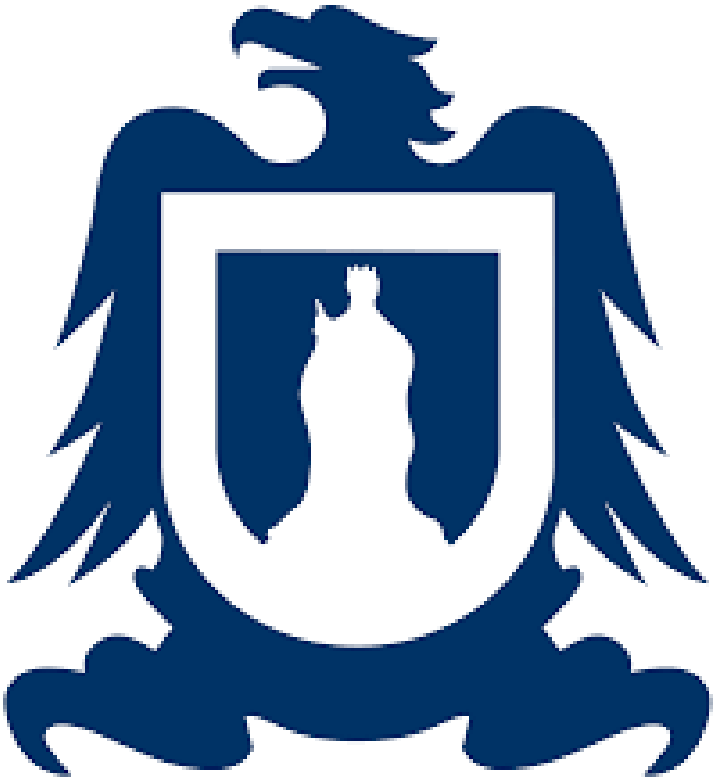


The Normal Knight

MANUEL DEL PROGRAMADOR

Profesor: Dr. Francisco Javier Torres Reyes.
Materia: Tecnología Orientada a Objetos.
Facultad de ingeniería.
Carrera: Ingeniería en Computación.
Semestre: 2025 – 2026 I

Por:
Luis Alejandro Castillo Vessi | 301304 |
Oman León Aguilar | 336496 |

Ingeniería en Computación	
Objetivo Educacional	**Atributo de Egreso**
OE1: Los egresados son profesionales que se desempeñan con éxito en la práctica de la Ingeniería en Computación o de las áreas relacionadas con las Tecnologías de la Información y la Comunicación.	**AE1.** Aplicará los principios de las matemáticas, fundamentos de ingeniería e ingeniería computacional para la solución de problemas complejos en ámbitos sociales, públicos o privados
Ingeniería en Sistemas Inteligentes	
Objetivo Educacional	**Atributo de Egreso**
OE1: Los egresados son profesionales que se desempeñan con éxito en la práctica de las Ciencias Computacionales o de las áreas relacionadas con las Tecnologías de la Información y la Comunicación.	**AE1.** Aplicará los principios de las matemáticas, ingeniería y de las ciencias computacionales para la solución de problemas complejos en ambitos sociales, públicos o privados.

Ingeniería en Computación	
Objetivo Educacional	**Atributo de Egreso**
OE2: Los egresados son ingenieros competitivos a nivel nacional e internacional capaces de adaptarse a diferentes ámbitos de trabajo.	**AE2.** Analizará, diseñará e implementará el proceso del desarrollo de sistemas computacionales de software y hardware que resulten en proyectos que cumplen necesidades con requerimientos reales en ambientes tales como económicos, sociales, políticos, éticos, de seguridad, salud, manufactura y sustentables.
Ingeniería en Sistemas inteligentes	
Objetivo Educacional	**Atributo de Egreso**
OE2: Los egresados son ingenieros competitivos a nivel nacional e internacional capaces de adaptarse a diferentes ámbitos de trabajo.	**AE2.** Analizará, diseñará e implementará el proceso del desarrollo de software inteligente que resulten en sistemas computacionales que cumplen necesidades con requerimientos reales en ambientes tales como económicos, sociales, políticos, éticos, de seguridad, salud, manufactura y sustentables.

The Normal Knight

Sobre el diseño de las clases del proyecto

Las clases clave del proyecto son: GamePanel, Interfaz de Usuario, Jugador, ManejadorObjetos, ManejadorTiles, ControladorEventos, DetectorColisiones.

GamePanel, panel principal del juego, donde se maneja la lógica del juego, la actualización de entidades, el dibujo de gráficos y la gestión de la cámara. Administra todos los objetos requeridos para el buen funcionamiento del programa. Aquí es donde se instancian y configuran los objetos necesarios de las diferentes clases para que inicie el juego.

InterfazDeUsuario, gestiona la interfaz de usuario (UI) del juego. Se encarga de dibujar elementos en pantalla como mensajes, transiciones y otros componentes visuales relacionados con la UI. Se maneja y dibuja los mensajes hacia el jugador, dibuja menús, pantallas de diálogo, crea las transiciones entre mapas y mundo, configura y dimensiona imágenes en el cuadro de la pantalla.

Jugador, representa al jugador principal del juego. Hereda de Entidad. Maneja la lógica específica del jugador, como la carga de sus sprites, la actualización de su estado basada en la entrada del teclado, y su dibujado en la pantalla, además de la interacción con objetos y NPC´s.

MajenadorObjetos, gestiona todos los objetos dentro del juego, tanto al jugador como NPC´s enemigos e ítems. Carga las imágenes de los objetos y las coloca en el mapa. Actualiza a los objetos que son entidades.

ManejadorTiles, gestiona todos los mosaicos (tiles) del juego. Se encarga de cargar las imágenes de los tiles, leer los archivos mapa (.txt), almacenar la estructura del mapa y dibujar el mapa en la pantalla en función de la posición de la cámara.

ControladorEventos, se dedica a gestionar los eventos dentro del juego, revisando constantemente si el jugador ha activado algún evento.

DetectorColisiones, es responsable de detectar colisiones entre entidades y el entorno de juego.

Sobre los métodos

Update y draw de cada clase son clave para el funcionamiento del proyecto.

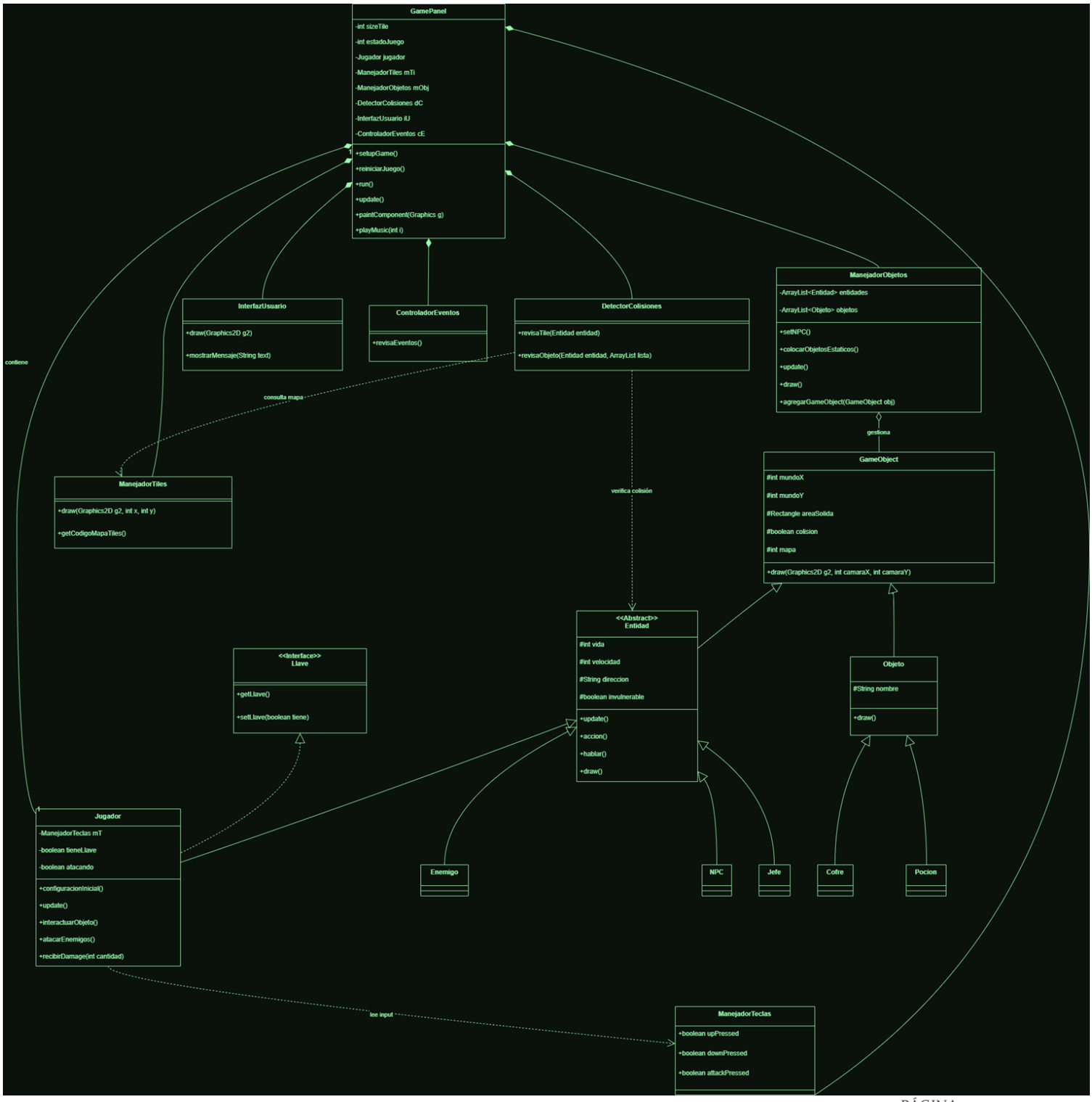
Update, actualiza el estado en tiempo real de cada evento hecho por el jugador, permitiendo fluidez del programa.

Draw, cada variante del método draw en cada clase, permite al IDE redibujar en pantalla cada evento hecho por el jugador, calculando y dimensionando respecto al cuadro de la pantalla.

Sobre el tiempo invertido

El tiempo invertido en total fue de 27 horas aproximadamente.

Diagrama de clases UML del proyecto



Sobre los problemas del proyecto y las soluciones encontradas

No aparece una vez que se muere y reinicia el juego

Al reiniciar juego, una vez que el jugador moría, se tiene que reiniciar la configuración inicial del jugador, donde se define en que mapa este se va a dibujar, pero al morir en una mazmorra, se reiniciaba el mundo y después se configuraba al jugador, ocasionando que el jugador no cambiara su configuración de ubicación nueva, por ende, se dibujaba el mundo "cementerio", pero el jugador no, ya que este estaba configurado en aparecer en la mazmorra, su última configuración. Se arreglo moviendo una línea de código, primero se resetea el mundo, se hace el cambio de mapaActual de mazmorra a mundo y luego se resetea la configuración inicial del jugador respecto al mapaActual.

Colisiones fantasmas

Se encontró un error con la lógica dentro de las mazmorras, el jugador colisionaba con objetos inexistentes dentro de ellas, se identificó que había un problema al redibujar el mapa "mazmorra" sobre el mundo "cementerio", al hacer la transición, las colisiones de los objetos no se validaban de forma local respecto a lo dibujado en la pantalla, se comparaba colisiones con jugador uniendo mapa "mazmorra", con mundo "cementerio". Se arreglo, indicando que, al dibujar un mapa nuevo, se comparara solo las colisiones de ese mapa en específico.

Música

Al morir el jugador, la música no se detenía, se detectó que se estaba usando un solo objeto de clase soundtrack para manejar todos los sonidos, tanto la música de ambiente como los efectos de sonido. Se soluciono creando dos objetos de clase soundtrack, una para la música y otro para los efectos, evitando problemas de lógica.

Sprites

Al tener los gráficos limitados a 16x16 pixeles, ocasionaba problemas al crear nuevos sprites, debido a que el arte perdía calidad por no tener suficiente resolución,

ocasionando que los personajes perdieran sustancia, siendo muy difícil reflejar acciones o hasta las propias siluetas del personaje. Se soluciono dibujando a los personajes en una resolución mucho más alta, 128x128 pixeles, después, usando las herramientas de la aplicación Pixel Studio, se redimensionaba el Sprite a 32x32, y por código se hacia la redimensión final a 16x16 usando el método drawImage y la escala de tamTile.