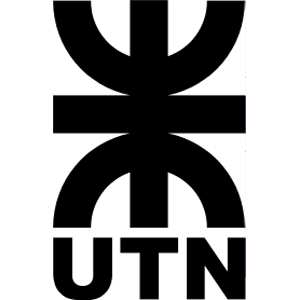
Trabajo Práctico Final

Juego de Rol: Wolf Forest



Laboratorio de Computación III

Ciclo Lectivo 2019

Profesor: Gonzalo Benoffi

Integrantes: Florencia Excoffon, Julián Aga, Lucas Acosta, Dante Scrosoppi

Nosotros desarrollamos un juego de rol (RPG) en donde tenemos clases de personajes que son: Perro Salvaje, Paladín y Ladrón. En este juego, cada clase de personaje tiene distintos stats y habilidades que sirven para un combate. El desarrollo fue paulatino, durante las clases de Laboratorio, también nos reunimos en la casa de uno de los integrantes para poder ponernos de acuerdo y organizarnos. La mayoría de la comunicación fue a través de Whatsapp, discord y pudimos hacer uso de GitHub para simplificar la manera de compartir el código, ya que los primeros días usábamos los documentos de google, copiando y pegando, y si bien esta herramienta tiene una ventaja muy importante que es que sea en tiempo real, quedaban muchos archivos sin pasarse, errores, etc.

En nuestro trabajo utilizamos clases para los gráficos y clases para el juego por consola. Tenemos un paquete “biblioteca” en donde se encuentran todos los escenarios del juego de consola, en ellos se avanza o retrocede. Otro paquete es el de “combate”, en el cual se encuentran los métodos de pelea de los personajes, es decir, donde se calcula la experiencia que gana el ganador de la pelea, cuál de los que pelean gana por stats, el daño que hacen ciertas habilidades, entre otras cosas, y directamente la clase que contiene el método “atacar” que es el cual se ejecuta para que la pelea entre dos personajes ocurra.

Tenemos una clase “Teclado” que es la cual capta los movimientos de las flechas del teclado, shift y esc para moverse, correr y salir respectivamente, que es utilizada en la parte gráfica de nuestro sistema.

Luego tenemos el paquete de “entes”, que contiene a todas las clases de personajes que vamos a utilizar y también al concepto abstracto de ente que es el que se utiliza en la parte gráfica.

La clase jugador contiene los stats y métodos de un personaje más métodos para poder utilizarlo en la parte gráfica. En cambio, jugadorP contiene lo mismo pero sin las cosas de la parte gráfica ya que será lo utilizado para el juego en consola. Ladrón, Paladín y Perro Salvaje heredan de jugadorP, ya que JugadorP es nuestra clase base de personaje de pelea.

Luego, tenemos el paquete “gráficos” que se encarga de leer todos los sprites que tenemos cargados desde una imagen png, básicamente se leen desde hoja sprite y luego se va desmenuzando la hoja de a 32x32 píxeles (ya que todos los sprites tienen el mismo tamaño) y luego son mostrados desde la clase Pantalla que define la posición de cada sprite.

En el paquete “habilidades” está la clase habilidad que contiene todos los atributos de las habilidades como el nombre, la cantidad de daño que hace, etc, sus constructores, getters y setters.

En “herramientas” hay tres clases que sirven para poder cargarle habilidades a los personajes, una clase que es para calcularStats y otra que es el funcionamiento de un dado.

La clase juego inicia el juego en su parte gráfica.

El paquete manager contiene al manager que es básicamente el que controla el juego, lo inicia, hace continuar o no, dependiendo de la otra clase llamada interaccionUsuario en la cual se le muestran cosas por pantalla al usuario y se escanean sus decisiones.

El juego empieza mostrando su parte gráfica, en una ventana sin marcos de 800x600 píxeles, en la mitad de la pantalla, el jugador debe avanzar hasta la puerta y por consola se le da la bienvenida, ahí se detiene el thread de gráficos y sigue con la parte de batalla y movimiento entre pueblos.

La interface del juego comienza con un menú y se le solicita que se elija la clase del personaje a utilizar.

Luego de haber elegido una clase, se podrá elegir el camino a seguir.

Al avanzar por el camino norte se sale del pueblo inicial y ya se puede tener encuentros con algún npc.

Al tener estos encuentros podemos elegir entre luchar o escapar, si elegimos escapar, la probabilidad de que podamos irnos o no será aleatoria.

Si no podemos escapar o elegimos luchar empezara el duelo, cada uno elegirá su habilidad y atacará. El primero en dar el golpe será quien tenga una mayor velocidad.

Si salimos victoriosos ganaremos experiencia y podremos continuar. Si morimos se acaba el juego.

Diario Semanal:

Semana 1 (06/05): Se formó el grupo con Lucas Acosta y Dante Scrosoppi. Eligieron el tema del proyecto: juego de rol. Realizaron un primer diseño de clases primitivo.

Semana 2 (13/05): Se sumaron Florencia Excoffon y Julián Aga al grupo al saber que era si o si de a tres o cuatro. Se realizan los primeros códigos, se crea el drive comunitario y nos comunicamos por discord.

Semana 3 (20/05): Se sigue con el código. Empezamos a investigar código para la interfaz gráfica, guiándonos por videos tutoriales de youtube.

Semana 4 (27/05):

Semana 5 (03/06): Finalizamos el código gráfico. Comenzamos a ajustar cosas del código del programa implementando las cosas pedidas para el mismo. El día 5 de junio nos juntamos en la casa de Dante para organizar las últimas cosas que faltan del trabajo. Hacemos el código de Json e implementamos hashmaps al trabajo.

Semana 6 (10/06) : Nos comunicamos a través de discord para ultimar detalles y hablar del código, mejoras, etc. Comenzamos a utilizar Github eficientemente (antes solo Julián comprendía bien su uso), nos simplifica mucho las cosas.

Semana 7 (17/06): Seguimos ultimando detalles, redactando el informe final, agregamos la documentación correspondiente en formato jdoc, implementamos más conceptos que no habíamos implementado antes.

Tabla de Problemas y Soluciones:

|  |  |  |
| --- | --- | --- |
| Fecha | Problema | Solución |
| 19/05 | Problemas compilando cuando se crea la función mostrarEnPantalla | Añadir restricciones que no permitan bajar de 0 ni subir más allá del ancho de pantalla permitido. |
| 22/05 | Problemas para cargar los Sprites | Se acomodaron la variable “lado” de cada función y se posicionaron bien los sprites, también se acomodo el valor de los Sprites a 32 px x 32 px |
| 29/05 | Problemas mostrando Sprites en pantalla | Los Sprites no se muestran o no se acomodan porque se debe poner de forma inversa |
| 03/06 | Error cuando el jugador sale del mapa | Más restricciones de hasta dónde puede ir el jugador y hasta donde se imprimen los Sprites en pantalla |
| 15/06 | Error de compilación intentando mezclar códigos |  |
| 18/6/1985 | Problema al pasar el repositorio de GitHub a eclipse. | arreglar el build path |
| 18/06 | No nos sale como hacer un get de un hashmap para poder ponerlo como valor al pasar los stats al formato JSON. | No buscar en los hashmap. |

|  |  |  |
| --- | --- | --- |
| 15/06 | Error con los constructores de la clase jugador del código de Dante y el de Julián, el parámetro que carga el sprite, al no necesitarlo el código de dante rompía el juego | Se creó una clase auxiliar para las batallas de esa parte del código |

Bibliografía:

<https://www.youtube.com/watch?v=qa6GA5p9nQ0&list=PLN9W6BC54TJJr3erMptodGOQFX7gWfKTM>

<https://marcusman.com/>

<http://javadevone.com/>

<https://stackoverflow.com/questions/15424834/java-lang-illegalargumentexception-input-null-when-using-imageio-read-to-lo>

<https://stackoverflow.com/questions/27420945/java-lang-nullpointerexception-attempt-to-invoke-virtual-method-on-a-null-objec>

<https://es.stackoverflow.com/questions/3547/por-qu%C3%A9-no-puedo-ejecutar-una-clase-java-creada-y-compilada-desde-netbeans-por>

<https://foro.elhacker.net/java/ayuda_error_al_ejecutar_un_java-t356083.0.html>

<https://www.youtube.com/watch?v=coK4jM5wvko>