# **Momento III - Informe**

Jorim de Jesus Saltarin Villamizar

Informática II

Augusto Enrique Salazar Jiménez

Universidad de Antioquia

09 de julio de 2025

**Dragon Ball: Camino al Torneo**

El presente proyecto, titulado “Dragon Ball: Camino al Torneo”, consiste en el desarrollo de un videojuego inspirado en el universo de Dragon Ball, centrado en la etapa de entrenamiento de los personajes antes de participar en el Torneo de las Artes Marciales.

Este juego fue implementado utilizando C++ Y Qt, aplicando principios de programación orientada a objetos, manejo de gráficos 2D, animaciones, eventos de teclado, temporizadores (QTimer) y reproducción de sonidos mediante QMediaPlayer. El jugador puede seleccionar entre 2 personajes (Goku y Krilin), cada uno con sus propias animaciones, características de movimiento y acciones como caminar, saltar, golpear y esquivar obstáculos.

**Nivel 1: Entrenamiento en la Isla del Maestro Roshi**

En este primer nivel, el jugador asume el rol de Goku o Krilin niño, previamente seleccionado mediante una pantalla de selección interactiva. El escenario está ambientado en la Isla del Maestro Roshi, con una vista lateral 2D, que combina elementos naturales como vegetación, palmeras y rocas.

El objetivo principal es completar un circuito de entrenamiento recolectando 10 piedras marcadas, mientras se evitan rocas, con una cantidad limitada de vidas y un tiempo limitado de 60 segundos para cumplir con las 10 piedras recolectadas. El entorno y la jugabilidad simulan una rutina de entrenamiento supervisada por el Maestro Roshi, antes del Torneo de las Artes Marciales.

**Vista del Nivel**

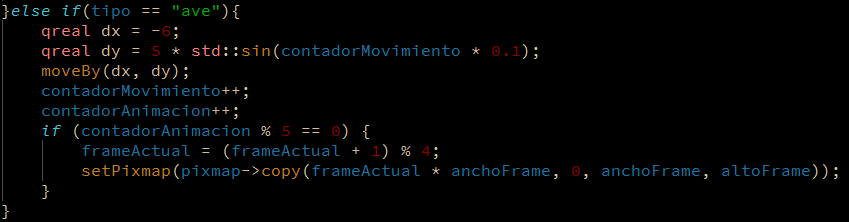
* **Estilo visual:** side.scroller 2D con vista lateral
* **Desplazamiento del fondo:** animación de parallax al avanzar (scroll del fondo hacia la izquierda al moverse a la derecha)
* **Ambientación:** playa tropical con vegetación, arena, palmeras, troncos, piedras.
* **Personaje jugable:** Goku o Krilin en su etapa infantil, con animaciones (caminar, saltar, recibir daño)

**Interacciones y Mecánicas**

* **Jugador:**
  + Movimiento lateral controlado por el teclado (←, →)
  + Salto con trayectoria parabólica (espacio)
  + Animaciones para caminar, saltar, aterrizar, atacar y recibir daño
  + Capacidad para recolectar objetos al colisionar con ellos
  + Reacción visual al ser golpeado + efecto de sonido
  + HUD: contador de piedras recolectadas y barra de vidas en pantalla
* **Entorno:**
  + **Obstáculo:** rocas dispersas
  + **Colisiones:** Si el jugador choca con un roca o un ave, reduce su barra de vida
  + **Objetos:** piedras marcadas que incrementa el contador

**Físicas y Colisiones**

* **Movimiento horizontal:** basado en eventos de teclado. Se animan los sprites con ayuda de QTimer o control frame a frame, utilizando una hoja de sprite organizadas por filas y columnas.
* **Salto parabólico:** implementado con una velocidad inicial y una aceleración gravitatoria (velocidadY += gravedad), en donde se actualiza la posición vertical del personaje.
* **Colisiones:** detección por collidesWithItem(). el cual se utiliza en, si el personaje colisiona con una roca, se reduce una vida. Si colisiona con una piedra, se recolecta y suma al contador.
* **Movimiento tipo senoidal:** se implementó un movimiento ondulatorio usando una función senoidal. Simula el subir y bajar de una ave mientras se desplaza horizontalmente. dy = A \* sin(fase \* frecuencia), en donde A es la amplitud y representa que tanto sube y baja el ave. Fae se incrementa con cada actualización del movimiento del objeto con ayuda de un contador. Y la frecuencia controla la rapidez al subir y bajar. En el codigo se represento de la siguiente manera



En donde también cuenta con una animación visual basada en un sprite de 4 frames que simulan el vuelo del ave, esto mediante un contador que logra que se vea animado mientras vuela (abrir y cerrar alas), construyendo un efecto físico y visual

**Nivel 2 - Combate**

En este segundo nivel el jugador continúa la aventura participando en el 21° Torneo de las Artes Marciales, es un estilo de combate 1 contra 1. El jugador puede controlar a Goku o Krilin niño (dependiendo del personaje que eligió en el nivel 1) y enfrentarse a Jackie Chun (Maestro Roshi disfrazado) manejado por la computadora.

**Vista del Nivel**

* Vista lateral fija 2D
* Fondo del **Martial Arts Tournament Arena**
* El personaje controlado por el jugador (Goku o Krillin) se ubica a la izquierda, y Jackie Chun a la derecha.

**Interacciones y jugabilidad**

* **Jugador:**
  + **Movimiento:** izquierda, derecha y salto
  + **Ataques:** puño (X), patada (Y)
* **Jackie Chun:**
  + Movimiento automático simple
  + Ataque con golpes o patadas aleatorios
  + Detecta colisión con el jugador y aplica daño
  + Reacciona a los golpes con animación, efecto de sonido, opacidad y retroceso.

**Sistema de Combate**

* 1 contra 1 con barras de vida visibles para ambos personajes
* Al recibir daño:
  + Disminuye la barra
  + El personaje reacciona con animación de golpe, movimiento hacia atrás, efecto de sonido de golpe recibido y opacidad reducida
* Al llegar a 0 vidas:
  + Efecto de sonido de “muerte”
  + Se detienen los movimientos
  + Se emite la señal según sea el caso



**Físicas y colisiones:**

* **Saltos:** mismo sistema del primer nivel
  + velocidadY se incrementa con la gravedad en cada iteración
  + Al llegar al suelo, se detiene
  + Puede moverse de forma horizontal mientras salta, simulando un salto parabólico
* **Colisión - impulso:** 
  + El personaje golpeado se desplaza en el eje X, dando como resultado un efecto de retroceso al recibir el golpe

**Arquitectura**

Se desarrolló aplicando principios de la POO, lo que permitió mayor reutilización del código y una posible escalabilidad.

* **Herencia:** se creó una clase padre **Personaje,** que define atributos y comportamientos comunes como:
  + Movimiento
  + Salto y caída
  + Reacción de colisiones
  + Vidas y muerte
  + Animaciones
  + Contadores
  + Señales

De esta clase heredan los 3 personajes del juego, Goku, Krilin y Roshi. Cada subclase puede especializar sus propias animaciones y atributos al redefinir variables como tamaño del sprite (ancho, alto)

* + Posición inicial
  + Cantidad de cuadros por animacion
  + Rutas
  + Coordenadas X,Y

Ejemplo:

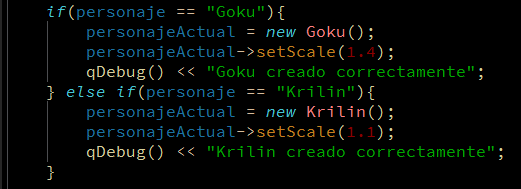


Aunque los métodos como *animar(), animarSalto()* o *actualizar()* son similares, cada personaje adapta su lógica dependiente de spritesHeet, número de cuadros y tamaño

* **Polimorfismo y uso de virtual:** En la clase padre Personaje, varios métodos fueron declarados como virtual, lo cual habilita polimorfismo en tiempo de ejecución, lo que permite que las clases hijas sobreescribir métodos como:
  + iniciarAnimacion()
  + detenerAnimacion()
  + perderVida()
  + reaccionGolpe()
  + animarPuno()
  + animarPatada()
  + spriteVictoria()
  + actualizarBarraVida()

Esto debido a que los personajes pueden variar su comportamiento, además se evita repetir código ya que muchas funcionalidades comunes se heredan directamente de Personaje.

* **Memoria Dinámica:** 
  + **Objetos:** En todo el desarrollo se implementó múltiples elementos con new. Por ejemplo el personaje principal (Goku o Krillin) se crea dinámicamente según la elección del jugador. Lo que permite que el flujo del juego funcione con distintas clases.



* + Elementos como la barra de vida y textos, se crean y administran dinámicamente desde la clase Personaje, permitiendo que:
    - Se adapten a cada instancia
    - Puedan ser eliminados al morir el personaje o reiniciar el nivel

**Evolución del Proyecto**

El desarrollo del videojuego “Dragon Ball: Camino al Torneo” se llevó a cabo en distintas etapas, inicialmente por el diseño e idea del flujo general, siguiendo los capítulos seleccionados y lo planteado en los momentos I y II. Comenzando con el primer nivel de Entrenamiento, con su respectiva clase, en la cual se se lleva el control de todo el nivel; desde creación de la escena y añadir los respectivos objetos a la misma. Culminando con un nivel dos, el cual se basa en un combate 1 vs 1, la cual también cuenta con su respectiva clase, que almacena toda la lógica del nivel. Teniendo en cuenta todo este proceso que se llevó a cabo, partimos de una simple escena de movimiento lateral hasta incorporar más personajes, animaciones personalizadas, recolección de objetos, gestión de vidas, detección de colisiones y combate, en donde cada proceso que se realizó se refinaron y afianzaron aspectos técnicos y visuales.

Inicialmente, se creó una estructura base para el personaje (movimiento, salto y recolección). A medida que se avanzaba, se consolidó la idea de la clase Personaje, la cual se extendió para integrar nuevas funcionalidades, lo que permitió la importancia de la POO y la herencia, al permitir que las mejoras aplicadas a la clase padre impactaran de forma positivas en todas las clases hijas (Goku, Krilin y Roshi).

**Desafíos**

Durante el desarrollo se presentaron varios retos técnicos y de diseño, que también ayudaron a la apropiación de algunos conceptos clave:

* **Manejo correcto de memoria dinámica:** creación y destrucción de objetos sin generar desbordamientos de memoria o errores por punteros al vacío.
* **Animaciones específicas:** debido a que cada personaje posee diferentes dimensiones y frames en su hoja de sprites, fue necesario sobreescribir métodos heredados.
* **Flujo entre niveles:** coordinación de eventos que indican la victoria o derrota de un nivel o un round.
* **Colisiones y físicas:** sistema básico de salto parabólico, detectar colisiones con obstáculos o rival y reaccionar según sea el caso (daño, recolección).
* **Sincronización entre acciones y sonidos.** ya que se investigó la forma de agregar efectos de sonido, se debía coordinar el cambio de sprite con la reproducción del sonido adecuado.

**Flujo**

El jeugo está compuesto por dos niveles principales con transiciones claras y fluidas, lo que permite mantener coherencia en la narrativa y la jugabilidad.

**Nivel 1 - Entrenamiento en la Isla del Maestro Roshi**

* El jugador selecciona entre Goku o Krilin
* Se enfrenta a un circuito de obstáculos (rocas) y debe recolectar 10 piedras marcadas
* Al complementar la recolección, se emite una señal que da paso al siguiente nivel.
* El jugador cuenta con 3 intentos para pasar el nivel, el cual si un intento sus vidas llegan a 0, se reinicia el nivel automáticamente, si pierde los 3 intentos vuelve a la pantalla de selección, para decidir volver a jugar o no. Además, cuenta con un tiempo determinado de 60 segundos, el cual sí también llega a 0 perderá el intento.

**Nivel 2 -Combate contra Jackie Chun**

* Aparece Jackie Chun como enemigo controlado automáticamente, con código de seguimiento al rival (Goku o Krilin) básico, lanzando ataques de patada o puño.
* El jugador puede golpear, saltar para esquivar y perder vidas. Los dos cuentan con la barra de vida.
* Consiste en 3 rounds, cada uno con un tiempo de 30 segundos, al vencer al rival o quedarse sin vidas, se muestra un mensaje de victoria, derrota o empate, en donde quien salga victorioso de 2 rounds será el ganador.

**Diagrama de Clases actualizado**

