Proyecto TDP

Tower Defense

Documentación-Patrones de diseño

En el diseño del juego podemos ver que tenemos diferentes módulos o paquetes donde cada uno tiene las clases correspondientes según el tipo de relación u orden que le quisimos dar como por ejemplo, algunos paquetes solo contienen las clases de su patrón de diseño y otros contienen todas las clases que heredan de una interfaz. Para la realización de este juego utilizamos diferentes tipos de patrones de diseño. A continuación describiremos los paquetes en los que haya clases con dichos patrones y explicaremos el porqué de su elección.

En Juego

Utilizamos el patrón de diseño singleton ya que solamente queremos que se cree una instancia de las clases GUI, Controlador, Mapa, Tienda y Jugador. No sería un buen diseño para nuestro juego si se pudieran crear más instancias de dichas clase porque no corresponden a objetos que debamos clonar ni obtener más de una instancia por nivel.

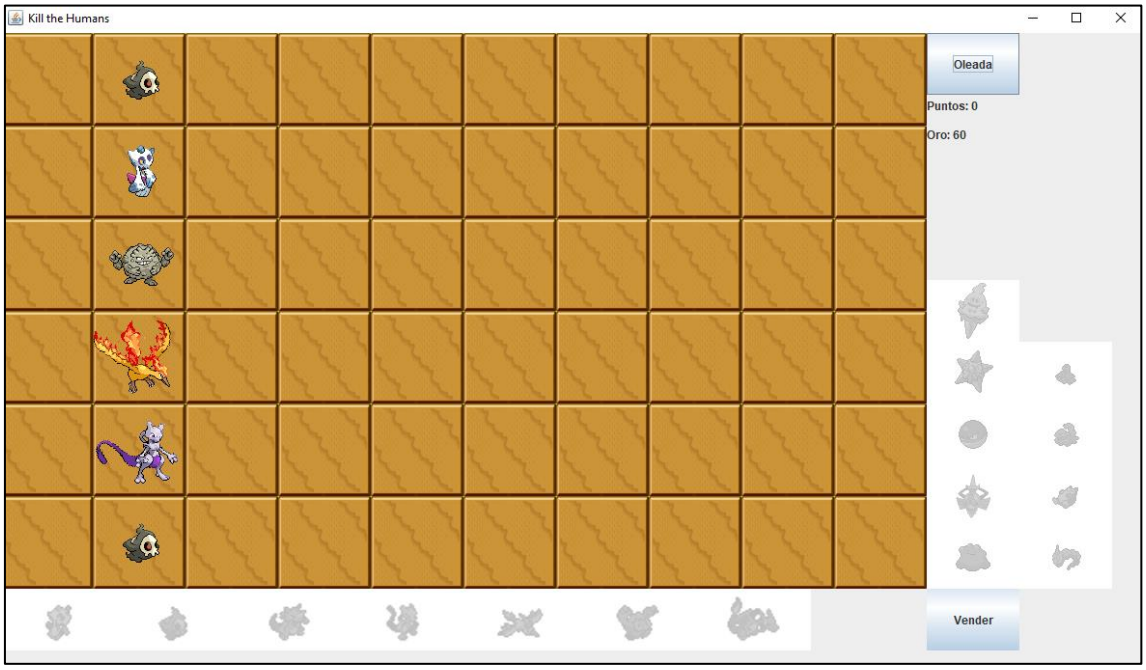
En Personajes

Utilizamos el patrón de diseño state porque, como lo dice su nombre, es adecuado que ciertos personajes presentes diferentes estados para poder modelar adecuadamente su situación en el juego.

Tenemos las clases abstractas de personaje, torres, enemigo y proyectil con sus respectivas clases heredadas como el tipo de proyectil y tipo de enemigo y torre.

En el caso de las *torres* aplicamos el patrón de diseño state para poder modelar dos tipos de estados que van a presentar las mismas. Las clases interface de estado son EstadoDefensa y EstadoAtaque. Cada estado tiene sus dos clases concretas de estado, que son las que implementan el comportamiento de Torre. Para cada estado se presenta un EstadoDefault y dependiendo lo que suceda van a cambiar a otro estado

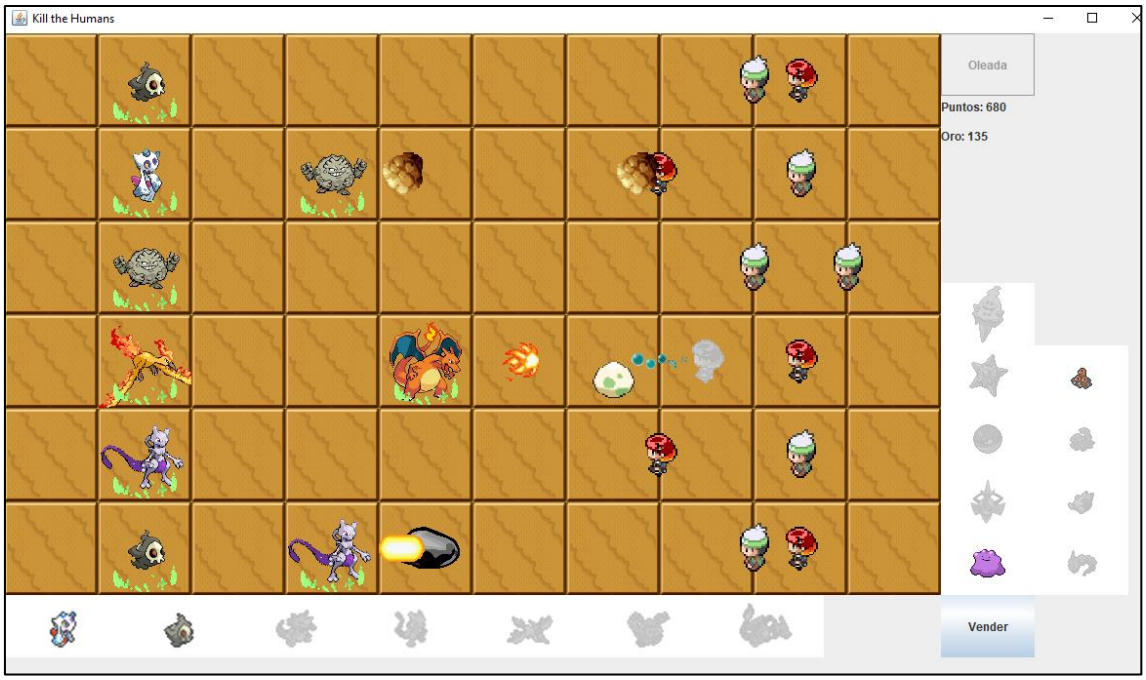
tanto en defensa como en ataque. La clase torre va a mantener una instancia con el EstadoDefault.



Ejemplo EstadoDefault

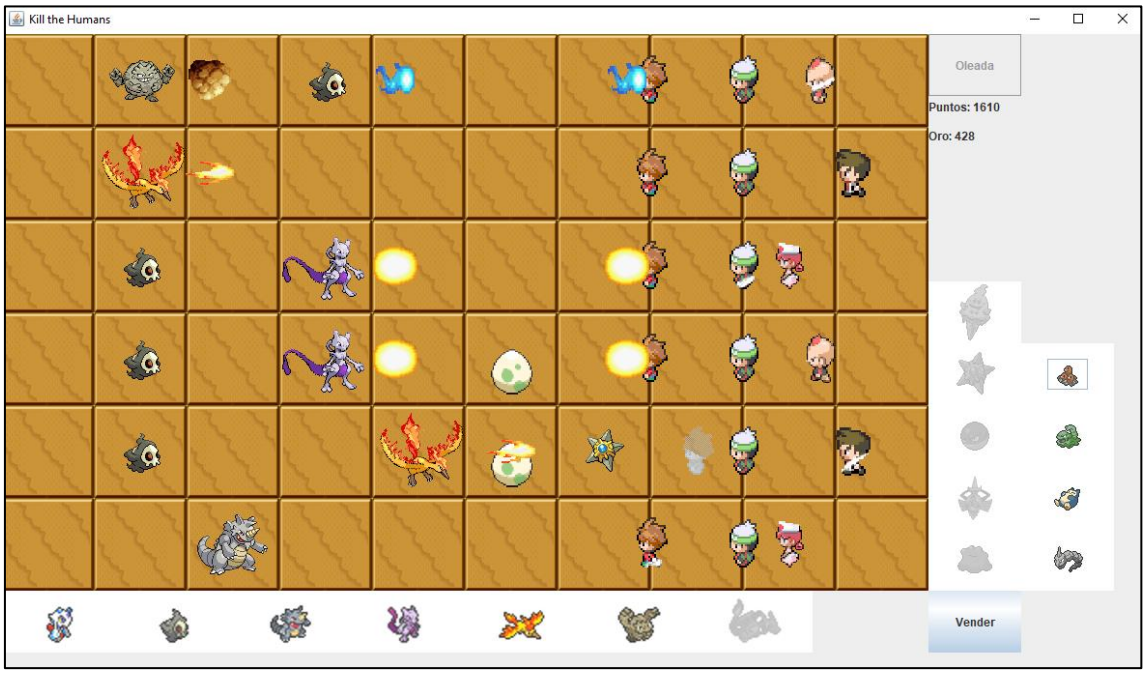
Ejemplo EstadoProtegidoTorre

En este caso se puede observar dos torres “Alien”. La de arriba con estado default y la de abajo tiene el estadoProtegidoTorre que gráficamente genera un brillo en dicha torre.



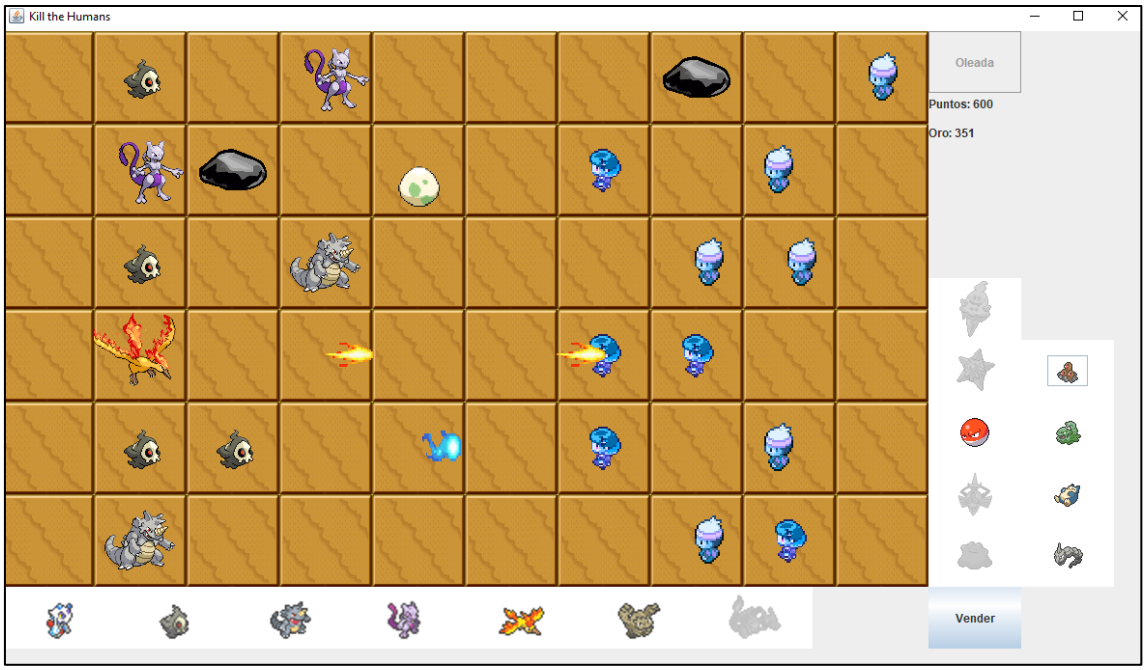
Ejemplo EstadoDobleFuerzaAtaque

Los *enemigos*, al igual que las torres, van a tener sus respectivos estados que también están diseñados con el patrón state. Los enemigos tienen los estados de defensa y de actuar. En defensa los estados concretos corresponden a dañar a las torres y los de actuar corresponden a caminar hacia las torres o estar congelados porque se les aplico un powerup utilizado por el jugador para su beneficio. Estos estados concretos heredan de las clases abstractas EstadoDefensaE y EstadoActuarEnemigo.



Ejemplo EstadoProtegidoEnemigo

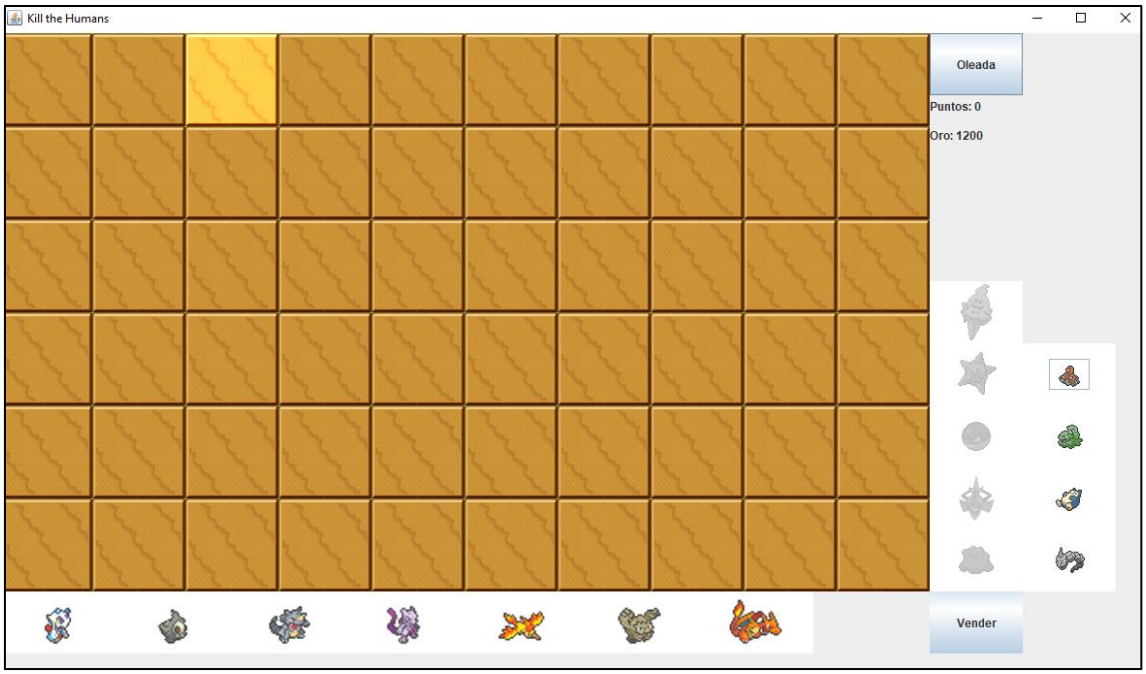
En este ejemplo algunos enemigos brillan indicando que tienen el estado protegido y los que no están en estadoDefault que heredan de EstadoDefensaE. A su vez el otro estado que presentan es EstadoActuarEnemigo que corresponde a que los enemigos están disparando y haciendo daño y los que están mas atrás están en EstadoActuarDefault que ambos heredan de EstadoActuarEnemigo



Ejemplo EstadoCongeladoEnemigo

En Tienda

La clase tienda va a tener un patrón de diseño state para modelar sus estados. Cada uno va a corresponder según lo que el jugador vaya seleccionando. Tenemos los estados concretos que son TiendaDefaultEstado, TiendaVenderTorre, tiendaComprarObjeto, TiendaPowerUpTorre, TiendaComprarTorrem TiendaPowerUpCelda. Estas heredan de su clase abstracta TiendaEstado. Aplicamos este patrón porque según el estado en el que se encuentre la tienda, va a corresponder con ciertos botones habilitados o deshabilitados, que podrá usar el jugador. Estos se habilitan dependiendo del oro que tenga el jugador y pueda comprarlos , de los powerup que este haya obtenido por lo cual se le habilitan para poder aplicarlos donde desea. Y la tienda va a comenzar con un EstadoDefault.



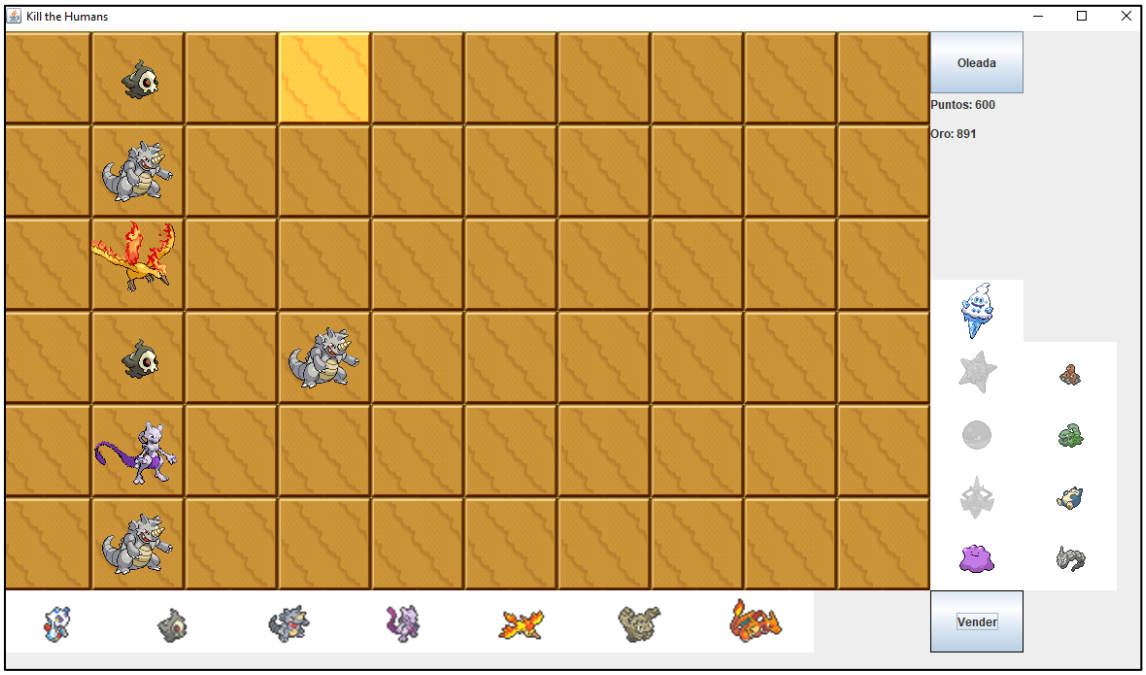
Ejemplo TiendaEstadoDefault

Ejemplo TiendaPowerUpCelda

Tenemos porwerUps habilitados, entonces al clickear sobre Bomba o el de TorreAleatoria que en este caso no esta habilitado, la tienda pasa al TiendaPowerUpCelda.

Ejemplo de TiendaComprar

Podemos seleccionar algún objeto de los que está habilitado y la tienda pasa al estado TiendaComprarObjeto que gráficamente no cambia pero habilita donde ubicar ese objeto. Tambien podemos seleccionar una torre y cambia a TiendaComprarTorre y al igual que el anterior habilita para ubicarla en el mapa.



Ejemplo TiendaVenderTorre

Tambien si se clickea en BotonVender se cambia al estado TiendaVenderTorre y se puede seleccionar la torre que se desea vender

En Visitor

Utilizamos el patrón de diseño visitor para varias clases y las guardamos todas en el mismo paquete para mantener un orden con respecto al patrón de diseño. Utilizamos este patrón de diseño ya que para cuando se colisionan elementos, cada uno debe saber con quién esta colisionando para poder actuar adecuadamente. Este patrón nos permite saber de qué tipo de clase son aquellas que se han visitado. Por lo cual cada visitor visita a elementos y a su vez estos elementos en sus clases tienen el correspondiente método accept que recibe al objeto visitante.

Tenemos la clase abstracta visitor y sus visitor concretos son:

VisitorEnemigo cuando visita una torre, según la distancia a la que se encuentre, la ataca. Cuando visita un objeto con vida que es el que usa el jugador a su favor también lo ataca, al igual que hace con los objetos aleatorios que va incorporando el mapa al juego a modo de ayuda.

VisitorTorre este solo realiza algún método cuando visita a un Enemigo y lo ataca.

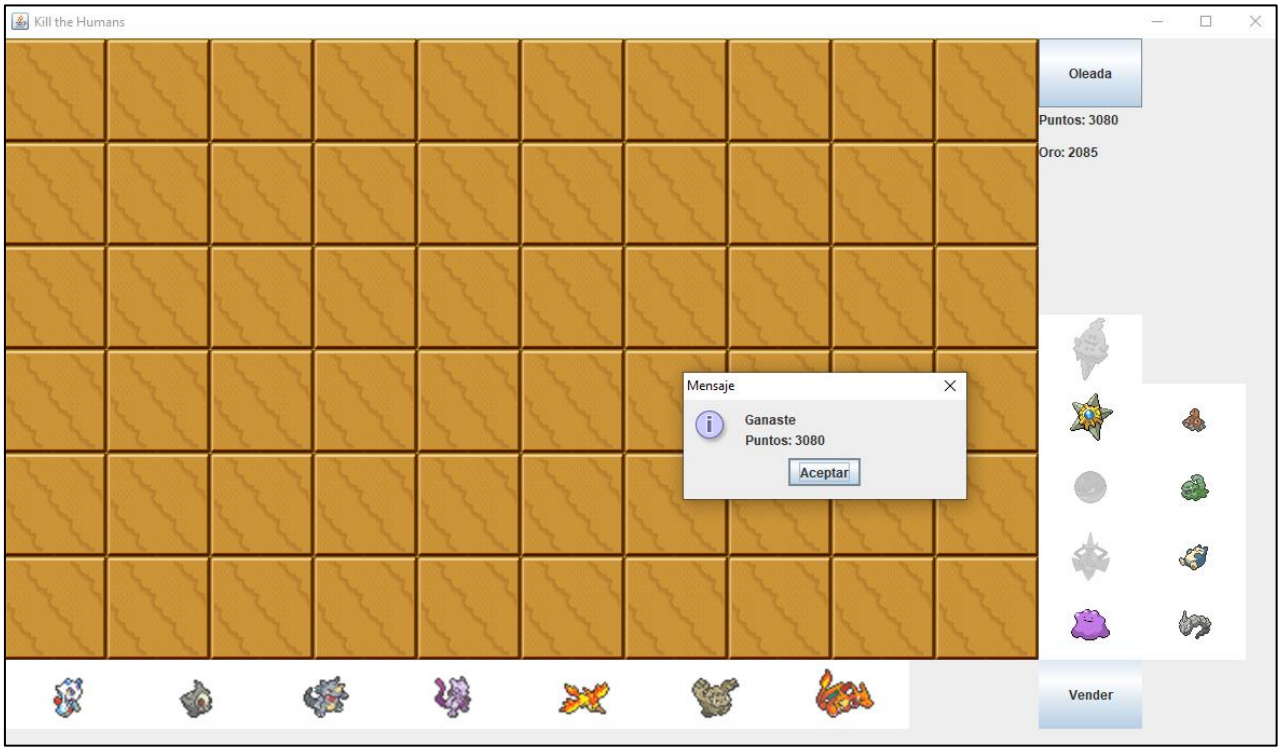
VisitorProyectilTorre cuando visita a un enemigo lo daña, disminuyendo su vida.

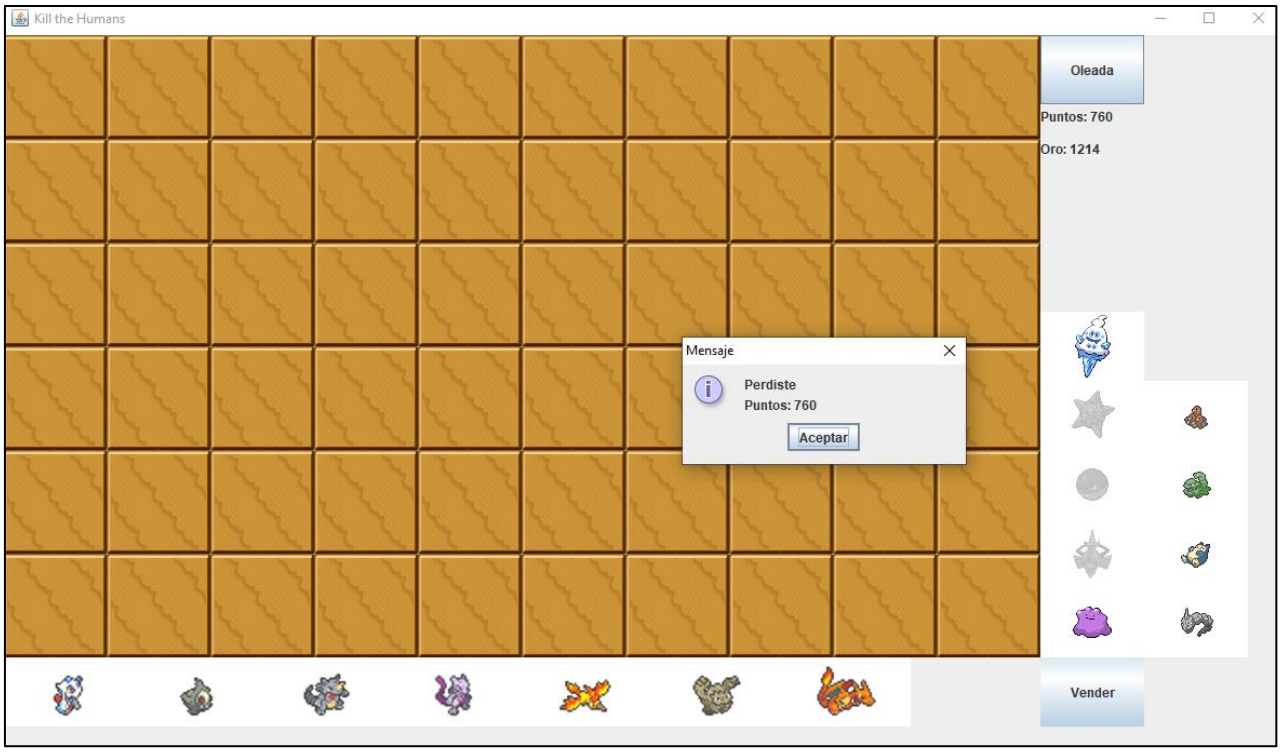
VisitorProyectilEnemigo cuando visita a una torre u objeto con vida los daña, disminuyéndoles la vida. En cambio cuando visita a un objeto con tiempo el que se destruye es el proyectil.

VisitorPowerUpGlobal cuando visita a una torre o enemigo les aplica la magia asociada al powerUp correspondiente.

VisitorClick está relacionado con la tienda. Al seleccionar algo, este visitor visita al elemento que se clickeo y puede habilitar botones o simplemente seleccionar el elemento.

Por último las imágenes de perder o ganar.




Kill the Humans
Oleada
Puntos: 760
Oro: 1214
Mensaje
Perdiste
Puntos: 760
Aceptar
Vender