02
OTT
2020

SCUOLA ITALIANA MODERNA

Rivista
per la scuola
primaria

SPECIALE

GIOCHI LINGUISTICI GIOCHI LOGICI

- LET'S PLAY WITH CLIL
- GIOCHIAMO CON LA PIXELART
- ITALIANO PER GIOCO
- PROBLEMI E GIOCHI MATEMATICI

EDITRICE LA SCUOLA

CODING
step-by-step

GIOCHIAMO CON LA PIXELART

Antonio Faccioli
Formatore e volontario CoderDojo

La Pixelart è un'attività di coding?
No, perché la Pixelart per come viene spesso proposta da molti non può considerarsi un'attività di coding.
Mi spiego meglio: la semplice consegna di un codice da seguire e da cui ricavare un disegno attraverso una griglia è sicuramente un ottimo esercizio che mette in gioco diverse competenze, come la lateralità o la capacità di eseguire istruzioni, ma di certo non sviluppa il pensiero logico-computazionale.
Se ci fermiamo solo all'abilità di trascrizione, al massimo possiamo considerarla un approccio introduttivo al coding.
La Pixelart, inoltre, non potrebbe essere definita coding perché manca di alcuni elementi fondamentali e importanti: le condizioni (se… allora... altrimenti) e le ripetizioni (ripeti finché, ripeti 10 volte ecc.), elementi che, normalmente non presenti nella Pixelart, sono invece di estrema importanza quando vogliamo fare coding.
Quindi abbandoniamo questa tipologia di attività?
Assolutamente no: possiamo introdurre alcuni accorgimenti che possono far divenire la Pixelart un'attività di coding.
Io preferisco utilizzare il termine Pixelcoding in contrapposizione a Pixelart, proprio per distinguere i due approcci strumentali.
Il risultato finale sarà sempre un disegno su una griglia, ma, giocando sul tipo di istruzioni, sulla sua sintassi e sulla tipologia di attività, possiamo arrivare ad avvicinarci molto al coding, se non addirittura a farlo entrare tra i possibili strumenti.

STEP 1 • IMPARIAMO UN LINGUAGGIO

Sicuramente il primo passaggio è quello di **definire un linguaggio per le istruzioni** (**codice**) che andremo a utilizzare.
Prendiamo, come esempio, la **Figura 1**: si tratta di predisporre un **codice** che da una parte sia **semplice da utilizzare**, **veloce da comprendere** e che dall'altra **non contenga istruzioni ambigue**. Per questo propongo un paio di soluzioni e rinnovo, come sempre, l'invito a farle vostre e a personalizzarle.

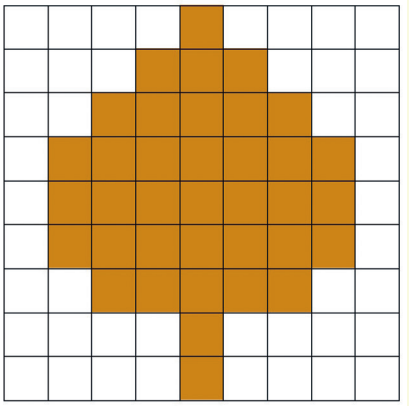


Figura 1

Step 1.1 • Linguaggio visuale

La prima soluzione è un **codice** basato su alcune palline colorate che può essere scritto in **forma estesa** (**Figura 2**): il **numero delle palline corrisponde esattamente alle caselle da colorare nella griglia**. Per questo lavoro consegneremo il codice e i bambini dovranno colorare con lo stesso colore le celle corrispondenti sulla griglia vuota. È un primo approccio per i più piccoli.
La seconda soluzione è un **codice** più "complesso", una **forma raccolta** (**Figura 3**): le **palline ci comunicano le istruzioni di colore e quantità di caselle da colorare nella griglia**.

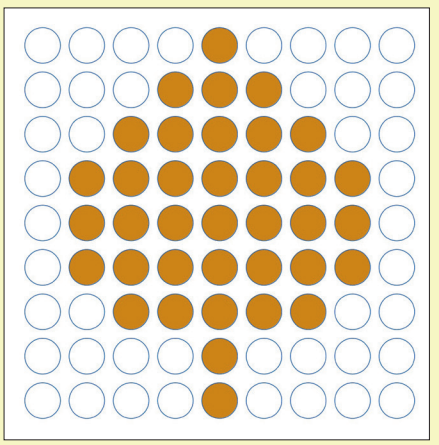


Figura 2

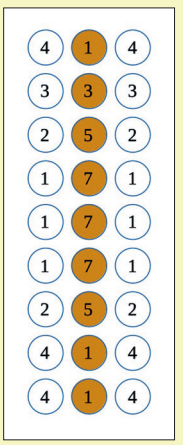


Figura 3

Step 1.2 • Linguaggio testuale

Un'altra soluzione, che può essere considerata un ulteriore incremento rispetto alle precedenti, è un **codice testuale** (**Figura 4**): al posto delle palline, il codice è composto da **una sequenza di istruzioni costituite da una coppia numero e lettera**; il primo indicherà quante caselle dobbiamo colorare e la seconda invece il colore da utilizzare.
Per questa soluzione sarà importante consegnare agli alunni una **legenda** che li aiuti a **interpretare correttamente la lettera del colore**.
Di solito proposte simili sono disponibili anche in rete.

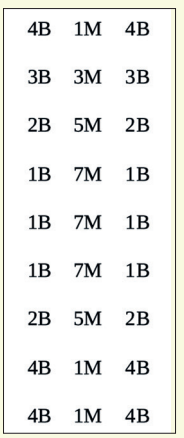


Figura 4

SUGGERIMENTI

Prendiamo in considerazione le tre soluzioni, soprattutto per i più piccoli: si tratta di **progressive implementazioni di un unico linguaggio procedurale**.
Quando iniziamo queste attività, proponiamo **Pixelart semplici con pochi colori**, anche uno solo, e su griglie ridotte. Man mano che le attività proseguono, potremo aumentare la dimensione della griglia e i colori.

STEP 2 • INVERSIONE

Come dicevo all'inizio del tutorial, non possiamo fermarci alla semplice consegna di un codice da seguire, non è coding. Possiamo però introdurre un'attività che in informatica si chiama **ingegnerizzazione inversa**: nel nostro caso partiamo da un disegno finito e ricaviamo il codice (**Figura 5**).
Non consegniamo nessun codice, ma una Pixelart finita e invitiamo gli alunni a **scrivere le istruzioni** necessarie per poi disegnarla. Questo permetterà di **"metabolizzare" il linguaggio procedurale**, di farlo loro e di scontrarsi già con una prima difficoltà: la **codifica**.
Attenzione: in questo caso dovremo indicare **quale tipologia di linguaggio** – visuale o testuale – utilizzare.

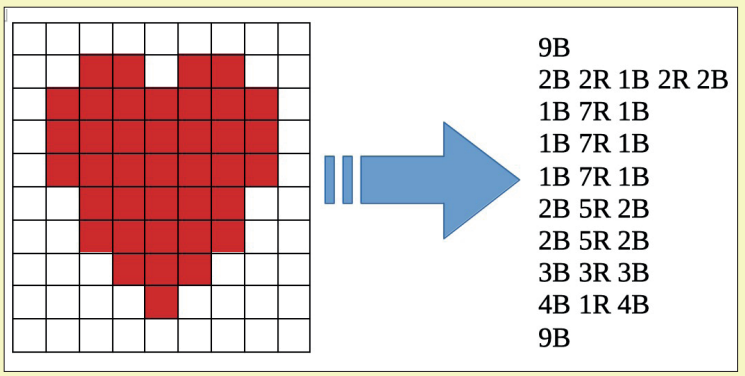


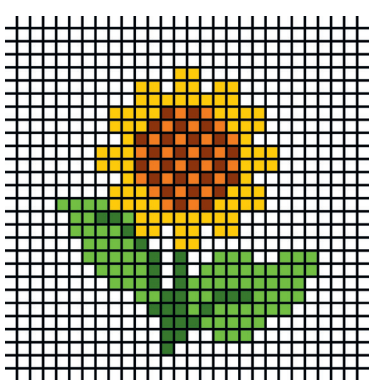
Figura 5

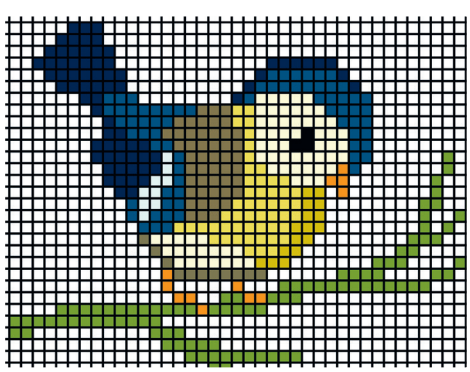
STEP 3 • IL DEBUG

A questo punto, le cose iniziano a farsi molto interessanti. Il passo successivo è naturalmente quello di **provare il codice scritto dagli alunni**, con l'avvertenza che **il test deve essere fatto da una persona diversa dal "programmatore"**, cioè colui o colei che l'ha scritto.
Questa attività è tipica dell'ambito dell'informatica e si chiama **debug**.
Si tratta di una fase molto importante nello sviluppo di un programma: chi ha scritto il codice lo consegna a un compagno o a una compagna e gli/le chiede di ricostruire il disegno. **Se il risultato finale sarà identico al disegno originale vuol dire che la codifica è perfetta**, altrimenti dovremo andare a cercare gli errori, e da questi si impara.

STEP 4 • PROGRAMMIAMO

Dopo aver appreso un linguaggio che ci permette di codificare delle informazioni da cui ricavare un disegno su una griglia, dobbiamo mettere a frutto queste conoscenze creando dei disegni personali: lasciamo liberi gli alunni di disegnare ciò che vogliono, dovranno produrre sia la Pixelart sia il codice. Realizzano un disegno, scrivono le istruzioni e poi consegnano il proprio codice a un compagno o a una compagna.
Naturalmente possiamo utilizzare i modelli realizzati per lo STEP 2 e lo STEP 3.





STEP 5 • LE RIPETIZIONI

Possiamo provare ad arricchire le istruzioni del codice con **alcuni simboli per inserire l'elemento della ripetizione**, in modo da avvicinarci ancora di più al coding vero e proprio.
Vi propongo anche qui un paio di soluzioni basate sulle versioni di codifica illustrate nello STEP 1.

Step 5.1 • Linguaggio visuale

Possiamo utilizzare un simbolo a inizio della riga, una freccia con un numero, che ci indicherà che quella riga andrà ripetuta *n* volte (**Figure 6-7**).

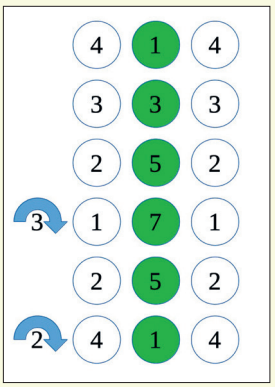


Figura 6

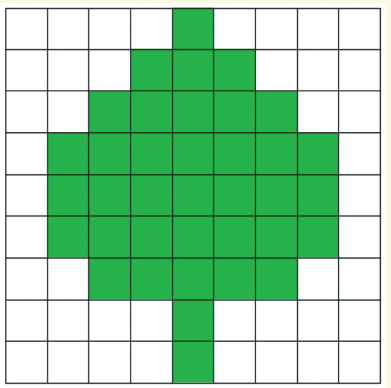


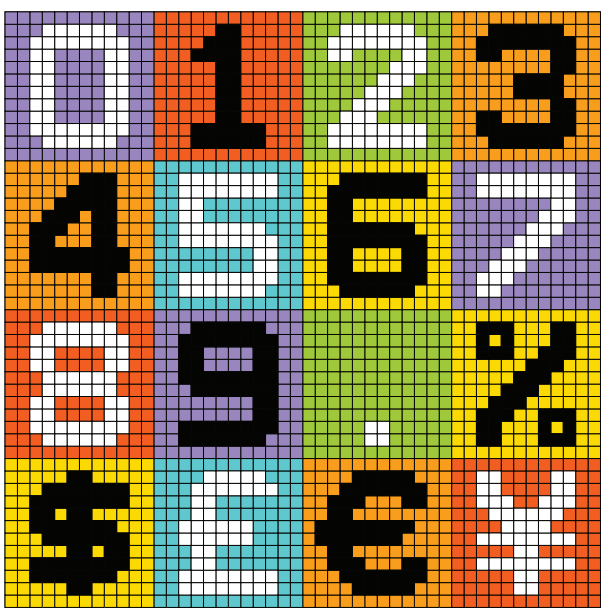
Figura 7

Step 5.2 • Linguaggio testuale

Nella versione testuale, invece, potremmo prendere spunto dai veri linguaggi di programmazione e inserire l'istruzione **Rip** con un numero prima della riga da ripetere, racchiudendo questa tra parentesi.
Osserviamo la **Figura 8**: troviamo i comandi **Rip 3 (...)** / **Rip 2 (...)** che ci indicano che le istruzioni contenute tra parentesi dovranno essere ripetute 3 o 2 volte.

```
      4B  1V  4B
      3B  3V  3B
      2B  5V  2B
Rip 3 (1B  7V  1B)
      2B  5V  2B
Rip 2 (4B  1V  4B)
```

Figura 8



CONCLUSIONI

In questo tutorial abbiamo cercato di andare oltre alla classica Pixelart e di introdurre degli elementi di coding.
Nella mia esperienza laboratoriale di aula ho notato che questi passaggi ci permettono di usare poi altri strumenti di coding con più facilità.
Naturalmente **le attività di coding si possono fare sia in modalità unplugged**, senza uso di un computer, **sia in modalità digitale**, con diversi strumenti disponibili in rete.
A questo proposito vi suggerisco di utilizzare il sito **zaplycode.it** che consente di lavorare molto bene dallo STEP 1 allo STEP 4 utilizzando un codice testuale.
Buon coding!