ODD
Object Design Document

DASHING CUBE

Riferimento	Nc11_dashingcube-odd.docx
Versione	0.4
Data	08/01/2025
Destinatario	Prof. Carmine Gravino
Presentato da	Vincenzo Beniamino Fresa, Francesco Botta
Approvato da	

Revision History

Data	Versione	Descrizione	Autori
22/12/2024	0.1	Stesura dell'introduzione	Francesco Botta Vincenzo Beniamino Fresa
28/12/2024	0.2	Packages	Francesco Botta
03/01/2025	0.3	Design Pattern	Vincenzo Beniamino Fresa
06/01/2025	0.4	Prima Revisione	Vincenzo Beniamino Fresa

Sommario

1. Introduzione

Dashing Cube si propone come un progetto facile da costruire e intrattenente da giocare per un grande bacino di pubblico.

Lo scopo dell'ODD è quello di presentare lo "scheletro" del progetto, fornendo tutte le funzionalità descritte negli altri documenti cercando di comprendere come sono stati integrati. Ciò porterà all'elenco dei packages che dividono il progetto e i metodi utilizzati all'interno di essi. Verrà poi inserito una riproposizione del class diagram realizzato nel documento RAD.

1.1 Componenti off-the-shelf

Il progetto sfrutta diverse "componenti off-the-shelf", chiamate così perché sono prodotti offerti dal mercato per poter essere utilizzati da chiunque nella creazione di progetti del genere.

Il prodotto più impattante è Unity: esso è un engine di gioco utile a evitare di scrivere direttamente sul codice il lato grafico del progetto e alcune funzionalità d'interfaccia come l'interazione di pulsanti.

Vengono inoltre utilizzate alcuni prodotti presenti nello "Unity Asset Store" (UAS) per diverse funzionalità. L'Asset Store può infatti fornire un'enormità di prodotti acquistabili in grado di semplificare alcuni passaggi della costruzione del progetto.

1.2 Linee Guida per la documentazione del progetto

Di seguito è elencata la lista delle linee guida e delle convenzioni utilizzate all'interno dell'intero progetto per una corretta implementazione degli strumenti utilizzati:

Unity:
https://docs.unity.com/
C#:
https://learn.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/
Unity Scripting API:
https://docs.unity3d.com/6000.0/Documentation/ScriptReference/index.html

1.3 Definizioni, Acronimi e Abbreviazioni

- **Package:** raggruppamento di scripts, oggetti e assets utili al progetto.
- **Design pattern:** template di soluzioni applicati a problemi ricorrenti, utili per avere riutilizzo ed espandibilità del sistema.
- **UAS:** Unity Assets Store, negozio in cui è possibile includere differenti oggetti di disparate funzionalità direttamente nel proprio progetto Unity.

2 Packages

In questa sezione viene mostrata la divisione del sistema in differenti packages. Il modo in cui è stato diviso riflette la struttura Three Layer definita nell'SDD:

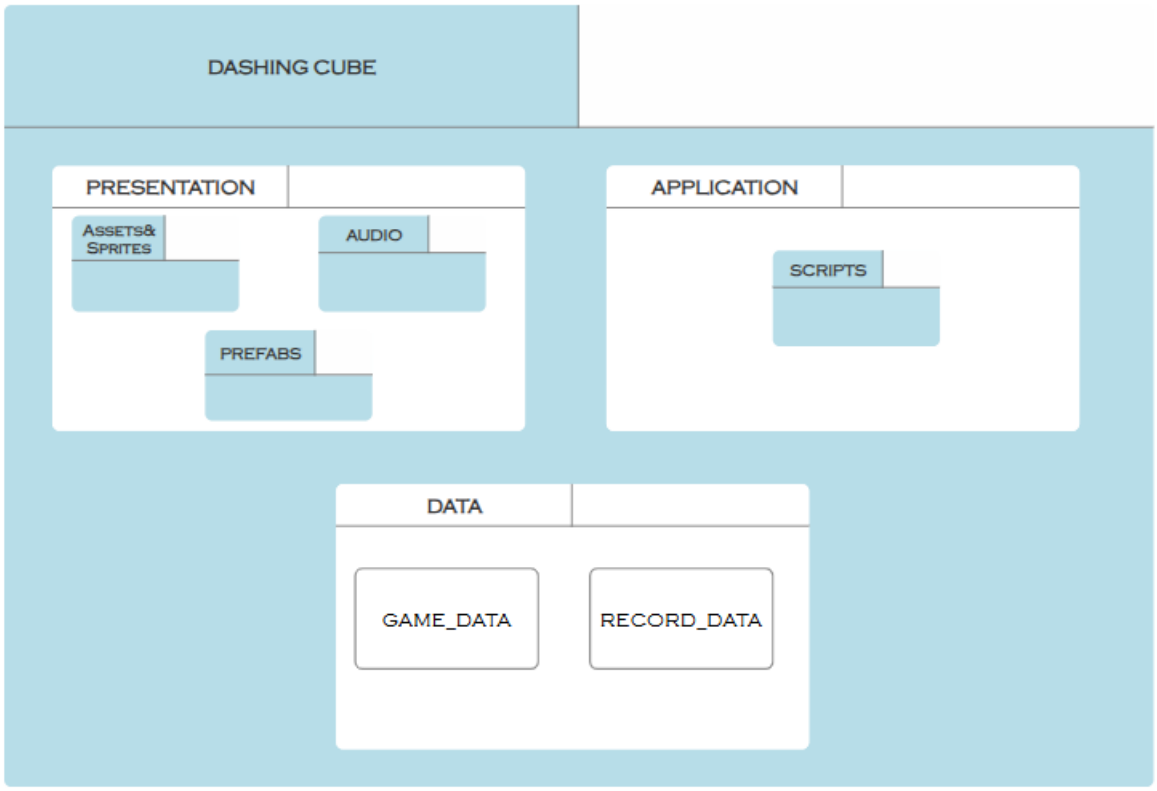
- **System:** Cartella madre che contiene tutte le altre cartelle.
  LAYER PRESENTATION
  - **Assets&Sprites:** Cartella che contiene gli assets presi dall'UAS e gli sprites che costituiranno il lato grafico del gioco.
    - **Texts:** Cartella che conterrà tutto ciò che riguarda i font e la creazione di testi nell'UI.
      - **Fonts:** I font che vengono utilizzati nel sistema.
      - **TextMesh Pro:** Asset utile alla creazione di testi.
    - **UI:** Cartella che contiene tutti gli oggetti che formano il lato grafico del sistema.
      - **Background:** Cartella che contiene i background utilizzati dal sistema.
  - **Audio:** Cartella che contiene tutto ciò che concerne l'audio del gioco.
    - **Music:** gestione delle musiche presenti.
    - **Sound Effects:** gestione degli effetti sonori presenti.
  - **Prefabs:** Cartella che contiene la lista di Prefabs utilizzati.

  LAYER APPLICATION
  - **Scripts:** Cartella che contiene tutti i frammenti di codice utili al sistema.
    - **AudioScripts:** Cartella che contiene il codice utile all'audio.
    - **FileScripts:** Cartella che contiene il codice utile ai file di memorizzazione.
    - **SessionsScripts:** Cartella che contiene il codice utile ad una sessione di gioco, intesa come livello di gioco.

  LAYER DATA
  - **Dashing_Cube_Data:** Cartella che contiene i 2 file di memorizzazione del gioco utili alla ripresa di una nuova sessione.

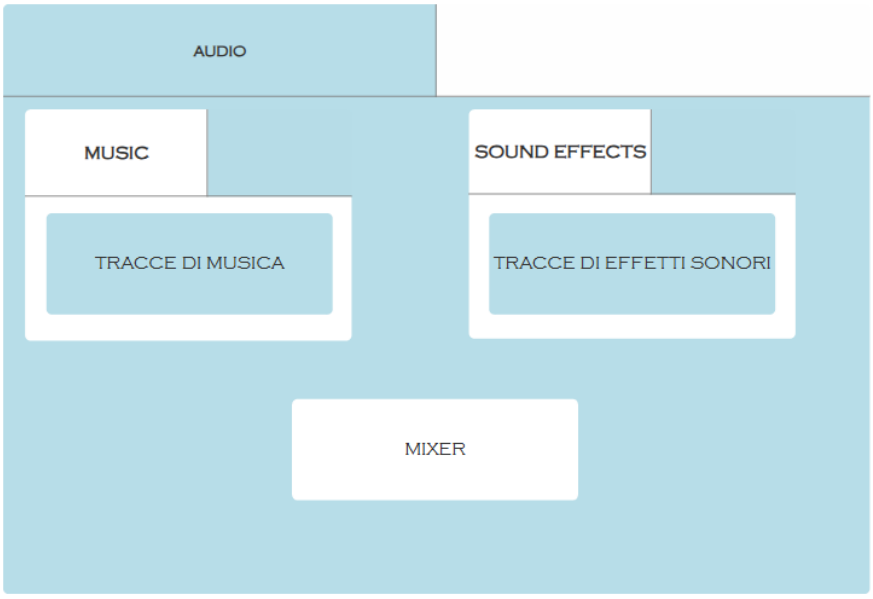
- **Unity_Packages:** Cartella che contiene core assets e frammenti di codice che Unity fornisce alla creazione di un progetto.

2.1 Diagramma di Package UML

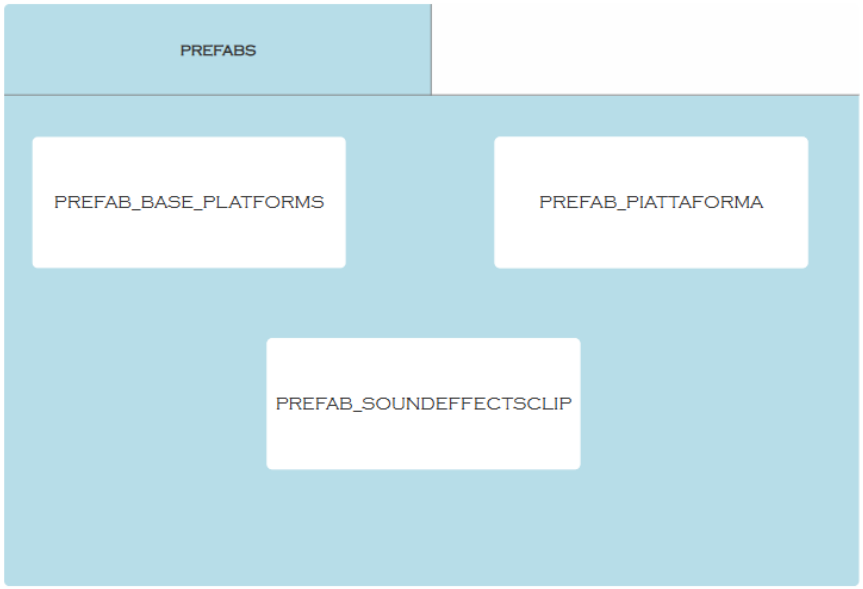


PACKAGE ASSETS & SPRITES

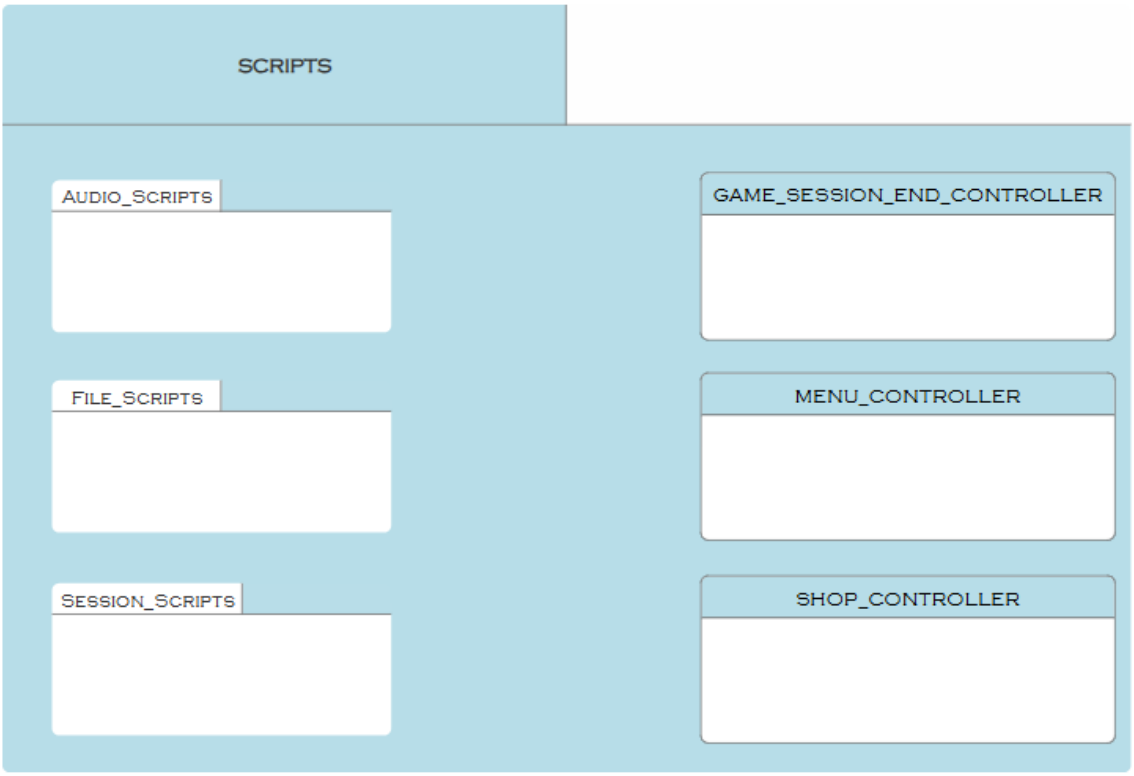
PACKAGE AUDIO



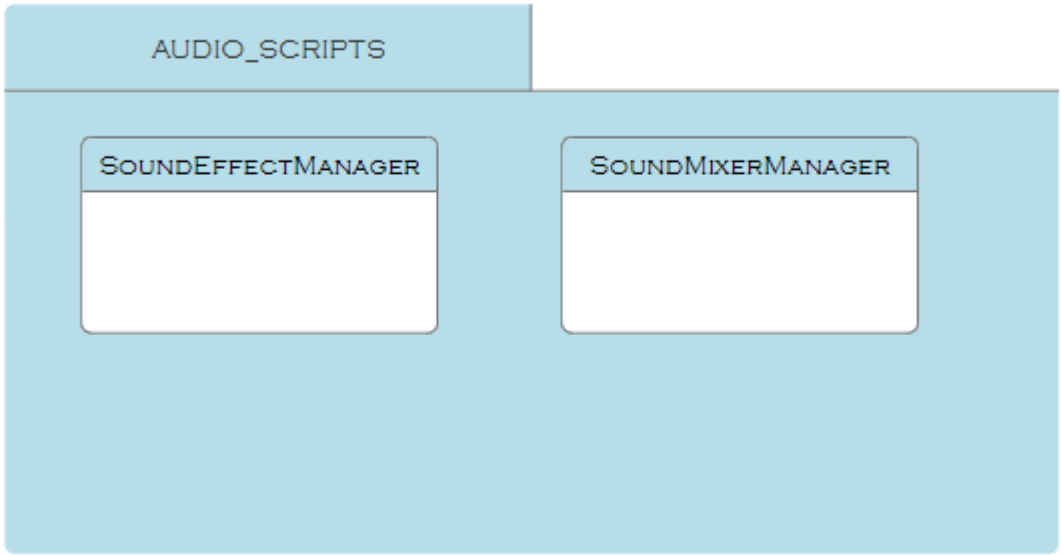
PACKAGE PREFABS


PREFABS
PREFAB_BASE_PLATFORMS
PREFAB_PIATTAFORMA
PREFAB_SOUNDEFFECTSCLIP

PACKAGE SCRIPTS

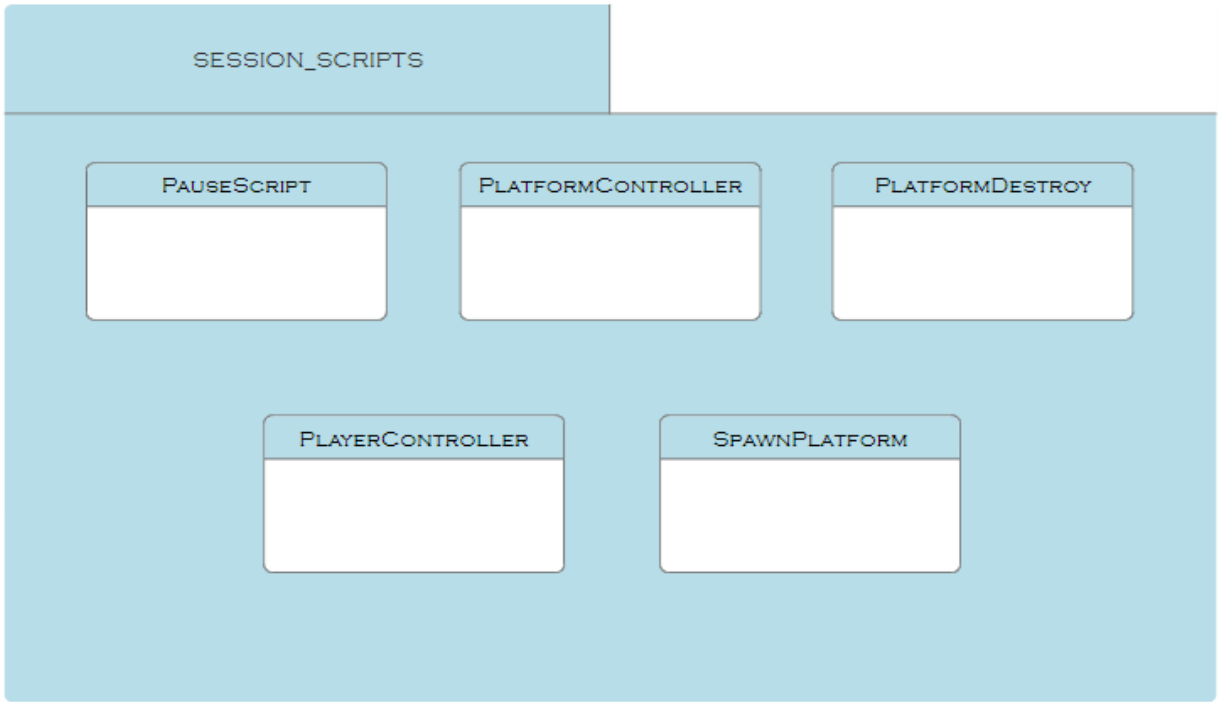


PACKAGE AUDIO_SCRIPTS



PACKAGE FILE_SCRIPTS

PACKAGE SESSIONS_SCRIPT



3 Design Pattern

In questa sezione verranno esaminate le due componenti di design pattern utilizzate per la costruzione di questo progetto: **Observer** e **Singleton**.

Observer

Dato che all'interno del progetto avevamo bisogno di un elemento che controllasse la schermata attuale in cui il giocatore è presente, abbiamo scelto di utilizzare un Observer, tramite quest'ultimo infatti, è possibile navigare tramite i vari menu del gioco, attivando e disattivando le varie scene del gioco a seconda dei pulsanti che vengono premuti.

Come precedentemente accennato, l'Observer ha il compito di osservare quale flag è attivo e comunicare ad altri script quest'ultimo per mostrare all'utente la scena corretta, ad esempio se il giocatore cliccherà sul pulsante delle impostazioni, si attiverà il metodo goToSettings() che disabiliterà il menu principale e mostrerà la schermata delle impostazioni.

Singleton

Nello sviluppo del prodotto abbiamo avuto necessità di implementare effetti sonori, visto che dobbiamo accedere molto spesso a questa risorsa, abbiamo deciso di dare ad un Singleton la gestione di questi ultimi, grazie ad esso l'effetto sonoro è facilmente accessibile, salvaguardia il carico di memoria evitando a UNITY il lavoro di andare a cercare per tutto il programma i metodi di esso, e inoltre permette di memorizzare il cambio di volume che l'utente potrebbe effettuare, dato che memorizza tutti i cambi effettuati ad una qualunque istanza dell'effetto sonoro.