RETROEDICOLA

3 VIDEOWORLD LUG 2015

IL MENSILE GRATUITO SUL RETROGAMING

Trent'anni di... Zzap!

Programmare un Game & Watch

I segreti di Gyruss

Tutto su Retrocampus!

ASSOCIAZIONE CULTURALE RETRO CAMPUS

#retrogame #retrocomputing #eventi #internet

PROGRAMMARE UN GAME & WATCH

Parte 1 - intro e preparazione

*TUTORIAL
*GAME&WATCH
*PROGRAMMARE

Luca "MADrigal" Antignano

GIOCO ELETTRONICO

Al suono di queste due parole, a molti di noi (specialmente quelli più attempati...) viene subito alla memoria un periodo ben preciso del secolo scorso: i primi anni '80, caratterizzati da sperimentazioni più o meno pionieristiche nella storia del videogame casalingo.

A quel tempo un po' tutte le fabbriche di giocattoli si cimentavano nella corsa all'intrattenimento elettronico portatile, sfruttando perfino semplici lampadine colorate che, combinate a scatolotti elettromeccanici, diventavano talvolta un divertente gioco di Baseball portatile, altre volte un piccolo flipper casalingo.

A partire dalla seconda metà degli anni '70 troviamo prima i giochi elettro-meccanici, poi quelli basati su piccole lucette LED, ed infine lui, il mitico "schiacciapensieri" con schermo a cristalli liquidi, che avrebbe spianato la strada al nascituro GameBoy di Gunpei Yokoi, a tutti i suoi avversari ed alle sue derivazioni. Non a caso gli odierni Nintendo DS ricordano esteticamente proprio i primi modelli di gioco elettronico a due schermi, messi in commercio dalla stessa Nintendo a partire dal 1982.

A questo punto consentitemi di sfatare un vecchio luogo comune.

Forse non tutti sanno (o non ricordano) che il nome "schiacciapensieri", che ancora oggi è sinonimo di gioco elettronico tascabile "tout court", deriva in realtà da una geniale trovata dei responsabili marketing della Polistil (...sì, proprio i produttori delle piste di automobiline!), che nei primi anni '80 distribuiva, con quel nome, i giochi prodotti dalla VTech di Hong Kong.

Il nome "schiacciapensieri" deriva da una storpiatura dello strumento musicale della tradizione siciliana (lo scacciapensieri), qui modificato per dare l'idea di un oggetto che potesse farci dimenticare i problemi e gli impegni della vita quotidiana, in favore di un sano relax grazie ad un passatempo da giocare comodamente anche in giro per la città.

Pertanto, più correttamente parleremo di "gioco elettronico" per individuarlo come oggetto e fenomeno in generale, di "schiacciapensieri" riferendoci ai soli giochi VTech/Polistil, di Game & Watch per quelli targati Nintendo, e così via.

TECNOLOGIA

I primi giochi elettronici dotati di display a lucine (LED o VFD), e quelli più moderni basati su schermetti a cristalli liquidi (LCD) non hanno poi grandi differenze, tranne quella evidente della resa visiva: tutti condividono stessa architettura, stesse CPU, stessa manifattura. La logica di funzionamento dei giochi è assolutamente identica: tutto viene gestito da piccoli (ed economici!) processori a 4-bit, che attivano o disattivano la corrente su un insieme di elementi, talvolta composti da lucine colorate, tal altra volta da uno schermo LCD. Troviamo la medesima "non differenza" tra le primissime calcolatrici, basate su numeri luminescenti, e quelle più moderne con i cristalli liquidi... ma se vogliamo spingerci ancora oltre, troviamo simili analogie anche nei modernissimi telefoni cellulari.

Aprendo uno qualsiasi tra le centinaia di differenti giochi elettronici di quegli anni, troviamo sempre alcuni elementi comuni. Per primo, un minuscolo quarzo che, vibrando, "dà il tempo" alla piccola CPU. Si tratta ovviamente di quello che oggi nei nostri PC di casa viene chiamato "clock" (orologio). Si vede poi sempre una (o talvolta addirittura due) minuscole casse audio, solitamente piattissime e con una membrana di materiale cartaceo.

Ma l'altra cosa che non manca è ovviamente la CPU, cioè il cuore pulsante del gioco, quello che ne contiene la logica. Ad un occhio più attento, facendo un paragone con i nostri computer di

MADrigal, che potremo definire il piccolo Gunpey Yokoi italiano, ha creato anni fa il primo simulatore dei mitici Game & Watch.
Da allora lavora come freelance per molti progetti hardware legati a questa tipologia di videogiochi.

MADrigal Design

casa, viene spontanea la domanda: "ma dove stanno la RAM ed il BIOS? OK il processore lo vedo, lo schermo c'è, l'audio pure, ma... il programma dove risiede?". Ecco, il "bello" degli schiacciapensieri è proprio qui: RAM e ROM ci sono... ma non si vedono!

Per un discorso di ottimizzazione di costi, ridurre l'ingombro sulla schedina del gioco e minimizzare il consumo di elettricità, si è scelto di impiegare un microprocessore a 4-bit che l suo interno contenesse anche la RAM, la logica per la gestione dei tasti ed avesse una certa quantità di memoria scrivibile per memorizzarci il programma del gioco... ovvero la cosiddetta ROM (Read-Only Memory).

Dunque guardando le schedine interne a due Game & Watch scelti a caso ma della stessa serie, possiamo constatare che sono assolutamente identiche, tranne il codice seriale che campeggia sulla CPU: quello individua il programma ROM contenuto all'interno, che è unico e gestisce lo schermo e l'audio in modo apposito per quel gioco.

SIMULAZIONE ED EMULAZIONE

Chi si intende un po' di "emulatori" (per i pochi lettori che non avessero mai sentito questa parola, magari leggendo "MAME" capiranno...) è molto familiare al concetto di ROM. Negli ultimi dieci anni, dopo il boom del fenomeno, per molti utenti ROM è sinonimo di "gioco gratis", con tutti i problemi legati alla legalità del loro possesso.

In realtà l'emulatore in sé non è un prodotto pirata o illegale, in quanto si tratta di un software interamente realizzato da appassionati e distribuito liberamente per la rete. Il problema è che l'emulatore da solo non serve a niente senza i programmi da emulare (le ROM appunto). D'altronde a che servirebbe avere un computer vuoto, senza i programmi installati dentro? Tramite l'ausilio di apposite interfacce, il contenuto delle ROM viene estratto dalle cartucce-gioco delle varie console, e riversato in file binari (detti "ROM files" appunto) destinati poi ad essere gestiti dagli emulatori. Il problema, per quel che riguarda i giochi elettronici, è che i programmi ROM non sono accessibili dall'esterno, essendo "blindati" dentro il chip del microprocessore. L'unico modo in cui i piccoli chip comunicano con l'esterno, sono semplicemente i tasti gommati che gestiscono l'input, le uscite video sullo schermo LCD e ed audio verso la minuscola cassa audio: nessuna interfaccia sarebbe capace di estrarre il contenuto ROM dal chip di un gioco elettronico.

Dunque non essendoci ROM da estrarre, come si fa oggi a realizzare un emulatore di uno schiacciapensieri? Semplice... non si realizza un emulatore, ma un "simulatore". La differenza non è propriamente sottile, anzi si tratta di tecniche di programmazione quasi diametralmente opposte! Da un lato l'emulatore, che si occupa di creare un "ambiente virtuale" in cui eseguire un programma ROM prelevato da un hardware identico a quello che si sta emulando. Dall'altro il simulatore, che non si cura dell'hardware originale, ma solo di ciò che viene da esso generato (il gioco), e cerca di "replicarlo" in modo da somigliarci infinitamente: il programma ROM originale non viene usato, ma viene integralmente "clonato" usando un

moderno linguaggio di programmazione. E' un po' come vedere Pac Man in sala giochi, e decidere di cimentarsi a realizzarne una copia fedelissima, usando Flash o Java installato sul PC di casa.

Il problema è però appunto la fedeltà di questa replica: per quanto io possa impegnarmi a farlo uguale, non verrà mai la stessa cosa fino a quando non sarò in grado di replicare al 100% il comportamento del gioco originale in qualsiasi delle condizioni di funzionamento (compresi gli eventuali bug). Per questa ragione, è necessario conoscere quanti più dettagli possibile sul gioco da clonare, in modo da ridurre al minimo le differenze tra originale e copia. E ovviamente per avere informazioni così dettagliate è consigliabile avere in mano il gioco da replicare, e non accontentarsi magari di avere una semplice descrizione di esso.

IL TUTORIAL: PREMESSE

Se fino a questo momento non siete ancora confusi dai concetti di emulatore, simulatore, ROM, cloni, etc... potreste essere invece incuriositi dal "come" in realtà si possa realizzare un software di questo tipo. D'altronde un po' tutti noi abbiamo come sogno nel cassetto il desiderio di produrre il nostro videogame personale. Questo è stato anche sempre il mio grande sogno, ma prima di arrivare (e ci sono arrivato...) a creare un gioco completamente mio, ho voluto fare un po' di pratica, usando come spunto la realizzazione di alcuni di questi "cloni", spinto dal fascino di queste vecchie scatolette di plastica con schermetto monocromatico. E, potete credermi, è stato tutt'altro che difficile!

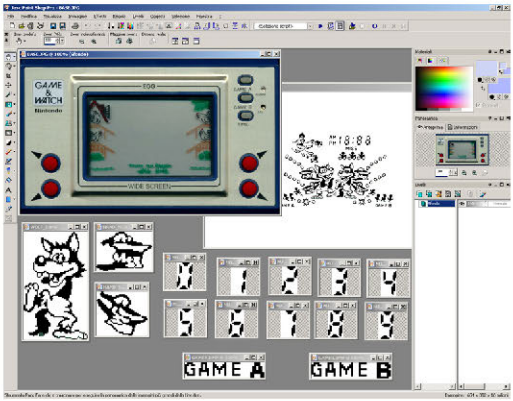
Programmare un simulatore/imitazione di gioco elettronico già esistente ci consente da subito di risolvere tutta una serie di problemi: la grafica è già bella pronta (sarà sufficiente uno scanner o una macchina fotografica per immortalare lo schermo di gioco acceso) e non va nemmeno animata dato che si tratta di oggetti fissi, i suoni sono già inclusi (si registrano con un modestissimo microfono collegato al PC), inoltre c'è l'innegabile vantaggio che gli oggetti non dovranno "muoversi" sullo schermo, ma solo accendersi o spegnersi! Questo ci consente di evitare passaggi complessi come l'impiego di librerie grafiche tipo DirectX ed OpenGL, o tecniche di "double buffering" per fare spostare oggetti sul video senza i fastidiosi effetti di sfarfallio.

Vi illustrerò come si realizza, a grandi linee, un simulatore impiegando uno dei tanti ambienti di sviluppo per Windows. Per semplicità, dato che è quello che uso maggiormente, mi riferirò a Borland Delphi, che potremmo definire una sorta di Visual Basic o C++ Builder però basato sul linguaggio Pascal - che oltretutto è particolarmente didattico.

Per prima cosa di cosa abbiamo bisogno? Sarà sufficiente un qualsiasi PC con almeno 200MHz, Windows 98 o superiore, 32MB di memoria, una scheda audio qualsiasi, un economico microfonino, uno scanner (meglio quelli di tipo vecchio, belli grossi e con una lente e luce al neon) ed ovviamente il gioco elettronico da riprodurre. A livello software ci serviranno il tool di sviluppo (Borland Delphi 4 o superiore), un programma di fotoritocco (Photoshop, Paint Shop Pro, The Gimp, etc.) ed un programma per l'editing audio (Nero Wave Editor, Goldwave, etc.).

LA PREPARAZIONE DELLE IMMAGINI

Procediamo per gradi: la preparazione delle immagini è la primissima cosa da fare. Sarà sufficiente scansionare il gioco due volte: una con lo schermo spento, in modo da ottenere un'immagine completa dello "sfondo". La seconda immagine invece va realizzata avendo tutti i segmenti LCD (gli oggetti) accesi e ben visibili, che poi provvederemo a "ritagliare" e trasformare in oggetti da accendere/spegnere nella nostra applicazione. E' comodo ricordare che nella gran parte dei giochi elettronici esiste una modalità chiamata "ACL" (All CLear) che si attiva per qualche minuto dopo l'inserimento delle batterie, oppure premendo un apposito tastino: ovviamente sfrutteremo questa opzione per realizzare l'immagine con tutti gli oggetti accesi. E' importantissimo, quando si lavora con immagini di questo tipo, salvare i file con formati senza perdita di qualità e colori: dunque scartate



GIF o JPG ed usate solo BMP o PNG a 24-bit. Per la sola immagine che andremo ad usare come sfondo assoluto, potremo salvare in formato JPG con altissima qualità (al massimo 5% di fattore-compressione), in modo da ridurre la quantità in KByte della nostra applicazione - che la conterrà.

Alcune considerazioni importanti vanno fatte sul tipo di schermo che vogliamo scansionare: ad esempio gli schermi LCD sono infatti "orientati" in modo che gli oggetti si vedano solo se la luce proviene dall'alto, con un'angolazione massima di 60°. Sarà dunque nostra cura scansionarlo "a testa in su", in modo che la luce dello scanner arrivi dall'alto al basso dello schermo, altrimenti i cristalli non saranno visibili.

Volendo invece scansionare uno schermo a LED o VFD, la questione si complica: le luci LED accese contrastano con quella dello scanner, rovinando l'esito dell'immagine. A questo punto c'è solo una strada ed è un po' laboriosa: aprire il gioco, scansionare lo schermo spento e colorare manualmente i vari oggetti di cui la scansione ha rilevato pochi dettagli monocromatici.

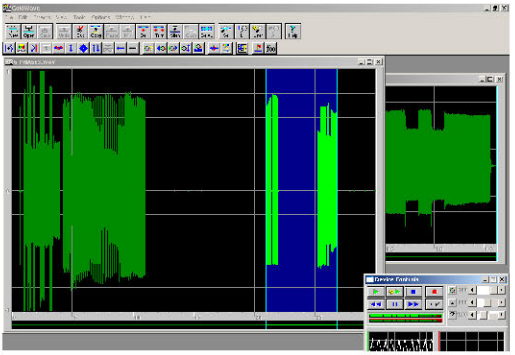
Una volta che le due immagini (lo sfondo e quella con gli oggetti accesi) saranno pronte, non resterà che dimensionarle per la risoluzione che vogliamo raggiungere: per esempio se supponiamo che l'utente medio avrà un desktop a 800x600 pixel, potremo decidere che il nostro simulatore dovrà vedersi in quello schermo, per cui magari potrà essere di 650x500 pixel in modo che dia l'effetto di oggetto "posato" sul desktop.

L'ultima fase della preparazione delle immagini è un po' lunga ma personalmente la trovo molto divertente: il "ritaglio" di ogni singolo oggetto onde poterlo trasformare in un singolo file immagine. Il nostro scopo ultimo è portare in Delphi una immagine grande da usare come sfondo, su cui poi faremo "poggiare" tantissime piccole immagini, ciascuna rappresentante un singolo elemento LCD, e che potrà essere acceso o spento all'occorrenza, a seconda di quello che la logica del gioco originale richiede. Possiamo fare uno "strappo alla regola" per i soli elementi che compongono i numeri: sarebbe impensabile pensare di gestire ogni cifra come combinazione dei suoi sette segmenti LCD, pertanto ci limiteremo a fare 10 immagini, ciascuna contenente i segmenti formanti le cifre da 0 a 9 già pronte. Per gestire quelle immagini useremo poi un sistema differente rispetto a quello degli oggetti del gioco.

LA PREPARAZIONE DEI FILE AUDIO

Possiamo ora passare alla prima fase di registrazione dei suoni. Si avvia il registratore di suoni, si prende il microfono e si "lega" al gioco elettronico nella parte dove sta la mini-cassa, usando eventualmente un pezzetto di scotch. A questo punto bisogna avviare la registrazione audio e fare una partita cercando di farla durare il più possibile, facendo sì che il gioco riproduca il maggior numero possibile di suoni - ovvero i soliti "beep beep" del gioco, le musichette di fine livello, le esplosioni, le musiche di avvio e fine partita, etc.

Si otterrà dunque un file audio (da salvare in formato WAV a qualità almeno 22KHz, 16bit, mono) che ci sarà utile per due cose. La prima è ottenere, dopo averli opportunamente "ritagliati", tanti file audio quanti sono i nostri effetti sonori (mediamente da 5 a 10 per gioco). La seconda è farci capire quanti millisecondi intercorrono tra un effetto sonoro e l'altro, ovvero possiamo capire alla perfezione l'esatta velocità con cui si svolge



il gioco! Questo è assolutamente fondamentale in quanto solitamente il gioco diventa più veloce man mano che la partita procede, e con il metodo che vi ho qui descritto riuscirete a sapere in maniera precisissima la relazione tra la velocità del gioco e gli eventi che solitamente la fanno aumentare o diminuire. Tutte queste informazioni le annoteremo su un blocco note e le terremo pronte all'occorrenza.

Continua sul prossimo numero!!!

Chi avesse dubbi o domande, gradisse approfondimenti, o intendesse cimentarsi nella realizzazione del proprio "videogame tascabile", non esiti e mi scriva pure: cercherò di aiutarvi al meglio che posso! Vi aspetto sul mio sito web *www.madrigaldesign.it*.