PIXEL ART

da **CODE MOOC**

Considerazioni preliminari

I computer per rappresentare le immagini hanno bisogno di costruire una griglia e di colorare i quadretti. Non sanno fare diversamente. Ogni quadretto è un **pixel**. Chiamiamo **pixel art** ogni disegno che mette in evidenza la struttura a quadretti e ne fa un espediente artistico, facendo di necessità virtù.

Tanto più piccoli e numerosi sono i pixel, tanto meno evidente è la quadrettatura e tanto più definita e continua ci appare l'immagine. I video ad alta risoluzione hanno 1920 colonne e 1080 righe, quindi circa 2 milioni di pixel. Le fotocamere dei nostri cellulari e gli schermi di cellulari, tablet, computer, anche di più.

Definita la griglia e scelto un punto di partenza (per convenzione il pixel in alto a sinistra) e un ordine (per convenzione la scansione per righe da sinistra a destra e dall'alto al basso), per rappresentare un'immagine basta dire il colore di ogni pixel. Non c'è neppure bisogno di rappresentare il cambio di riga, perchè basta aver deciso una volta per tutte che quando si arriva in fondo si ricomincia dall'inizio della riga successiva. Proprio come quando si legge. La sequenza dei colori di ogni pixel è una mera **descrizione** dell'immagine. Non è una **procedura** per disegnarla. Per renderla tale bisogna anche spiegare come usarla. Ad esempio: partendo dal pixel in alto a sinistra e procedendo per righe, finchè ci sono colori nella sequenza, leggi un colore, usalo per colorare il pixel su cui ti trovi e vai avanti.

Esistono anche descrizioni più compatte. In particolare la codifica **RLE** (run length encoding) usa il concetto di "run" per evitare di ripetere più volte di seguito lo stesso colore. Un "run" è proprio una sequenza di pixel contigui dello stesso colore, che viene rappresentata indicando il colore una sola volta e specificando per quanti pixel va mantenuto.

Essendoci diverse possibili descrizioni, per interpretarle correttamente occorre sapere che convenzione adottano. Nelle immagini che usiamo abitualmente l'**estensione** del file (le

due o tre lettere che seguono il nome del file, dopo il punto) ne indica il formato. Il programma che usiamo per aprire il file applica la procedura giusta per interpretare la descrizione e mostrarci l'immagine.

Usando la pixel art come attività di **coding unplugged** possiamo prendere ispirazione dai formati di descrizione delle immagini adottando una convenzione e descrivendo l'immagine in modo coerente. In pratica è come se la procedura facesse parte della convenzione adottata una volte per tutte e la descrizione dell'immagine permettesse all'esecutore di ricostruirla. Nel gergo informatico, la descrizione dell'immagine è di tipo **dichiarativo.**

Possiamo usare un approccio **imperativo** dotandoci di un repertorio di istruzioni che descriva direttamente cosa fare per disegnare l'immagine. In tal caso la descrizione dell'immagine e le azioni da compiere per riprodurla sono un tutt'uno. Ad esempio possiamo usare CodyRoby per spostare un robottino sulla scacchiera dei pixel aggiungendo una nuova istruzione che dica a Roby di colorare il pixel su cui si trova. Il disegno è l'intera sequenza di istruzioni di movimento e di colorazione.

In ogni caso, il disegno che si produce colorando i pixel è una cosiddetta **pixelmap**, o **bitmap**, caratterizzata da una risoluzione predefinita. Per contro, un disegno potrebbe essere descritto attraverso senza limiti di risoluzione come combinazione di figure geometriche e curve di cui siano dati i parametri o le regole di costruzione. Se vogliamo disegnare una circonferenza ci basta indicare il centro, il raggio, lo spessore della linea e il colore. Qualunque sia la risoluzione del disegno, con questa descrizione potremo sempre ricostruirla sfruttandola pienamente. Questa tecnica si chiama **grafica vettoriale**. Da un disegno vettoriale si può sempre generare una bitmap di qualsiasi risoluzione. Da una bitmap non si può rigenerare un disegno vettoriale o una bitmap di risoluzione più alta.

Anche con la grafica vettoriale si può fare coding. E anche in questo caso si può definire una convenzione e usare un linguaggio dichiarativo per descrivere il disegno, oppure si può usare direttamente un linguaggio imperativo per descrivere le azioni necessarie a disegnarlo. E' un po' come il disegno tecnico. Se dobbiamo spiegare a qualcuno come usare il compasso per disegnare il cerchio gli diamo istruzioni precise e seguendo le

nostre indicazioni disegnerà il cerchio (**approccio imperativo**). Ma se tutti sappiamo già come di usa il compasso per disegnare un generico cerchio, per far disegnare un cerchio ad un altro ci basterà fornire i parametri del cerchio che vogliamo (**approccio dichiarativo**).

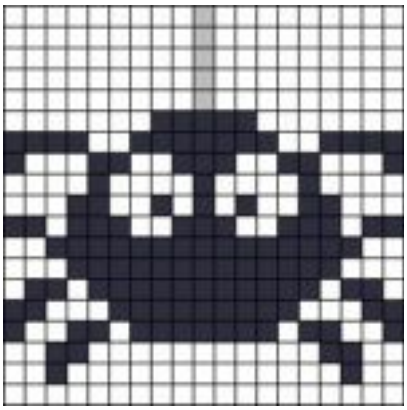
L'approccio imperativo è quello dell'artista di code.org o di Scratch. Se volete sperimentare la grafica vettoriale e toccare con mano il potere espressivo di un linguaggio dichiarativo, vi consiglio di usare **inkscape** (è libero e open source e consente di fare di tutto) per generare un file .**svg** (simple vector graphics). Poi aprite il file .svg con un editor di testo per guardarci dentro e provate a riconoscere la descrizione dei vari oggetti che avrete disegnato.

PIXEL ART

da **Maestramarta**

Continuando a parlare di coding e programmazione nelle varie classi, abbiamo parlato anche dei **PIXEL**…cosa sono?

*“Il termine inglese **Pixel** ha origine dalla contrazione delle parole picture ed element (pict-el, dunque pixel) e serve a identificare ogni singolo puntino che compone un’immagine all’interno della memoria di un computer.*
*I **punti riprodotti sono così piccoli e numerosi** da non essere distinguibili ad occhio nudo, apparendo fusi in un’unica **immagine** quando vengono stampati su carta o visualizzati su un monitor.*
Ogni pixel dunque rappresenta il dato più minuscolo dell’intera immagine e ha un valore preciso dato dalla sua posizione, dal colore e dall’intensità.



Come **scrivere un codice** che permetta a una macchina, o nel nostro caso a un compagno, di riprodurre l'immagine?
Cominciamo **numerando o nominando con lettere le varie righe dell'immagine** e procediamo segnando su un foglio quanti e quali quadretti colorare e di che colore…
Quindi segniamo per la riga A quanti quadretti bianchi, quanti rossi ecc e procediamo in questo modo per tutte le righe che compongono la nostra immagine…

Alla fine del lavoro scambiano i codici tra compagni e leggendolo riproduciamo le immagini…seguiamo le indicazioni…

Alla fine c'è la fase di controllo ed eventualmente correzione del codice detta in gergo tecnico "**Debugging**"…ognuno confronta il proprio disegno con l'originale e vede se è uguale, in caso contrario è necessario trovare dove sta l'errore? nel codice o nella realizzazione? Si trova l'errore e lo si corregge…**FASE MOLTO IMPORTANTE PER OGNI PROGRAMMAZIONE!!!!**

*"Il **debugging** (o semplicemente **debug**), in informatica, indica l'attività che consiste **nell'individuazione** da parte del programmatore della **porzione di software affetta da errore** (bug) rilevata nei software a seguito dell'utilizzo del programma.*
*L'attività di **debug** è una delle operazioni più importanti per la messa a punto di un programma, spesso estremamente difficile… " (da Wikipedia)*

I lavoro ha coinvolto moltissimo i bambini che hanno realizzato tantissimi codici per tantissime immagini e ne hanno anche portate a casa, fatte fare a parenti e amici e realizzate alcune da soli…inventate…

Naturalmente **questo non è l'unico modi di scrivere il codice** per riprodurre l'immagine…si può fare attraverso le coordinate, numerando righe e colonne e dando le varie coordinate da colorare, oppure numerare i quadretti della griglia e dire quale quadretto colorare e di che colore…oppure approntando un codice simile a quello della griglia di CodyRoby con spostamenti e riempimenti…

Io li ho stimolati e fatti riflettere per cercare la soluzione e il modo più comodo (secondo loro) per scrivere il codice e loro hanno scelto questo…

In **facebook** una miniera di attività di questo genere ma anche altro legato al **coding** sono i gruppi **CODING IN YOUR CLASSROOM NOW**, e **CODING E PENSIERO COMPUTAZIONALE** in cui si trovano spunti e attività e si condividono volentieri esperienze e materiali e in cui trovare molti modi diversi di scrivere codici e presentare attività ai bambini.

Abbiamo lavorato con i PIXEL anche facendo matematica grazie alle schede del sito **http://www.coloringsquared.com/ che propone di risolvere semplici o complesse operazioni e colorare seguendo il codice colorato**