
OPEN CONSOLE
N°5 Novembre/Dicembre - 2011
BENNU
IMPARIAMO A PROGRAMMARE UN VIDEOGIOCO
PARTE 2
CANOO: BEST OF - PARTE 2
OC
Benny
Game Development
OpenConsole.it
LA PRIMA WEBZINE ITALIANA SUL MONDO DELLE CONSOLE OPEN

Indice:

INTRO

di Zip

Tempi duri per la scena open,
con l'ormai annunciata uscita di scena della GPH, l'unica fonte di speranza sembra essere il progetto OpenPandora, anche se con l'avvento dei nuovi tablet lowcost anche esso è sul ciglio del baratro. Nessuna grossa novità risolleva la fetta di utenti Dingoo a parte l'annunciato progetto Opendingux, che sarebbe una continuazione di Dingux, che soffre di molte incompatibilità con i software precedentemente rilasciati.
Ma ad un tratto dalla Cina spunta Ygd-18 una nuova consolina open che promette bene, ma ne parleremo in un prossimo numero.

In questa edizione continueremo con lo splendido tutorial del nostro RZZ su come programmare un videogioco in Bennu, e poi tratteremo dei migliori giochi in circolazione per Caanoo nella seconda parte del nostro best of.

OpenConsole webzine
&
OpenConsole.it community
vi augurano buone feste
ricordandovi che noi resteremo sempre a disposizione degli utenti Open italiani che hanno bisogno di un punto di riferimento su questo fantastico mondo.

WWW.OPENCONSOLE.IT

Adesso è possibile leggere i numeri della nostra webzine anche online all'indirizzo:
www.issuu.com/openconsole

Per questo numero hanno collaborato:

Ally copertina

Kayuz recensioni

Kensirou Funzone

Roberto_Ranieri recensioni

RZZ recensioni; tutorial di programmazione

Zip recensioni; impaginazione; coordinamento

Bennu GD

www.bennugd.org

Di RZZ

Impariamo programmare un videogioco 2D attraverso un esempio pratico - PARTE 2

Nello scorso numero abbiamo creato uno sfondo animato e la navicella con relativo controllo dei movimenti, ora è tempo di dargli il dono dello sparo:

Iniziamo aggiungendo queste variabili globali:

```
int shot_time = 210;                    // tempo tra uno sparo e il successivo
int lastShoot  = 210;
int velocita_sparo_normale = 85;        // velocità a cui si muove il proiettile
int Shoot_power= 2;                     // energia tolta dal proiettile
```

Al processo *Ship()*, all'interno del ciclo loop, aggiungiamo:

```
if((jkeys_state[_JKEY_A]))                  // se premo il tasto A...
   if(get_timer()-lastShoot>tempo_sparo)
      Shoot();                              // ...esegue il processo Shoot che è il
                                               proiettile vero e proprio
       lastShoot=get_timer();
   end
elseif(!(jkeys_state[_JKEY_A]))
   lastShoot=-tempo_sparo;
else
end
```

Quando viene premuto il tasto A, prima di creare il proiettile, viene fatto un controllo su quanto tempo è trascorso dall'ultimo sparo, se il tempo, espresso in millisecondi è superiore al valore della variabile shot_time, allora lo sparo viene creato, altrimenti no, Questo ovviamente serve ad evitare che venga a crearsi un flusso continuo e incontrollato di proiettili.

Passiamo al proiettile e creiamo un nuovo processo:

```
process Shoot ();
begin

        resolution=res;
        file=common;                        // file da cui prelevare lo sprite
        region=1;                           // dove visualizzarlo
        graph=6;                            //  numero dello sprite

        x=ship_id.x+170;                    // viene creato a 170 pixel a destra del
        y=ship_id.y-10;                        centro della nave e 10 verso il basso

        fiamma_sparo_normale(x,y);          // viene creato l'effetto fiamma dello sparo
        repeat
              x+=velocita_sparo_normale;    // velocità a cui si muove il proiettile
              frame;
        until(out_region(id,region));       // se il proiettile esce dalle schermo
```

```
        signal(id,s_kill);                    viene eliminato
end


process fiamma_sparo_normale (int a, int b);
  begin
       resolution=res;
       file=common;
       region=1;
       graph=rand(99,101);
       x=a+58;
       y=b;
       frame(200);
       signal(id,s_kill);
end
```

Per dare una miglior sensazione di potenza di sparo aggiungiamo una fiamma in prossimità del punto in cui il proiettile esce dalla nave:

```
process fiamma_sparo_normale (int a, int b );

begin
resolution=res;
file=common;
region=1;
graph=rand(99,101);
x=a+58;
y=b;
frame(200);
signal(id,s_kill);
end
```

Passiamo ad un processo fondamentale: la struttura del livello, che si occupa di creare i nemici e i fondali (oltre a quello scorrevole creato nella puntata precedente)

```
process Level()
begin

frame(2000);
       text[2]=write(0,160,110,4,"MISSION 1");   // facciamo apparire delle scritte
       frame(7000);                                 che indicano il numero del livello
       delete_text(text[2]);
       text[2]=write(0,160,140,4,"START !");
       frame(3000);
       delete_text(text[2]);
       frame(1000);
       text[2]=write(0,160,140,4,"START !");
       frame(3000);
       delete_text(text[2]);

              frame(5000);
              Nemico(3250,1200,0);              // viene creato il nemico tipo "0"
              frame(7000);                         in posizione 325 x 120, ovvero a metà
              Nemico(3250,800,0);                schermo, leggermente fuori dall' area visibile
              frame(7000);
              Nemico(3250,1700,0);
              frame(5000);
       end
```

Questo processo è semplificato al massimo, in quanto fa apparire solo tre nemici uguali e poi termina. In un gioco completo i nemici saranno molti di più ed inoltre, una volta terminato il livello 1 si passa al successivo, ma quanto presente in questo esempio è più che sufficiente per capirne la meccanica.

Vediamo cosa fa il processo Nemico(x, y n)

```
process Nemico(int xpos,int ypos,int numero);
       private
       int timeShot=0;
       int ang=180000;
       int energia;
       int timeanim;
       int animazione=50;                          // sprite di partenza del nemico 0

       begin

             if (numero == 0)                      // serve a scegliere il tipo del nemico
                   energia = 5;                    // assegna l'energia relativa
                   else energia = 2;
             end

       timeShot=get_timer();
       resolution=res;
       flags=0;
       region=1;
       x=xpos;
       y=ypos;
       file=lv1;

       loop

         if(collision(ship_id))                    // se il nemico tocca la nostra nave
           if(get_timer()-respawn>tempo_respawn)   // ed è passato sufficiente tempo dal respawn
             Explode();                            // la nave esplode...
             Death();                              // ...e anche il nemico
           end
         end

         if(collision(type Shoot))                 // se il nemico collide con il nostro proiettile
              energia -= Shoot_power;              // la sua energia viene decurtata di un valore
         end                                       // pari alla potenza del proiettile

              if (energia <= 0)                    // se l'energia del nemico raggiunge lo 0
                   score+=5;                       // aumenta il nostro punteggio
                   Death();                        // e il nemico esplode
              end

        if(numero==0)                              // caso nemico numero 0 (l'unico per ora...)
           if (get_timer()-timeanim>80)            // creiamo l'animazione: ogni 80 ms lo
             graph=animazione;                     // sprite del nemico viene cambiato, partendo
```

```
            animazione+=1;                          dal numero 50 fino al 56, poi ricomincia
            if (animazione>56)
               animazione=50;
            end
            timeanim=get_timer();
          end                                       // fine animazione

          if (x>2800)                               // controllo pattern di movimento
              xadvance(180000,9);                   // da 325 pixel a 280 si muove con angolo di
          end                                           180° e velocità 9

          if ((x<=2800)& (x>=1900))                 // da 279 pixel a 190 si muove con angolo di
              xadvance(200000,11);                      200° e velocità 11
          end


          if (x<1900)                               // da 189 pixel a 0 si muove con angolo di
              xadvance(180000,9);                       180° e velocità 9
          end                                       // fine pattern di movimento

           if ((get_timer()-timeShot>1500)&         // gestione sparo
              (x>nonsparare)&(numerospari<2))       // ogni 1500ms se le condizioni sono rispettate
              sparo_nemico(548, 549, 12);               il nemico spara
              timeShot=get_timer();                 // il contatore tempo-sparo viene azzerato
           end
        end                                         // fine della gestione del nemico numero 0


            if(out_region(id,region))               // se il nemico esce dallo schermo
                signal(id,s_kill);                      viene distrutto
            end

            frame;                                  // aggiorna il video
      end                                           // fine loop
end
```

Il nemico spara ogni 1500 millisecondi se e solo se ci sono meno di due proiettili nemici su schermo, e se si trova a più di "nonsparare" pixel (che è stato settato come variabile globale, a 40 pixel) dal bordo sinistro dello schermo. Queste sono due condizioni fondamentali per migliorare la giocabilità, perché senza la prima lo schermo potrebbe riempirsi di una quantità esagerata di proiettili e senza la seconda, il nemico potrebbe sparare troppo vicino alla nostra nave o addirittura quando è al 99% fuori dall'area visibile, rendendo il proiettile troppo difficile da evitare.
Da notare che in: `if(out_region(id,region)) signal(id,s_kill);`
non viene distrutto il nemico in quanto sprite, ma viene eliminato il suo processo, cosa molto importante per liberare memoria e processore.

Diamo un occhiata al processo dello sparo nemico. Quello che vogliamo creare è un oggetto che al momento della sua creazione rileva la posizione della nostra nave e si muove nella sua direzione:

```
process sparo_nemico(int gr, int gr2, int sp)      // come parametri ha: sprite iniziale,
                                                      sprite finale e velocità a cui muoversi.
private
int timeanim = 0;
int animazione;

begin
numerospari+=1;                                    // aggiorna  il numero di proiettili a schermo
animazione = gr;
        resolution=res;
        file=lv1;
        graph=gr;
        region=1;                                  // x e y vengono presi in riferimento alla
        x=father.x;                                   posizione della nave che lo ha generato
        y=father.y;
        angle=fget_angle(ship_id.x,ship_id.y,      // angolo con il quale si muoverà:
                         father.x,father.y);          determinato dalla posizione del nemico e
                                                      della nostra nave
        repeat

          if (get_timer()-timeanim>100)            // gestione animazione
              graph=animazione;
              animazione+=1;
                    if (animazione>gr2)
                       animazione=gr;
                    end
                    timeanim=get_timer();
           end
           advance(-sp);


           if(collision(ship_id))
              if(get_timer()
                        - tempo_respawn>respawn)
               Explode();
                x=10000;
              end
           end
           frame;
        until(out_region(id,region));              // fine ciclo repeat

        numerospari-=1;                            // aggiornato il numero spari a schermo

        signal(id,s_kill);

end
```

A questo punto rimangono solo due processi che ormai saprete capire benissimo, in quanto non fanno altro che visualizzare gli sprite relativi all'esplosione della nostra nave e del nemico:

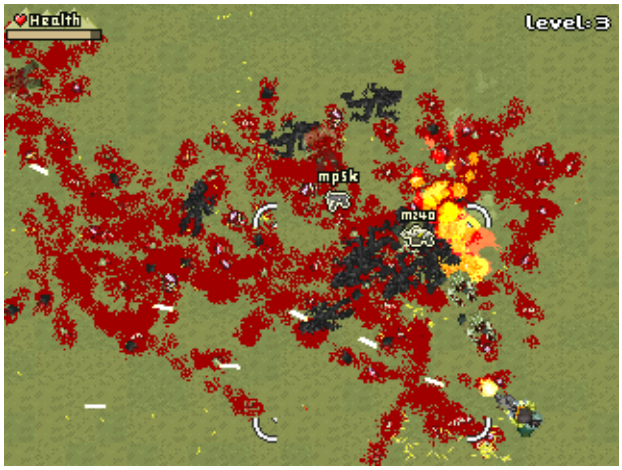
```
process Explode ( );                // Esplosione della nostra nave
begin
       resolution=res;
       file=common;
       region=1;
       graph=933;
       x=ship_id.x;
       y=ship_id.y;
                                    //  mette in pausa il processo della nave in modo
       signal(ship_id,s_sleep);         che non risponda ai comandi
       tempo_respawn=get_timer();
       frame(210);
       repeat
         graph+=1;                  // visualizza l'esplosione
         frame(210);
       until(graph==962)
       lives-=1;                    // diminuisce il numero di vite
       frame(3000);
       signal(ship_id,s_wakeup);    // riattiva la nave...
       ship_id.x=400;               // ...e la riposiziona
       ship_id.y=1200;
end


process Death ( );                  // esplosione del nemico
begin

        resolution=res;
        file=common;
        region=1;
        graph=450;

        x=father.x;                 // viene creata in base alla posizione
        y=father.y;                    del nemico che ha chiamato il processo

        signal(father,s_sleep);
        repeat
        graph+=1;
        frame(200);
        until(graph>458)
        father.x=-10000;
        signal(father,s_kill);
        frame;
        signal(id,s_kill);
 end
```

Il gioco è lungi dall'essere finito, ma la struttura portante è stata costruita. Se volete cimentarvi a continuare potete procedere con l'aggiungere i suoni, i menù, i boss e tutto quello che fa da contorno, come le armi secondarie la gestione dei proiettili, bonus ecc..
Buon lavoro !

Caanoo: Best of - parte 2

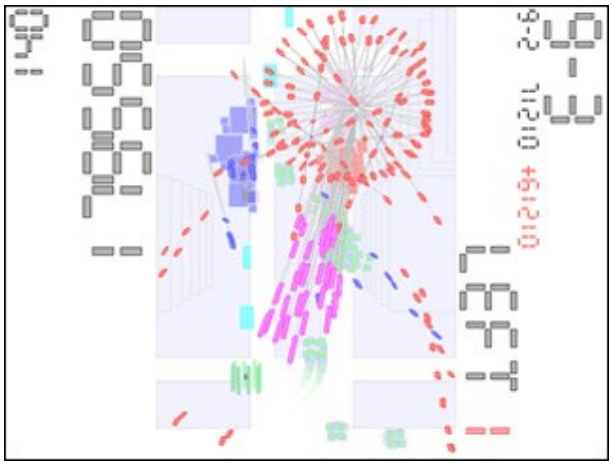
Raccolta dei titoli considerati migliori a nostro insindacabile giudizio...
...e se non siete d'accordo venite a discuterne su www.openconsole.it/forum/

Zombies On My Grass



Sebbene il titolo sia piuttosto ripetitivo, lo considero tra i migliori disponibili per Caanoo grazie all'ottimo uso che fa del touch screen. Con la croce direzionale muoveremo il nostro personaggio nello schermo, mentre toccando il display punteremo l'arma nella relativa direzione e faremo fuoco verso l'orda di zombies sempre più numerosa che ci attaccherà a 360°. E' davvero un piacere notare la cura per certi dettagli, ad esempio gli zombies uccisi con il lanciafiamme, al momento della loro (seconda) dipartita, lasceranno sul campo un corpo carbonizzato, mentre quelli colpiti da armi da fuoco una bella chiazza insanguinata. Per sua natura il gioco non prevede molte animazioni, ma le poche sono comunque di buona fattura. Musiche tamarre quanto basta ed effetti sonori tetri al punto giusto e armi in quantità rendono il titolo un vero piacere da giocare.
RZZ

Noiz2sa



Shoot'em up 2D a scorrimento verticale con grafica minimal-futuristica che ricorda i menù del mitico Wipeout. Il gioco non basa la struttura su di un level design complesso, ma sulla pura abilità del giocatore di schivare la moltitudine di proiettili che invade lo schermo, pressoché in ogni frangente.
L'essenzialità della grafica e lo stile di gioco istintivo e senza fronzoli, lo rendono un vero e proprio moderno retro-game, che merita un posto d'onore tra i titoli delle console open !
RZZ

Audiorace

È dura correre da soli..

Eccoci a parlare subito del gameplay, croce e delizia del gioco.

Per spiegarlo non ci vuole un professore, la vostra unica missione è quella di comandare una spazionave in un circuito infinito cercando di evitare gli ostacoli che ci faranno perdere punteggio, che invece andremo ad incrementare raccogliendo le "lattine" presenti nel circuito.

Ci sono tre sole corsie, che potremo agilmente cambiare; in queste corsie apparirannoi casualmente ostacoli e punti, dando quindi un certo livello di sfida.

Peccato che parlare di sfida pare davvero assurdo visto che, (se uno volesse) potrebbe giocarci all'infinito... sentendo la propria musica preferita!

È questa la vera novità carina di AudioRace... ovvero, prima di iniziare la nostra infinita corsa tra ostacoli e lattine potremo selezionare le nostre musiche preferite, dando quindi un senso al gioco: quello di ascoltare la musica avendo però qualche cosa da fare; un gioco di corse per l'appunto.

L'unica vera gemma di AudioRace è il suo engine; lo sviluppatore è infatti più un tecnico e quindi i suoi giochi sono più che altro delle prove tecniche; bisogna lodare Crow-riot per questa grafica, davvero fluida e dettagliata, anche se uno sfondo sarebbe stato il massimo.
Kayuz

Rhytmos

Il Buono, il Brutto, il Cattivo

Oggi, sono qui dopo ore passate giocando a Rhythmos per raccontarvi un po' com'è, nella speranza di essere più obbiettivo e utile possibile. Rhythmos è un gioco molto strano, se così lo possiamo definire e sarebbe stato difficile fare una recensione da quella che vi sto per proporre. Difatti, quello che state per leggere è una schematizzazione con molti pareri del recensore di quello che è il gioco.

I Pro:

-Selezione difficoltà davvero su misura: nel menu delle opzioni (cambiando questo valore non ho avuto nemmeno la percezione di cambiare la difficoltà, ad essere sinceri), selezione velocità delle note, numero di bottoni da premere, margine di "pignoleria" sulle note (in modalità easy, ad esempio è facile fare molte note in "perfect" e "Rhythmos"

-Due modalità di gioco, Touch Style e Button Style

-Canzoni (seppur coreane) molto belle

-Effetti Grafici Mozzafiato (ogni canzone ha un proprio video di sfondo)

-Artwork molto carini e in generale la grafica ben curata.

-Gameplay solido e divertente

-Possibilità di vedere i punteggi di altri giocatori online

Contro:

-Tv Out non supportato

-Eccetto la prima canzone, non si riescono ad ottenere i risultati (e quindi avanzare nel gioco) se si sta usando la memory card fornita con la Caanoo (e probabilmente anche altre memory card di fascia bassa)

-Caricamenti lunghi su una memory card di fascia bassa

-Per accedere ai punteggi online è necessario un account fungp (credo) ma non c'è lo spazio nella tastiera virtuale.

-Non è espandibile con nessuna canzone (il massimo è 10 canzoni)

-DRM restrittivo che non ci permette di eseguirlo su un'altra Caanoo

Bisogna considerare questo gioco (uno dei pochi giochi commerciali disponibili) un must have, essendo davvero curato nei minimi dettagli. Un gameplay solido e divertente accompagnerà le svariate ore di gioco offerte da questo titolo.

A questo punto vorrei analizzare questo gioco in 2 modi: come casual game e gioco di musica. Ci si aspetta, infatti, da un casual game di essere semplice, divertente e breve. Da un gioco di musica, invece, ci si aspetta un parco canzoni allucinante per poter suonare, volendo, anche le ultime hit del momento con accessori belli e, ovviamente, anche costosi.

Roberto_Ranieri

Jump to the moon

Questo gioco riesce a conquistare un posto nella rosa dei migliori titoli per Caanoo , grazie al suo gameplay semplice ed intuitivo.
Impersoneremo un astronauta che dovrà salire sempre più in alto tra le varie piattaforme e l'unico aiuto che avremo a nostra disposizione sarà il "kerosene" che ci servirà come "colpo di reni" nei momenti più difficili. Il titolo sfrutta a pieno la novità introdotta da GPH a suo tempo nella/nel Caanoo ovvero il sensore di movimento, quest'ultimo sarà il nostro controller, tenendo la console in verticale tra le nostre mani sposteremo il nostro astronauta a destra e a sinistra, e usando l'analogico sfrutteremo il kerosene, che potremo ricaricare andando a prendere i vari item stando attenti però ai diversi livelli di propulsione a seconda della piattaforma su cui atterreremo. Grazie a questo tipo di controlli ed al gameplay praticamente identico ai più famosi predecessori vedi: "doodle jump" o "icy tower", jump to the moon si presenta come apripista per i "casual game" su console open. Degna di nota è anche la grafica che si presenta curata per quei pochi sprites che sfrutta il gioco.
Zip

FUN ZONE!

di Kensirou

NAME THE GAME!

Benvenuti al nuovo gioco di Open Console, Name The Game!

Verranno inseriti una parte della copertina originale e uno parte di uno screenshot e a voi lettori spetterà il compito di indovinare i nomi dei due giochi ai quali si riferiscono.

Ecco le due domande di questa prima sessione :

Name The Game 1

Cover

Name The Game 1

Screenshot

Il primo che ci invierà entrambe le risposte esatte vincerà n°20 di O.C. Points e il suo nome verrà pubblicato sulla copertina del prossimo numero di Open Console.

Inviate le vostre risposte per messaggio privato a Zip sul forum!

Buon divertimento e in bocca al lupo a tutti! ^^/

Openjoke

I WANT YOU FOR OpenConsole

Collabora con noi! Inviandoci articoli, suggerimenti recensioni, faq, idee e critiche

WWW.OPENCONSOLE.IT