comentaris. La resta de línies contenen una de les comandes següents:

- `Objectiu;Descripció`
- `Sala;Nom sala;Descripció`
- `Item;Nom;Descripció;Valor;Encumbrance;Sala on està`
  o bé, si l'Item no està a cap sala
- `Item;Nom;Descripcio;Valor;Encumbrance`
- `Connecta;Origen;Destí;Nom sortida`
- `Porta;Nom;Clau;Origen;Destí`
- `Contenidor;Nom;Clau;Sala;Item amagat`
- `Inici;Sala inici`

Per a la lectura d'aquest fitxer de text, heu d'utilitzar un objecte de la classe **Scanner**, la qual proporciona mètodes per llegir fàcilment línies senceres de text. Tingueu en compte que el mètode constructor d'aquesta classe defineix un paràmetre de tipus **InputStream**. Com que nosaltres volem llegir un fitxer, li passarem aleshores un objecte de tipus **FileInputStream**, que haurem
creat prèviament, inicialitzat per a que llegeixi del nostre fitxer `laberint.txt`. Podeu consultar la documentació d'Oracle d'ambdues classes disponible a Internet si ho creieu necessari, o veure l'exemple que s'adjunta.

En cas que succeeixi qualsevol problema a l'hora de llegir la informació del laberint (error a l'hora d'obrir i llegir el text del fitxer, línia de text que no correspon amb cap dels tipus de línia especificats anteriorment, nombre d'arguments incorrectes, argument amb valor invàlid, no s'ha especificat habitació inicial, etc...) el mètode ha de

generar i llançar una excepció de tipus **`DungeonBuildingException`**, que es capturarà com sempre a la interfície d'usuari.

Finalment, a la classe **UserInterface** definirem una constant booleana **DUNGEON_FROM_FILE**. Si aquesta constant val "true", el laberint es carregarà de fitxer amb "loadDungeon()", i si val "fals", el laberint es crearà a partir del mètode "createDungeon()".

La resta de detalls els trobareu al javadoc.