Approfondimenti

Provare a scrivere un gioco con Scratch può essere un'esperienza interessante.
Prendiamo spunto da come operano i professionisti di questo settore.
E facciamo conoscenza con i programmatori di questo linguaggio.

La sceneggiatura nei videogame

Qualche spunto di design e tecniche di sceneggiatura per creare videogame semplici ma interessanti

Che si tratti di un prodotto commerciale in cui si investono decine di milioni di euro o di un minuscolo programmino costruito con qualche blocco di Scratch, progettare un gioco interessante non è **mai banale**. Anche il più semplice dei videogame richiede arte, tecnica e psicologia.

Ciò vale non solo per un gioco informatico, ma anche per un gioco da tavolo, per un gioco di carte e non solo: anche nella scrittura di un racconto o di una sceneggiatura cinematografica o teatrale bisogna far sì che il giocatore/lettore/spettatore trovi l'opera gradevole.

Naturalmente nei videogame si usano tecniche differenti rispetto ad altre forme artistiche, anche se vi sono diversi punti in comune: l'animo umano è lo stesso e uguali sono i meccanismi della psiche che determinano un'impressione positiva o negativa di un'**esperienza** (ludica).

Alcuni "trucchi del mestiere" riguardano la storia e la presentazione, ossia la sceneggiatura, altri il progetto (design). Forniamo qui qualche spunto, in ordine sparso e senza pretesa di completezza: sull'argomento sono stati scritti, infatti, numerosi articoli e libri.

Non va dimenticato che queste tecniche hanno grande importanza anche per i giochi educativi, e in generale per rendere più attraente la trasmissione del sapere.

STORIA E OBIETTIVO

Uno stesso gioco, anche di azione, diventa più interessante se contiene un elemento di narrazione, qualcosa che fornisca un motivo per le azioni che si devono compiere. Facciamo un esempio, considerando tre possibili casi:

1. il cerchio A deve raggiungere il cerchio B;
2. il gatto deve catturare il topo;
3. il gatto deve catturare il topo, che porta una pericolosa malattia.

Il caso 1 è alquanto banale; si tratta semplicemente di far incontrare due forme geometriche. Che la cosa sia più o meno facile non ha importanza (almeno non in questo contesto): non c'è un vero motivo per farlo.

Nel caso 2 il meccanismo di gioco è assolutamente identico, abbiamo semplicemente sostituito le immagini geometriche con immagini di animali. Già la presenza di esseri viventi

cambia la percezione del gioco, ma c'è un ulteriore aspetto: anche senza alcuna spiegazione, ci aspettiamo automaticamente che il compito del gatto sia quello di catturare il topo: vi è un chiaro scopo da raggiungere.

Il caso 3 aggiunge un'altra componente, un coinvolgimento emotivo: bisogna salvare gli abitanti della casa dal possibile contagio. Affinché ciò si verifichi, l'obiettivo dev'essere dichiarato esplicitamente o implicitamente all'inizio del gioco. Per esempio, potrebbe apparire brevemente un bambino che dice «Aiuto, quel topo è contagioso! Gatto, salvami!» (nota: per leggibilità usiamo "bambino", senza per questo intendere distinzione di genere).

In questo modo il giocatore si sente fin da subito protagonista in prima persona, identificandosi in questo caso col gatto; è più motivato ad agire per cercare di catturare il topo e maggiore sarà la sua soddisfazione nel riuscirci.

L'IMPORTANZA DEL FINALE

L'**ultima scena** ha grande importanza nel determinare il grado di soddisfazione del giocatore e il ricordo più o meno piacevole del gioco. Torniamo all'esempio precedente e vediamo cosa può succedere quando il giocatore-gatto cattura il topo:

1. il gioco finisce senza dire nulla;
2. appare una scritta «Hai vinto»;
3. il gatto dice «Finalmente l'ho preso!»;
4. appare il bambino e dice «Grazie gatto, mi hai salvato!».

La soluzione 1 è assai poco gradevole: manca la conferma esplicita, il riconoscimento di avere raggiunto l'obiettivo, di avere fatto qualcosa che valeva la pena di fare. Il giocatore rimane come in sospeso e sostanzialmente deluso.

Anche la soluzione 2 non è granché: il gioco è finito, ma il giocatore non ne ricava un particolare appagamento.

La 3 è già più interessante: non dimentichiamo che il giocatore si identifica col gatto, per cui è come se fosse il giocatore stesso a esprimere la propria soddisfazione per avere raggiunto l'obiettivo.

La 4 è ancora meglio: non solo la conferma-elogio arriva da un altro personaggio (e ha quindi maggior valore) ma c'è anche la gratificazione per avere fatto qualcosa di buono.

LINGUAGGIO NON VERBALE

Informazioni ed emozioni possono essere espresse, anziché con la parola scritta, per mezzo di semplici **effetti grafici**.

Invece di dichiarare l'obiettivo del gioco facendo dire al bambino «Aiuto, quel topo è contagioso! Gatto, salvami!», possiamo comunicare la stessa informazione alterando leggermente i due sprite.

Ad esempio possiamo far ricoprire il topo di macchie rosse, meglio ancora se in due o tre stadi progressivi e magari aggiungergli un alone verdastro, duplicando il costume di Scratch e modificandolo; inoltre potremmo partire con un topolino, ingrandirlo e far lampeggiare le macchie due o tre volte (alternando il costume originale con quello "infetto").

A questo punto cambiamo l'espressione del bambino, inizialmente tranquilla e sorridente, in modo che appaia preoccupato o impaurito oppure, ancora più semplicemente, facciamo soltanto lampeggiare alcune linee tracciate a raggiera dalla sua testa per rendere l'idea di "allarmato" (in entrambi i casi con un cambio di costume dello sprite).

Abbiamo così espresso in forma non verbale due concetti: il topo è contagioso e pericoloso, il bambino è spaventato. La forma grafica è per certi versi più efficace, in quanto non richiede un'analisi cognitiva del testo, ma trasmette la sensazione in modo diretto.

Nulla vieta, naturalmente, di **combinare grafica e testo**, ad esempio facendo lampeggiare una scritta «Contagioso!» insieme ai pallini: non esistono regole fisse, in questo campo c'è sempre ampio spazio per la fantasia e la sperimentazione.

PICCOLI SUONI, GRANDE EFFETTO

Finora abbiamo giocato prevalentemente con gli sprite, ossia con la grafica, ma la vista non è l'unico senso che possediamo. L'udito è altrettanto importante: lo possiamo usare per comunicare, integrare e sottolineare; avremo creato in questo modo un gioco ancora più multimediale e di conseguenza più efficace.

Non occorre essere tecnici del suono: pochi semplici tocchi sono sufficienti a cambiare in modo significativo la percezione di un gioco, ossia a migliorare nettamente quella che con un termine un po' abusato si chiama ***player experience***.

Possiamo ad esempio aggiungere un suono breve ed aspro, ripetuto per le canoniche tre volte, quando il topo diventa "infetto"; una finezza potrebbe consistere nell'aumentare ogni volta il volume, di pari passo con l'ingrandimento del topo.

Per produrre il suono con Scratch abbiamo tre possibilità: usare uno dei suoni predefiniti, registrarne uno nuovo, oppure semplicemente suonare delle note musicali scegliendo opportunamente lo strumento per ottenere l'effetto desiderato.

Nel nostro caso potremmo usare tre note basse di tonalità crescente per il topo e magari invece registrare la voce di un alunno per il bambino che chiede aiuto al gatto. I vari eventi del gioco potranno poi essere sottolineati con suoni appropriati. Non dimentichiamo il finale, del quale abbiamo già avuto modo di sottolineare l'importanza. Anche qui, invece del (o in aggiunta al) ringraziamento verbale, una musichetta rafforza il "premio" che il giocatore riceve per il raggiungimento dell'obiettivo.

Non servono suoni elaborati: **bastano poche note** per rendere eventi e personaggi più realistici; anzi, a voler strafare si rischia di ottenere l'effetto contrario, annoiando il giocatore

con suoni troppo frequenti e rendendo di conseguenza meno efficaci quelli significativi.

DARE UN NOME AI PERSONAGGI

Un altro ritocco piccolo ma significativo consiste nel rendere i protagonisti del gioco ancora più realistici con un semplice trucco: dare loro un nome.

Non serve farlo esplicitamente: se prima avevamo scritto, ad esempio «Gatto, salvami!», è sufficiente sostituirlo con «Nerino, salvami!».

Il gatto generico viene così ad assumere una personalità propria e si **identifica** automaticamente con il gatto di casa, non solo nel gioco ma anche col nostro (reale o immaginario) animale di compagnia.

L'IMPORTANZA DEI TEMPI

Le attese possono cambiare in modo sensibile la **percezione** di una scena. Consideriamo ad esempio questo frammento di dialogo, tra due personaggi che si guardano:

– Dove sei stato?

– A scuola.

In Scratch lo possiamo rendere mostrando la prima frase per 2–3 secondi, poi sostituendola con la seconda (pronunciata ovviamente dall'altro personaggio). Fin qui nulla di particolare, la risposta appare neutra e informativa.

Proviamo però a introdurre una pausa di 3–4 secondi tra la scomparsa della prima frase e l'apparizione della seconda, in modo che i due si guardino in silenzio: questa volta, benché le frasi siano esattamente le stesse, abbiamo la netta impressione che la risposta non sia sincera.

Un'altra importante funzione delle temporizzazioni consiste nel dare al giocatore il tempo di digerire le informazioni (o le emozioni). Tornando all'esempio del gatto, se l'animazione che mostra graficamente che il topo è contagioso dura troppo poco e il topo inizia subito a muoversi senza una breve attesa, il giocatore si concentrerà sulla **meccanica del gioco** senza averne capito bene la motivazione.

Per contro, se l'animazione mostrata all'inizio del gioco dura troppo tempo, il giocatore si stancherà presto di doversela sorbire ogni volta che rigioca. Una possibile soluzione consiste nel dare la possibilità di saltare la scenetta premendo un tasto (in Scratch è un po' macchinoso da fare); oppure si può mostrare l'animazione solo alla prima partita (usando una variabile).

Una soluzione meno elegante, ma tecnicamente assai più semplice da realizzare può essere quella di aggiungere due pulsanti: "Introduzione" e "Gioca". In tal modo la scenetta iniziale può essere lunga a piacere, senza annoiare.

LIVELLO DI DIFFICOLTÀ

Perché sia gradito, un gioco non dev'essere né troppo facile né troppo difficile: se è troppo facile il giocatore si annoia, se è troppo difficile smette di giocare. In entrambi i casi si è mancato l'obiettivo.

Solo se il **premio finale** è raggiunto a prezzo di un certo impegno il giocatore si sente gratificato, ma raggiungerlo non dev'essere troppo frustrante. Come tutte le altre considerazioni, ciò vale tanto per i giochi di azione che per quelli di riflessione.

Si tratta di uno dei tanti problemi di bilanciamento con cui devono confrontarsi i progettisti di giochi, problema reso più difficile dal fatto che non tutti i giocatori hanno lo stesso livello di abilità (manuale o intellettuale).

Per un gioco d'azione si tratta solitamente di aumentare o diminuire la **velocità di movimento** o i tempi di reazione richiesti, mentre in un gioco di riflessione cambierà la difficoltà degli enigmi.

In ogni caso, chi crea un gioco tende generalmente a farlo troppo difficile: questo perché, a differenza del giocatore, lo conosce molto bene e l'ha giocato mille volte o, nel caso di problemi "intellettuali", conosce già la soluzione o la strategia migliore.

Nel dubbio è meglio sbagliare dal lato della facilità, ossia fare in modo che l'obiettivo sia più facilmente raggiungibile: ottenere il premio con facilità è infatti poco soddisfacente, ma sempre meglio che non ottenerlo affatto.

Il metodo migliore per ottenere un corretto bilanciamento è quello di osservare in silenzio diverse persone mentre giocano, prendendo nota delle difficoltà che incontrano e delle loro emozioni.

Tratto da *Coding - Primi passi*, De Agostini Scuola

I mestieri nell'industria videoludica

Una "fabbrica" di giochi richiede diverse competenze professionali e capacità di lavorare in team

C'era una volta il mito del ragazzino un po' goffo e introverso che, nella sua cameretta, creava da solo un gioco per computer di grande successo mondiale.

Quei tempi sono passati. Intendiamoci, gli sviluppatori solitari esistono ancora, ma le loro chance di successo sono molto basse, nonostante i pochi casi eclatanti, ampiamente pubblicizzati dai mezzi di informazione.

Il settore dei videogame è ormai una vera e propria **industria**, con un fatturato dello stesso ordine di grandezza di quella cinematografica (e forse persino superiore). Proprio come nel cinema, produrre una nuova opera richiede un notevole investimento di denaro e di lavoro, coinvolge un gran numero di partecipanti, ma soprattutto richiede diverse abilità... che si possono iniziare a imparare con Scratch!

I GRANDI E I PICCOLI

Per dare un'idea, la realizzazione di un gioco "AAA" (cioè uno dei più complessi, prodotto da una grande software house) può costare oltre cento milioni di euro e richiedere il lavoro di centinaia di addetti per 2–3 anni.

Naturalmente esistono aziende dalle dimensioni più ridotte, spesso capaci di realizzare giochi più interessanti perché non legate a strette logiche di marketing. Alla base della piramide c'è un gran numero di piccoli gruppi di giovani che si divertono, riescono magari a guadagnare qualcosa, ma soprattutto si preparano per lavorare nel settore che li appassiona ed essere pagati per quello che hanno sempre sognato di fare.

Dalle grandi software house ai piccoli team spontanei, tuttavia, le **competenze necessarie** per realizzare un gioco completo sono le stesse; quando i partecipanti sono pochi, ciascuno dovrà svolgere più di un lavoro (oppure qualcosa viene tralasciato), mentre nella grande industria c'è magari un'intera squadra assegnata a ciascun compito. Vediamo quali sono i principali "mestieri" e alcune delle loro specializzazioni.

IL DESIGNER

Per creare un gioco occorre per prima cosa avere un'idea, per esempio una corsa tra **Achille e la tartaruga**. Ma l'idea non basta: va sviluppata in dettaglio, arricchita fino a farla diventare un progetto completo: è lì che davvero si crea il gioco.

Il designer deve ideare una storia e i personaggi, definire il loro aspetto e carattere, scegliere un'ambientazione, descrivere il gameplay, cioè il meccanismo di gioco con le azioni e gli eventi (per esempio in quel punto piove, Achille può scivolare e slogarsi il tallone; se la tartaruga trova una molla avanza a grandi balzi per 10 secondi, e così via).

Deve indicare esattamente i comandi a disposizione del giocatore e come essi agiranno, stabilire le varie schermate che appariranno e stendere una **documentazione scritta** che descriva completamente il gioco: questo sarà il progetto su cui tutti lavoreranno.

Il suo lavoro non finisce qui, perché deve risolvere gli innumerevoli problemi che si presenteranno durante lo sviluppo, chiarire le ambiguità, aggiungere, togliere, definire, ristrutturare o riprogettare quando necessario. In pratica è il regista dell'opera.

Se il gioco è formato da più livelli o ambientazioni differenti (per esempio il bosco, il deserto, la montagna), ciascuno di essi può essere progettato, nell'ambito di quanto definito, da una diversa persona: un "level designer".

Infine, se c'è una storia da narrare, lo "script writer" (scrittore della sceneggiatura) ha il compito di prepararne tutti i testi.

I GRAFICI

Nei giochi a **due dimensioni** (2D) vi possono essere due tipi diversi di immagini e sfondi, come in Scratch: vettoriali (fatte di elementi come cerchi e linee) oppure bitmap (fatte di pixel); nel primo caso il mestiere è simile a quello di un illustratore pubblicitario; nel secondo si parla di "pixel art" e le competenze richieste sono assai diverse.

La **grafica 3D** è ben più complessa da realizzare: per prima cosa occorre preparare un modello tridimensionale di ciascun elemento del gioco (personaggi, oggetti, costruzioni, alberi, ecc.) e questo è compito del modellatore.

I personaggi avranno anche uno "scheletro" che definisce come possano muoversi l'una rispetto all'altra le varie parti che li compongono.

A questo punto l'esperto di "skinning" può applicare la "pelle" sul modello 3D, trasformando una forma geometrica

Tratto da *Coding - Primi passi*, De Agostini Scuola

tridimensionale in un colorato oggetto o personaggio, come appare al giocatore.

L'animatore sta a metà tra la grafica e la programmazione: deve fare in modo che il personaggio si muova in modo realistico (nell'ambito del gioco) quando cammina, corre, salta, si tuffa e così via.

Per ogni scena, c'è chi si occupa dell'illuminazione ("lighting"), definendo tonalità di fondo, posizione, colore e caratteristiche delle fonti di luce, in modo da dare verosimiglianza e coerenza al tutto.

Può servire anche un esperto di video editing, che realizzerà il filmato di presentazione e gli eventuali video clip (spezzoni di video) mostrati nel corso del gioco.

I PROGRAMMATORI

Le immagini da sole non fanno un gioco: occorre scrivere del codice, cioè un programma che esegua quanto indicato dal designer. Detto così sembra una cosa semplice, ma per arrivare a completare con successo un gioco commerciale servono diverse specializzazioni di "coder".

Può valere la pena di sottolineare agli allievi quanto la matematica e la geometria siano importanti per la programmazione dei giochi: oggi si parte dalle tabelline e domani magari si arriva... ai giochi in 3D (che sono costruiti a base di moltiplicazioni di matrici quadridimensionali).

Tornando alle specializzazioni, un "engine programmer" costruisce il "motore" del **meccanismo di gioco**, quello che trasforma i modelli tridimensionali in immagini sullo schermo, applica l'illuminazione e via dicendo; questa parte è talmente complessa che quasi sempre si preferisce utilizzarne una già fatta (ad esempio, Scratch ci fornisce un engine 2D).

Il "gameplay programmer" si occupa della **parte logica e funzionale del gioco**, ossia fa in modo che esso risponda alle richieste del designer ("se il giocatore preme la freccia in giù, la tartaruga deve fermarsi e scavare una piccola buca"). Per integrare il lavoro preparatorio di animazione svolto dai grafici, ad esempio combinando animazioni in vario modo in una certa scena, può esserci anche un "animation programmer" specializzato in questo compito.

Un "UI programmer" realizza e fa funzionare l'**interfaccia utente**: si tratta di tutti i menu e le schermate che non fanno parte del gameplay, ossia del gioco propriamente detto.

Il mestiere di "AI programmer" consiste invece nel progettare e realizzare l'**intelligenza artificiale**, cioè tutto il necessario perché il gioco possa prendere decisioni autonome, ad esempio trovare la migliore tattica per Achille quando il giocatore controlla la tartaruga. Un tipico esempio di AI è il "pathfinding", cioè il tracciamento del percorso migliore verso una data destinazione.

Abbiamo poi un "FX programmer" che realizza gli **effetti speciali** (viene giù un fulmine, Achille fa scintille, la tartaruga lascia una scia multicolore).

Non è finita: serve un "audio programmer" che inserisca musica, voci ed effetti sonori, facendo in modo che il giocatore li senta provenire dalla parte giusta e col volume adeguato alla distanza (di solito l'engine aiuta in questo secondo compito).

Se il gioco è multigiocatore, magari via Internet, occorre anche un "network programmer" che si occupi di far funzionare la **comunicazione tra i diversi computer**, minimizzare i ritardi quando la linea non è ottimale, fare in modo che il gioco possa proseguire anche nel caso che qualche informazione venga persa in transito.

Ultimo, ma non per importanza, è il "tool programmer": il suo lavoro consiste nel costruire gli **strumenti software** necessari agli altri partecipanti, per esempio l'editor dei livelli con il quale piazzare laghi, montagne, strade, ostacoli, personaggi e tutti gli altri elementi del gioco, oppure un programma per convertire un modello 3D da una forma di rappresentazione a un'altra.

AUDIO E MUSICA

Se l'audio programmer deve integrare correttamente i suoni nel gioco, qualcuno deve realizzare questi suoni.

Serve quindi un compositore che crei delle musiche su misura: per la scena iniziale, per il sottofondo, per gli eventi importanti, per il finale. Non è solo un lavoro puramente musicale, perché deve anche rispettare requisiti tecnici e di durata.

Quello che nel cinema è il "rumorista", nei giochi è l'esperto di **effetti sonori**: invece di scuotere un foglio di lamierino per simulare un tuono (come si usava una volta), deve saperlo fare con mezzi elettronici... ma nessuno gli proibisce di usare anche i vecchi trucchi del mestiere!

Le eventuali voci dei personaggi sono registrate da attori, basandosi sul lavoro dello script writer. Se occorre sincronizzare il movimento delle labbra al suono delle parole ("lip-syncing"), potrà essere necessaria una coordinazione sia con i grafici sia con i programmatori, ad esempio la costruzione di uno strumento (tool) per facilitare questo compito.

I COLLAUDATORI

Difficilmente un programma funziona perfettamente al primo colpo, a maggior ragione se è complesso e prodotto dalla collaborazione di diversi partecipanti.

I "tester" hanno il compito di **scovare errori** e problemi di ogni genere, grandi e piccoli, di programmazione e di design, lavorando in parallelo allo sviluppo del gioco in modo da poterli identificare e correggere quanto prima possibile. Ad esempio: se Achille inciampa nell'ultimo sasso a sinistra della prima curva poi non si rialza; la tartaruga non risponde alla

freccia a sinistra dopo una capriola; nel secondo livello non c'è abbastanza da mangiare; e così via.

I problemi più gravi, tutt'altro che rari nelle fasi di sviluppo, sono i "crash": il programma si "pianta" e non è più possibile continuare a giocare. Alcuni di essi sono ripetibili, cioè avvengono sempre quando si compie una certa azione, ma altri si verificano in modo apparentemente casuale; in quest'ultimo caso, il tester deve trovare un modo per far succedere sempre o quasi sempre il problema, in modo che i programmatori possano analizzarne la causa e porvi rimedio.

A differenza degli altri partecipanti, il lavoro del tester prosegue anche dopo la fine dello sviluppo, quando il gioco è in vendita: dovrà infatti trovare le cause dei problemi segnalati dai giocatori. Può sembrare un lavoro divertente (giocare di continuo) ma è in realtà un compito assai impegnativo, faticoso e ripetitivo. È tuttavia la tipica porta di ingresso per iniziare a fare esperienza nell'industria dei videogame.

L'ORGANIZZAZIONE

Coordinare il lavoro di più persone esperte in diverse discipline non è cosa facile, specialmente se va fatto in tempi ristretti e se il progetto da realizzare si modifica continuamente nel corso dello sviluppo. Una buona **struttura organizzativa** è essenziale.

Ciascun sotto-team ha il proprio dirigente: vi possono essere un direttore artistico ("art director"), un capo programmatore ("lead programmer") e via dicendo, esperti nel proprio lavoro e capaci di dirigere e organizzare quello degli altri.

Il "project manager" coordina l'intera struttura e dialoga con i committenti, cercando di far sì che si raggiungano determinati obiettivi ("milestone") nei tempi prefissati.

Nel corso dello sviluppo si realizzano diverse versioni "alfa" (molto preliminari) e "beta" (già più simili al prodotto finito), finché si arriva finalmente al "gold master", cioè l'originale del DVD (o altro supporto) pronto per la duplicazione. Il lavoro di sviluppo è finito.

Ma non è finito il lavoro dell'azienda: c'è tutta la parte di marketing, pubbliche relazioni, localizzazione (traduzione in altre lingue), distribuzione, supporto... insomma, realizzare un gioco non è un gioco!

I programmatori di Scratch

Anche un linguaggio di programmazione ha bisogno di essere programmato

Scratch è un software completo e complesso, che per essere sviluppato e costantemente aggiornato ha bisogno del contributo di molte persone.

Questo gruppo di ricercatori universitari prende il nome di **Scratch Team** e opera a Boston, negli Stati Uniti, presso l'università Massachusetts Institute of Technology.

Il gruppo ha una pagina web ufficiale, che è questa: https://wiki.scratch.mit.edu/wiki/Scratch_Team.

In questo momento sono tutti al lavoro sulla prossima versione di Scratch.

Si ringrazia Carmelo Presicce dello Scratch Team per la fotografia.