Programación orientada a objetos
TPE
Dungeon Game

June 6, 2011

Autores:

Tomás Mehdi	Legajo: *51014*
Alan Pomerantz	Legajo: *51233*

1 Desarrollo y justificaciones del diseño e implementación del juego

Cuando comenzamos a hablar sobre el diseño surgieron varias ideas, a pesar de las variadas ideas, las buenas opciones para un buen diseño orientado a objetos eran pocas. Para comenzar se decidio hacer una jerarquia para los monstruos y los jugadores, estas dos clases heredan de "Character"; la cual es una clase abstracta. La justificacion a esta jerarquia es que monstruo y player tienen comportamientos parecidos y muchas variables en comun, pero no son lo mismo ya que el player puede realizar algunas cosas mas que un monstruo. Se creo la interfase "Putable" lo que nos permite tratar a todas las cosas que pueden ponerse en un tablero de la misma manera. Por cuestiones de implementacion se creo también la clase abstracta Cell la cual tiene una variable de instancia "isVisible" y los getters y setters de la misma. Esto facilito la opcion del juego de estar todo escondido hasta que el jugador lo descubre. A esta clase la heredan todos las clases que pueden ser insertadas. Aclaro que hice extender a Character de Cell y no al mismo Monster, porque existe la posibilidad de en un futuro en ves de solo poner monstruos escondidos tambien poner jugadores escondidos. La otra opcion era implementar directamente los metodos en la clase Monster, pero parecio mejor la idea de que un Character pueda ser visible o no. Se realizo un parseo con una gran orientacion a objetos, lo cual permitio utilizar esa clase para realizar la carga de un juego luego de ser guardado. No hay mucho que acotar a esta implementacion, hay partes imperativas, no hay forma de evitarlo, pero se realizo de una forma ordenada. El front se basa en los metodos del "DungeonGameListener", lo cuales son implementados en "DungeonGameFrame" en una clase inner privada "DungeonGameListenerImp". "DungeonGameFrame" hereda de "GrameFrame", una clase abstracta que podria utilizarse para crear el frame de distintos juegos. "DungeonGameFrame" contiene una instancia de un "DataPanel" y una de "DungeonPanel". Solo el Dungeon Panel hereda de la clase "GamePanel" otorgada por la catedra. La opcion de pasar el mouse por encima de un character en estado visible y que aparesca en el DataPanel complico un poco las cosas. Se creo una clase inner a "DungeonGameFrame" llamada "DungeonPanelListener", en la cual se aplican los eventos a las diferentes posisciones del mouse. Esta jerarquia ademas de ser muy logica favorece mucho la implementacion, ya que se reutiliza mucho codigo y simplifica la creacion de la parte grafica con JavaSwing. El topico de guardado y cargue del un juego al principio parecia

que debia ser una cualidad del juego, "el juego sabe guardarse y cargarse", pero luego descubrimos que si no era una "cualidad" del juego podria modularizarse para guardarse como cada uno quisiera sin tener que modificar la clase juego sino creando una propia clase para el guardado del juego. Lo cual nos guio a crear una interfase e implementar nuestra propia clase que la implementara.