

DEV.

La rivista che ti insegna a programmare

SHELL

Programmazione & Amministrazione

Monad: la futura shell di Windows

Implementazione shell in Perl

Utilizzo API shell in C#

.NET

Interazione tra codice managed e unmanaged

DB

ADO.NET, istruzioni per l'uso

DB

PostgreSQL 8

Intelligenza Artificiale

Agenti software

VBA

Realizzare un add-in solver per OpenOffice.org

Provato per voi
OpenOffice.org 2.0



GRUPPO EDITORIALE
INFOMEDIA

THE SOFTWARE SIDE OF YOUR WORLD

ISSN: 1124-9488 - CONTIENE I.R.
Sped. in abb. post. 40% art. 2 comma 20/b legge 662/96
Posta di Roma



Programmo
Subito

Corso di Database Design.

Un caso pratico

Realizzare una GUI con Qt



Buffer overflow e dintorni

Game DEV
Introduzione a Game Maker 6

Linux
Fusi orario su Linux con PHP

di Roberto Casarini
rcasarini@infomedia.it

Novità in DEV

Nello speciale di questo numero trattiamo un argomento ben noto agli informatici: la Shell. Ho sempre sentito dire "perché ricordarsi tutti questi strani comandi per effettuare un'operazione banale?". Un informatico conosce bene questa risposta, ma difficilmente ve la dirà. Di interpretazioni vere e proprie della shell ce ne potrebbero essere diverse: la più usuale è intesa

come console per interagire con il sistema. Infatti, il termine stesso "conchiglia" indica il contenitore dei possibili comandi per interagire con il sistema. In particolare, in questo speciale viene fornita una panoramica del termine stesso, partendo dalla futura shell di casa Microsoft per passare ad una possibile implementazione in Perl di shell, arrivando alle cosiddette API shell di .NET utilizzandole con C#.

Grazie a Microsoft, siamo soliti pensare soltanto ad un ambiente costituito da finestre, dove con un pizzico di intuito è possibile utilizzare il proprio calcolatore, senza dover ricordare nulla di complicato o che richieda grandi difficoltà.

E dopo aver viziato l'utente non esperto e trascurato il sistemista, Microsoft torna di scena con una nuova shell: MSH, alias Monad, è la nuova shell del prossimo sistema operativo di Redmond. Le novità introdotte, come vedremo negli articoli dello speciale, sono notevoli, senza per questo abbandonare la filosofia che ha portato al successo il sistema operativo di casa Microsoft.

In questo numero, viene introdotta una nuova rubrica dedicata allo sviluppo di videogiochi. Gli strumenti presenti sul mercato sono numerosi e a volte sconosciuti. Questa rubrica, propone una panoramica degli strumenti essenziali, cercando di introdurre le principali tecniche per realizzare un videogioco con la maggiore semplicità possibile. Questo argomento, in parte interessa gli appassionati di videogiochi, ma senza dubbio interesserà il "comune informatico" visto che le future applicazioni saranno sempre più apprezzabili tramite un'interfaccia 3D.

Partiamo subito con *Game Maker*, uno strumento dal costo molto contenuto, che negli anni si è sviluppato ed ha appassionato amanti e sviluppatori di videogiochi.

Ulteriore novità su DEV, che vi segnalo (anche se in ritardo), è la rubrica J2ME World, dedicata alla programmazione dei cellulari. L'obiettivo di questa rubrica è discutere le API specifiche di sviluppo che permettono di accedere ad alto livello alle caratteristiche dei dispositivi cellulari e handheld.

Infine, tra le novità non vi sarà certamente sfuggito un dettaglio, il mio ingresso nella redazione di DEV. Le presentazioni non mi sono mai piaciute, ma in questa occasione un cenno è d'obbligo.

Essendo anch'io un informatico, come coordinatore cercherò di proporre tematiche d'interesse sia per il programmatore esperto sia per chi è alle prime armi, fornendo degli approfondimenti utili a tutti per rimanere sempre aggiornati. Lo stesso coinvolgimento "viscerale" lo aspetto anche da voi lettori, grazie al vostro senso critico e alla passione che ci accomuna, pronto a recepire consigli e proposte anche dai programmatori più esigenti.

Ho in serbo altre novità, per rendere sempre più interessante la vostra lettura, che mi riservo di presentare di volta in volta: a partire da rubriche caratterizzate da articoli che trattino più in dettaglio lo sviluppo di progetti open source, senza trascurare un'approfondita esplorazione dei principali ambienti e tool di sviluppo.

Non mi resta che augurarvi buona lettura!

Roberto Casarini



Lo speciale di questo numero: Shell: programmazione e amministrazione

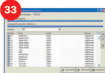
- 12 **Monad: Microsoft Windows command shell**
di Antonio Cisternino
- 18 **MONAD: Il linguaggio di scripting** di Fausto Zanasi
- 22 **Scripting o Shell?** di Antonio Cisternino
- 26 **Utilizzare le API di shell in C#** di Gian Maria Ricci

Programmazione

- 29 **C++ - Buffer overflow e dintorni** di Gianluca Insolubile
- 33 **Cellulari - Il protocollo AT+ dei telefoni cellulari.**
Introduzione alla gestione delle rubriche (PARTE 15)
di Vito Vessia
- 39 **Database - PostgreSQL 8** di Gabriele Bartolini
- 41 **Database - ADO.NET, istruzioni per l'uso** (PARTE 4)
di Lorenzo Vandoni
- 44 **Intelligenza Artificiale - Agenti software** di Paolo Besana
- 47 **Sicurezza - Detection verso Prevention** di Nicola Sotira
- 53 **VB.NET - Utilizzare il controllo DataGridView**
di Visual Basic .Net 2005 (beta) di Emanuele Mattei
- 56 **Visual Objects - SQL Manager 4.1.3** di Danilo Giuliani
- 59 **VBA - Realizzare un add-in solver per OpenOffice.org**
di Andrea Sella
- 62 **.NET - Interazione tra codice managed e unmanaged**
di Ferruccio Bongiani
- 85 **Delphi - Borland Delphi 2005. Il passato e il presente**
di Vito Vessia

Rubriche

- 6 **Developer's Tools** a cura di Roberto Casarini
- 8 **Extended Mode - The Moment of Silence**
a cura di Stefano Arcidiacono
- 10 **SourceForge News** a cura di Gabriele Ganzi
- 11 **CrackMe - Image: quanto quello che sembra non è quello**
che sembra a cura di Matteo Baccan
- 50 **Normative - La tutela della proprietà intellettuale su internet:**
la legge sul Diritto D'autore di Massimo Farina
- 94 **Game DEV - Introduzione a Game Maker 6** di Mauro Pesci
- 98 **Horizon - Il piccolo scrivano fiorentino** di Luigi Morelli



Sei agli inizi? Non riesci a risolvere i primi problemi? Niente paura! Iscriviti alle mailing-list Informedia!

programma_subito@informedia.it
o linux.it@informedia.it

I listati e i programmi di utilità che corredano gli articoli sono prelevabili gratuitamente dal sito <http://www.informedia.it/DEV> oppure da <http://ftp.informedia.it/pub/DEV/Listati/>



Introduzione a Game Maker 6

Realizzare videogiochi non è mai stato così semplice. Game Maker, infatti, giunto alla versione 6 è divenuto un potente strumento in grado di offrire risultati professionali, con pochi sforzi, anche ai meno esperti di programmazione

Realizzare il proprio videogioco è un sogno che molti inseguono, spesso senza successo. Per farlo da soli oltre a possedere notevoli capacità ed esperienza di programmazione è necessario avere competenze in materia di librerie per l'utilizzo della grafica, del suono, dei controlli ecc. Game Maker ci permette di superare tutti questi problemi in un colpo solo. Il software che presentiamo oggi riunisce, in un solo e semplice ambiente di sviluppo, tantissimi strumenti che ci permettono di ottenere risultati stupefacenti con pochi clic anziché migliaia di righe di codice.

L'idea portante di Game Maker, infatti, è di sfruttare le potenzialità di un'interfaccia grafica *drag&drop* che rende possibile caricare *sprite* e *suoni* in maniera semplice e veloce, creare i personaggi del gioco definendone i comportamenti, progettare l'aspetto dei vari livelli ed aggiungerne gli *oggetti*.

La versione 6, inoltre, introduce altre caratteristiche importanti come l'utilizzo delle OpenGL per accelerare la gestione della grafica, gli strumenti per la creazione di giochi 3D tipo *First Person Shooter* o *FPS*, la possibilità di aggiungere funzionalità di gioco multiplayer in rete, di applicare ai suoni effetti di riverbero, coro, eco, proiezione 3d e tantissime altre che rendono questo ambiente un potente strumento per la realizzazione di videogiochi professionali.

Ai più esperti ed esigenti Game Maker mette a disposizione il GML, un linguaggio di scripting che risulta molto semplice da utilizzare. Se nemmeno GML riuscisse a supplire alle esigenze di sviluppo, un'altra soluzione è rappresentata dalla possibilità di creare DLL nel linguaggio che si ritiene più opportuno ed interfacciarle ai giochi Game Maker.

Anatomia di un videogioco Game Maker

Un videogioco creato con Game Maker prende vita all'interno di un'area detta *stanza*, paragonabile ad un livello di gioco, dove vengono posizionati i vari *oggetti*. Esempi tipici di *oggetti* sono il 'personaggio principale', gli 'oggetti volanti', i 'nemici' oppure un 'ostacolo' lungo il percorso.

Possiamo considerare gli 'ostacoli' come *oggetti* inanimati che non hanno, cioè, alcun comportamento. Negli altri casi citati, invece, gli *oggetti* sono in grado di percepire gli *avvenimenti* che possono accadere durante il gioco e reagire ad essi nella

maniera più opportuna. Tali *avvenimenti* sono chiamati *eventi*, mentre le reazioni agli *eventi* sono dette *azioni*. In Game Maker esistono numerosi *eventi*, e tantissime tipologie di *azioni* da svolgere in relazione ad essi. Ogni *oggetto* visibile sullo schermo è rappresentato da un'immagine o da una serie di immagini; tale rappresentazione si chiama *sprite*. Esempi di *sprite* sono un'astronave, un 'fantasma' oppure un'esplosione. Se uno *sprite* è composto da una serie di immagini allora il risultato visivo può essere più significativo dando origine ad un'animazione.

Una volta definiti *sprite*, *oggetti* e *stanze* il nostro gioco è pronto per essere compilato e distribuito in un file eseguibile il quale contenendo al suo interno tutte le immagini e suoni specificati nel progetto, non avrà bisogno di particolari procedure di installazione per funzionare correttamente.

L'ambiente

A prima vista l'ambiente visuale può apparire semplice e spoglio data l'assenza delle decine di icone e menu che si trovano in altri software. La finestra principale, infatti, è fornita di pochi menu a tendina, alcuni pulsanti e un menu ad albero posizionato in verticale sulla sinistra (Figura 1). Ma ciò non deve essere motivo di preoccupazione o di credere che Game Maker sia un prodotto poco funzionale.

Ogni voce presente nel menu ad albero, infatti, rappresenta un aspetto fondamentale nella realizzazione dei giochi ed è su questo che focalizzeremo il nostro interesse. Nel **Riquadro 1** viene presentato un riassunto delle più importanti di queste.

Gli oggetti

Un *oggetto* è la costituente fondamentale e 'viva' di un gioco. Per crearne uno è sufficiente posizionarsi con il mouse sull'apposita cartella chiamata *objects*, premere il

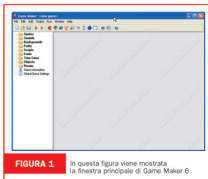


FIGURA 1

In questa figura viene mostrata la finestra principale di Game Maker 6

RIQUADRO 1

Le componenti cruciali che intervengono alla creazione di un videogioco in Game Maker

Oggetto

Rappresenta l'entità principale di un gioco

Stanza

L'area in cui gli oggetti possono esistere

Sprite

Immagine o sequenza di immagini usate per rappresentare un oggetto

Sfondo

Un'immagine da utilizzare come sfondo di una stanza

Suono

Può essere una musica di sottofondo o un effetto sonoro

Script

Un programma scritto in GML (Game Maker Language)

tasto destro del mouse e selezionare la voce 'Add Object'. Subito comparirà la finestra 'Object Properties' come mostrata in **Figura 2**. Qui, partendo da sinistra, sono presenti una casella di inserimento del nome dell'oggetto, il menu di scelta dello sprite (visibile cliccando la piccola icona a destra della casella di testo), una checkbox che specifica se si desidera che l'oggetto sia visibile, un'altra che indica se l'oggetto deve essere considerato solido ed altri parametri che per il momento non sono necessari.

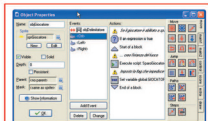
Subito a destra si può vedere una lista con a capo la label 'Events'. Premendo il pulsante 'Add Event', è possibile scegliere a quali tra i tanti eventi disponibili l'oggetto deve rispondere. In **Figura 3** è mostrata un'immagine della finestra di selezione degli eventi.

Passiamo ora alla lista successiva, quella con label 'Actions'. Questa serve ad indicare la sequenza di azioni che devono essere effettuate al verificarsi dell'evento selezionato. Le azioni possibili, e ce ne sono tantissime, si trovano sotto forma di icone nel menu a tabelle a destra, organizzate per categorie.

Si occupano del movimento, della creazione/distruzione di istanze di oggetti, dell'esecuzione dei suoni, del salvataggio/caricamento automatico della partita in corso, della creazione di variabili, della creazione di flussi di controllo, della gestione delle vite, dell'energia, del punteggio e altre mansioni. Per esempio, se si seleziona l'evento 'Key Press' e poi

**FIGURA 3**

Ogni pulsante in questa finestra rappresenta una categoria di eventi

**FIGURA 2**

Così appare la finestra di inserimento delle proprietà di un oggetto

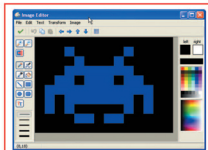
Left ogni qualvolta che il tasto freccia destra viene premuto, potremmo indicare all'oggetto di spostarsi verso quella direzione. Ciò è realizzabile semplicemente trascinando sulla lista delle azioni l'icona con otto frecce rosse nella tabella move e impostando i parametri di direzione e velocità. In questa maniera, semplicemente trascinando le azioni appropriate e impostando i parametri corretti è possibile far compiere agli oggetti qualunque tipo di operazione.

Gli sprite

Game Maker dà la possibilità di utilizzare *sprite* esistenti caricandoli da file ed è in grado di riconoscere i formati grafici più comuni come JPG, BMP, PNG, PSD, WMF, ICO, PSP e altri. È possibile anche modificarli o crearli di nuovi utilizzando l'*Image Editor*, un editor ben fornito e ricco di strumenti per il disegno, il ritocco e la trasformazione delle immagini. Per un esempio vedere in **Figura 4**.

Chi ha già dimestichezza con programmi di disegno e/o fototocco troverà un ambiente molto semplice e familiare. Se si vuole aggiungere un nuovo *sprite* ad un progetto basta procedere come per gli oggetti posizionandosi con il mouse sull'apposita cartella e selezionando la voce 'Add Sprite' dal menu popup.

La finestra di inserimento degli *sprite* presenta due pulsanti 'Load Sprite' e 'Edit Sprite'. Il primo serve a caricare un file esistente tramite la comune finestra di dialogo di Windows,

**FIGURA 4**

L'editor di immagini integrato presenta le stesse caratteristiche di un programma di fototocco

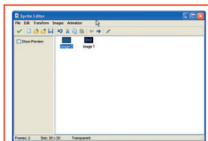


FIGURA 5

Come si vede dalla figura, le immagini sono disposte in sequenza e per ognuna è possibile accedere all'Image Editor tramite un doppio clic

mentre il secondo apre la finestra chiamata 'Sprite Editor'. Questo editor possiede tutte le funzioni necessarie per creare, organizzare e modificare le sequenze di immagini che compongono lo *sprite*. Nei vari menu sono presenti i comandi per cambiare l'ordine delle immagini o per generare nuove animazioni ruotando, ridimensionando o trasformando con effetto di *morphing*, una singola immagine.

Le stanze

Come abbiamo già detto una *stanza* è un'area bidimensionale che rappresenta la "finestra" visibile del videogioco all'interno della quale è possibile posizionare gli oggetti. Qui essi possono muoversi, interagire tra loro, essere modificati, creati e distrutti. Quando si crea una *stanza*, tramite la solita procedura, appare l'editor contenente una serie di menu a tabella ed una rappresentazione della stessa costituita da un rettangolo di colore grigio.

Se si seleziona la tabella *objects*, alla voce 'Object to add with left mouse' è possibile scegliere un oggetto tra quelli definiti e inserirlo nella posizione voluta tramite la pressione del tasto sinistro del mouse, all'interno del rettangolo. Al momento dell'esecuzione del programma ogni oggetto occuperà le coordinate impostate. Nella tabella *settings* si possono inserire il nome di uso interno della *stanza*, le sue dimensioni in pixel, il titolo da applicare alla finestra e la velocità di disegno espressa in fotogrammi al secondo. Tramite questa ultima impostazione è possibile ottenere che il nostro videogioco vada alla stessa velocità su tutti i computer sulla quale è lanciato. La tabella *backgrounds* permette di associare un colore di fondo o un'immagine scelta, tra quelle inserite nell'apposita sezione, e decidere come adattarla in caso questa sia di dimensioni diverse da quelle della *stanza*.

La tabella *tiles* permette di creare una stanza componendola tramite tante piccole immagini come se fosse un mosaico. L'ultima tabella, quella chiamata *views* o *view*, rappresenta un'altra grande caratteristica di Game Maker.

Spesso in un videogioco in 2D gli oggetti e lo sfondo si spostano seguendo i movimenti del personaggio principale, mentre questo rimane sempre posizionato al centro dello schermo. Impostando l'oggetto da seguire ed entro quale area deve rimanere, Game Maker svolgerà tutto il lavoro necessario per ottenere questo effetto.

Gli sfondi

Per quanto riguarda gli sfondi c'è ben poco da dire. Nella cartella *backgrounds* si possono caricare tutte le immagini

che si vuole utilizzare come sfondo per le stanze. Quando si sceglie di creare un nuovo sfondo Game Maker presenta una finestra simile a quella vista per gli *sprite* contenente i pulsanti 'Load Background' ed 'Edit Background'. Il primo serve a caricarne uno da un file esterno mentre il secondo a lanciare l'Image Editor già visto.

I suoni

Nella casella *sounds* vengono caricati tutti gli effetti speciali e musiche di sottofondo. Game Maker riconosce i formati di file audio Wav, Midi e MP3 e permette di applicare loro vari effetti come coro, riverbero, eco ecc. Possiamo anche immergere un suono in un ambiente 3D simulato ed impostarne le coordinate spaziali x, y, e z.

Game Maker non include un editor per modificare i suoni. Per supplire all'inconveniente permette di scegliere il programma esterno preferito da lanciare su pressione del tasto 'Edit Sound'.

Gli script

Il GML è un linguaggio di script costituito da un sottoinsieme della sintassi del C. Chi possiede già familiarità con questo linguaggio non troverà particolari problemi a scrivere uno script.

Basta solo tenere a mente che in GML non esistono tipi di dato per cui dichiarando una *variabile*, questa, potrà essere trattata come intero, numero in virgola mobile, codice di istanza di un oggetto, stringa ecc.

In assenza di un vero e proprio meccanismo di dichiarazione, inoltre, la memoria necessaria al loro utilizzo viene allocata quando l'interprete, in un blocco di codice, incontra la prima occorrenza di utilizzo della *variabile*. È possibile usare la parola 'global' per indicare che la visibilità della *variabile* è estesa a tutto il progetto, come per esempio in questo script:

```
{
// Creo una variabile locale che memorizza l'istanza di
// un nuovo oggetto giocatore
giocatore = instance_create(100, 250, objGiocatore);

// Creo una variabile globale che memorizza
// il numero delle vite rimanenti del giocatore
global.NumeroVite = 3;
}
```

Alla stessa maniera funziona per gli *array*, infatti, per crearne uno di n elementi basta indicare un indice tra parentesi quadre appena dopo il nome della *variabile*, in questa maniera:

```
{
// Creo 10 istanze di nuovi mostri e utilizzo un array
// per memorizzarne l'id
for ( i = 0; i < 10; i += 1){
mostri[i] = instance_create(i*16,100, objMostro1);
}

// Sposto i mostri appena creati di 16 pixel in
// orizzontale utilizzando gli id memorizzati nell'array
for ( i = 0; i < 10; i += 1){
mostri[i].x += 16;
}
}
```

Non sono presenti gli operatori unari di incremento (++) e

decremento (--), mentre i costrutti condizionali if (...) else (...), switch() ... case, i cicli for e while e le parole chiave break, continue, exit, return risultano identici a quelli del C. Sono state aggiunte due parole chiave la prima delle quali è with con lo stesso significato che ha in Visual Basic dove, all'interno del blocco di codice seguente, è possibile omettere il nome dell'oggetto, come in

```
{
// Senza blocco with
giocatore.x = 100;
giocatore.y = 100;
giocatore.image_speed = 10;

// Con blocco with
with (giocatore)
{
x = 100;
y = 100;
image_speed = 10;
}
}
```

La seconda è la parola repeat, che permette di ripetere un comando n volte. Esempio:

```
{
// Creo 100 istanze di ostacoli in posizione
// casuale
repeat (100) instance_create(random(room_width),
random(room_height), objOstacolo);
}
```

Anche i commenti sono identici a quelli del C, cioè, i simboli /* ... */

```
/*
Testo
del
commento
*/
```

per i commenti multilinea e //

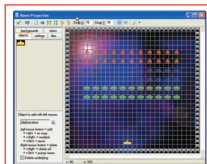


FIGURA 6

All'interno della stanza gli oggetti vengono posizionati tramite il mouse

```
// Teste del commento
```

per quelli a linea singola.

La funzione *instance_create*, serve per creare, a livello di codice, una nuova istanza di un oggetto esistente nel progetto, e rappresenta una delle centinaia di funzioni che GML ha a disposizione.

GML, inoltre, dispone di numerose variabili e costanti interne alle quali è possibile fare riferimento durante il gioco per avere il controllo su vari aspetti, come le dimensioni della stanza attuale

```
room_width
room_height
```

oppure la velocità di visualizzazione delle sequenze di immagini di uno sprite

```
<nome_sprite>.image_speed
```

Come procurarsi Game Maker

La homepage di Game Maker è www.gamemaker.nl, dalla quale è possibile scaricare oltre che i file di installazione anche varie risorse (sprite, sfondi e suoni di esempio), alcuni tutorial e diversi giochi di esempio.

Bisogna, però, chiarire che Game Maker non è un prodotto freeware, ma richiede il pagamento di una modesta cifra per la registrazione, attualmente 15Euro. La versione scaricabile dal sito, infatti, impedisce l'utilizzo di alcune funzionalità e mostra un logo di avvertimento ogni volta che viene eseguito un progetto.

Conclusioni

Esistono molte caratteristiche di cui non si è parlato, come i percorsi, le time line, la gestione dei tabelloni dei punteggi, la possibilità di scrivere sul registro di sistema ecc. che permettono la realizzazione di giochi molto complessi.

E' da considerare che il codice prodotto da Game Maker è già testato e sicuro; di conseguenza anche i tempi di *debugging* sono ridottissimi e permettono di distribuire i propri lavori in totale sicurezza.

Un difetto di Game Maker è che non possiede un efficace sistema di debug per cui risulta, a volte, ostico tracciare il comportamento degli script.

Bibliografia

- [1] Mark Overmars, "Designing Games with Game Maker", www.gamemaker.nl

Mauro Pesci

Si occupa di Analisi e Sviluppo del software in vari ambienti e piattaforme. Ha lavorato per diverse aziende nel settore, mentre ora è impegnato in un progetto presso la IHP di Roma. Attualmente studia Informatica presso l'università di Tor Vergata (Roma).

Download del progetto

Scarica il progetto completo di questo articolo all'indirizzo: <http://ftp.infomedia.it/pub/DEV/Listati/Dev129/gamedev.zip>