**Capolavoro Anno 2023-2024**

**la relazione presuppone la conoscenza dell’ Unity engine e che venga tenuto d’occhio il proggetto del gioco durante la lettura.**

|  |  |
| --- | --- |
| **Titolo:** | Breakout Chaotic Edition. |
| **Github:** | https://github.com/galactic1106/Breakout-Chaotic-Edition |
| **Unity Play:** | https://play.unity.com/mg/other/webgl-builds-386067 |
| **Breve Descrizione:** | replica del gioco breakout con un alterazione ad una meccanica del gioco. |
| **Origine Titolo:** | Il titolo deriva dalla nuova meccannica inserita all’interno del gioco erroneamente. La pallina che normalmente rimbalza e si muove in modo simile ad una reale adesso ha un movimento caotico che risulta in situazioni assurde ed esileranti insieme ad una maggior difficoltà del gioco. |
| **La nuova meccanica:** | La pallina non viene afflitta dalla gravità, subisce le force d’attrito, ha un coefficente di rimbalzo del 100% cioè se la dovessi lasciar cadere sul pavimento continuerebbe a rimbalzare all’infinito ed ha velocità costante nella direzione in avanti relativa alla pallina.  In particolare la velocità costante e le forze di attrito causano un movimento caotico; infatti la pallina pre i primi istanti della partita andra dritta in una Direzione casuale verso la racchetta; al contatto con la racchetta essa comincerà a ruotare e con essa anche le sue direzioni relative; da questo momento in poi la pallina avrà traiettorie che disegnano archi, spirali, linee quasi rette e tutto questo in sia in verticale che orizzontale a causa della mancanza di gravità.  Queste traiettorie possono sembrare casuali ma non lo sono, anzi con un po’ di pratica sono prevedibili anche se farlo perfettamente e costantemente è pressochè impossibile. |

|  |  |
| --- | --- |
| **Descrizione asset** | |
| **Materials:** | Questa cartella contiene tutti i materialia li usati per dare colori ai GameObjects; il materiale RGB è un materiale usato come indicazione in quanto tutti gli oggetti che lo possiedono saranno arcobaleno; nella sottocartella BlockColors sono presenti tutti i colori assegnati ai blocchi da distruggere. |
| **Physics**  **Materials:** | In questa cartella sono contenuti tutti materiali che danno delle proprietà fisiche ai oggetti in questo caso è presente solo quello della pallina ma ber convenzione ed organizzazione ho preferito aggiungere una cartella dedicata. |
| **Prefabs:** | Questa è la cartella dedicata ai prefabbricati cioè oggetti che potranno essere istanziati più volte e saranno tutti uguali in aggiunta un cambiamento ad uno di essi può affliggere tutti gli altri. Qui troviamo lo sfondo con le pareti, la palla, la racchetta e i blocchi. |
| **Scenes:** | La cartella dedicata alle scene nel mio caso non è di assolutà priorità in quanto le scene sono strumenti utili solo a proggetti più grandi e complessi ma io necessito solo di una scena; quella in cui si trova il gioco. |
| **Scripts:** | Questa cartella contiene tutti i coice creati dagli sviluppatori per gestire tutto quello che non viene gestito dal motore di gioco che sia istanziare un oggetto, cambiarne il colore, gestire il modo in cui si muove o qualunque altra cosa. Scenderò nei dettagli nella sezione dedicata ad essi. |
| **Sprites:** | La cartella degli sprite è quella che contiene gli sprite cioè modelli a strati 2d. Ne uso solo uno quello delle frecce che nelle prime versioni si trovavano al bordo dello schermo ma adesso occumano metà dello schermo ciascuna e sono invisibili. |

|  |  |
| --- | --- |
| **Scripts** | |
| **Ball:** | Questo script viene aggiunto all’oggetto rappresentante la palla.  Variabili: nell’attributo speed viene memorizzata la velocità, in rb viene memorizzato il Rigidbody2D della palla, in lose viene memorizzata la scritta che compare in caso di sconfitta e in reloadButton il tasto per ricominciare il gioco a fine partita.  Start: la funzione rnadomizza la direzione iniziale della palla in modo tale da farla rimbalzare sulla racchetta. Poi recupera dell’oggetto della palla il componente RigidBody2D e cerca nella scena l’oggetto con il tag Lose e quello con il tag Reset.  Update: la funzone semplicemente applica una forza relativa di tipo impulso nella direzione up.  OnCollisionEnter2D: nel momento della collisione con uno switch viene determinata l’azione da svolgere nel caso che la collisione avvenga con un oggetto con il tag Block allora esso verrà distrutto, se la ccollisione avviene con un oggetto con il tag Ground allora verranno attivati gli oggetti lose e reloadButton e la pallina verrà distrutta. |
| **Button Functions:** | Questo script contiene tutte le funzioni e attributi da associare agli eventi di un bottone (button up/down...); esso è stato pensato per dare una buona espandibilità al gioco anche se molto piccolo.  Reload: la funzione fa il get della scena corrente per poi ricaricarla |
| **Check Win:** | Questo script viene aggiunto al GameObject che contiene come figli i blocchi da distruggere per vincere.  Variabili: win è la scritta che compare nel momento in cui si vince.  Start: al primo istante viene trovato l’oggetto con il tag Win e viene disattivato  Update: controlla quanti oggetti figli ci sono al interno del contenitore dei blocchi se il loro numero è zero distrugge la palla e attiva l’oggetto che contiene la scritta vittoria. |
| **Movement Functions:** | Variabili: speed rappresenta la velocità con cui muovere il Rigidbody2D che viene chimato rb; leftPressed e rightPressed sono due variabili booleane che indicano se il bottone(non i tasti della tastiera ma i button di Unity) è nello stato basso.  Start: viene cercato e assegnato il Rigidbody2D della racchetta identificata con il tag Paddle.  Update: se uno dei tasti per muovere la racchetta a destra o a sinistra è premuto muovi la racchetta.  moveLeft/moveRight: applicano delle forze rispettivamente verso sinistra e verso destra al Rigidbody2D  onAttackLeft/onAttackRight: nel momento in cui il bottone unity cambia stato inverti lo stato delle variabili booleane che indicano se è nello stato basso o alto. |
| **Place Blocks:** | Lo script si occupa di posizionare i blocchi all’inzio del gioco.  Variabili: xStart/yStart indicano il la posizione iniziale istanziare i blocchi. xEnd/yEnd indicano la posizione del blocco finale. xJump/yJump indicano offset tra un blocco e l’altro. blockPrefab è il prefabbricato del blocco. materials è l’array che contiene tutti i materiali con cui colorare i blocchi. randomColors è un booleano che indica se assegnare i materiali in ordine randomico. line è un intero il cui ruolo è fare da indice per l’array dei materiali.  Start: con una coppia di cicli for simili a quelli unsati per iterare una matrice utilizza le variabili menzionate prima per piazzare i blocchi su più rige; se randomColors è settato a true allora i colori saranno randomizzati altrimenti ogni riga avra un colore e il loro ordine dipende da quello dei colori nell’array materials. |
| **RGB:** | Si occupa di rendere arcobale gli oggetti che si vogliono avere arcobaleno.  Variabili: gos è un array che contiene tutti i GameObject il cui colore sarà arcobaleno; lastColorChange è una variabile che contiene l’orario dell’ultimo aggiornnamento del colore; colorChageSpeed è un valore che indica il tempo tra un aggiornamento del colore e l’altro; compressedColor contiene i valore RGB che normalmente sono una terna di valori tra 0 e 255 compressa in una terna di valori tra 0 e 1;  Update: se abbastanza tempo è passato aggiorna il colore e lo comprime per poi assegnarlo ai componenti che lo contengono.  RGBLoop: questa funzione è un sistema a stati finiti che si occupa di far ciclare il colore. Ha 5 variabili dedicate; r/g/b sono i valori RGB che verranno poi compressi; offset è il salto ad ogni iterazione del codice; stato è un variabile che può assumere solo alcuni valori dichiarati nel insieme chiamato Stati.  Compress: ha come argomento un intero di valore tra 0 e 255 che verra trasformato in un decimale di valore corrispondente tra 0 e 1. |