Những kỹ năng lập trình với Scratch

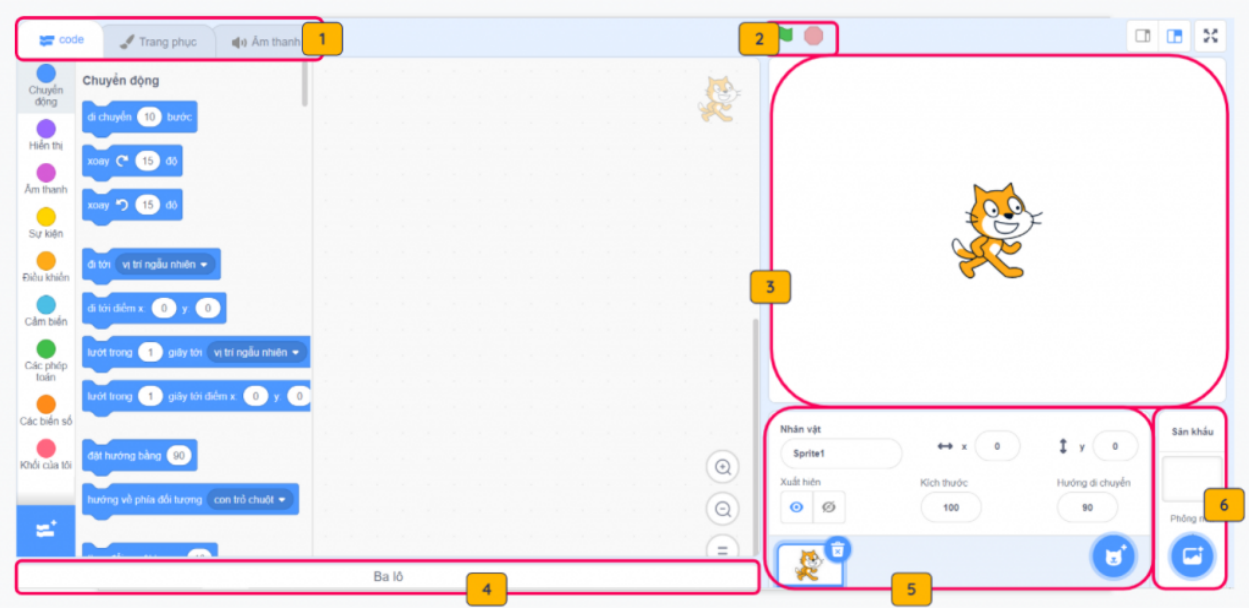
Mục Lục

1. Giới thiệu chung về giao diện và khối lệnh trong Scratch: 3
1.1. Bố cục màn hình của Scratch: 3
1.2. Màn hình code của nhân vật: 3
1.3. Chi tiết về từng loại khối lệnh: 4
1.4. Trang phục của nhân vật: 6
1.5. Âm thanh của nhân vật: 7
2. Những thói quen nên có trong lập trình: 9
2.1. Tạo thói quen bắt đầu một chương trình bằng cách kéo khối Lá cờ xanh vào: 9
2.2. Thói quen "Chia trang vở làm 2": 9
2.3. Kiểm tra các khối lệnh cơ bản cho nhân vật: 10
2.4. Đặt tên rõ ràng cho Nhân vật (Sprite), Trang phục (Costume), Ảnh nền (Background): 10
2.5. Kiểm tra và chỉnh sửa ngay: 10
2.6. Làm chương trình tinh gọn hơn: 10
2.7. Ghi các chú thích: 11
2.8. Học hỏi từ các nguồn khác: 11
3. Tạo nhân vật trong Scratch: 12
3.1. Chọn nhân vật có sẵn trong thư viện: 12
3.2. Vẽ một nhân vật: 14
3.3. Chọn một nhân vật ngẫu nhiên: 15
3.4. Tải nhân vật từ máy tính: 16
3.5. Các thông số của nhân vật: 17
4. Thứ tự thực hiện các khối lệnh: 19
4.1. Thực hiện tuần tự: 19
4.2. Thực hiện cùng lúc: 20
4.3. Kết hợp cả hai để phân tích một chương trình: 21
5. Các khối lệnh với âm thanh: 24
5.1. Phát âm thanh: 24
5.2. Ngừng mọi âm thanh: 24
5.3. Thay đổi âm lượng: 25
6. Cách viết vòng lặp: 26
6.1. Tại sao phải sử dụng vòng lặp trong Scratch: 26
6.2. Các loại vòng lặp trong Scratch: 26
7. Hướng di chuyển và Di chuyển: 29
7.1. Di chuyển: 29
7.2. Xoay: 29
7.3. Giới thiệu tọa độ: 30
7.4. Bộ câu lệnh đi tới: 33
7.5. Bộ câu lệnh lướt: 34

7.6. Bộ câu lệnh hướng: 35
7.7. Câu lệnh bật lại khi tiếp xúc với cạnh: 36
7.8. Câu lệnh đặt kiểu xoay: 37
7.9. Thao tác trực tiếp lên tọa độ: 38
8. Câu lệnh Nếu... Thì... 40
9. Câu lệnh "Đang chạm ...?" và "Phím .... được bấm?": 43
9.1. Câu lệnh "Đang chạm ...?": 43
9.2. Câu lệnh "Phím ... được bấm?": 45
10. Khối của tôi: 47
10.1. Cách tạo Khối của tôi 47
10.2. Cách sử dụng Khối của tôi 49
11. Bút vẽ: 51
11.1. Cách thiết lập hiển thị các khối bút vẽ trong Scratch: 51
11.2. Chức năng của các khối lệnh trong "Bút vẽ": 52
12. Khối lệnh "Phát tin" và "Nhận tin": 56
12.1. Khối lệnh "Phát tin": 56
12.2. Khối "Nhận tin": 57
12.3. Phân biệt khối "Phát tin" và "Phát tin và đợi": 58

1. Giới thiệu chung về giao diện và khối lệnh trong Scratch:

1.1. Bố cục màn hình của Scratch:



(1) Chuyển giữa **code**, **trang phục** và **âm thanh** của **nhân vật đang chọn.**

(2) Nhấn l**á cờ màu xanh** để **bắt đầu trò chơi**, nhấn **nút màu đỏ** để **dừng trò chơi.**

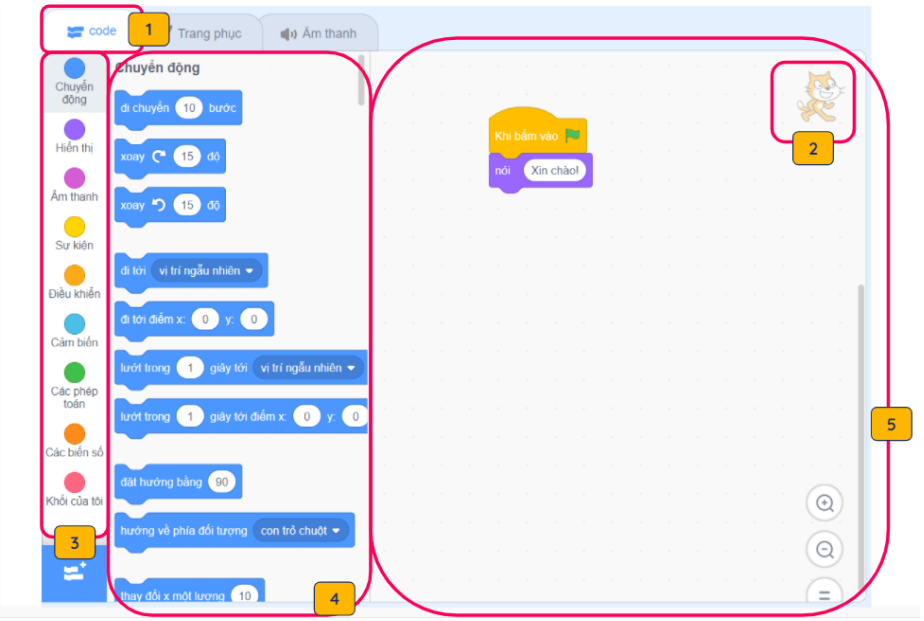
(3) Màn hình trò chơi / Sân khấu.

(4) Balo có thể dùng để **sử dụng lại các khối lệnh**, **nhân vật** hoặc **trang phục.**

(5) Thông tin nhân vật đang chọn.

(6) Nhấn vào để **lập trình phông nền.**

1.2. Màn hình code của nhân vật:



(1) Nhấn vào nút **Code** để **xem code của nhân vật.**

(2) Hình ảnh của **nhân vật đang được lập trình.**

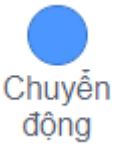
(3) Danh sách các loại khối lệnh, khi nhấn vào 1 loại thì các khối lệnh thuộc loại đó sẽ được liệt kê trong phần danh sách các khối lệnh.

(4) Danh sách các khối lệnh.

(5) Khu vực lập trình.

1.3. Chi tiết về từng loại khối lệnh:

a. Nhóm khối lệnh chuyển động:



- **Có thể dùng để lập trình nhân vật:**

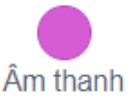
- Đi thẳng
- Xoay trái, phải
- Thay đổi tọa độ của nhân vật

b. Nhóm khối lệnh hiển thị:

- **Có thể dùng để cho nhân vật:**

- Ẩn/hiện
- Cho nhân vật nói hoặc nghĩ một câu nào đó
- Thay đổi trang phục cho nhân vật / Thay đổi phông nền cho trò chơi
- Thay đổi kích thước nhân vật (kích thước nhân vật được tính theo đơn vị phần trăm – %)
- Thay đổi màu sắc nhân vật
- Thay đổi vị trí trước sau của các nhân vật trên màn hình.

c. Nhóm khối lệnh âm thanh:



- Có thể dùng để **phát âm thanh**, trong đó có **2 khối lệnh dễ nhầm lẫn** với nhau là **phát âm thanh đến hết** và **bắt đầu âm thanh**.
- Trong trường hợp có một hoặc nhiều câu lệnh phía sau khối lệnh phát âm thanh cho đến hết và bắt đầu âm thanh thì chúng sẽ thể hiện sự khác biệt. Cụ thể,
    - Nếu chúng **đặt sau câu lệnh phát âm thanh cho đến hết**, chúng sẽ được **thực hiện sau khi toàn bộ âm thanh được phát sóng**.
    - Nếu chúng **đặt sau câu lệnh bắt đầu âm thanh**, chúng sẽ được **thực hiện ngay sau khi bắt đầu phát âm thanh**, không quan trọng khi nào âm thanh đó kết thúc.

d. Nhóm khối lệnh sự kiện:

- Có chứa một khối lệnh hay được sử dụng là khối **"Khi bấm vào lá cờ xanh"**.
- Những khối lệnh trong nhóm này cũng được dùng để **phát tin hoặc nhận tin.**

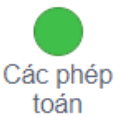
e. Nhóm khối lệnh điều khiển:

- Có chứa các câu lệnh tạo ra **vòng lặp** trong chương trình lập trình.
- Ngoài ra nhóm này còn có các khối lệnh giúp **nhân bản nhân vật.**

f. Nhóm khối lệnh cảm biến:

- Những khối lệnh trong nhóm này có thể **nhận ra tín hiệu từ chuột hoặc bàn phím**. Thường được **dùng kết hợp với các khối lệnh trong nhóm khối lệnh điều khiển**.
- Ngoài ra trong nhóm này còn có một số khối lệnh dùng để **đếm giờ** hoặc **đặt câu hỏi** và **nhận câu trả lời** từ người chơi.

g. Nhóm khối lệnh các phép toán:



- Giúp **thực hiện các phép** toán trong Scratch.

h. Nhóm khối lệnh các biến số:

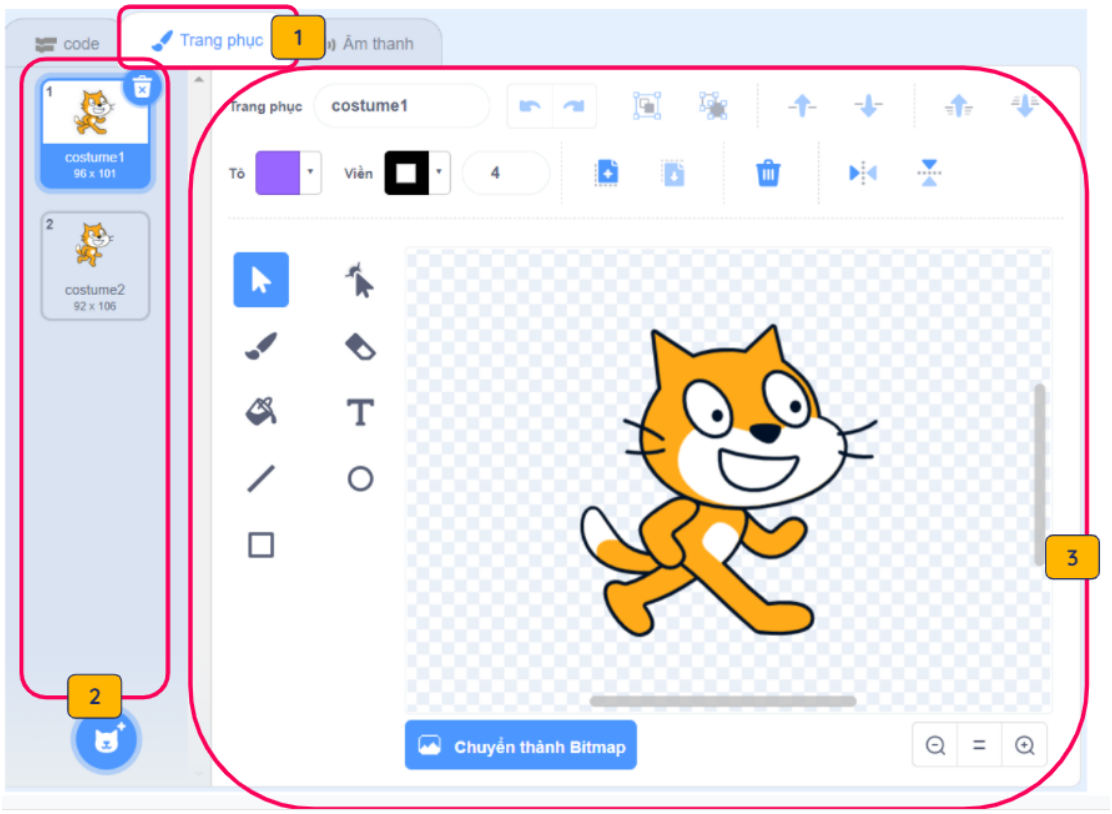
- Bao gồm những chức năng có liên quan đến **biến số và danh sách (list).**

i. Phần Khối của tôi:

- Giúp tạo ra các câu lệnh mới, không có sẵn trong Scratch. Ngoài ra, khối của tôi còn được sử dụng để tạo ra các **Hàm (Function)**, giúp việc lập trình trở nên ngắn gọn và tường minh hơn.

1.4. Trang phục của nhân vật:

- Trang phục có thể được hiểu là **các hình dáng khác nhau của từng nhân vật.**

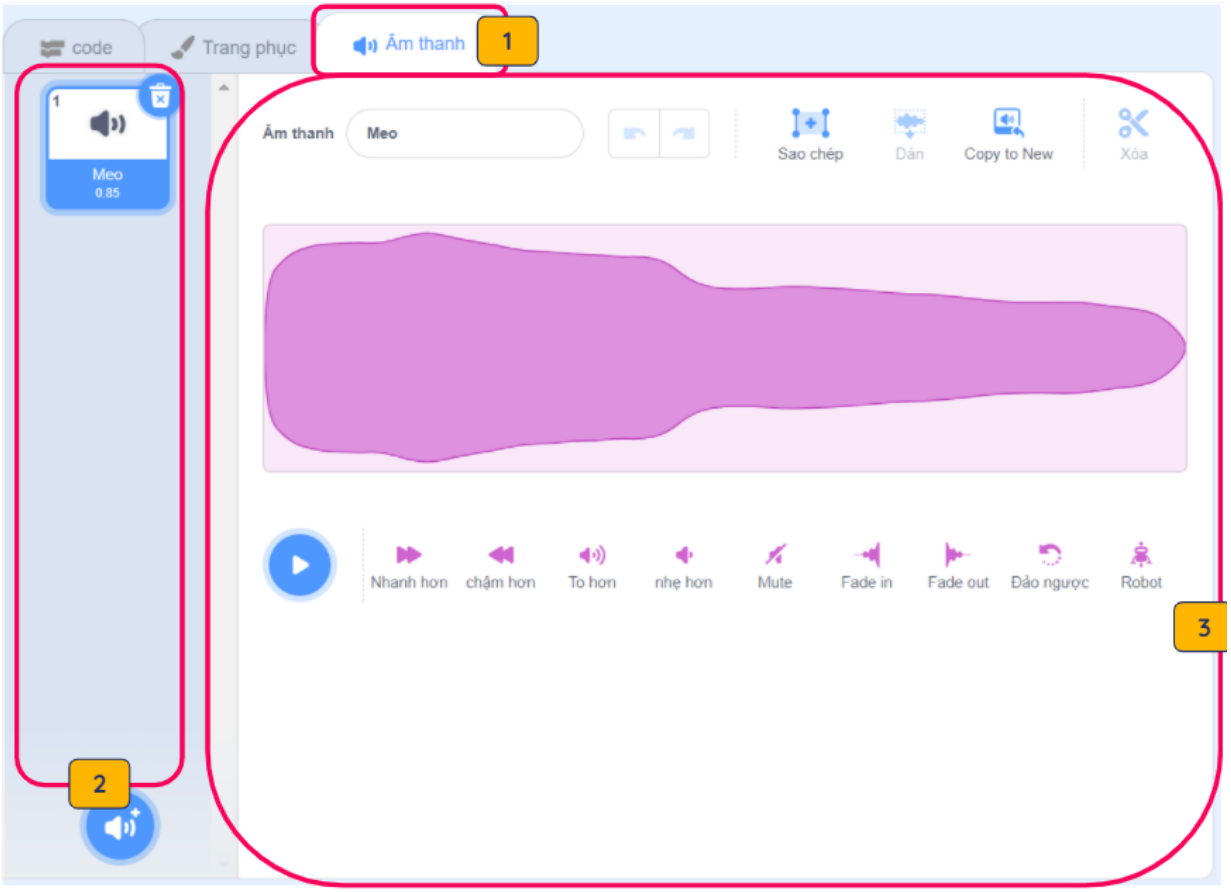


(1) Nhấn vào nút **Trang phục** để **xem các trang phục** của nhân vật.

(2) Danh sách các trang phục của nhân vật. Khi nhấn vào 1 trang phục thì trang phục đó sẽ được hiện ra ở phần chỉnh sửa trang phục (3) nằm bên phải.

(3) Phần chỉnh sửa trang phục.

1.5. Âm thanh của nhân vật:



(1) Nhấn vào nút **Âm thanh** để **xem các âm thanh** của nhân vật

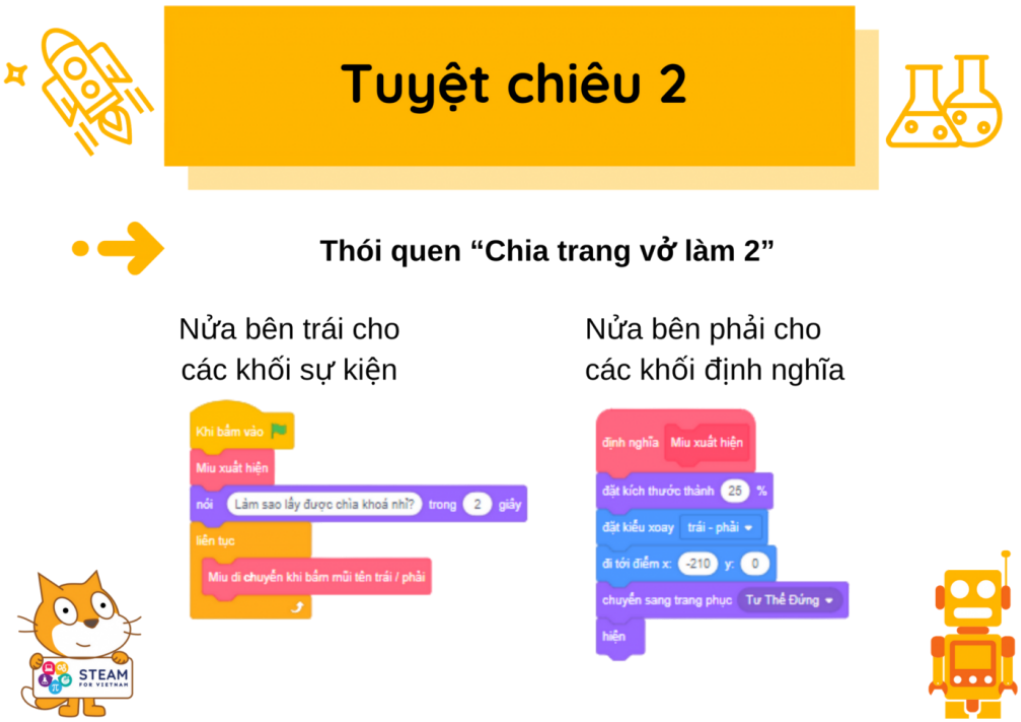
(2) Danh sách các âm thanh của nhân vật. Khi nhấn vào 1 âm thanh thì âm thanh đó sẽ được hiện ra ở phần chỉnh sửa âm thanh (3) nằm bên phải.

(3) Phần chỉnh sửa âm thanh.

2. Những thói quen nên có trong lập trình:

2.1. Tạo thói quen bắt đầu một chương trình bằng cách kéo khối Lá cờ xanh vào:

2.2. Thói quen "Chia trang vở làm 2":



- Nửa **bên trái** cho các **khối sự kiện (hat blocks)**, nửa **bên phải** là cho các **khối định nghĩa functions.**
- Để tất cả các khối sự kiện sang 1 bên (bắt đầu bằng khối Lá cờ) làm cho việc đọc code dễ dàng hơn vì thấy ngay nhân vật/sân khấu sẽ phản ứng với các sự kiện nào.

- Lợi ích nữa của việc này là chỉ cần đọc code từ trên xuống dưới giống hệt như đọc văn.

2.3. Kiểm tra các khối lệnh cơ bản cho nhân vật:

- Thầy/Cô nên kiểm tra để **đảm bảo mỗi nhân vật** trong chương trình **đều nên được định nghĩa xuất hiện với Lá cờ Xanh**, bao gồm vị trí, hướng, trạng thái ẩn/hiện.

2.4. Đặt tên rõ ràng cho Nhân vật (Sprite), Trang phục (Costume), Ảnh nền (Background):

- Khi thầy/cô làm trò chơi **có nhiều nhân vật**, hãy **đặt tên cụ thể** để **dễ dàng nhận dạng nhân vật** trong bài code. Đặt tên luôn là một vấn đề khó nhằn, ngay cả với những kỹ sư lâu năm.

- Khi lập trình Scratch, thầy/cô nên đặt tên **Nhân vật** là **Danh từ**, tên **Hàm (Function)** là **Động từ.** Chẳng hạn, tên **nhân vật "Miu", "Quả bóng", "Mê cung"** sẽ rõ ràng hơn nhiều việc đặt tên **"Nhân vật 1", "Nhân vật 2", "Nhân vật 3"**. Đối với tên Hàm, hãy thử đặt là **"Đi sang trái", "Nói chuyện"** thay vì nói một cách chung chung như **"Hành động X", "Hành động Y", v.v**. Đối với tên hàm, thầy/cô cũng có thể dùng dạng **Danh từ + Động từ**, ví dụ **"Miu xuất hiện", "Chú gấu di chuyển", v.v...**

2.5. Kiểm tra và chỉnh sửa ngay:

- **Luôn kiểm tra** xem **chương trình có hoạt động đúng ý** thầy/cô không **mỗi khi thêm các khối mới vào**, nếu không thầy/cô hãy chỉnh sửa ngay.

2.6. Làm chương trình tinh gọn hơn:

- Thông thường, nếu thầy/cô có thể làm chương trình hoạt động với **ít khối hơn**, thì chương trình **có lẽ sẽ "tốt" hơn** vì các công việc tương tự nhau có thể được gộp vào cùng một Khối.

2.7. Ghi các chú thích:

- Thêm các **Chú thích (comment)** nếu chú thích **làm chương trình** trở nên **dễ hiểu hơn.**

2.8. Học hỏi từ các nguồn khác:

- Học hỏi từ chương trình của các thầy/cô khác; tuy nhiên, cũng nên xem thầy/cô có thể làm chương trình tốt hơn được không (sử dụng ít khối hơn, viết chương trình cho dễ hiểu hơn,...)

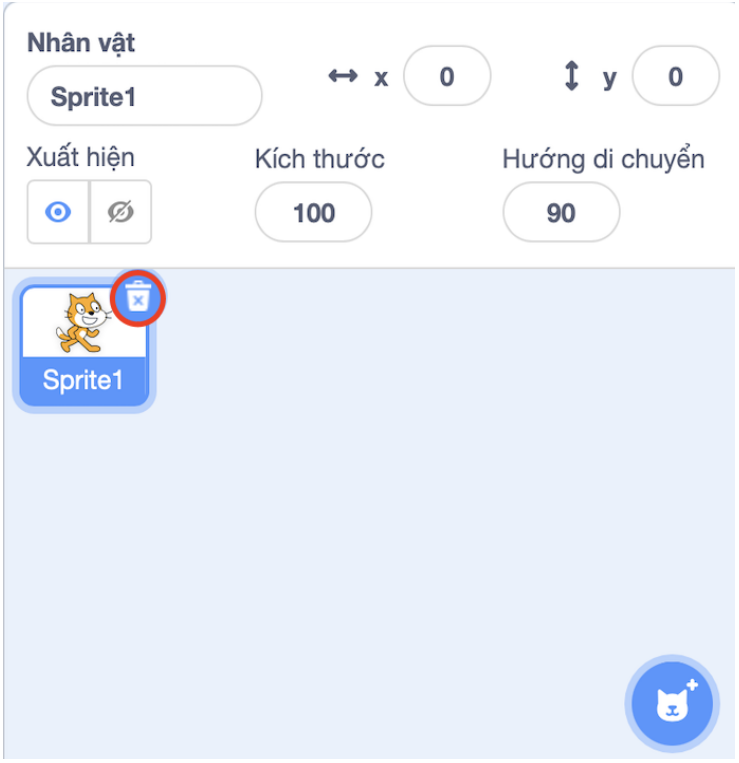
3. Tạo nhân vật trong Scratch:

- **Nhân vật trong Scratch** (còn được gọi là các **Đối tượng / Sprite**) đơn giản chính là những hình ảnh do Scratch cung cấp hoặc thầy/cô tự tạo ra để dựng nên dự án của mình.

- **Các nhân vật** trong Scratch sẽ **hoạt động nhờ những đoạn lệnh được lập trình cho chúng**. Với vô vàn những kết hợp khác nhau, thầy/cô có thể tạo ra những hành động, cử chỉ, tương tác... giữa các nhân vật với nhau hoặc giữa các nhân vật với môi trường xung quanh... tùy theo mong muốn của mình.

Lưu ý:

Khi tạo một dự án mới, nhân vật hình con mèo với tên là Sprite1 sẽ được thêm vào sẵn. Nếu muốn xóa nhân vật này, chúng ta **chọn nút có biểu tượng thùng rác** như hình dưới đây và sau đó thực hiện thêm các nhân vật tùy ý.



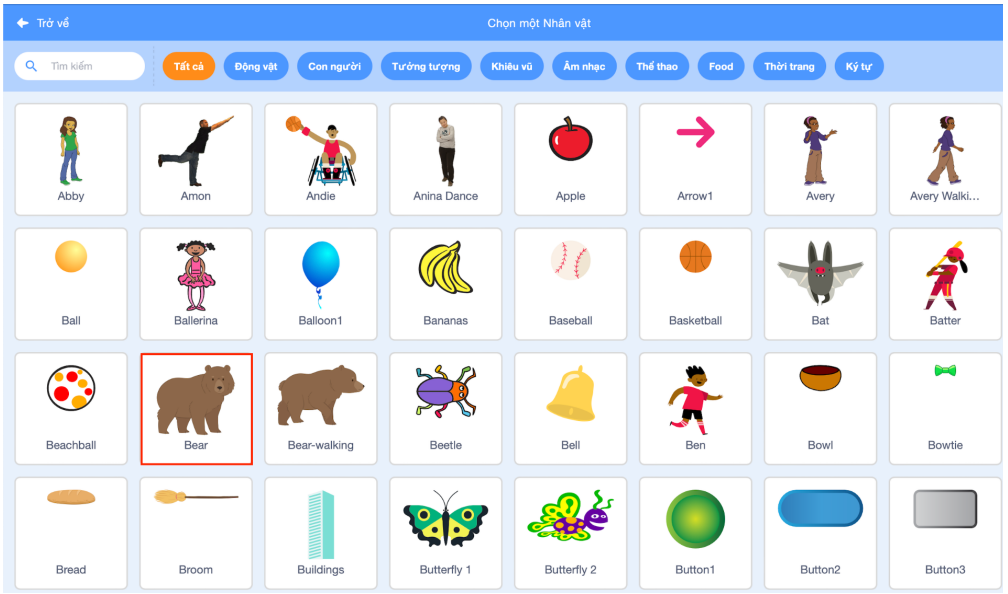
3.1. Chọn nhân vật có sẵn trong thư viện:

a. Bước 1:

- Di chuột đến biểu tượng **"Thêm nhân vật"** rồi nhấn vào nút "**Chọn một Nhân vật"** hoặc chọn **nút có hình kính lúp**  . Đây là nút đưa chúng ta đến thư viện nhân vật của Scratch với nhiều chủ đề khác nhau và ở đó ta có thể chọn một nhân vật tùy ý.

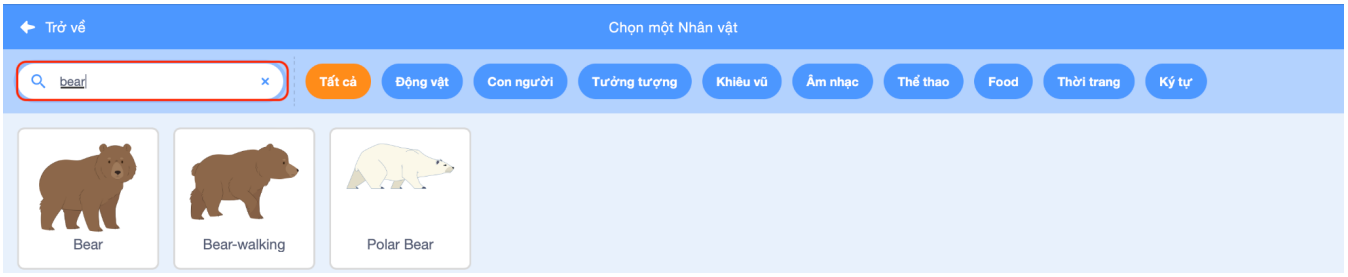
b. Bước 2:

- **Chọn một nhân vật muốn sử dụng.** Trong ví dụ này, chúng ta chọn nhân vật có tên là ***Bear***.

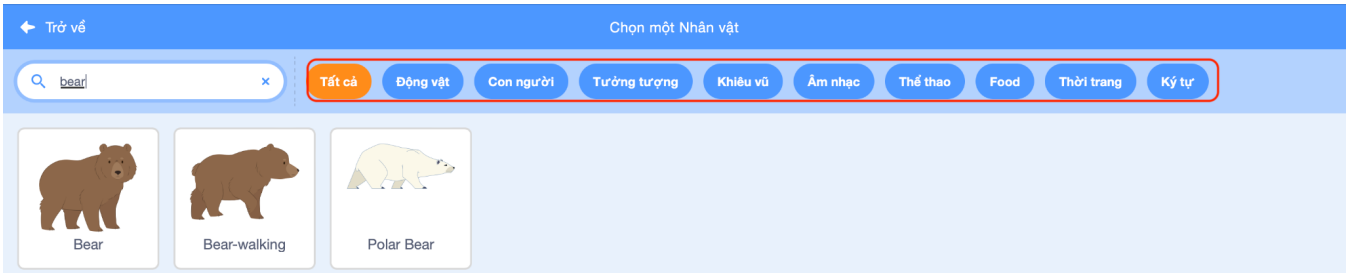


Lưu ý:

- **Ô tìm kiếm:** Có thể **tìm nhanh tên nhân vật** khi gõ vào ô này. Do **tên các nhân vật đều bằng tiếng Anh** nên khi tìm kiếm nhân vật, thầy/cô nên nhập các từ tiếng Anh như trong hình minh hoạ.



- **Các ô chủ đề:** Các nhân vật được phân loại thành nhiều chủ đề khác nhau, thuận tiện cho việc tìm kiếm.



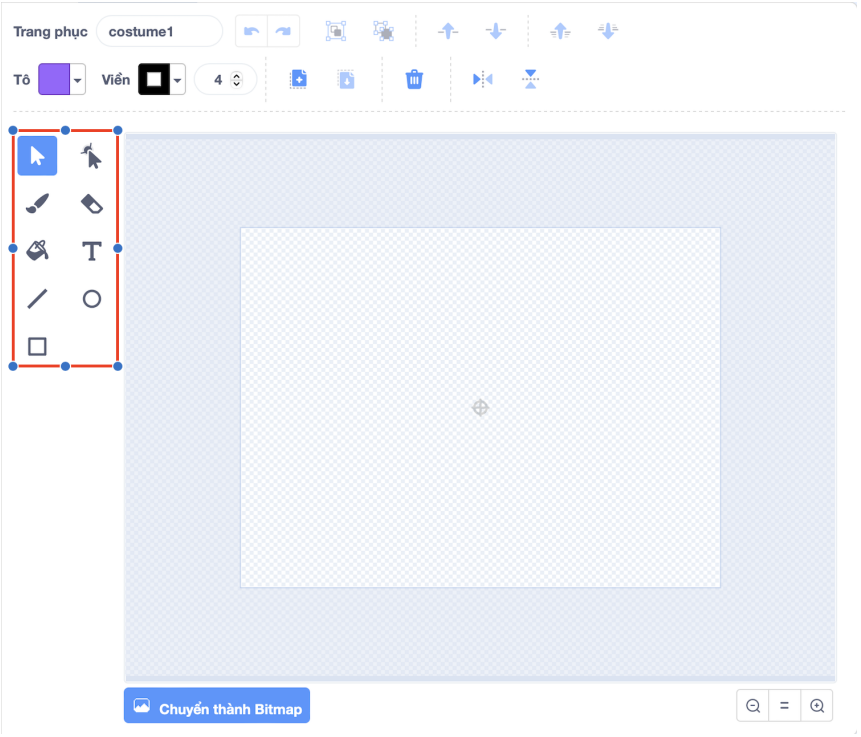
3.2. Vẽ một nhân vật:

a. Bước 1:

- Di chuột đến biểu tượng **"Thêm nhân vật"** và sau đó chọn nút có **hình bút vẽ** . Đây là nút đưa chúng ta đến thuộc tính trang phục của nhân vật và ở đó **có thể tùy ý vẽ các nhân vật với các công cụ vẽ có sẵn.**

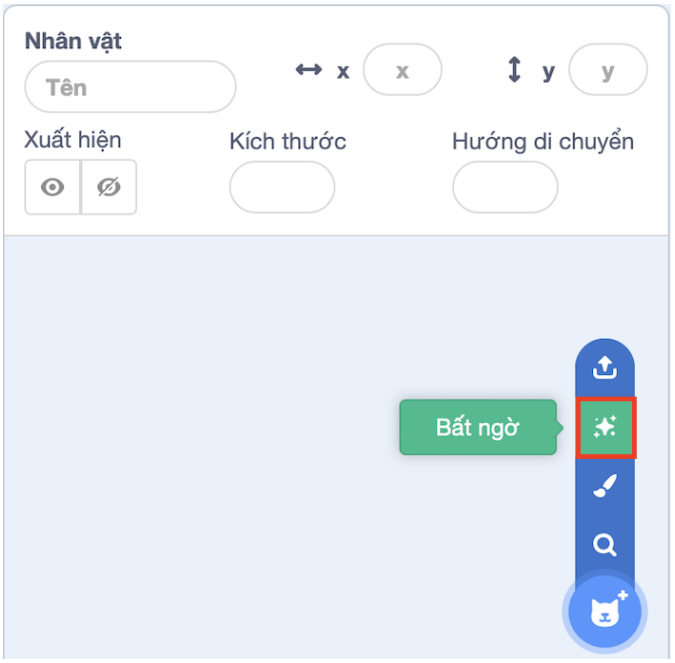
b. Bước 2:

- **Sử dụng các công cụ vẽ có sẵn** như cọ vẽ, tẩy, đổ màu, hình tròn, hình vuông và thậm chí **có thể thêm cả chữ vào phông nền.**



3.3. Chọn một nhân vật ngẫu nhiên:

- Di chuột đến biểu tượng **"Thêm nhân vật"** và sau đó chọn nút **Bất ngờ (có hình ngôi sao)**  . Đây là nút giúp **chương trình tự chọn một nhân vật bất kỳ** có trong thư viện của Scratch. **Mỗi lần ấn vào nút này, một nhân vật ngẫu nhiên sẽ được chọn** để hiển thị trên Sân khấu.



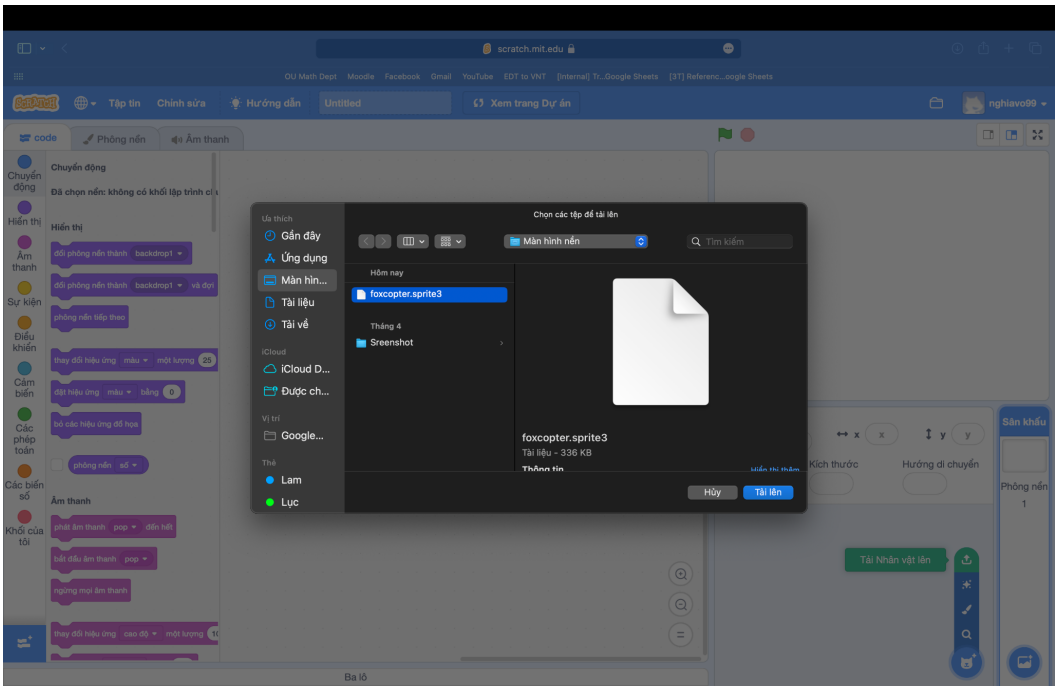
3.4. Tải nhân vật từ máy tính:

a. Bước 1:

- Di chuột đến biểu tượng **"Thêm nhân vật"** và sau đó chọn nút **Tải nhân vật lên** . Sau đó màn hình sẽ hiển thị một hộp thoại.

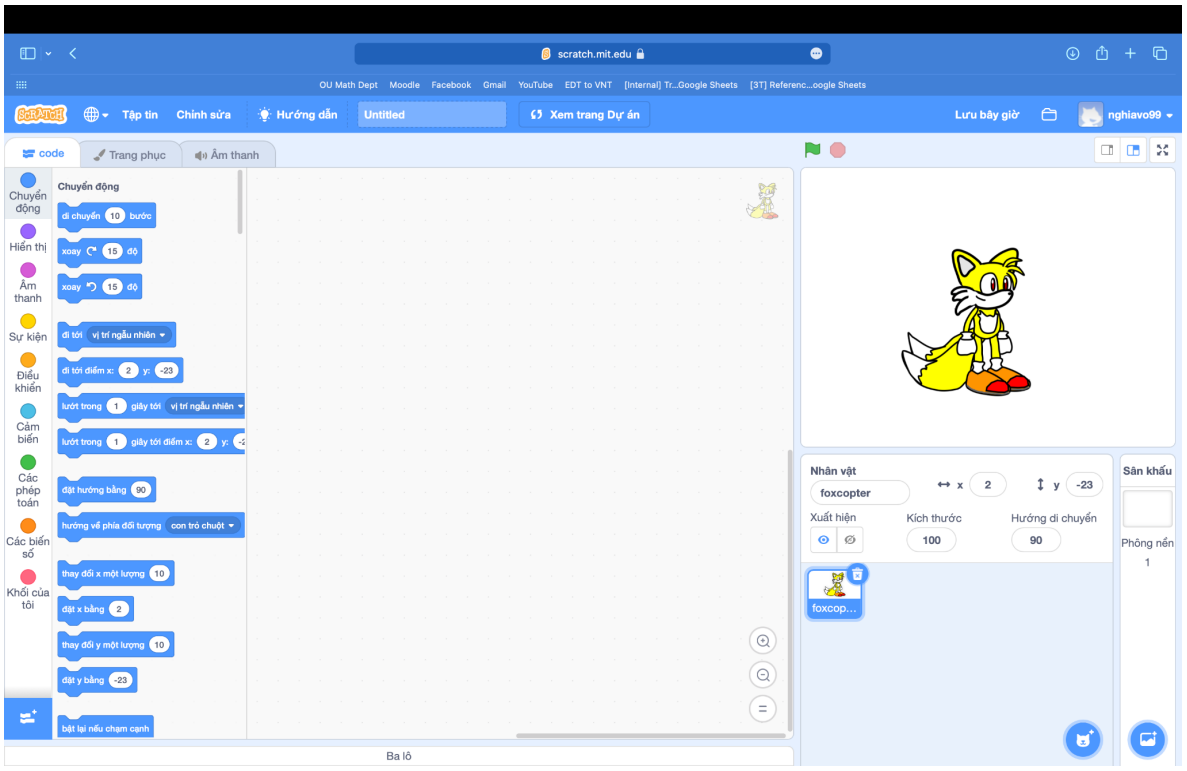
b. Bước 2:

- **Tìm vị trí và chọn nhân vật muốn tải lên từ máy tính.** Trong ví dụ này, chúng ta chọn tải lên một nhân vật đã có sẵn, tên là ***"foxcopter".***



c. Bước 3:

- Nhấn nút **"Tải lên",** kết quả sẽ là nhân vật đã được tải lên trang web của Scratch.



3.5. Các thông số của nhân vật:

- Để **xem thông số của nhân vật**, trước hết chúng ta cần **chọn nhân vật tương ứng trong danh sách** nhân vật.

(a) Tên nhân vật.

(b) Hoành độ của nhân vật (so với trung tâm của sân khấu).

(c) Tung độ của nhân vật (so với trung tâm của sân khấu).

(d) Ẩn/Hiện nhân vật.

(e) Kích thước nhân vật.

(f) Hướng di chuyển (Góc của nhân vật so với trục hoành).

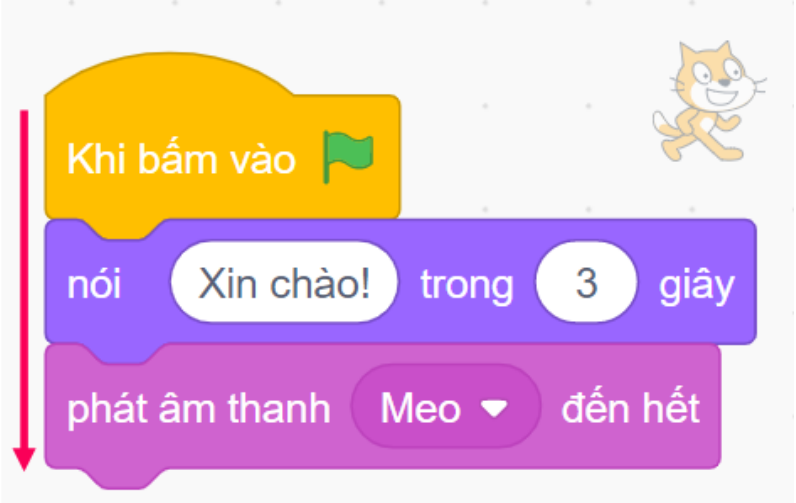

Nhân vật
a
foxcopter
b
x
2
c
y
-23
Xuất hiện
d
Kích thước
e
100
Hướng di chuyển
f
90

4. Thứ tự thực hiện các khối lệnh:

4.1. Thực hiện tuần tự:

- **Cách nhận biết các khối lệnh được thực hiện tuần tự:** Các khối lệnh được thực hiện tuần tự khi chúng được **xếp chồng lên nhau. Khi bắt đầu** chương trình, **khối lệnh ở trên cùng sẽ được thực hiện trước.** Sau khi **hoàn thành khối lệnh ở trên thì khối lệnh ngay dưới đó mới được thực hiện.**

- **Ví dụ:**



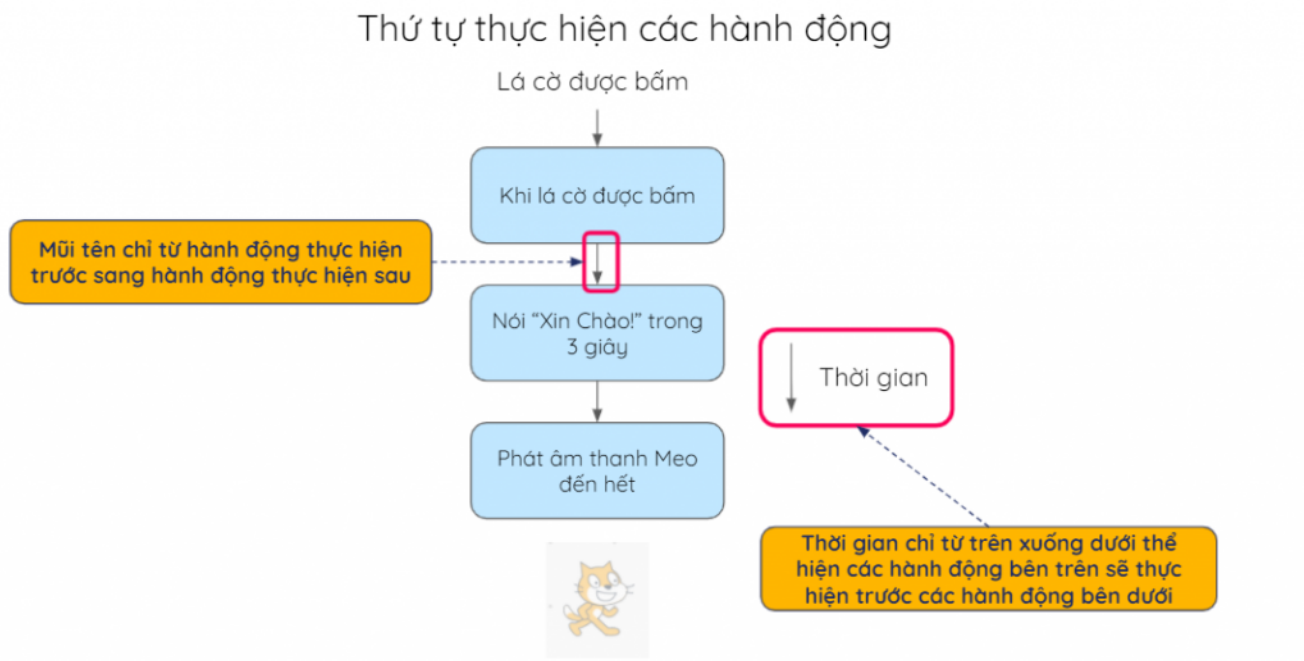
- Trong chương trình trên, chúng ta thấy các khối lệnh **được xếp chồng lên nhau**, do đó các khối lệnh này sẽ **được thực hiện tuần tự**. Thứ tự các khối lệnh được thực hiện sẽ là:

(1) Khối lệnh **"Khi bấm vào lá cờ xanh"** được thực hiện khi người chơi bấm vào nút là cờ.

(2) Tiếp đến, khối lệnh **"nói *Xin chào!* trong 3 giây"** được chạy để cho bạn Mèo hiển thị bong bóng nói trong vòng 3 giây.

(3) **Sau 3 giây** (tức là sau khi khối lệnh "nói" được thực hiện xong) thì bạn Miu mới **phát âm thanh meo.**

- **Chúng ta có thể biểu diễn trình tự trên như sơ đồ dưới đây:**



- **Khi nào nên cho các khối lệnh thực hiện tuần tự:** Chúng ta cho các khối lệnh thực hiện tuần tự khi chúng ta muốn các hành động của nhân vật được thực hiện theo đúng một thứ tự nhất định. Thực hiện tuần tự là mặc định của Scratch.

4.2. Thực hiện cùng lúc:

- **Cách nhận biết các khối lệnh được thực hiện cùng lúc:** Các khối lệnh được thực hiện cùng lúc sẽ không xếp chồng lên nhau. Thay vào đó, chúng sẽ cùng nằm dưới các khối lệnh sự kiện (khối lệnh màu vàng tươi) giống nhau.

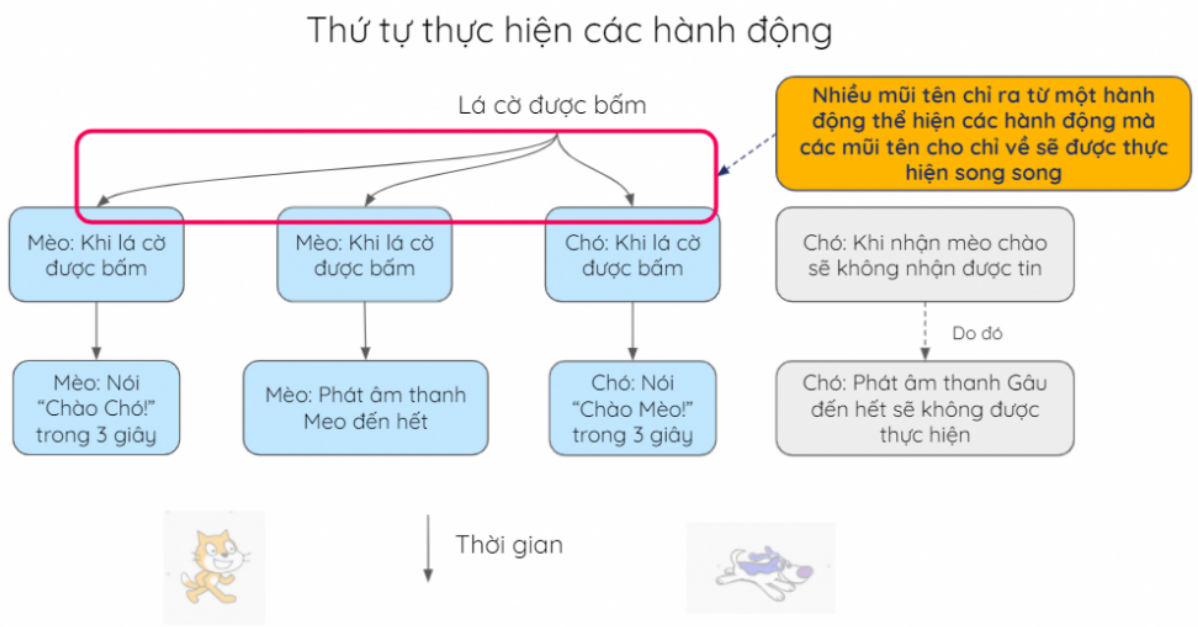
- **Ví dụ:**

- Trong hình trên, 4 khối lệnh **"Mèo nói *Chào chó!*", "Mèo phát âm thanh *Meo*", "Chó nói *Chào mèo!*" và "Chó phát âm thanh Gâu" không được xếp chồng lên nhau**, do đó chúng sẽ **không thực hiện tuần tự** với nhau.

- Thêm vào đó, chúng ta nhận ra 3 khối lệnh **"Mèo nói *Chào chó!*", "Mèo phát âm thanh *Meo*" và "Chó nói *Chào mèo!*" cùng nằm dưới** khối lệnh **"Khi bấm vào lá cờ xanh"** giống nhau, nên nếu **bấm vào lá cờ xanh** thì **3 khối lệnh này sẽ được thực hiện cùng lúc.**

- Khối lệnh **"Chó phát âm thanh Gâu"** tuy cũng nằm dưới một khối lệnh màu vàng, những khối lệnh màu vàng đó lại khác với 3 khối lệnh màu vàng còn lại nên hành động "Chó phát âm thanh Gâu" sẽ không được thực hiện cùng lúc với 3 hành động kia.

- **Chúng ta có thể biểu diễn trình tự trên như sơ đồ dưới đây:**



- **Khi nào nên sử dụng thực hiện cùng lúc:** Các khối lệnh của các nhân vật khác nhau hoặc các khối lệnh trong các hành động khác nhau của cùng một nhân vật có thể được cho thực hiện cùng lúc với nhau.

4.3. Kết hợp cả hai để phân tích một chương trình:

- Bây giờ chúng ta thử áp dụng kiến thức ở trên vào một chương trình cơ bản như sau: chương trình gồm có **2 nhân vật Mèo và Chó**, mỗi nhân vật được lập trình như hình dưới:

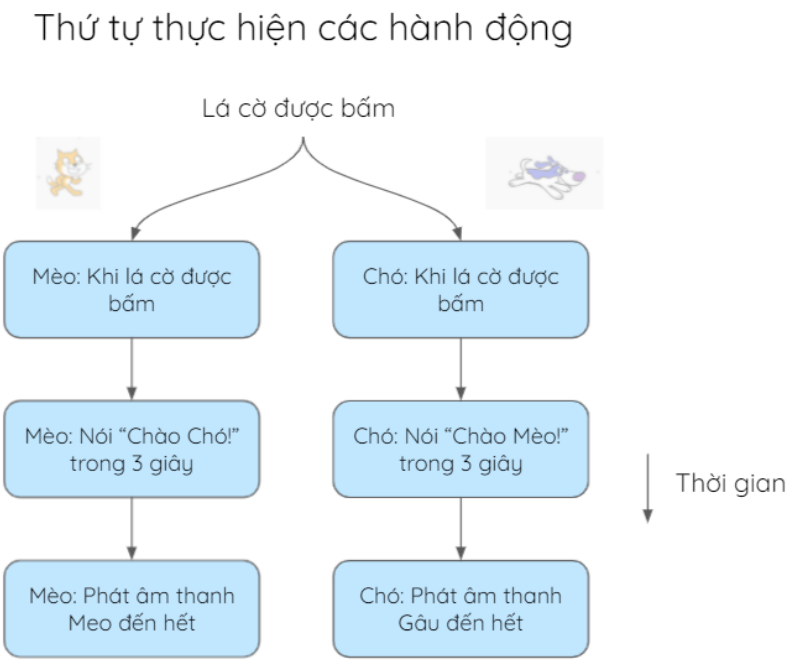
- **Chúng ta nhận thấy 3 điểm sau:**

(1) Khối lệnh **"nói *Chào Chó!* trong 3 giây"** của Mèo và khối lệnh **"nói *Chào Mèo!* trong 3 giây"** của Chó đều **nằm dưới** một khối lệnh màu vàng giống nhau đó là **"Khi bấm vào lá cờ xanh"**. Do đó 2 khối lệnh này sẽ **thực hiện cùng lúc với nhau (thực hiện cùng lúc).**

(2) Ở nhân vật Mèo, khối lệnh **"phát âm thanh *Meo* đến hết" nằm dưới** khối lệnh **"nói *Chào Chó!* trong 3 giây"** nên sẽ được **thực hiện sau** khi khối lệnh **"nói *Chào Chó!* trong 3 giây"** thực hiện xong **(thực hiện tuần tự).**

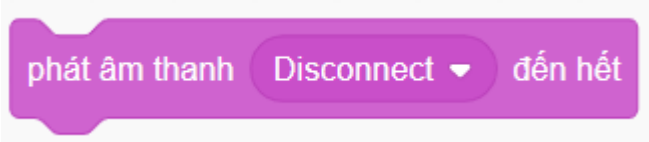
(3) Ở nhân vật Chó, khối lệnh **"phát âm thanh *Gâu* đến hết" nằm dưới** khối lệnh **"nói *Chào Mèo!* trong 3 giây"** nên sẽ được **thực hiện sau** khi khối lệnh **"nói *Chào Mèo!* trong 3 giây"** thực hiện xong **(thực hiện tuần tự).**

- **Từ 3 phân tích trên, chúng ta sẽ có sơ đồ về cách chương trình sẽ chạy như sau:**

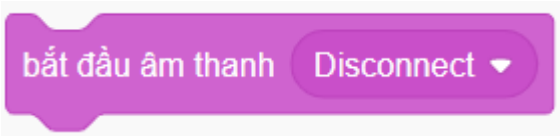


5. Các khối lệnh với âm thanh:

5.1. Phát âm thanh:



- Bắt đầu **phát âm thanh cho đến khi âm thanh kết thúc.**



- Bắt đầu **phát âm thanh.**

Lưu ý:

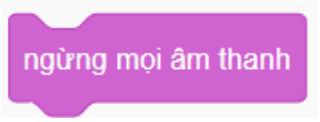
Phân biệt hai khối lệnh trên. Khối lệnh **"phát âm thanh … đến hết"** sẽ bắt đầu phát âm thanh và đợi cho đến khi âm thanh kết thúc thì sẽ thực hiện tiếp khối lệnh ở dưới. Khối lệnh "bắt đầu âm thanh" sẽ bắt đầu phát âm thanh và thực hiện luôn khối lệnh phía dưới mà không cần chờ âm thanh đến khi kết thúc.

Ví dụ:

- Ở khối bên trái, **âm thanh *Disconnect*** sẽ được **phát hết** rồi **nhân vật mới chuyển sang trang phục *Trâu 1.***

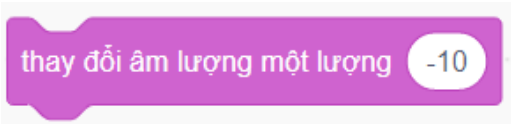
- Còn ở khối bên phải, **âm thanh *Disconnect*** sẽ **bắt đầu phát** và **trong lúc âm thanh được phát, nhân vật sẽ được chuyển sang trang phục *Trâu 1.***

5.2. Ngừng mọi âm thanh:



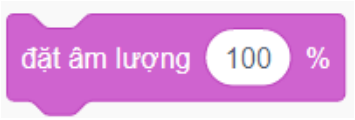
- **Dừng lại tất cả âm thanh** hiện đang phát

5.3. Thay đổi âm lượng:



- **Thay đổi âm lượng một lượng tùy ý.**

- **Ví dụ** ban đầu âm thanh ở mức 50. Để **tăng âm lượng lên 10 mức,** ta sẽ đổi thành ***thay đổi âm lượng một lượng 10.*** Lúc đó âm thanh sẽ ở mức 60. Còn để **giảm âm lượng 10 mức,** khối lệnh sẽ là *t****hay đổi âm lượng một lượng -10.*** Lúc đó âm thanh sẽ ở mức 40.



- Đặt âm lượng của âm thanh về bao nhiêu. Mặc định ban đầu âm lượng là 100%. Giảm thành số nhỏ hơn 100 thì âm lượng sẽ giảm, tăng lên số lớn hơn 100 thì âm lượng sẽ tăng.

Ví dụ:

- Mặc định âm thanh luôn là **100%**. Nếu chuyển thành ***đặt âm lượng 50%***, âm thanh sẽ chỉ to **bằng một nửa** âm thanh ban đầu. Còn nếu khối lệnh là ***đặt âm lượng 300%***, âm thanh sẽ **to gấp ba lần** âm thanh ban đầu.

6. Cách viết vòng lặp:

6.1. Tại sao phải sử dụng vòng lặp trong Scratch:

- **Vòng lặp (loop)** là một khái niệm cơ bản trong lập trình.
- **Chức năng** của vòng lặp là **lặp đi lặp lại những hành động bên trong vòng lặp. Thay vì cứ nối đoạn chương trình với các câu lệnh giống y hệt nhau** thì người ta **có thể sử dụng vòng lặp** để **giảm số câu lệnh** và **khiến chương trình rõ ràng hơn.**

Ví dụ:

- Thay vì nối 5 câu lệnh ***trang phục kế tiếp,*** chúng ta có thể lập trình như sau:

6.2. Các loại vòng lặp trong Scratch:

- Tùy theo số lần lặp mà người ta phân ra làm ba loại: Lặp lại, Lặp lại cho đến khi và Liên tục.

a. Vòng lặp Lặp lại:

Ý nghĩa: Lặp lại với số lần cố định biết trước

Ví dụ:

- Một bạn nhỏ đang ở dưới chân cầu thang, cầu thang dài 10 bậc. **Thay vì** nối liền **10 câu lệnh *di chuyển 1 bước***, chúng ta **có thể lặp lại câu lệnh đó 10 lần**:

b. Vòng lặp Lặp lại cho đến khi:

Ý nghĩa: Lặp lại cho đến khi thỏa mãn một điều kiện nào đó (có thể hoặc không biết trước số lần lặp, miễn là thỏa mãn điều kiện thì vòng lặp sẽ dừng lại)

Ví dụ:

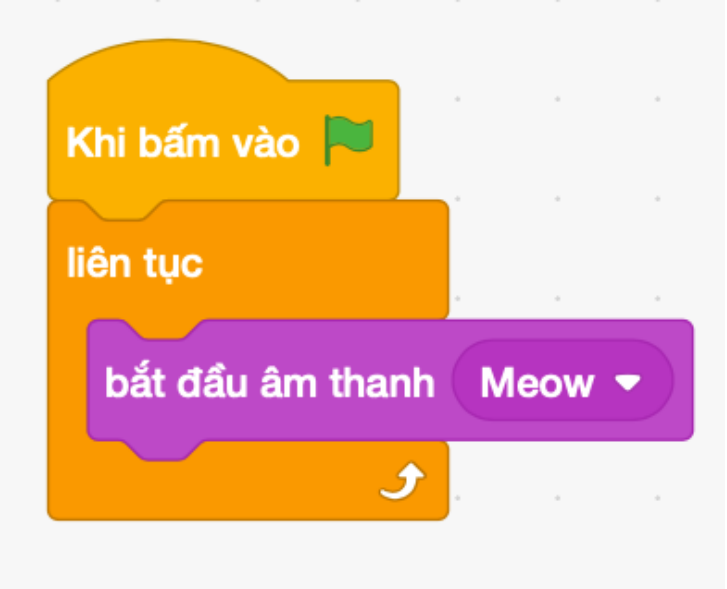
- Một bạn nhỏ đang ở dưới chân cầu thang, **không biết trước số bậc là bao nhiêu**. Miễn là bạn đó leo hết cầu thang, đến tầng gác mái thì vòng lặp dừng lại.

c. Vòng lặp Liên tục:

Ý nghĩa: Lặp lại mãi mãi (Cho đến khi nào chương trình dừng thì sẽ dừng vòng lặp. Chừng nào chương trình còn chạy thì vẫn tiếp tục lặp)

Ví dụ:

- Khi thầy/cô muốn chơi một đoạn nhạc nền trong chương trình, mình sẽ sử dụng vòng lặp liên tục (chơi nhạc) đến khi chương trình kết thúc.

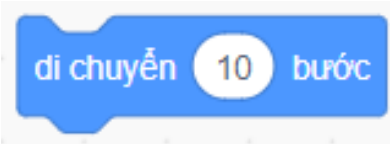


Lưu ý:

- Ta **vẫn có thể lập trình tiếp sau** câu lệnh **Lặp lại** hay **Lặp lại cho đến khi.** Còn **đối với liên tục**, ta **không thể nối bất cứ câu lệnh nào phía sau** nó.

7. Hướng di chuyển và Di chuyển:

7.1. Di chuyển:



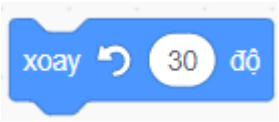
- Câu lệnh này dùng để **làm nhân vật di chuyển về phía trước** (Phía trước của nhân vật được quy định bởi yếu tố hướng di chuyển trong mục Các thông số của nhân vật).
- **Số bước** mà nhân vật di chuyển là **số ở trong câu lệnh**. Thầy/Cô có thể tùy ý chỉnh sửa số này để làm nhân vật di chuyển dài/ngắn theo ý mình mong muốn.
- **Ví dụ:** Như trong hình, câu lệnh này làm **nhân vật di chuyển về phía trước mặt 10 bước**.

7.2. Xoay:

a. Xoay phải:

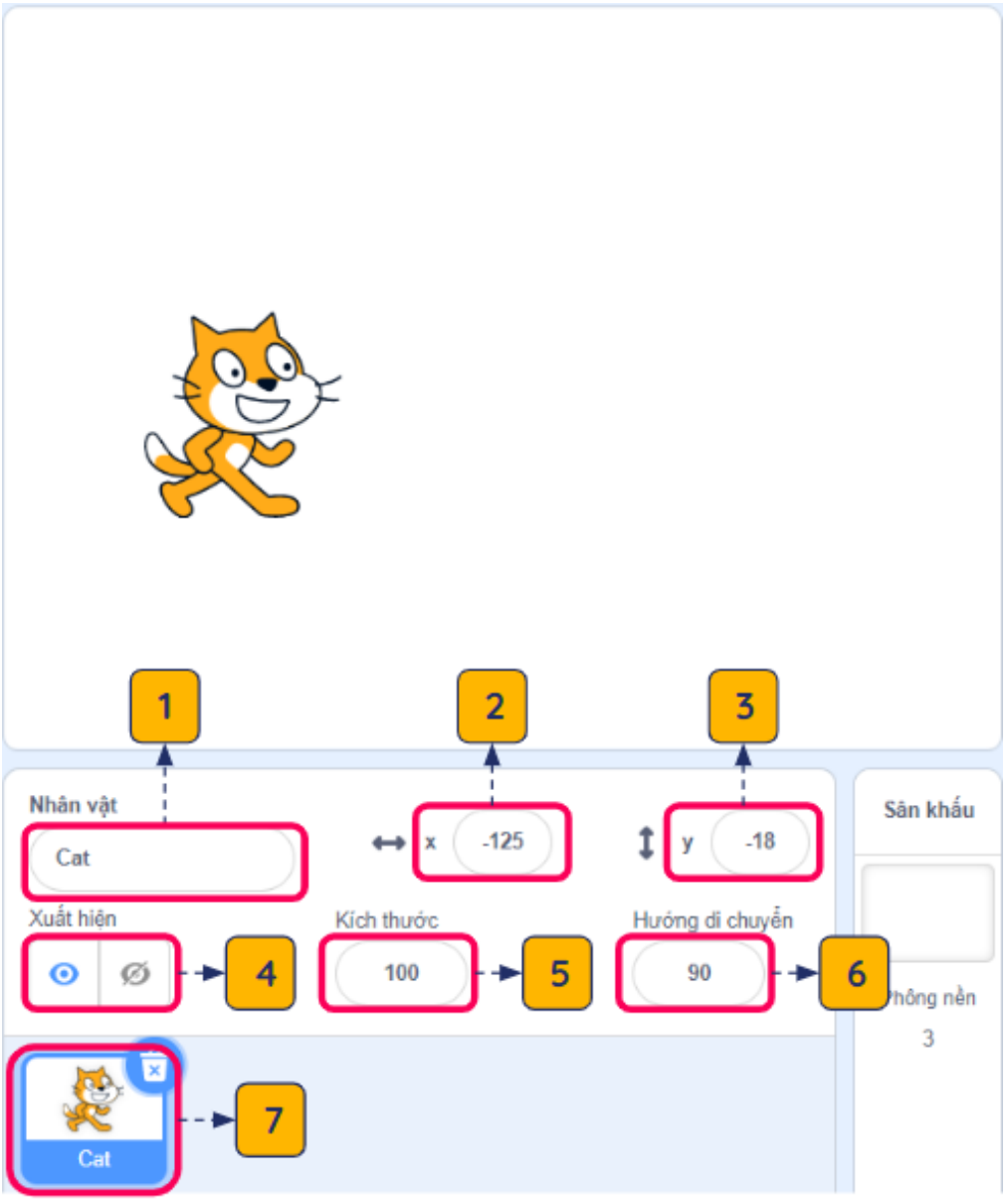
- Câu lệnh này dùng để **làm nhân vật quay về bên phải.** Nhân vật quay bao nhiêu độ so với vị trí hiện tại phụ thuộc vào số trong câu lệnh. Số này có thể tùy chỉnh để nhân vật quay theo ý mình.
- **Ví dụ:** Khi kích hoạt câu lệnh trên, **nhân vật sẽ quay một góc 30 độ sang bên phải** như thầy/cô thấy trong hình dưới đây.

b. Xoay trái:



- Tương tự như câu lệnh quay phải, câu lệnh quay trái này hoạt động với quy tắc giống hệt như câu lệnh quay phải.
- **Ví dụ:** Khi kích hoạt câu lệnh trên, **nhân vật sẽ quay một góc 30 độ sang bên trái** như thầy/cô thấy trong hình dưới đây.

7.3. Giới thiệu tọa độ:



(1) Tên nhân vật.

(2) Hoành độ của nhân vật (so với trung tâm của sân khấu).

(3) Tung độ của nhân vật (so với trung tâm của sân khấu).

(4) Ẩn/Hiện nhân vật.

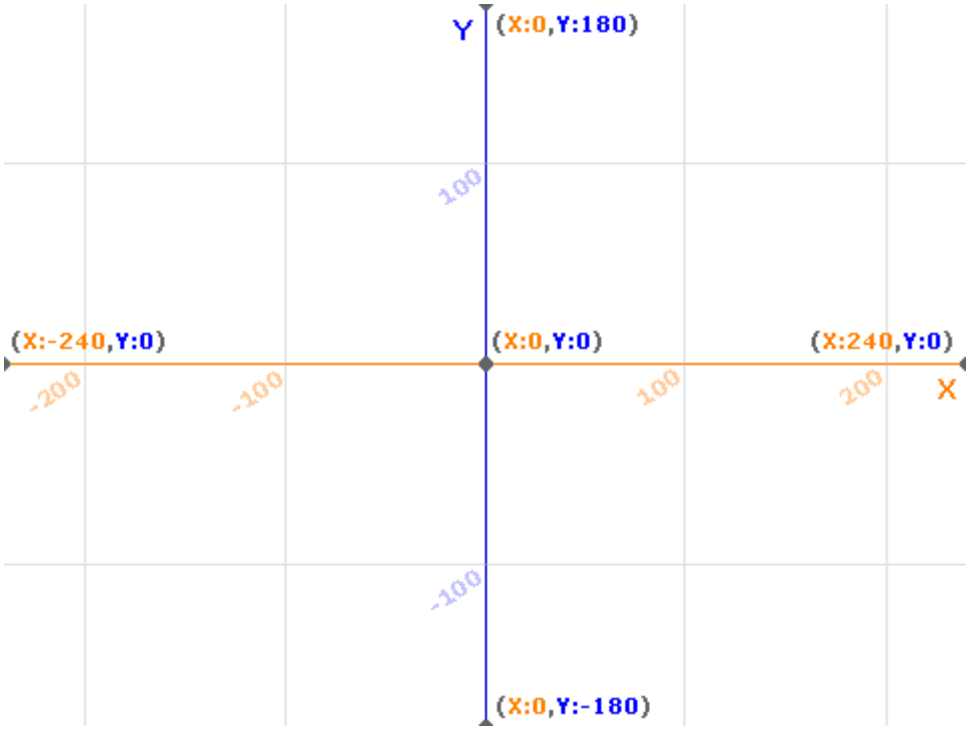
(5) Kích thước nhân vật.

(6) Hướng di chuyển (Góc của nhân vật so với trục hoành).

(7) Nhân vật đang chọn.

Khái niệm:

- Hệ tọa độ Oxy gồm 2 trục Ox và Oy.



- **Trục Ox** là trục **nằm ngang**, kéo dài từ bên trái sang bên phải tương ứng với **giá trị x từ -240 (bên trái màn hình) đến 240 (bên phải màn hình)**. Tọa độ x (hay còn gọi là hoành độ) để **biểu thị vị trí của nhân vật theo phương ngang.**

- Tương tự như vậy, **trục Oy** là trục **thẳng đứng**, kéo dài từ dưới lên trên tương ứng với **giá trị y từ -180 (phía dưới màn hình) đến 180 (phía trên màn hình)**. Tọa độ y (hay còn gọi là tung độ) để **biểu thị vị trí của nhân vật theo phương thẳng đứng.**

- Kết hợp tọa độ x và y, ta được một **bộ (x, y)** dùng để **biểu diễn chính xác vị trí của nhân vật** trên màn hình.
- Ngoài ra, **điểm O** sẽ là **điểm nằm chính giữa sân khấu** và có **tọa độ (0,0).**

Ứng dụng

1. Tọa độ của nhân vật:

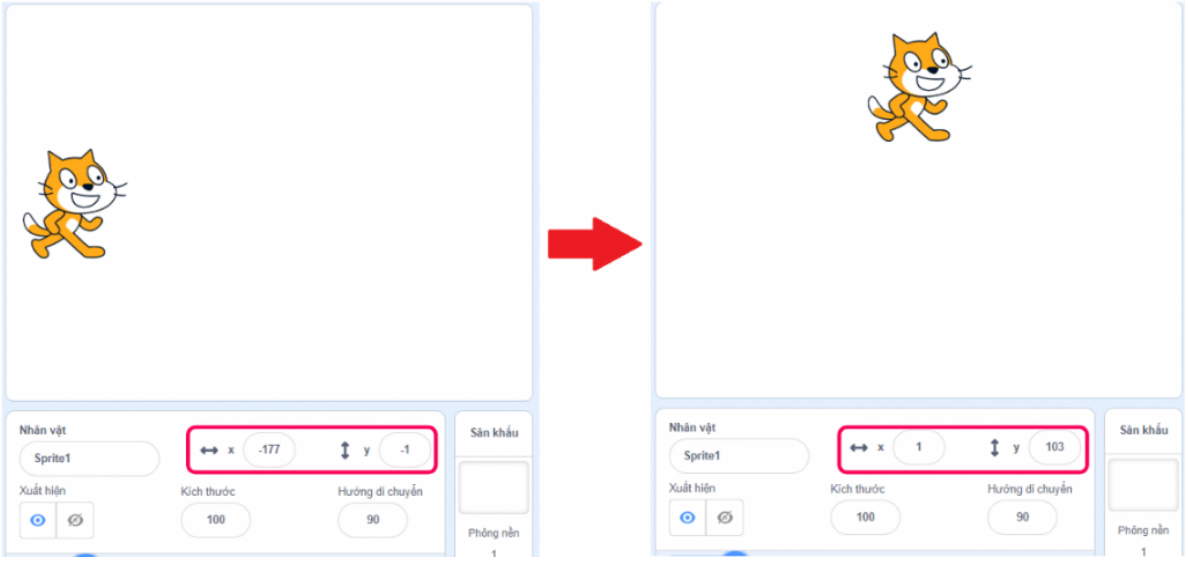
- Để biết được tọa độ của một nhân vật, ta **nhấn chọn nhân vật đó** và **nhìn ở phần thuộc tính nhân vật.**

Ví dụ: Nhân vật mèo Miu dưới đây có tọa độ x bằng -125 còn tọa độ y bằng -18.

2. Tìm tọa độ của nhân vật ở vị trí bất kỳ:

- Muốn biết tọa độ của nhân vật ở vị trí bất kỳ, bạn hãy **nhấn chọn nhân vật và di chuyển nhân vật đến vị trí đó và thả chuột.** Phần **thuộc tính của nhân vật sẽ tự động cập nhật vị trí tọa độ (x;y) tương ứng.**

Ví dụ:

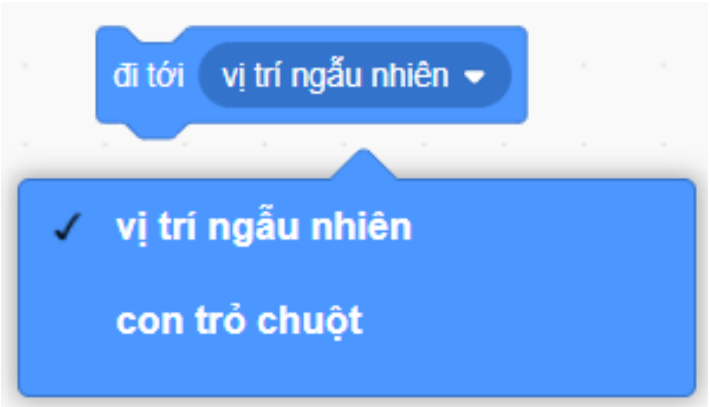


- Vị trí ban đầu của nhân vật là (-177,-1) (hoành độ x có giá trị là -177, tung độ y có giá trị là -1). Sau khi di chuyển nhân vật đến vị trí mới thì tọa độ của nhân vật lúc đó là (1,103) (hoành độ x có giá trị là 1, tung độ y có giá trị là 103).

- **Lưu ý:** Khoảng giá trị của hoành độ x và hoành độ y tương ứng với màn hình của Scratch.

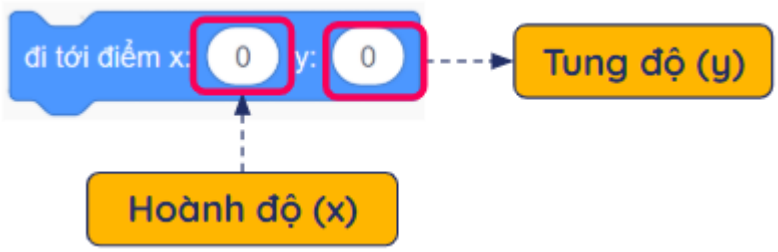
7.4. Bộ câu lệnh đi tới:

a. Câu lệnh đi tới:



- Câu lệnh **đi tới** có tác dụng **làm cho nhân vật lập tức di chuyển đến vị trí nào đó** dựa vào lựa chọn của câu lệnh:
    - **vị trí ngẫu nhiên:** khi kích hoạt, nhân vật **đi tới vị trí ngẫu nhiên.**
    - **con trỏ chuột:** khi kích hoạt, nhân vật **đi tới vị trí của con trỏ chuột.**

b. Câu lệnh đi tới vị trí xác định:



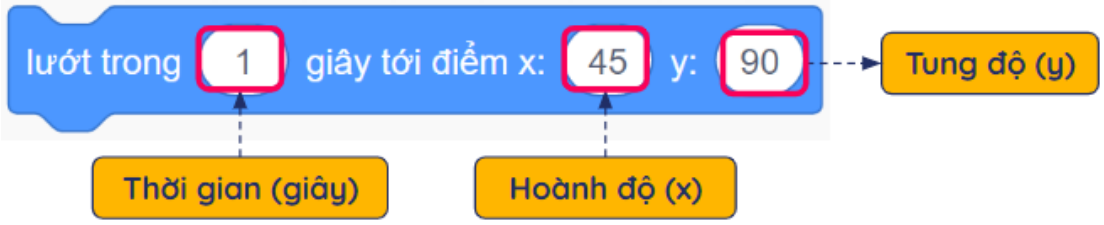
- Câu lệnh **đi tới điểm x: … y: …** có tác dụng **làm cho nhân vật lập tức di chuyển đến vị trí tương ứng** với hoành độ (x) và tung độ (y) đã xác định trên sân khấu.

7.5. Bộ câu lệnh lướt:

a. Câu lệnh lướt trong … giây tới …

- Nếu như câu lệnh **đi tới** làm nhân vật **LẬP TỨC** nhảy đến vị trí nào đó, thì câu lệnh **lướt trong … giây tới …** làm cho nhân vật **DI CHUYỂN TỪ TỪ** đến một vị trí trong số giây được nhập vào.
- **Số giây** để nhân vật di chuyển đến nơi chỉ định được nhập trong ô số, số giây được **đặt mặc định** là **1 giây**.
- Khi kích hoạt câu lệnh:
    - **vị trí ngẫu nhiên:** nhân vật **di chuyển đến vị trí bất kì.**
    - **con trỏ chuột:** nhân vật **di chuyển đến vị trí con trỏ chuột.**

b. Câu lệnh lướt xác định tọa độ:



- Sau khi **nhập số giây và tọa độ (hoành độ x, tung độ y)** của vị trí mà thầy/cô mong muốn và thực hiện câu lệnh này, nhân vật sẽ **di chuyển đến vị trí có tọa độ (hoành độ x, tung độ y)** tương ứng trên phông nền t**rong thời gian mà thầy/cô nhập.**

7.6. Bộ câu lệnh hướng:

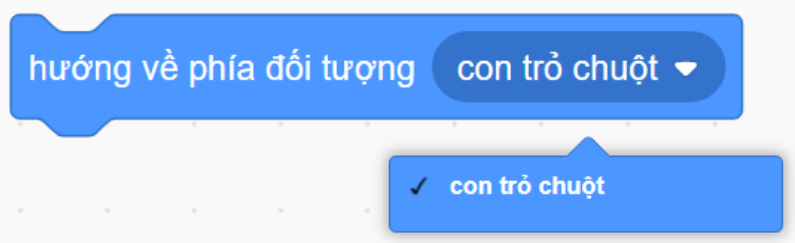
- **Hướng của nhân vật** được đo bằng **góc** giữa **phía mà nhân vật đó đang hướng về** và **trục hoành (trục Ox).**

a. Câu lệnh đặt hướng bằng:

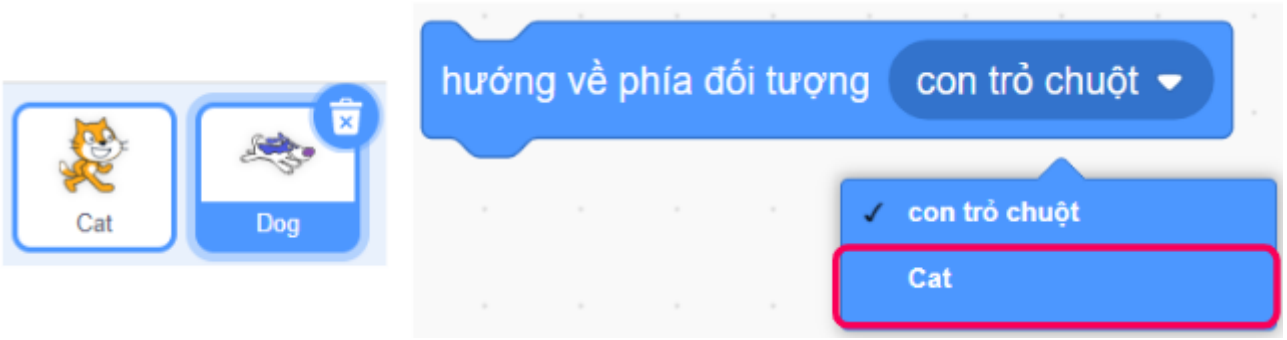
- Câu lệnh **đặt hướng bằng ...** có tác dụng làm cho nhân vật quay về hướng thầy/cô mong muốn bằng cách nhập số vào ô.

- Thầy/Cô có thể nhập trực tiếp số hoặc quay mũi tên về hướng mình muốn.

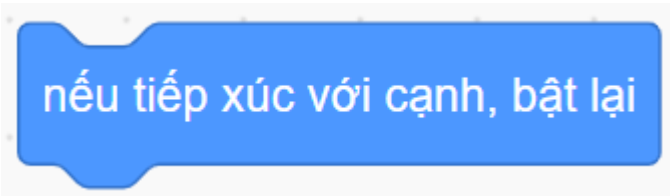
b. Câu lệnh hướng về phía đối tượng:



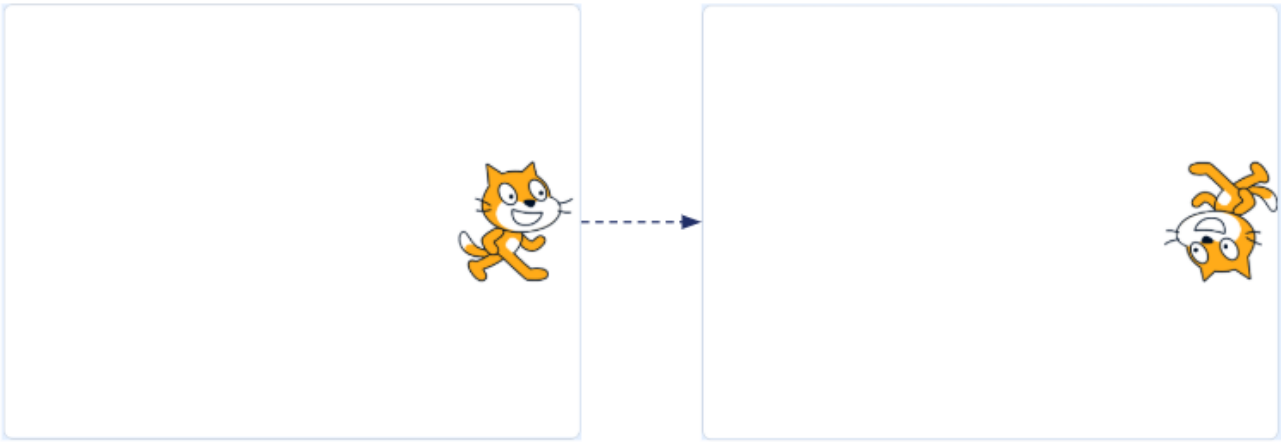
- Khi chạy câu lệnh này, nhân vật sẽ **hướng về phía vị trí của con trỏ chuột** trên phông nền.
- Trong trường hợp dự án (Project) có nhiều hơn 1 nhân vật (sprite), ta có thể lập trình để nhân vật này hướng về nhân vật khác.
- **Ví dụ** trong hình dưới đây, có 2 nhân vật: Cat (mèo) và Dog (chó). Để lập trình cho Dog hướng về phía Cat, ta chọn câu lệnh *hướng về phía đối tượng Cat.*



7.7. Câu lệnh bật lại khi tiếp xúc với cạnh:

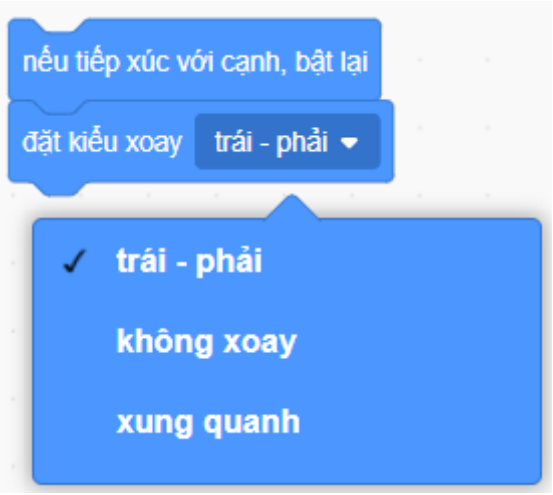


- Câu lệnh này là **một câu điều kiện,** nó sẽ được **thực thi khi nhân vật** của thầy/cô **di chuyển đến cạnh của màn hình sân khấu.**
- Cụ thể, khi nhân vật của thầy/cô di chuyển và chạm vào cạnh màn hình, nhân vật sẽ bị bật ngược ra, như ví dụ bên dưới.



7.8. Câu lệnh đặt kiểu xoay:

- Khi nhân vật của thầy/cô di chuyển và chạm vào cạnh của màn hình, sử dụng câu lệnh **nếu tiếp xúc với cạnh, bật lại** sẽ làm nó bật ngược lại, nhưng với hướng ngược lại như thầy/cô thấy trong ví dụ trên. Đây là lúc thầy/cô nên sử dụng câu lệnh **đặt kiểu xoay.**
- Câu lệnh đặt kiểu xoay có **3 lựa chọn**:
    - **Trái-phải:** có nghĩa là hình ảnh của nhân vật **chỉ quay sang trái và quay sang phải.**
    - **Không xoay:** có nghĩa là hình ảnh của nhân vật l**uôn hướng mặt sang một phía dù thay đổi** thông số **Hướng di chuyển** trong phần Thuộc tính của nhân vật.
    - **Xung quanh:** có nghĩa là hình ảnh nhân vật **xoay 360 độ.**
- Khi sử dụng câu lệnh này, nhân vật của mình sẽ bật ra nhưng theo hướng mà thầy/cô chọn, chứ không chỉ bật ngược như ví dụ trên



- Thầy/Cô nên ghép câu lệnh **đặt kiểu xoay** với câu lệnh **nếu tiếp xúc với cạnh, bật lại** như thế này.

- **Lưu ý:** Câu lệnh **đặt kiểu xoay** chỉ ảnh hưởng đến hướng của hình ảnh nhân vật. Nó không thể hiện hướng chuyển động của nhân vật.

7.9. Thao tác trực tiếp lên tọa độ:

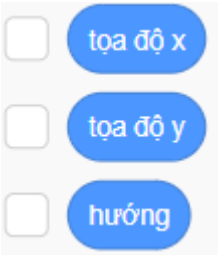
a. Câu lệnh đặt trực tiếp tọa độ:

- Có 2 câu lệnh đặt tọa độ, áp dụng cho hoành độ x và tung độ y
- **Sau khi nhập tọa độ** mình mong muốn **và chạy câu lệnh**, **nhân vật** của thầy/cô sẽ **được đưa đến vị trí có tọa độ tương ứng.**

b. Câu lệnh thay đổi tọa độ:

- Có 2 câu lệnh thay đổi tọa độ (hoành độ x, tung độ y)
- Câu lệnh này có tác dụng **tăng/giảm tọa độ x hoặc y lên số đơn vị** mà thầy/cô nhập.
- **Ví dụ:** Trong hình trên:
  - Câu lệnh **thay đổi x một lượng 10** có tác dụng **tăng tọa độ x hiện tại lên 10 đơn vị.**
  - Câu lệnh **thay đổi y một lượng -10** có tác dụng **giảm tọa độ y hiện tại xuống 10 đơn vị.**

c. Hiển thị vị trí lên màn hình:



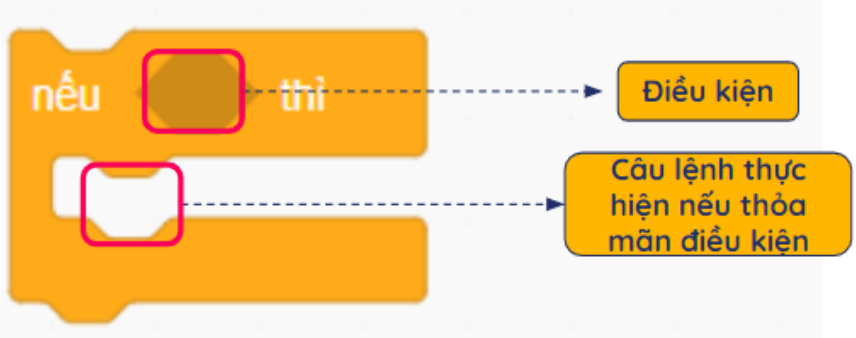
- Các mục **có thể tích** như trên **để hiển thị tọa độ (hoành độ x, tung độ y)** và **hướng** của nhân vật lên màn hình.

- **Ví dụ:** Khi thầy/cô tích chọn các thông số, trên màn hình sẽ hiện

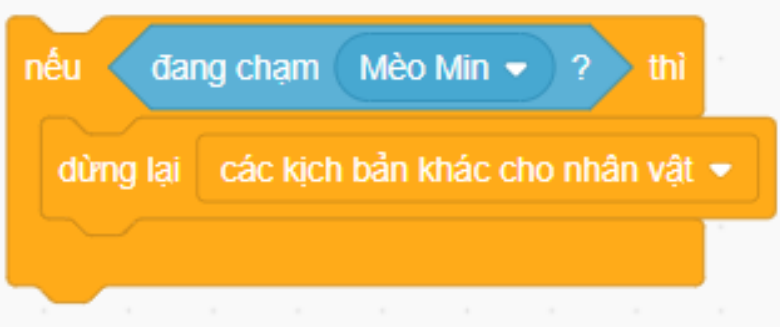
8. Câu lệnh Nếu... Thì...

- Một trong câu lệnh cơ bản của tất cả các ngôn ngữ lập trình, bao gồm cả Scratch, chính là câu lệnh **Nếu ... thì ...**

- **Câu lệnh Nếu** được gọi là **câu lệnh điều kiện**. Câu lệnh này dùng để kiểm tra một điều kiện nào đó, **nếu thỏa mãn điều kiện thì thực hiện câu lệnh bên trong nó.**



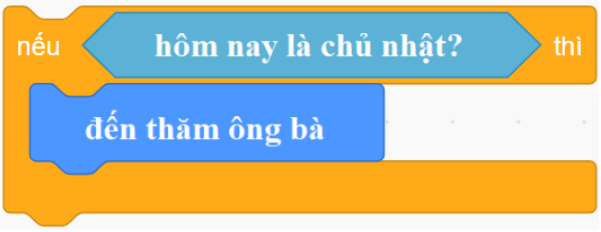
Ví dụ:



- Trong trường hợp này, **nếu nhân vật đang chạm Mèo Min** thì chương trình sẽ dừng lại các kịch bản khác cho nhân vật.
- Ở đây **"Dừng lại các kịch bản khác cho nhân vật"** được hiểu đơn giản là các nhân vật khác sẽ dừng ở trạng thái hiện tại và không thực hiện các hành động sau nữa.

- **Nếu điều kiện sai** (đang không chạm Mèo Min) thì sẽ không làm gì cả.

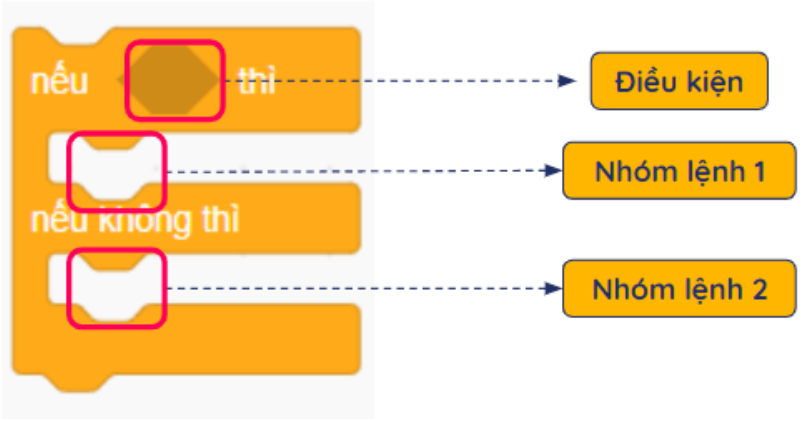
Một ví dụ khác đơn giản hơn:



- Chương trình sẽ kiểm tra *"Nếu hôm nay là chủ nhật? Thì đến thăm ông bà"*. Trong trường hợp hôm nay không phải là chủ nhật, thì không làm gì.

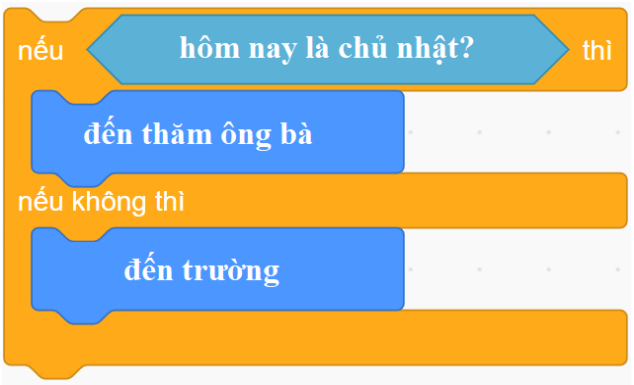
MỞ RỘNG:

- Ngoài ra còn có câu lệnh **Nếu … thì … Nếu không … thì …**



- Câu lệnh này **gần giống** với câu lệnh **Nếu … thì …** nguyên bản, nhưng điểm đặc biệt ở chỗ: Sau khi kiểm tra điều kiện, nếu **thỏa mãn đúng điều kiện thì thực hiện nhóm lệnh 1**, nếu **không thỏa mãn điều kiện thì thực hiện nhóm lệnh 2**.

Ví dụ:



- Chương trình sẽ kiểm tra **"Nếu hôm nay là chủ nhật? Thì đến thăm ông bà"**. Trong trường hợp hôm nay không phải là chủ nhật thì sẽ **đến trường.**

9. Câu lệnh "Đang chạm ...?" và "Phím .... được bấm?":

9.1. Câu lệnh "Đang chạm ...?":

- Một trong những câu lệnh đầu tiên trong khối "cảm biến" là câu lệnh "đang chạm....?". Câu lệnh này kiểm tra xem nhân vật của nó có chạm vào con trỏ chuột, cạnh hoặc một nhân vật khác hay không (ví dụ như nhân vật "winter hat" trong hình minh hoạ). **Nếu nhân vật hiện tại có đang chạm vào đối tượng đã chọn, khối sẽ trả về "đúng"; nếu không, nó trả về "sai".**

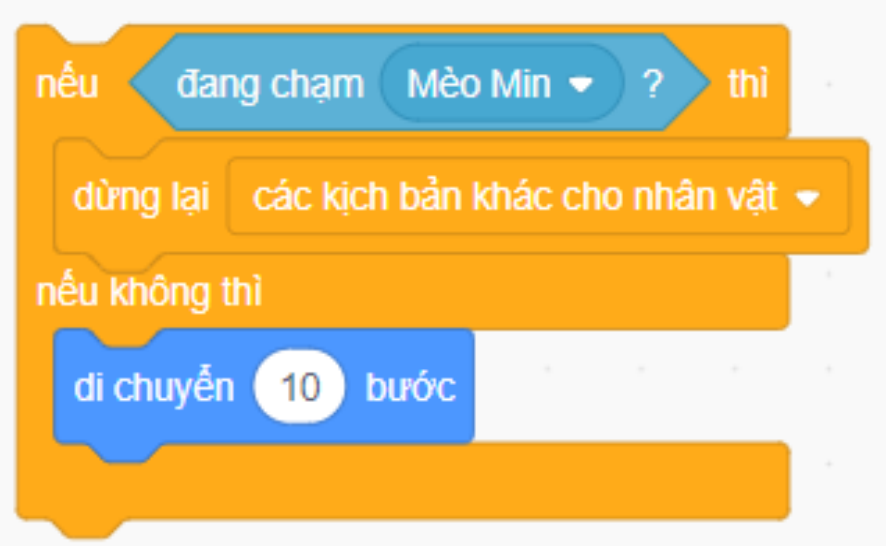
- Việc một nhân vật có chạm vào người khác hay không phụ thuộc vào việc một phần kết cấu hình ảnh của nhân vật đó (được xác định bởi **"Trang phục"** của nhân vật ấy) có chạm vào người khác hay không.

  Ví dụ: Nếu "Trang phục" của một nhân vật có hình vuông, nó sẽ có một **"hộp va chạm"** hình vuông khi phát hiện nếu nó chạm vào các nhân vật khác.

- Trong hình trên, Miu phía bên trái không có hộp va chạm thì Miu sẽ dừng lại khi râu của Miu chạm vào tường. Khi đó, tuy chỉ có râu của Miu chạm vào tường nhưng trò chơi sẽ tính là cả nhân vật chạm vào tường. Như vậy người chơi sẽ có thể cảm thấy trò chơi chưa đủ chân thật và sống động.

- Ngược lại, Miu ở phía bên phải sử dụng "hộp va chạm", Miu sẽ không dừng lại khi râu của mình chạm vào tường nữa. Miu sẽ chỉ dừng lại và bị tính là chạm tường khi "hộp va chạm" chạm vào tường.

Ví dụ: Kết hợp **"Đang chạm...?"** cùng với câu lệnh **"Nếu ... thì ..."** và **"Nếu không ... thì"**

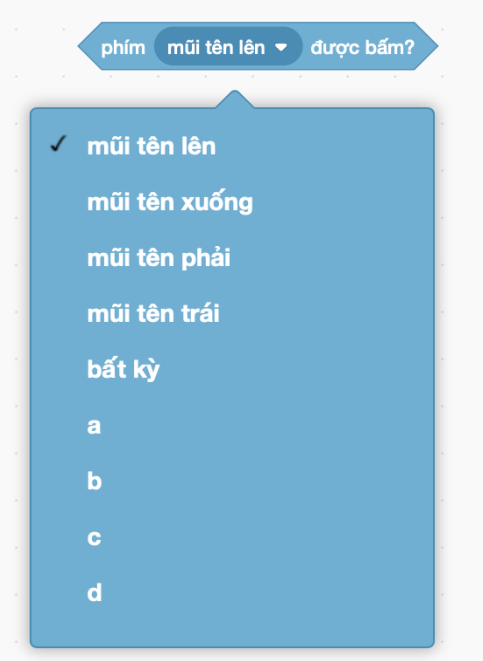


- Ở ví dụ này, nếu nhân vật đang chạm Mèo Min thì sẽ thực hiện dừng lại các kịch bản khác cho nhân vật. Trong trường hợp ngược lại (nếu đang không chạm Mèo Min) thì nhân vật sẽ di chuyển 10 bước.

9.2. Câu lệnh "Phím ... được bấm?":

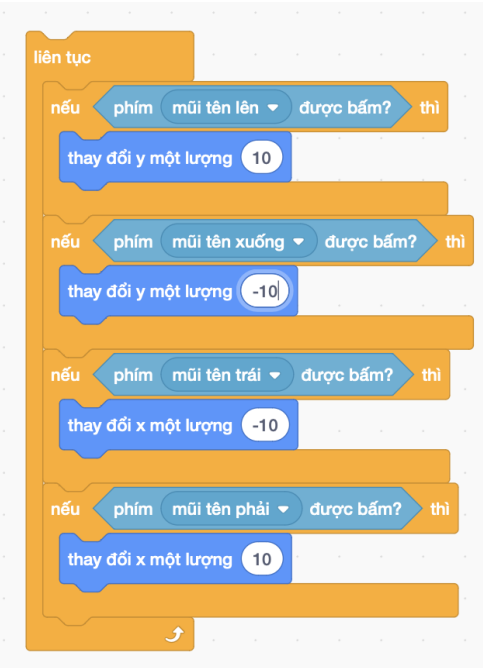
- Câu lệnh **"Phím ... được bấm?"** là thuộc **khối "Cảm biến"**, câu lệnh này kiểm tra xem phím được chỉ định có được nhấn hay không. Nếu phím đang được nhấn, khối trả về "đúng"; nếu không, nó trả về "sai".

- Các phím có sẵn để sử dụng trong cùng với câu lệnh này bao gồm **toàn bộ bảng chữ cái tiếng Anh (a, b, c, v.v.), các phím số (0, 1, 2, v.v.), các phím mũi tên (← ↑ → ↓) và phím cách**. Câu lệnh này còn có thể cho phép **chọn một phím bất kỳ**, cho phép một người nhấn bất kỳ phím nào để vận hành khối.



- Ngoài ra, câu lệnh **"Phím … được bấm?"** có thể được sử dụng để thay thế cho câu lệnh **"Khi bấm phím …"** vốn thuộc khối lệnh **"Sự kiện".**

Ví dụ: Một chức năng hay được dùng của câu lệnh **"Phím … được bấm?"** là dùng để điều khiển chuyển động của nhân vật một cách mượt mà.

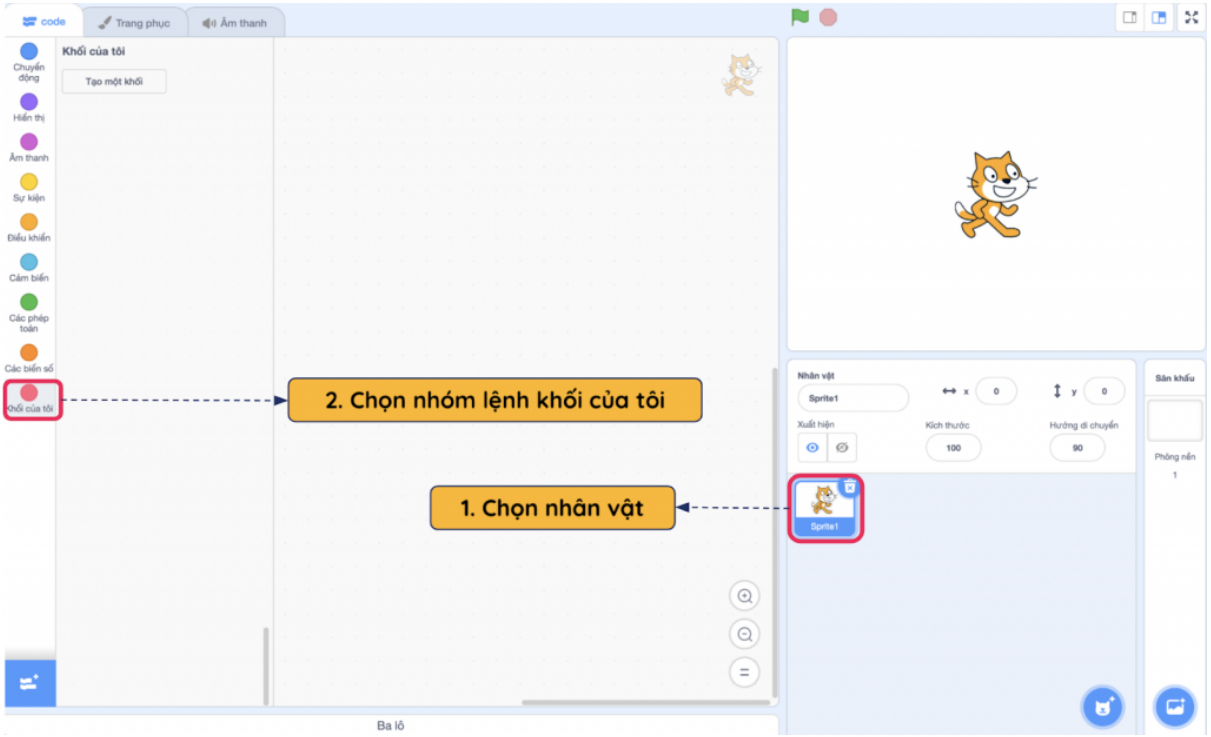


- Trong ví dụ này, câu lệnh **"Phím … được bấm?"** được kết hợp với câu lệnh **"Nếu … thì …"** và **vòng lặp liên tục**. Ý nghĩa của chương trình này là **liên tục** đợi chúng ta lệnh nhấn **phím mũi tên (← ↑ → ↓)** và thực hiện các lệnh làm thay đổi hoành độ x hoặc tung độ y của nhân vật một lượng + 10 (hoặc -10) đơn vị.
- Kết quả là nhân vật của chúng ta sẽ có hiệu ứng di chuyển sang các hướng phải, trái, lên, xuống tuỳ vào phím bấm phù hợp được chọn lựa.

10. Khối của tôi:

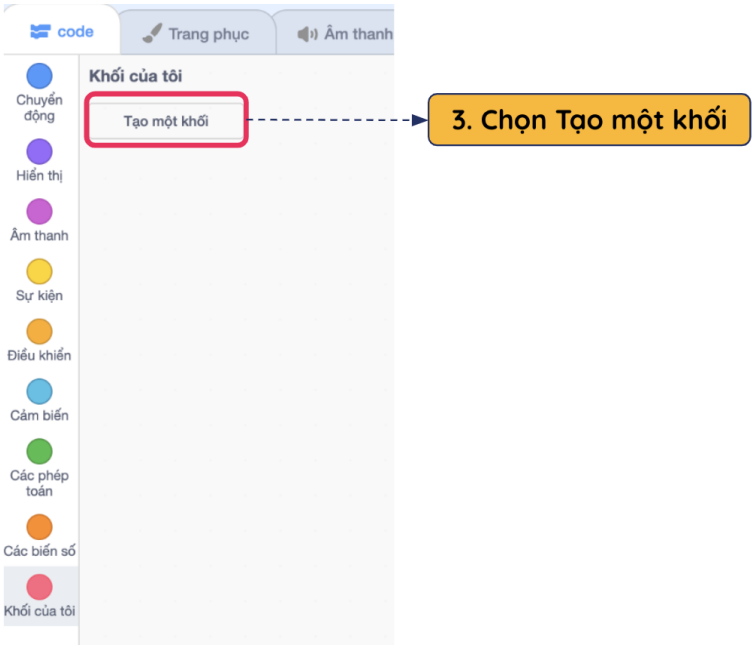
10.1. Cách tạo Khối của tôi

Bước 1: Chọn nhân vật bạn muốn tạo khối của tôi.



- **Lưu ý:** Khi tạo khối của tôi cho nhân vật nào thì chỉ nhân vật đó mới có thể sử dụng khối lệnh này.

Bước 2: Chọn nhóm lệnh Khối của tôi

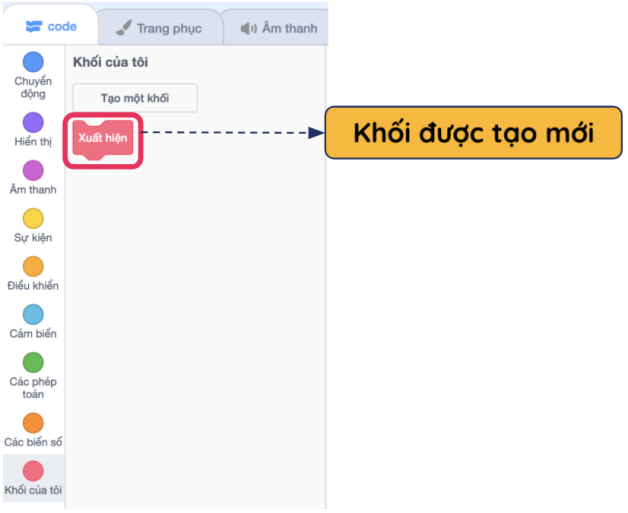


Bước 3: Chọn Tạo một khối. Sau đó màn hình sẽ hiển thị hộp thoại

Bước 4: Điền tên khối rồi nhấn nút OK

- Sau khi **nhấn OK**, một khối mới của tôi sẽ được tạo ra và xếp ở dưới như hình vẽ.

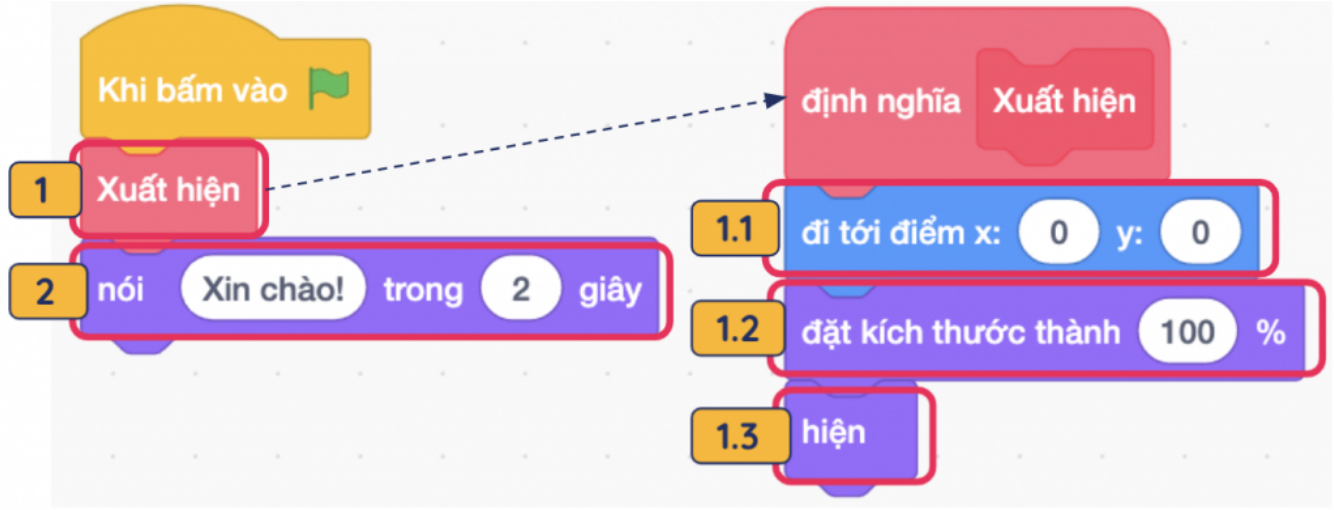
Kết quả:



10.2. Cách sử dụng Khối của tôi

- **Khối của tôi** là một khối lệnh mới mà chúng ta tự tạo ra. Khối lệnh này được định nghĩa dựa trên những khối lệnh có sẵn trong Scratch.

- **Ví dụ:** Khối lệnh Xuất hiện được định nghĩa là khi nhận vật sẽ xuất hiện ở chính giữa màn hình, có kích thước chính xác khi được tạo ra. Khi khối lệnh này được sử dụng ở chương trình chính (nối vào bên dưới khối lệnh sự kiện như Khi bấm vào lá cờ), chương trình sẽ chạy lần lượt tất cả các khối lệnh ở phần định nghĩa của khối lệnh này.



- Khi ấn vào biểu tượng lá cờ, khối xuất hiện sẽ được gọi. Khi đó toàn bộ các khối lệnh bên trong khối xuất hiện sẽ chạy cho đến hết. Sau đấy mới quay trở về chạy tiếp khối nói "xin chào!" bên dưới khối xuất hiện. Thứ tự chạy cụ thể là: đi tới điểm chính giữa màn hình → đặt kích thước → hiện → nói xin chào.

- **Lưu ý:** Khối lệnh của tôi chỉ chạy khi được gọi đến, nên thầy/cô nhớ sử dụng khối lệnh của tôi sau khi tạo. Nếu chỉ có phần định nghĩa khối của tôi thì khối lệnh này sẽ không được chạy

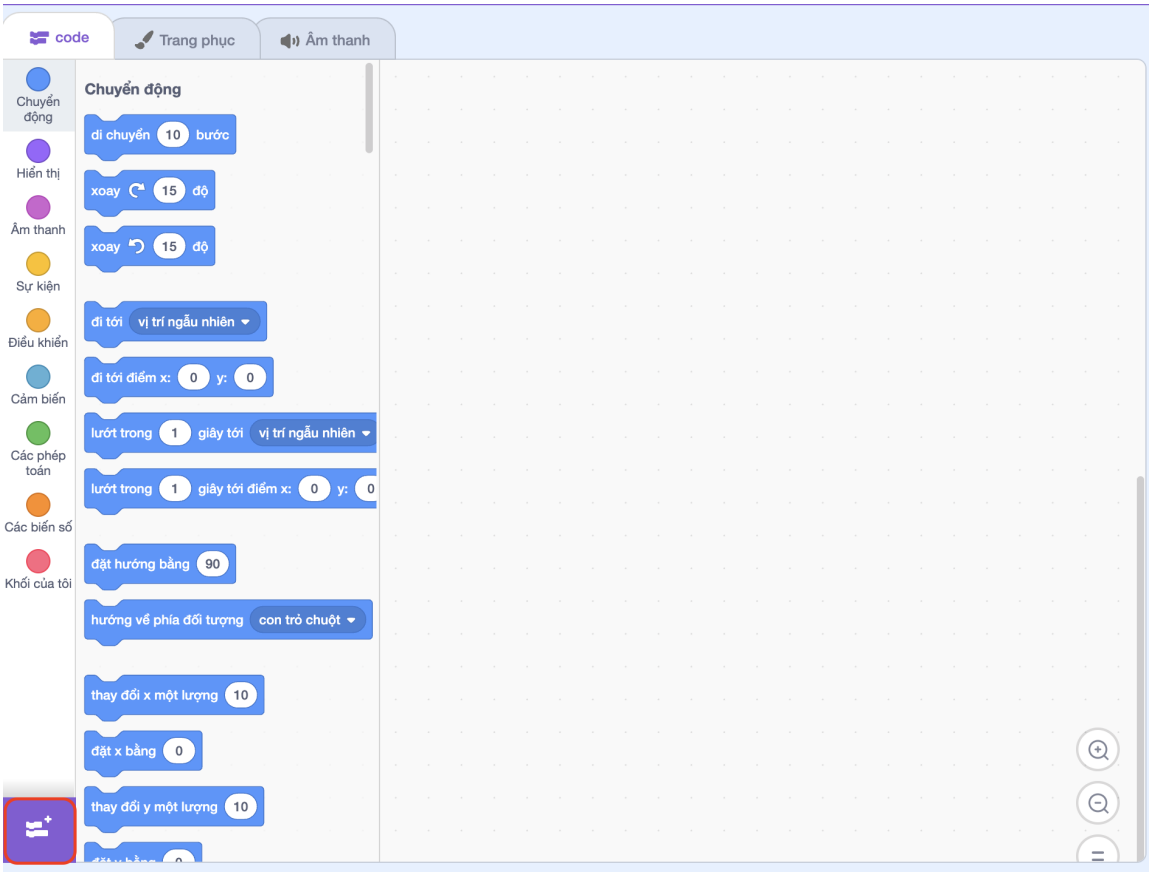
- Đây là đoạn chương trình giống như phần dùng khối lệnh xuất hiện ở trên, các khối lệnh cũng được chạy theo thứ tự cụ thể là: đi tới điểm chính giữa màn hình → đặt kích thước → hiện → nói xin chào.

- Thầy/Cô có thể thấy phần chương trình dùng khối lệnh xuất hiện nhìn rõ ràng và dễ hiểu hơn khi gọi chung các khối hành động. Đặc biệt khi tìm sắp xếp theo các khối của tôi, việc tìm bug sẽ được nhanh chóng và hiệu quả hơn.

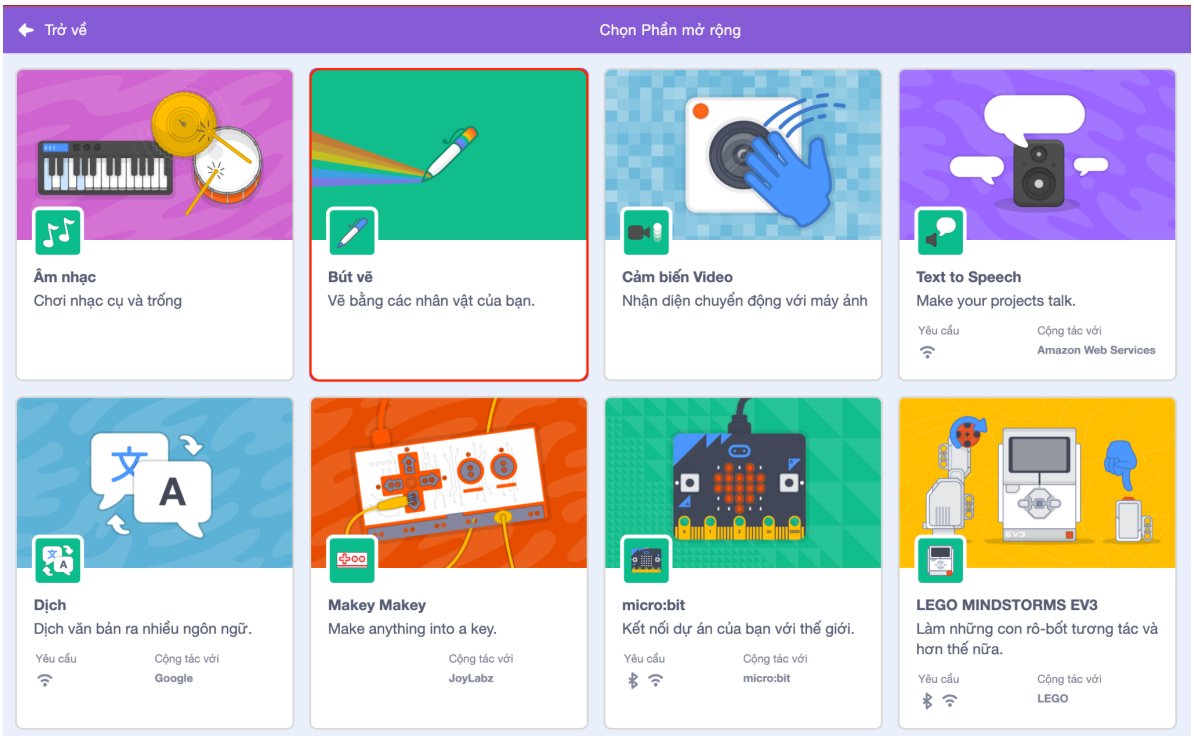
11. Bút vẽ:

11.1. Cách thiết lập hiển thị các khối bút vẽ trong Scratch:

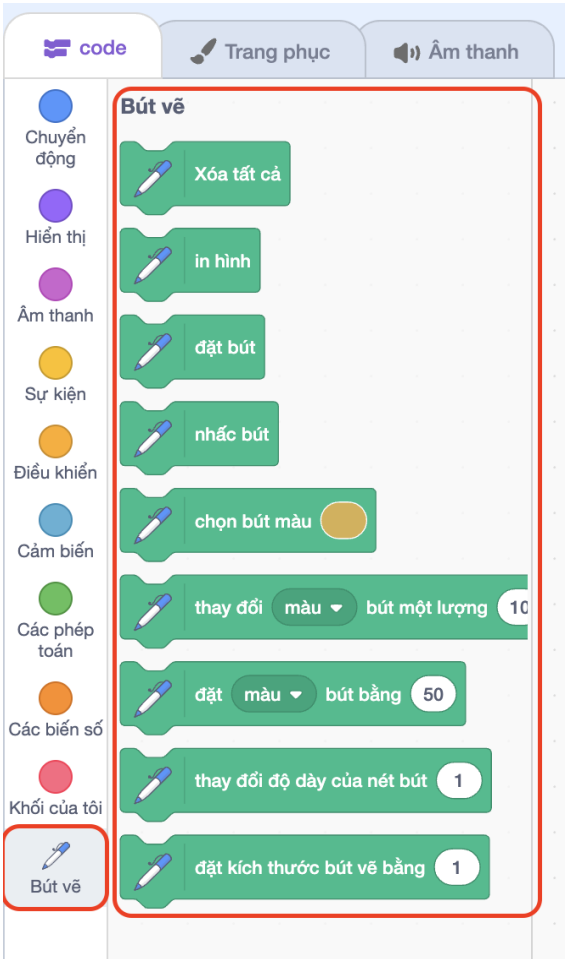
Bước 1: Chọn mục "Tạo thêm phần mở rộng" ở góc phía dưới màn hình bên trái



Bước 2: Chọn thêm Phần mở rộng "Bút vẽ"

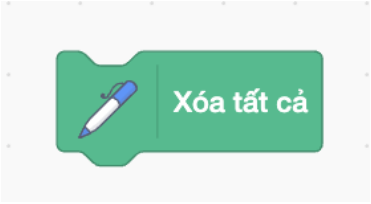


Bước 3: Phần mở rộng "Bút vẽ" đã được thêm vào trong màn hình code của thầy/cô



11.2. Chức năng của các khối lệnh trong "Bút vẽ":

a. Khối "Xoá tất cả":



- Khối **"Xoá tất cả"** có tác dụng xoá hết tất cả các nét bút mà đã được vẽ trên màn hình.

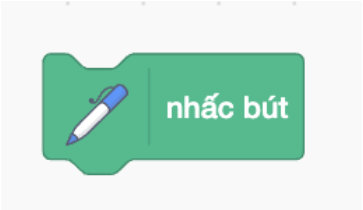
b. Khối "In hình":

- Khối **"In hình"** có tác dụng in lại hình ảnh của nhân vật ngay tại vị trí mà nhân vật đứng khi thực hiện câu lệnh. **Lưu ý:** hình ảnh bản sao được in lại đó không thể lập trình được mà chỉ đóng vai trò như một hình ảnh tĩnh của nhân vật được in lại trên sân khấu.

c. Khối "Đặt bút":

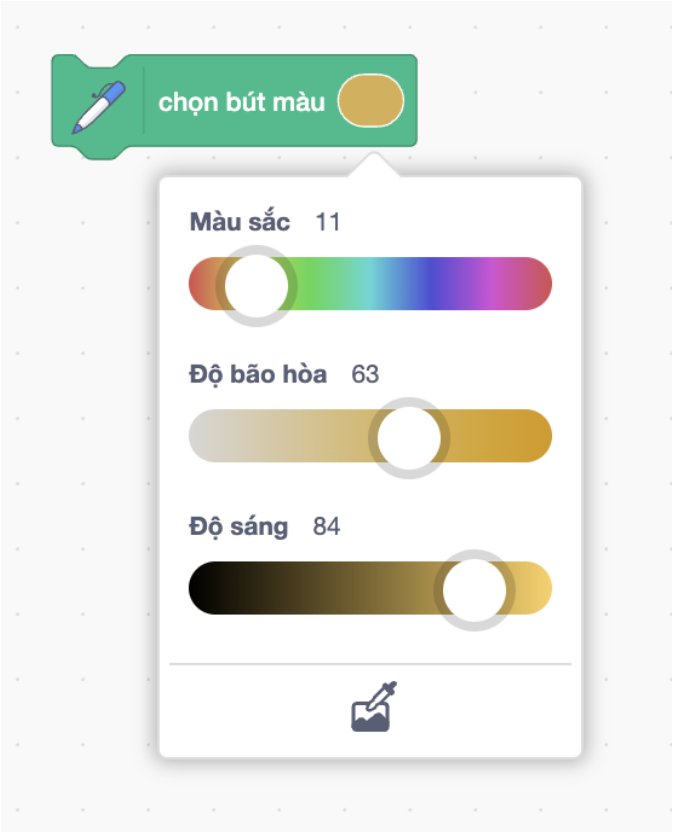
- Khối **"Đặt bút"** có tác dụng bật tính năng bút bên trong nhân vật. Câu lệnh này giống như hành động chúng ta đặt bút xuống giấy để bắt đầu viết. Nhân vật sẽ để lại dấu bút trên màn hình bất cứ nơi nào nó di chuyển cho đến khi nhấc bút.

d. Khối "Nhấc bút":



- Khối **"Nhấc bút"** giống như hành động chúng ta nhấc bút để ngừng viết.. Từ sau câu lệnh này nhân vật sẽ không để lại dấu bút khi di chuyển đến bất cứ đâu.

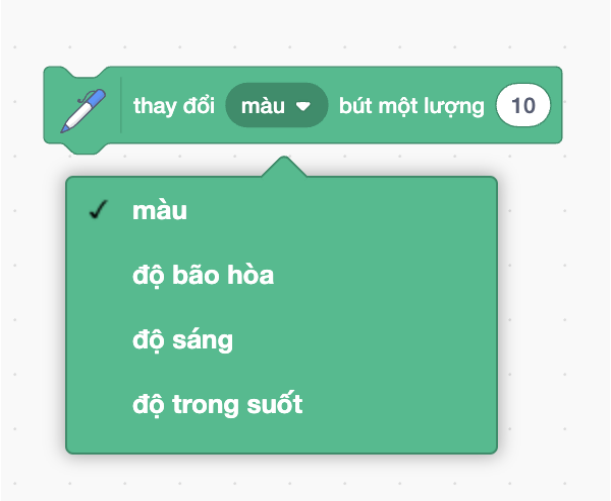
e. Khối "Chọn bút màu":



- Khối **"Chọn bút màu"** có tác dụng thay đổi màu sắc của bút theo ý thích của thầy cô.

- Thầy/Cô có thể kéo di chuyển trên các thanh thông số của màu sắc như **"Màu sắc", "Độ bão hoà", "Độ sáng"** để đạt được màu sắc theo ý mình muốn. Hình chọn màu giúp thầy cô chọn bất cứ màu nào trên sân khấu.

f. Khối "Thay đổi … bút một lượng …":

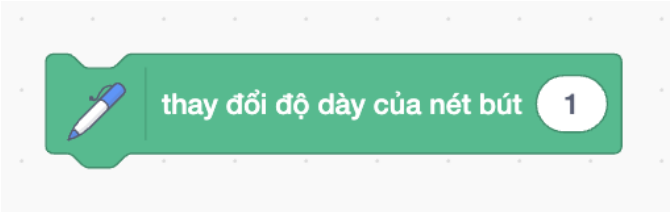


- Khối **"Thay đổi … bút một lượng …"** có tác dụng làm thay đổi lập tức các thông số của màu sắc ngay lập tức so với màu sắc hiện tại theo một lượng mà các thầy cô tuỳ chỉnh theo ý mình mong muốn..

g. Khối "Đặt … bút bằng …":

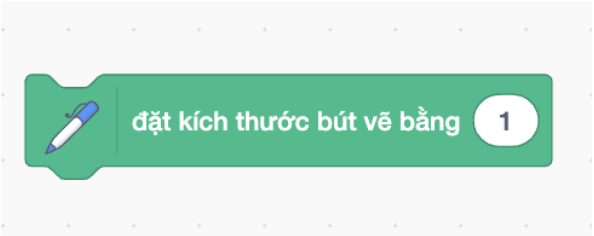
- Khối **"Đặt … bút bằng …"** có tác dụng chuyển đổi ngay lập tức các thông số của màu sắc từ các thông số hiện tại sang thông số mà các thầy cô đặt trong câu lệnh.

h. Khối "Thay đổi độ dày của nét bút …":



- Khối **"Thay đổi độ dày của nét bút …"** có tác dụng làm thay đổi độ dày của nét bút theo một lượng mà các thầy cô yêu cầu so với độ dày hiện tại của nét bút.

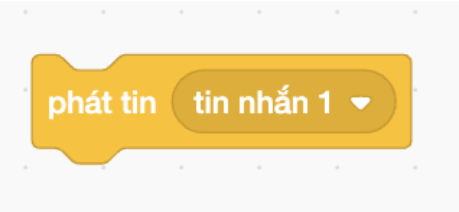
i. Khối "Đặt kích thước bút vẽ bằng …":



- Khối **"Đặt kích thước bút vẽ bằng …"** có tác dụng chuyển đổi ngay lập tức kích thước/độ dày của nét bút theo giá trị mà các thầy cô đặt trong câu lệnh.

12. Khối lệnh "Phát tin" và "Nhận tin":

12.1. Khối lệnh "Phát tin":

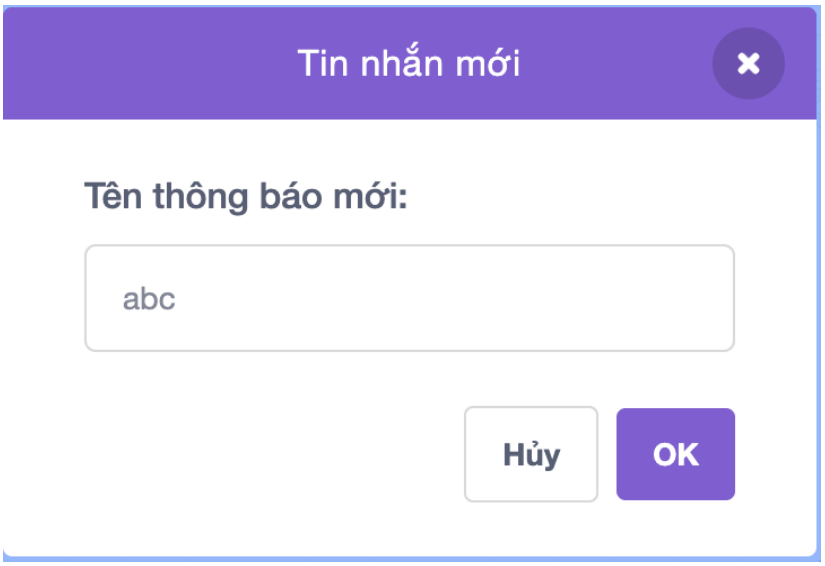


- Khối **"Phát tin"** sẽ giúp gửi ra một tin nhắn tới toàn bộ các nhân vật (bao gồm cả phông nền) trong dự án.
- Thầy/Cô có thể tạo tin mới theo các bước sau:

Bước 1: Chọn "Thông báo mới":

Bước 2: Đặt tên cho tin mới:

Lưu ý: Khi lập trình, các thầy cô nên đặt tên tin nhắn rõ ràng để tiện cho quá trình debug hoặc đọc-hiểu code sau này.



Bước 3: Chọn tin mới trong tuỳ chọn các tin:

12.2. Khối "Nhận tin":

- Khối **"Khi nhận tin ..."** là một khối lệnh **Sự kiện** có tác dụng kích hoạt các câu lệnh được gắn bên dưới nó khi nhận được tin nhắn phát ra bởi khối lệnh **"Phát tin"**.
- Nếu cùng một tin nhắn được gửi đi gửi lại trong đoạn chương trình nhận tin nhắn vẫn đang chạy, thì đoạn chương trình nhận tin nhắn cũng sẽ được chạy đi chạy lại và không thể đến được các khối lệnh ở cuối.
- Để tuỳ chọn các tin nhắn khác, thầy cô có thể thay đổi ngay trong khối lệnh bằng cách:

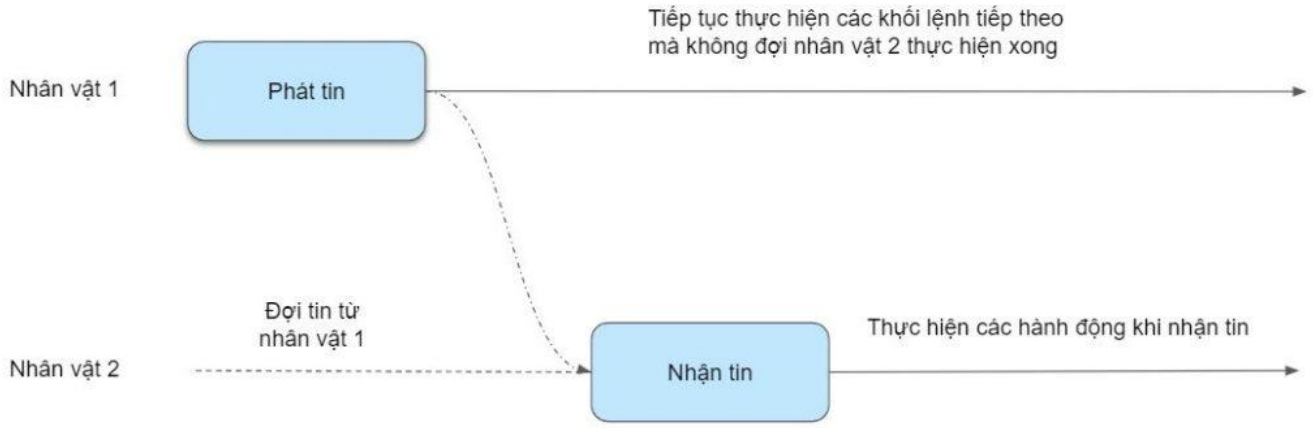
12.3. Phân biệt khối "Phát tin" và "Phát tin và đợi":

a. Điểm chung:

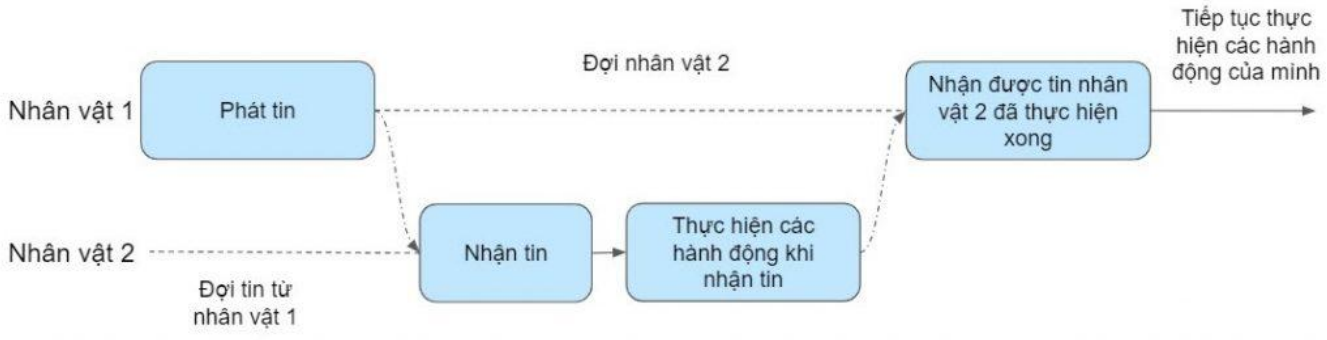
- Cả 2 khối lệnh **Phát tin** và khối lệnh **Phát tin và đợi** đều được dùng để **phát ra một tin** để một đoạn chương trình khác hoặc một nhân vật khác thực hiện hành động khi nhận được tin đó.
- **Ví dụ:** Trong ví dụ dưới đây, nhân vật Táo sẽ phát tin Chìa khóa xuất hiện khi chạm Miu. Nhân vật Chìa khóa sẽ có khối lệnh để nhận tin Chìa khóa xuất hiện và xuất hiện theo yêu cầu

b. Điểm khác nhau:

- Khối lệnh **Phát tin** sẽ **phát ra tín hiệu**, sau đó **tiếp tục thực hiện các khối lệnh tiếp theo đó mà không quan tâm liệu các hành động xảy ra khi nhận được tín hiệu đó có được thực hiện xong hay chưa.**
- Ở ngoài đời thật, việc phát tin tương tự như việc chúng ta báo cho thầy cô rằng chúng ta đã làm bài tập xong, sau đó tiếp tục làm các bài tập khác mà không cần đợi thầy cô

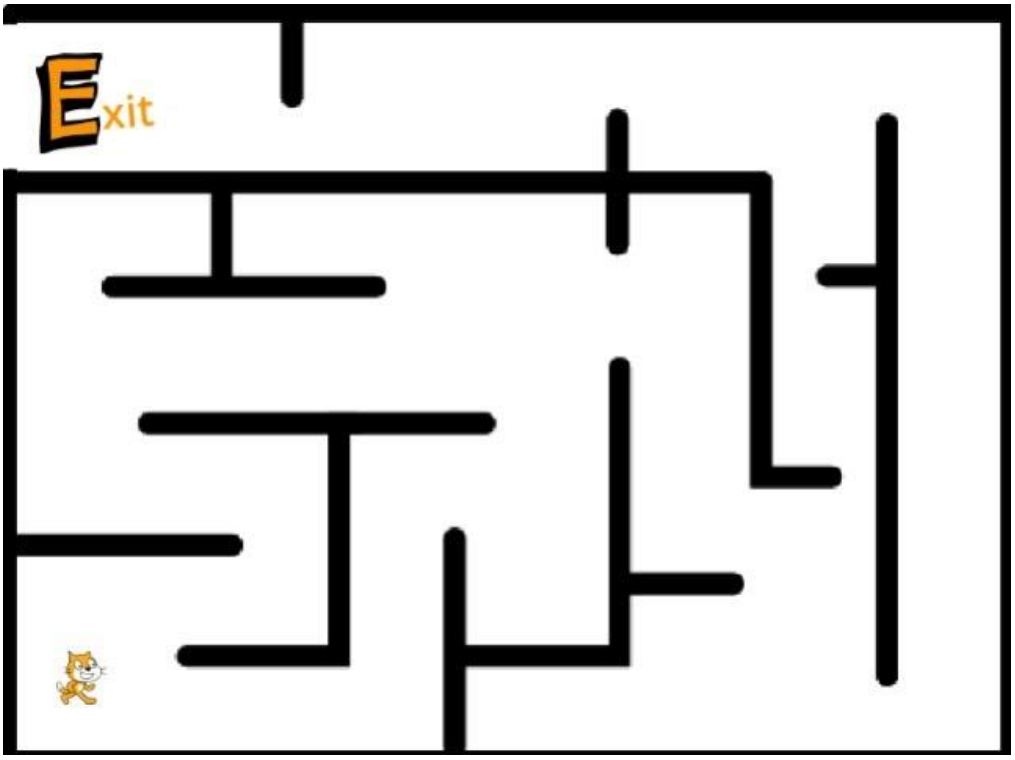


- Trong khi đó, khối lệnh **Phát tin và đợi** sẽ **phát ra tín hiệu**, sau đó **đợi cho các hành động xảy ra khi nhận được tín hiệu đó thực hiện xong rồi mới tiếp tục thực hiện** các khối lệnh tiếp theo đó.
- Ở ngoài đời thật, việc phát tin và đợi giống như chúng ta giơ tay (báo hiệu với các thầy cô rằng chúng ta muốn trả lời câu hỏi), sau đó sẽ đợi thầy cô. Chỉ khi nào thầy cô cho chúng ta phát biểu, thì chúng ta mới đứng lên trả lời câu hỏi.



c. Vậy khi nào chúng ta nên sử dụng khối lệnh nào?

- Chúng ta thử học cách sử dụng chính xác 2 khối lệnh này thông qua 2 ví dụ sau:
- Giả sự chúng ta đang muốn lập trình một trò chơi để Miu vượt qua mê cung và đi đến lối ra (Exit).



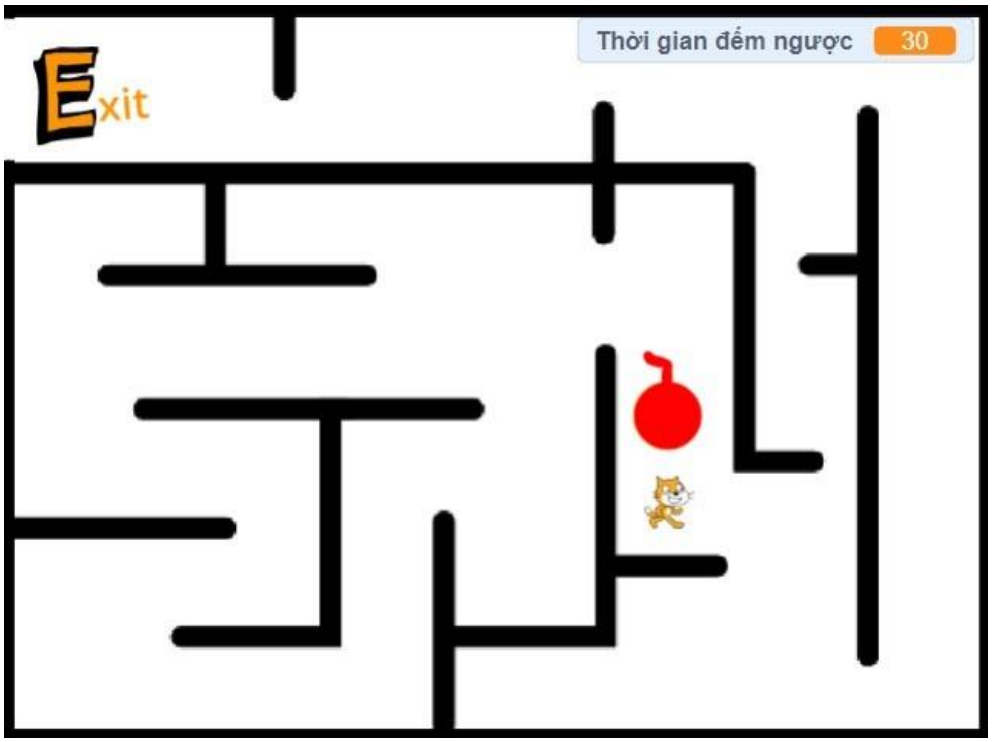
- Trong trò chơi này, xét 2 hành động sau:

- **Hành động 1:** Khi Miu đi tới Lối ra, Lối ra sẽ thông báo trò chơi chiến thắng trước khi dừng trò chơi. Khi nhận được tin trò chơi chiến thắng, nhạc chiến thắng sẽ phát ra trong vòng 10 giây, sau đó màn hình chiến thắng sẽ xuất hiện.

- Nếu chúng ta sử dụng khối lệnh Phát tin để lập trình hành động này của Lối ra, Lối ra sẽ dừng trò chơi ngay sau khi phát tin Chiến thắng. Điều này sẽ dẫn đến việc nhạc chiến thắng sẽ không kịp phát hết, và màn hình chiến thắng cũng sẽ không kịp xuất hiện.
- Nếu chúng ta sử dụng khối lệnh Phát tin và đợi để lập trình hành động này, sau khi phát tin chiến thắng, Lối ra sẽ đợi cho trò chơi phát nhạc chiến thắng xong, màn hình xuất hiện xong rồi mới dừng trò chơi

Vậy suy ra, khối lệnh đúng để lập trình cho hành động này là Phát tin và đợi

- **Hành động 2:** Trong khi tìm lối ra, Miu đã vô tình kích hoạt một quả bom. Miu thông báo Miu chạm bom. Khi nhận được tin Miu chạm bom, quả bom sẽ phát nổ trong vòng 30 giây nếu Miu không kịp thoát khỏi mê cung.



- Nếu chúng ta sử dụng khối lệnh Phát tin và đợi để lập trình hành động này của Miu, Miu sẽ thông báo Miu chạm bom, sau đó sẽ đứng yên đợi bom đếm hết 30 giây mới thực hiện được hành động tiếp theo. Như vậy là Miu sẽ không có cơ hội để thoát khỏi mê cung kịp thời gian rồi.
- Nếu chúng ta sử dụng khối lệnh Phát tin để lập trình hành động này, sau khi Miu thông báo Miu chạm bom, Miu sẽ không đợi bom đếm đến hết giờ. Thay vào đó, Miu sẽ tiếp tục việc tìm đường thoát khỏi mê cung của mình. Nhờ đó mà Miu kịp thoát khỏi mê cung trước khi bom phát nổ

Vậy suy ra, khối lệnh đúng để lập trình cho hành động này là Phát tin.