UNIVERSITÀ DEGLI STUDI
DI PERUGIA

Tesina Finale di

Programmazione di Interfacce Grafiche e Dispositivi Mobili

Corso di Laurea in Ingegneria Informatica ed Elettronica – A.A. 2022-2023

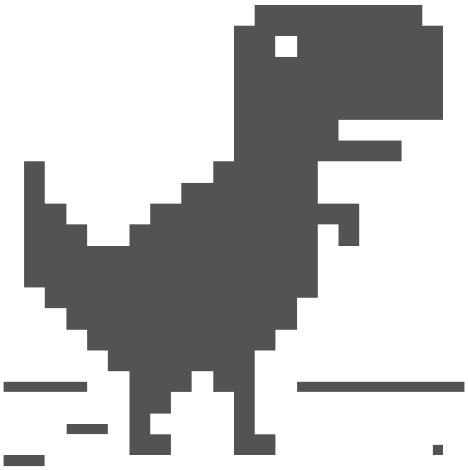
Dipartimento di Ingegneria

docente

Prof. Luca Grilli

Chrome Dino

applicazione desktop JFC/Swing



studente

314666 **Gian Marco Ferri** gianmarco.ferri@studenti.unipg.it

Data ultimo aggiornamento: 17 dicembre 2022

0.Indice

1.Descrizione del Problema

L'obiettivo di questo lavoro è lo sviluppo di un'applicazione desktop, denominata *Chrome Dino*, che realizza una versione semplificata dell'omonimo videogioco [1, 2].

L'applicazione sarà implementata utilizzando la tecnologia JFC/Swing in modo da favorire un'ampia portabilità su diversi sistemi operativi (piattaforme), riducendo al minimo eventuali modifiche al codice sorgente.

Di seguito sarà data una breve descrizione del videogioco originale *Chrome Dino* [1, 2], dopodiché si fornirà una descrizione della versione semplificata che si intende realizzare.

1.1 Il Videogioco Chrome Dino

Uno degli Easter Egg più famosi di Google Chrome è *Chrome Dino*, che appare quando si tenta di visitare un sito Web mentre si è disconnessi da Internet.

Questo gioco, noto anche come *The Dinosaur Game*, è un browser game sviluppato da Google e integrato nel browser web Google Chrome.

Il giocatore guida un Tirannosauro Rex pixelato attraverso un paesaggio desertico a scorrimento laterale, cercando di evitare degli ostacoli rappresentati da cactus per ottenere un punteggio più alto possibile.

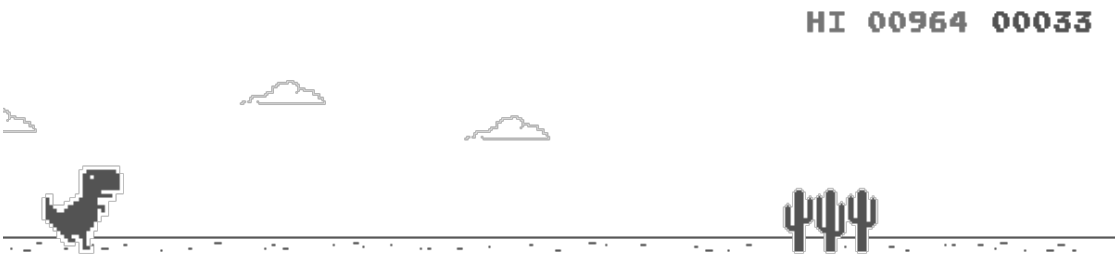


Figura 1.1: Schermata del videogioco *Chrome Dino* originale

Il T-rex è stato progettato e introdotto nel 2014 da Sebastien Gabriel, un visual designer noto anche per la progettazione di altre icone di Chrome, con l'idea di introdurre un messaggio di errore di mancata connessione ad Internet che si spera l'utente non veda mai. Se purtroppo dovesse accadere, si cerca di rendere questo inconveniente il più piacevole possibile distraendo l'utente, in modo che la frustrazione lasci il posto ad un momento di piacevole svago.

La scelta del tirannosauro come protagonista del gioco non è casuale, infatti il messaggio di fondo è: quando si perde la connessione a Internet è come essere catapultati indietro nell'era dei dinosauri.

Il gioco è un runner infinito che non ha né un epilogo, né limiti di tempo. Mentre cerca di andare avanti il più a lungo possibile, l'utente si dimentica che in realtà sta affrontando uno dei più grandi problemi dei nostri tempi: l'assenza di internet.

Attualmente ci sono 270 milioni di giocatori ogni mese, sia su laptop che su dispositivo mobile. Non sorprende che la maggior parte degli utenti provenga da zone con connessioni internet inaffidabili o costose, come India, Brasile, Messico o Indonesia.
Questo gioco è diventato così popolare che alcuni studenti disconnettevano intenzionalmente i loro Chromebook scolastici da Internet, consentendo al gioco del dinosauro di caricarsi all'interno del browser web Chrome, senza alcuna restrizione.
Gli sviluppatori di Google hanno preso molto sul serio questo problema e hanno sviluppato politiche per disabilitare il gioco sui Chromebook scolastici, e in seguito è stata fatta una cosa simile anche per altre piattaforme.

È stato anche creato l'URL `chrome://dino`, dove le persone possono giocare senza andare offline. La pagina offre una "modalità arcade" in modo che i gio-

catori possano allenarsi per cercare di ottenere dei punteggi più alti possibili in un'esperienza a finestra intera.

1.2 L'applicazione Chrome Dino

Il gioco può essere avviato premendo la barra spaziatrice.

Durante la partita il T-Rex si sposta continuamente da sinistra a destra attraverso un paesaggio desertico, con il giocatore che tenta di evitare gli ostacoli in arrivo rappresentati da cactus saltando o abbassandosi.

Premendo la barra spaziatrice il dinosauro salta, mentre premendo il tasto freccia giù (↓) il dinosauro si abbassa.

Il gioco continua fino a quando l'utente non sbatte contro un ostacolo, provocando il "Game Over".

A questo punto si può ricominciare a giocare semplicemente premendo la barra spaziatrice.

2.Specifica dei Requisiti

L'applicazione Chrome Dino che si intende realizzare dovrà soddisfare i seguenti requisiti.

1. Il programma è costituito da una semplice finestra grafica in cui all'inizio è rappresentato il dinosauro, personaggio protagonista del gioco, insieme alle scritte "CHROME DINO" e "Press SPACE BAR to start".
2. Premendo la barra spaziatrice il gioco si avvia.
3. Durante la partita il personaggio si sposta continuamente da sinistra a destra attraverso un paesaggio desertico con delle nuvole sullo sfondo.
4. L'animazione consiste nell'emulare sia la corsa del personaggio, con un frame specifico in caso di morte, che lo spostamento del terreno e dello sfondo.
5. Mentre il gioco è attivo, si può far saltare il personaggio premendo la barra spaziatrice, premendo invece il tasto freccia giù (↓) il dinosauro si abbassa.
6. Ogni volta che il personaggio salta, viene riprodotto il relativo suono.
7. Ad ogni ostacolo superato, il punteggio viene incrementato di 20 punti.
8. Quando il punteggio raggiunge 100 o un suo multiplo, viene riprodotto il relativo suono.
9. Il gioco finisce quando il dinosauro sbatte contro un ostacolo. A questo punto viene riprodotto il relativo suono e compare la scritta "Game Over" sullo schermo, assieme ad un'immagine del logo di replay che suggerisce di ricominciare il gioco.
10. Premendo la barra spaziatrice si può ricominciare a giocare.

3.Progetto

Viene ora descritta la struttura dell'applicazione realizzata, illustrandone prima l'architettura software per poi scendere nel dettaglio dei blocchi funzionali che la compongono.

3.1 Architettura del Sistema Software

Per la realizzazione di *Chrome Dino* si è scelto di basarsi sul pattern di programmazione Logic View (LV).
Questa particolare architettura software è ottenibile da un'altra architettura, Model View Controller (MVC), unendo Model e Controller.

Il Logic si occupa di rappresentare i dati gestiti dall'applicazione e gestisce la logica di funzionamento del programma.

Il View si occupa di rappresentare i dati all'utente raccogliendone l'input: in questo caso è costituito dall'interfaccia grafica.

Oltre alle classi appartenenti al Logic e al View, abbiamo altre tre classi contenute nel package "utils": Animation, Resource e AudioPlayer.
La classe Animation gestisce le animazioni del gioco, la classe Resource gestisce invece le varie immagini contenute nella cartella "data", mentre la classe AudioPlayer gestisce la riproduzione degli effetti sonori rappresentati da file audio, anch'essi contenuti nella cartella "data".

Di seguito vengono descritti nel dettaglio i moduli Logic e View con le relative classi.

3.2 Logic

Le classi appartenenti al blocco Logic di *Chrome Dino* si trovano nel package "logic". La loro struttura è rappresentata nel diagramma UML [3] in figura 3.1.

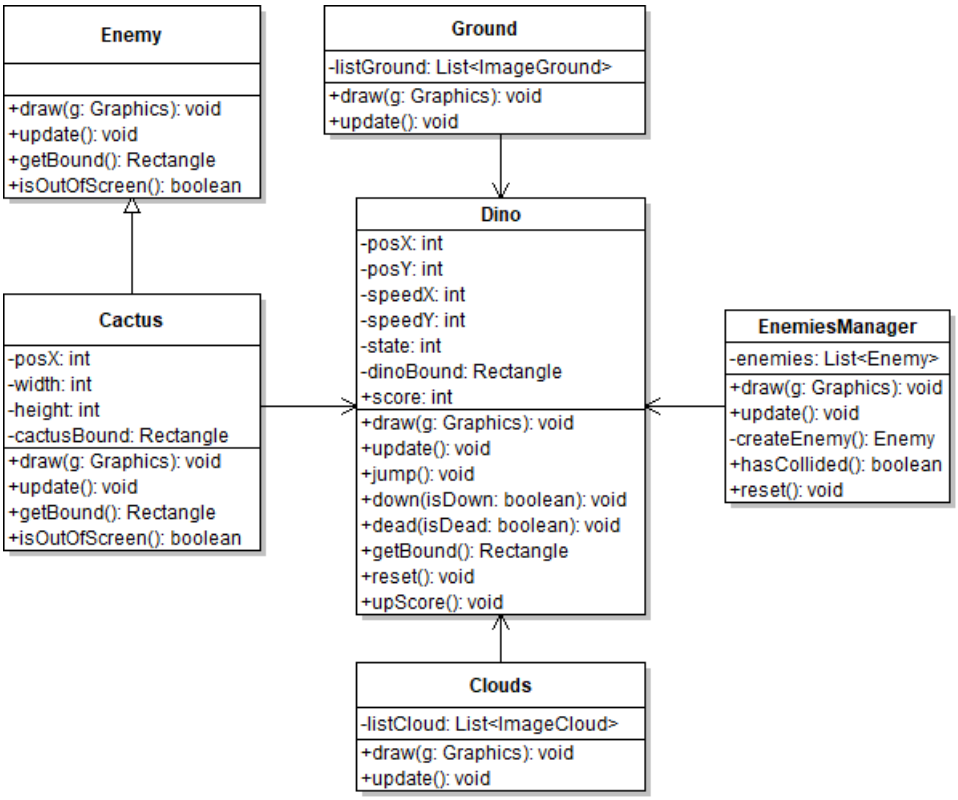


Figura 3.1: Diagramma UML del modulo Logic

In particolare:

- **Enemy** è una classe astratta che rappresenta un nemico generico.
  Dispone di quattro metodi astratti: `draw(Graphics g)`, `update()`, `getBound()`, `isOutOfScreen()`.

- **Cactus**, che estende la classe Enemy, rappresenta il "nemico" del personaggio principale. Memorizza la coordinata sull'asse X, l'altezza, la larghezza del cactus e il suo "bound", ovvero il rettangolo immaginario in cui è contenuto, la cui collisione con il dinosauro causa il "Game Over".
  Dispone del metodo `isOutOfScreen()`, volto a verificare se il nemico è stato superato correttamente.

Dispone inoltre del metodo `getBound()` che restituisce la dimensione del "bound" del cactus.
I metodi `draw(Graphics g)` e `update()` servono rispettivamente a disegnare e aggiornare il cactus sullo schermo.

- **EnemiesManager** è la classe che gestisce i nemici, ovvero i cactus.
  Essa si occupa di creare (`createEnemy()`), disegnare (`draw(Graphics g)`), aggiornare (`update()`), resettare (`reset()`) i nemici sullo schermo e verificare se è avvenuta una collisione con il personaggio principale (`hasCollided()`), evento che porta al "Game Over".
  Ci sono due immagini di cactus nella cartella "data" da cui EnemiesManager attinge mediante il metodo `getResourceImage(String path)` della classe Resource presente nel package "utils".
  I diversi tipi di cactus vengono creati e disegnati sullo schermo in maniera randomica.

- **Ground** è la classe che rappresenta il suolo su cui corre il dinosauro.
  Ci sono tre immagini di terreno nella cartella "data" da cui Ground attinge mediante il metodo `getResourceImage(String path)` della classe Resource.
  I metodi `draw(Graphics g)` e `update()` servono rispettivamente a disegnare e aggiornare il suolo sullo schermo.
  I diversi tipi di terreno vengono disegnati in maniera randomica e sono rappresentati da una lista.

- **Clouds** è la classe che rappresenta le nuvole presenti nello sfondo.
  C'è un'immagine di nuvola nella cartella "data" da cui Clouds attinge mediante il metodo `getResourceImage(String path)` della classe Resource.
  I metodi `draw(Graphics g)` e `update()` servono rispettivamente a disegnare e aggiornare le nuvole sullo schermo.
  Le diverse nuvole vengono disegnate sullo sfondo in maniera randomica e sono rappresentate da una lista.

- **Dino** è la classe che rappresenta il personaggio principale.
  Per rappresentare il dinosauro c'è bisogno delle sue coordinate spaziali, della sua velocità, del suo "bound", del punteggio, dei vari frame che vengono aggiornati per ottenere un'animazione fluida e degli effetti sonori.
  Per quanto riguarda l'animazione del dinosauro, Dino utilizza la classe Animation del package "utils", prendendo i vari frame presenti nella cartella "data" mediante il metodo `getResourceImage(String path)` della classe Resource.

Dino gestisce il salto, l'abbassarsi, la morte e l'aggiornamento del dinosauro. In particolare mediante il metodo `draw(Graphics g)` la classe disegna il personaggio nei vari stati: Run, Down Run, Jumping e Death.
Il metodo `update()` serve invece ad aggiornare i vari frame del personaggio, così da introdurre un'animazione fluida.
Il metodo `jump()` gestisce il salto del dinosauro, riproducendo anche il relativo effetto sonoro.
Il metodo `getBound()` invece restituisce il rettangolo immaginario in cui è contenuto il personaggio principale, utile nel gestire le eventuali collisioni.
Il metodo `upScore()` aggiorna il punteggio ad ogni ostacolo superato e riproduce un effetto sonoro quando si raggiunge uno score pari a 100 o un suo multiplo.
Il metodo `reset()` invece resetta il dinosauro e il punteggio in caso di "Game Over".

3.3 View

Le classi appartenenti al blocco View di *Chrome Dino* si trovano nel package "view". La loro struttura è rappresentata nel diagramma UML [3] in figura 3.2.

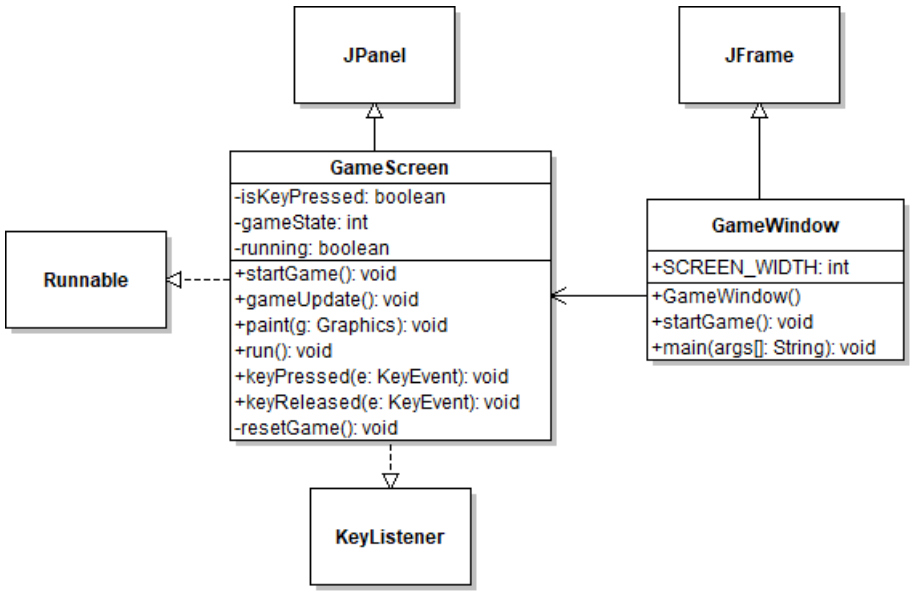


Figura 3.2: Diagramma UML del modulo View

In particolare:

- **GameWindow** estende JFrame e costituisce la finestra principale dell'applicazione grafica. Al suo interno si trova un pannello, istanza di GameScreen. Contiene il main e si occupa di far partire il gioco mediante il metodo `startGame()`.

- **GameScreen** estende JPanel e implementa Runnable e KeyListener.
  Questa classe contiene il metodo `startGame()`, invocato da **GameWindow**, che permette di far partire l'applicazione grafica su un thread specifico.
  La classe disegna e aggiorna il pannello grafico con tutte le sue componenti, ovvero il dinosauro, i cactus le nuvole e il terreno, tramite i metodi `paint(Graphics g)` e `gameUpdate()`. Quest'ultimo viene invocato dal metodo `run()` che rappresenta il "Game Loop" del videogioco.
  GameScreen implementa KeyListener, quindi è in grado di gestire tutti gli eventi relativi a qualsiasi input dell'utente tramite tastiera. Nel caso specifico di questa applicazione, l'utente può utilizzare solo la barra spaziatrice e il tasto freccia giù (↓).
  I metodi `keyPressed(KeyEvent e)` e `keyReleased(KeyEvent e)` permettono di specificare il comportamento del programma quando un tasto viene rispettivamente premuto o rilasciato. In particolare se l'applicazione è appena stata lanciata, si può iniziare a giocare premendo la barra spaziatrice; invece se il gioco è partito si può far saltare il dinosauro con la barra spaziatrice e lo si può far abbassare premendo il tasto freccia giù (↓). In caso di "Game Over", si può ricominciare a giocare premendo la barra spaziatrice
  Infine il metodo `resetGame()` permette di resettare il gioco in caso di "Game Over".

3.4 Problemi Riscontrati

Chrome Dino è piuttosto semplice dal punto di vista logico: sia la modellazione dei dati trattati che la gestione della logica applicativa non hanno comportato nessun particolare problema di implementazione.

L'aspetto che ha richiesto più tempo e più sforzi sicuramente è stato quello della gestione dell'animazione.
I vari frame devono venire aggiornati costantemente seguendo un'opportuna cadenza temporale per ottenere un'animazione fluida che conferisce una sensazione di movimento alla scena riprodotta. Per quanto riguarda l'animazione del dinosauro, la classe Dino utilizza la classe Animation del package "utils", prendendo i vari frame presenti nella cartella "data" mediante il metodo

`getResourceImage(String path)` della classe Resource.

Anche la gestione degli effetti sonori ha causato alcune problematiche. Innanzitutto alcuni file audio, nonostante fossero del formato corretto (.wav), hanno portato a diverse complicazioni. Alcune clip provocavano errori in fase di compilazione, altre invece non riuscivano ad essere lette e quindi utilizzate.
Un altro problema relativo all'audio è stato quello della riproduzione degli effetti sonori relativi ai salti ravvicinati del dinosauro. Utilizzando un solo oggetto Clip infatti, la clip audio non veniva riprodotta se si premeva la barra spaziatrice subito dopo averlo già premuta.
Si è quindi scelto di utilizzare un array di oggetti di tipo Clip anziché un solo oggetto. L'idea è quella di usare il primo oggetto per il primo evento di salto, il secondo oggetto per il secondo e così via procedendo secondo un ordine circolare, ovvero una volta giunti all'ultimo oggetto dell'array si deve ripartire dal primo.

4.Possibilità di estensione e personalizzazione

L'applicativo sviluppato rappresenta una versione minimamente semplificata del videogioco *Chrome Dino*, il quale, non essendo particolarmente complesso, non lascia molto spazio ad una eventuale espansione, se non introducendo una vera e propria personalizzazione non presente nel gioco reale.

Una possibile estensione per l'applicazione, non prevista nel gioco reale, potrebbe essere un sottofondo musicale che viene riprodotto durante la partita, interrotto da un eventuale "Game Over".
In questo caso però è sconsigliato utilizzare il formato .wav, già impiegato per la riproduzione degli effetti sonori rappresentati da clip audio di pochissimi secondi. Questo particolare tipo di formato infatti non è compresso, quindi potrebbe richiedere una significativa quantità di memoria. Un sottofondo musicale, al contrario di un effetto sonoro, può avere infatti una durata di diversi muniti e non di pochi secondi. Quindi se si vuole aggiungere un sottofondo musicale al videogioco bisogna utilizzare dei formati audio compressi, in particolare il formato .mp3, e utilizzare delle librerie che ne consentano, se possibile, la riproduzione in modalità streaming.
Una possibile alternativa invece è quella di mettere in loop un file audio di breve durata, creando così un sottofondo di lunghezza significativa. In questo caso è possibile usare il formato .wav, seguendo l'approccio già utilizzato per le clip audio di breve durata.

Un'altra possibile estensione, anch'essa non prevista nel gioco reale, è quella di permettere al dinosauro di muoversi a destra e sinistra, oltre che a saltare e abbassarsi soltanto. In questo caso però bisognerebbe cambiare l'intera dinamica del gioco. Nell'applicativo sviluppato infatti, il personaggio principale può saltare e abbassarsi, ma è fisso su una zona dello schermo. Sono i vari nemici, le nuvole

sullo sfondo e il terreno a muoversi verso di lui e non il contrario.

5.Bibliografia

[1] Wikipedia. Dinosaur game — Wikipedia, l'enciclopedia libera, 2022. [Online `https://it.wikipedia.org/wiki/Dinosaur_Game`; controllata il 28-ottobre-2022].

[2] Wikipedia. Dinosaur game — Wikipedia, the free encyclopedia, 2022. [Online `https://en.wikipedia.org/wiki/Dinosaur_Game`; controllata il 28-ottobre-2022].

[3] yUML. Create uml diagrams online in seconds, no special tools needed. [Online `https://yuml.me/`].