Manual de Proyecto

Integrantes

- 109186 - Esteban Ariel Brandán
- 106930 - Ignacio Viau
- 108230 - Lucas Ezequiel Villarrubia

División de tareas

Esteban	Ignacio	Lucas
• Implementación de armas y proyectiles en cliente y servidor • Control de sistema de físicas de proyectiles • Correcta visualización de armas con sprites correspondientes en el cliente con zoom ya implementado • Cambios de estado de personajes e interacciones con armas • Adaptación de protocolo para el ingreso de comandos de usuario en cliente y servidor para interacciones con armas	• Setup de librerías SDL para el renderizado de la demo y manejo de runtime errors en Renderer • Renderizado de personajes, texturas en runtime • Renderizado de mapa en cliente gráfico: zoom-in y zoom-out de cámara que sigue a patos en juego • Utilización de mapas en cliente y servidor en distintas clases • Movimientos de personajes y control de sistema de físicas primero en cliente • Construcción de mapas y entidades dentro de mapas: inserción por medio de archivos .yaml • Instalador shell de demo	• Modelado de esquema de clases e hilos utilizados en servidor y cliente • Modelado de protocolo de comunicación principal en clases y structs • Adaptación de servidor con estado de juego y personajes según estado inicial • Manejo de input de usuario por teclado y envío de información de cliente a servidor • Actualización de estado e información de cliente para renderizado con frame dropping adecuado • Construcción de lobby de juego y manejo de eventos de lobby • Motor de colisiones de personajes con elementos del terreno de juego • Documentación

Items faltantes

1. Patos: Se agacha/tira al piso, se cae y sigue resbalando por el piso cuando pisa una banana, interacciones con armaduras, apunta arma hacia arriba (a 90°).
2. Armas y Armaduras: Casco, Armadura, Granada, Banana, Pew-Pew Laser, Laser Rifle, Magnum (parcialmente implementada), Escopeta, Sniper. Explosiones, retroceso y desviación de proyectiles.
3. Sonidos: música, disparos, explosiones
4. Servidor: Partidas: lógica de victoria, rondas de juegos, tiempo de reaparición de cajas, daños del pato por arma, multipartidas.
5. Cliente: Escenario: Spawn places y Cajas: aparecen cada cierto tiempo, se rompen (cuando son disparadas), dan recompensas, explotan.
6. Configuración con YAML: Constantes de juego, daños, vida, velocidades, etc.
7. Cheats para easy-testing.
8. Instalador que cumple las directivas del enunciado.
9. Tests unitarios del protocolo de comunicación.

Recursos utilizados

Para este trabajo práctico se utilizaron las librerías de las clases Socket, Resolver, ResolverError, LibError, Queue, y Thread; y el esquema de directorios y archivos CMakeLists.txt provistos en tres repositorios de la cátedra:

- hands-on-sockets
- hands-on-threads
- template

Entre otras tecnologías, se utilizaron los IDEs de CLion y Visual Studio Code, así como el Editor de Texto para programar el código presentado.
Se utilizó el repositorio template provisto por la cátedra para el setup del proyecto con *cmake* y makefile.
En el caso de la visualización de la demo, se utilizaron las librerías *image*, *mixer* y *ttf* de SDL2, y en el código, clases de su versión SDL2pp.
Para el lobby de la demo, se utilizó la librería QT en su versión 6 y su personalización fue esquematizada al principio con QTCreator.