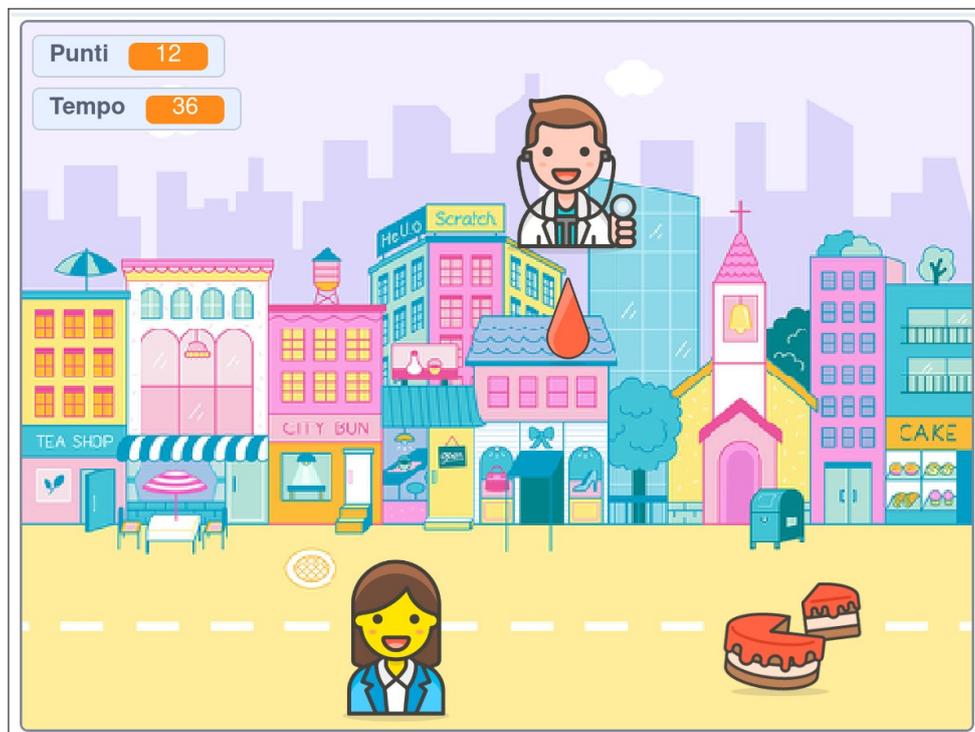


CoderDojo Online Edition

3 ottobre 2021



Il dono

- avviare Scratch e impostare la lingua italiana selezionandola dal menu delle lingue:



- trascinare il blocco `fai (10) passi` nell'area di lavoro – non succede niente:



- click sul blocco: il gattino si sposta verso a destra;
- selezionare il gruppo *Aspetto*;
- trascinare il blocco `dire (Ciao!) per (2) secondi` e agganciarlo sotto al blocco `fai (10) passi` – notare l'area grigia che appare quando i due blocchi sono vicini: evidenzia il punto di aggancio:



- click su uno qualunque dei blocchi: il gattino si muove e saluta;
- sganciare il blocco `dire (Ciao!) per (2) secondi` trascinandolo lontano dal blocco `fai (10) passi`;



- agganciarlo sopra al blocco `fai (10) passi`;
- click su uno qualunque dei blocchi, il gatto saluta e si muove: **l'ordine dei comandi conta!**
- sganciare i due blocchi spostando il blocco `fai (10) passi` (si sgancia sempre da sotto, se si trascina il blocco in alto si ottiene lo spostamento dell'intera catena di blocchi);
- eliminare il blocco `dire (Ciao!) per (2) secondi` trascinandolo nell'area dei blocchi – non importa la posizione precisa;
- facciamo fare una passeggiata al gattino: selezioniamo il gruppo *Controllo* e trasciniamo il blocco `per sempre` nell'area di lavoro; avvicinandolo al blocco `fai (10) passi` si incastra circondandolo; non succede nulla:



- click sul blocco e il gattino si sposta fino al bordo destro dello *Stage*;
- lo *Script* è in esecuzione, lo si riconosce dall'alone giallo che contorna il codice; se trasciniamo il gattino al centro dello *Stage* con il mouse, riprende a spostarsi verso destra;
- per fermare un blocco `per sempre` bisogna premere il pulsante rosso in alto; notare che l'alone giallo sparisce;
- vicino al pulsante di arresto c'è una bandierina verde: serve ad avviare i giochi;
- selezioniamo il gruppo *Situazione* e agganciamo il blocco `quando si clicca su` sopra al blocco `per sempre`;



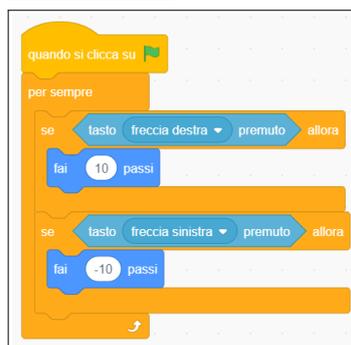
- riportiamo il gattino al centro dello *Stage*, e verificiamo che se ne sta fermo dove l'abbiamo lasciato; premiamo la bandierina ed esso riprende a muoversi;
- premere il pulsante d'arresto e riportare il gattino al centro della scena;
- vogliamo che il gattino si sposti solo quando premiamo il tasto cursore “freccia destra”; accedere al gruppo *Controllo* e portare un blocco `se <> allora` nell'area di lavoro, quindi spostare il blocco `fai (10) passi` al suo interno:



- notare lo spazio vuoto esagonale tra il *se* e l'*allora*: lì va indicata la condizione per la quale devono essere eseguiti i blocchi interni; le varie condizioni disponibili si trovano nel gruppo *Sensori*;
- selezionare il gruppo *Sensori* e trascinare il blocco `<tasto [spazio] premuto>` nello spazio, verificando che l'agganciamento sia avvenuto nel modo corretto (assicurarsi che il perimetro dell'esagono si evidenzi di bianco prima di rilasciare il blocco); aprire il menu a tendina e selezionare il tasto *freccia destra*;
- inserire il blocco `se <> allora` nel programma principale:



- premere la bandierina verde; il gattino resta fermo; premere il tasto cursore a destra, ed esso comincia a muoversi verso destra! fare alcune prove, al termine premere il pulsante d'arresto;
- in modo analogo si prepari un blocco `se <> allora` per lo spostamento verso sinistra:



- notare che per far muovere il gattino a sinistra occorre specificare una quantità di passi negativa.

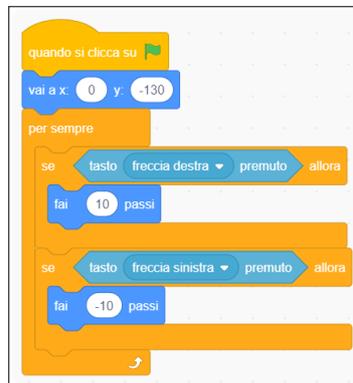
- verificare che ora il controllo sul gattino è completo; provare a modificare i parametri dei comandi: che effetto hanno sulla mobilità del personaggio?
- poiché il personaggio principale del nostro gioco è un donatore di sangue, carichiamo lo sprite **donatore.sprite3**: click sul pulsante *Scegli uno Sprite* in basso a destra, quindi *Importa Sprite*:



- il nuovo personaggio appare al centro della scena mentre la sua icona si aggiunge a quella di Scratch nell'area degli sprite:



- il programma che avevamo scritto è sparito! Niente paura, ogni sprite ha il suo programma: selezionare *Sprite1* ed ecco che il programma di prima ricompare; trasciniamolo sopra l'icona del donatore in modo da copiarlo (attendere che l'icona si evidenzi di blu prima di rilasciare il codice); selezionare lo sprite del donatore e verificare che il programma è stato copiato; a questo punto si può cancellare lo sprite di Scratch selezionandolo nell'area degli sprite e cliccando l'icona a forma di cestino;
- posizioniamo il personaggio principale in basso al centro all'inizio della partita con un blocco `vai a x: () y: ()`:



- lo sprite del donatore ha due costumi: chi desidera può cambiare l'aspetto con la figura maschile.
- è ora di salvare il gioco!

L'obiettivo

- scopo del gioco è far giungere al dottore quanto più sangue possibile senza sprecarlo; carichiamo allora lo sprite **dottore.sprite3** che rappresenta il destinatario del nostro prezioso sangue:



- all'inizio del gioco il dottore deve apparire in una posizione a caso, comunque sempre al di sopra del donatore; con un po' di pazienza si determinano gli estremi della porzione di schermo a lui dedicata:

```

quando si clicca su
per sempre
vai a x: numero a caso tra -190 e 190 y: numero a caso tra -20 e 120
    
```

- durante la partita il dottore appare per un attimo in una posizione dopodiché sparisce, per ricomparire da un'altra parte; per fare questo si sfruttano i blocchi `nascondi`, `mostra` e `attendi (1) secondi` esteso con l'operatore `numero a caso tra () e ()`:

```

quando si clicca su
per sempre
vai a x: numero a caso tra -190 e 190 y: numero a caso tra -20 e 120
mostra
attendi numero a caso tra 2 e 3 secondi
nascondi
attendi numero a caso tra 3 e 5 secondi
    
```

- poiché anche lo sprite del dottore contiene due costumi, facciamo in modo che ogni volta che si mostra cambi aspetto; serve il blocco `passa al costume seguente`:

```

quando si clicca su
per sempre
vai a x: numero a caso tra -190 e 190 y: numero a caso tra -20 e 120
passa al costume seguente
mostra
attendi numero a caso tra 2 e 3 secondi
nascondi
attendi numero a caso tra 3 e 5 secondi
    
```

La goccia di sangue

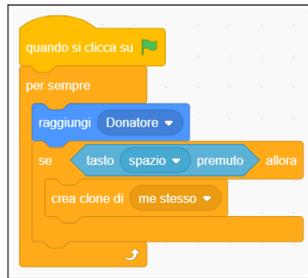
- aggiungiamo il terzo sprite, **goccia di sangue.sprite3**:



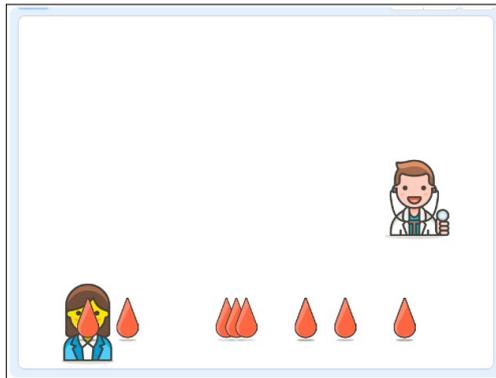
- facciamo in modo che la goccia di sangue segua sempre il donatore con il blocco `raggiungi (puntatore del mouse)` (ricordarsi di selezionare il nome dello sprite del donatore!); al momento giusto, quando il giocatore premerà la barra spaziatrice, la goccia si muoverà verso l'alto nella speranza di raggiungere il dottore:



- verificare che quando si muove il donatore con i tasti cursore la goccia segue i movimenti dello sprite;
- per dare la possibilità al giocatore di inviare più gocce di sangue al dottore creiamo dei cloni: saranno questi a muoversi, mentre lo sprite originale rimarrà sul donatore per originare sempre nuovi cloni; il blocco per creare un clone è `crea clone di (me stesso)`:



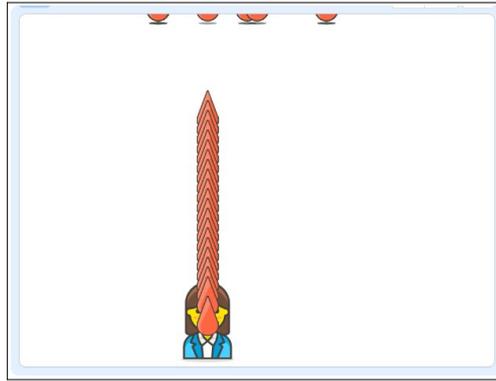
- verificare che ogni volta che si preme la barra spaziatrice una nuova goccia di sangue viene depositata nella stessa posizione in cui si trova il donatore; spostare lo sprite e ripetere:



- è ora di far muovere i cloni verso l'alto; **ATTENZIONE!** il programma eseguito dai cloni va predisposto sotto il cappello `quando vengo clonato`:



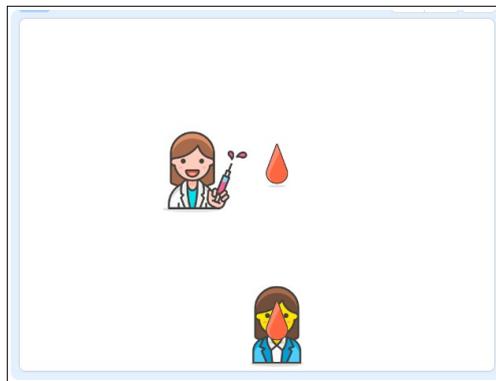
- il risultato che si ottiene purtroppo non è ottimale:



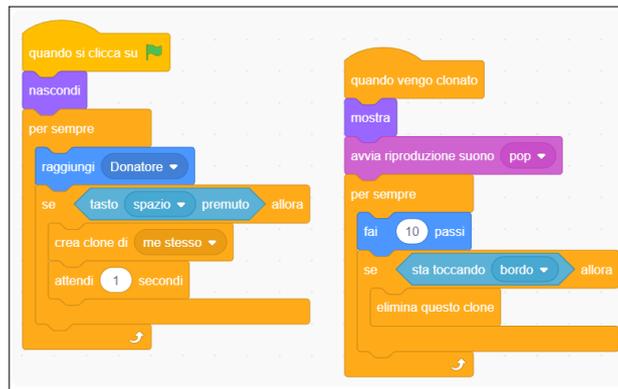
- le gocce rimangono appiccicate al soffitto e, se si mantiene premuta la barra spaziatrice, si formano delle orrende colonne di gocce sovrapposte; questo secondo aspetto si risolve inserendo una pausa dopo la creazione di un clone; la pausa si ottiene al solito con `attendi (1) secondi`, per cancellare gli sprite clonati si usa invece il blocco `elimina questo clone`:



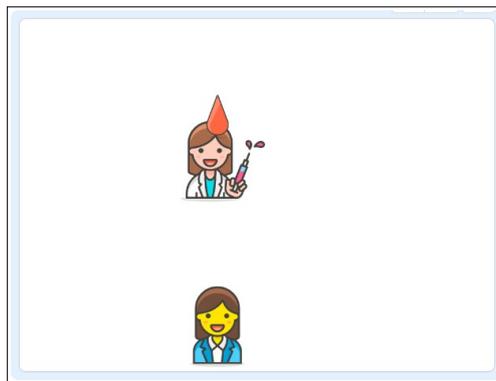
- resta un ultimo particolare da sistemare, quella antiestetica goccia di sangue che copre il donatore:



- nascondiamo allora lo sprite principale all'inizio del gioco con un blocco `nascondi`; questo però fa sì che anche i cloni nascano invisibili (verificare!) ed è dunque necessario utilizzare un blocco `mostra` per renderli visibili; ne approfittiamo per aggiungere un effetto sonoro alla creazione di un clone con il blocco `avvia riproduzione suono (pop)`:



- ora rimane un ultimo particolare da sistemare: quando la goccia raggiunge lo sprite del dottore deve sparire così il giocatore ha l'impressione che è stata raccolta e non è andata sprecata:



- usiamo un blocco `se <> allora` per cancellare il clone quando questo collide con lo sprite del dottore; aggiungiamo un effetto sonoro che sottolinea il fatto che la goccia ha raggiunto l'obiettivo:



Il punteggio

- è ora di aggiungere il punteggio: contiamo quante sono le gocce che sono state recapitate al dottore... selezionare *Variabili*, quindi *Crea una Variabile*, chiamarla "Punti" e confermare premendo *OK*; cosa succede?
- abbiamo detto che facciamo un punto ogni volta che una goccia di sangue raggiunge il dottore; selezioniamo lo sprite della goccia di sangue ed aggiungiamo il blocco `cambia [my variable] di (1)` all'interno del corpo del blocco `se <sta toccando (Dottore)> allora`, avendo l'accortezza di sostituire la variabile "my variable" con la variabile "Punti":



- opzionalmente possiamo penalizzare il giocatore quando spreca una goccia di sangue: è sufficiente diminuire il punteggio quando un clone raggiunge il bordo superiore, perché se arriva fin lassù vuol dire che ha mancato il dottore:



- provare il gioco e verificare che il punteggio cambia secondo le regole stabilite;
- notate un problema? quando il gioco riparte, il punteggio non si azzera! la soluzione è aggiungere il blocco `porta` `[Punti] a [0]` all'inizio del programma dello sprite della goccia di sangue:

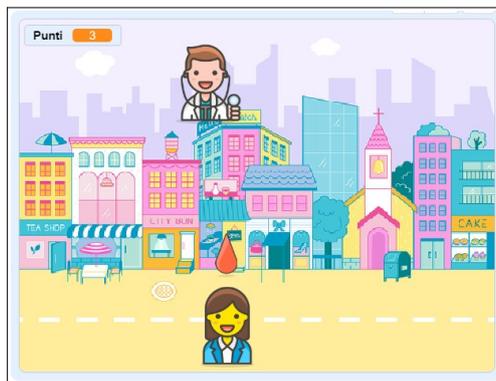


Lo sfondo

- aggiungiamo uno sfondo per dare un'ambientazione consona al gioco; premere il pulsante *Scegli uno Sfondo* in basso a destra e scegliere l'immagine preferita (es. **Colorful City**):



- l'aspetto del gioco ne guadagna assai:



- ora si può anche cancellare lo sfondo bianco iniziale, basta fare click sull'icona del cestino;
- si può giocare a schermo intero premendo il pulsante *Modalità presentazione* in alto a destra; premere il riquadro nella stessa posizione per tornare alla modalità di lavoro.

L'antagonista

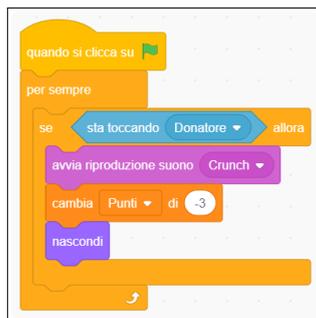
- rendiamo il gioco più difficile: poiché per poter donare bisogna essere in buona salute, il donatore deve evitare vizi ed eccessi; se fuma o esagera con i dolci o altri cibi molto calorici perde un punto;
- carichiamo l'ultimo sprite, **rischio.sprite3** e posizioniamolo in basso sullo *Stage* accanto al donatore:



- lo sprite dovrà apparire saltuariamente sulla stessa linea del donatore; conviene partire dal programma del dottore che copiamo selezionando lo sprite **Dottore** e trascinando il programma sullo sprite **Rischio**; lo si adatta al nuovo sprite apportando le seguenti modifiche:
 - introdurre un blocco `nascondi` e un blocco `attendi (10) secondi` prima del ciclo principale per far apparire lo sprite a gioco inoltrato (a piacere si può aumentare il ritardo a 15 o 20 secondi);
 - per far apparire lo sprite sulla stessa linea del donatore modificare il blocco `vai a x: () y: ()` impostando la coordinata y al valore fisso -130 (la stessa del donatore);
 - ridurre il tempo di apparizione dello sprite a un secondo eliminando il blocco `numero a caso tra () e ()` dal primo blocco `attendi () secondi`:



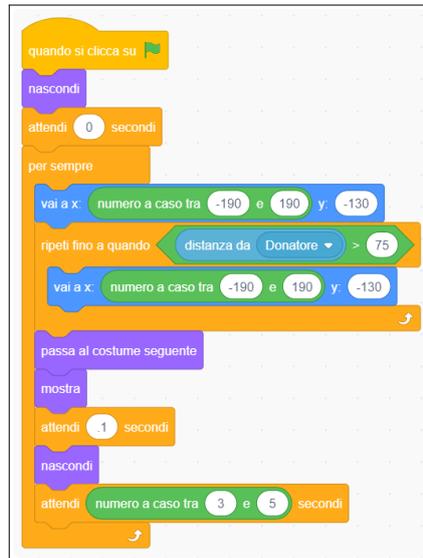
- aggiungere un nuovo programma nello sprite **Rischio** che si occupa della gestione della collisione con il donatore; quando i due sprite entrano in contatto si emette un suono e si applica una penalità di 3 punti (questo dimostra che uno sprite può eseguire più programmi contemporaneamente!):



- il blocco `nascondi` fa in modo che la collisione non si protragga nel tempo causando il precipitare del punteggio.

Ottimizzazione

- può accadere che lo sprite **Rischio** appaia proprio sopra il donatore, causando inevitabilmente una penalità immeritata; conviene invece riposizionare lo sprite in modo che si trovi ad una distanza ritenuta di sicurezza dal donatore; aggiungere allora un blocco `ripeti fino a quando <>` con la condizione `<(distanza da (Donatore)) > (75)>` che effettua lo spostamento dello sprite solo se necessario:



Partita a tempo

- facciamo in modo che il gioco abbia una durata predefinita... possiamo usare una nuova variabile, la impostiamo ad un valore iniziale che viene decrementato ogni secondo e, quando questa raggiunge lo zero, interrompiamo il gioco con il blocco `ferma [tutto]` (oppure la si incrementa a partire da zero e si interrompe il gioco quando raggiunge un valore predefinito);
- questo script può essere assegnato a uno sprite qualunque, anche allo sfondo!
- creare una nuova variabile "Tempo", quindi:



Fine partita

- prepariamo un fine partita più coreografico: allo scadere del tempo tutti gli sprite ad esclusione del donatore spariscono, appare la scritta "GAME OVER" e una musica avvisa che il gioco è terminato;
- la schermata finale si ottiene duplicando lo sfondo **Colorful City**, rinominando il nuovo sfondo in **Game Over**; questo sfondo andrà selezionato allo scadere del tempo:



- ricordarsi di impostare lo sfondo **Colorful City** all'inizio della partita, altrimenti ...
- come fermare e nascondere tutti gli sprite prima che inizi a suonare la musica del finale di partita? lo si può fare con inviando un messaggio a tutti gli sprite interessati:



- gli sprite **Dottore**, **Goccia di sangue** e **Rischio** si nascondono e fermano il programma quando ricevono il messaggio **Game over**:



- lo sprite del donatore invece resta visibile anche a fine partita:



Altri spunti

- è accettabile un punteggio negativo? come si potrebbe fare per evitare questa condizione?
- aggiungere una schermata iniziale di presentazione del gioco con le istruzioni;
- far terminare il gioco dopo tre collisioni con lo sprite **Rischio** anziché (o non solo) allo scadere del tempo;
- aggiungere una colonna sonora durante la partita;
- ...