

Animation *****

1. Que veux-tu animer? Les lettres de ton nom? Fais un lutin (ou *sprite*) par lettre. Une histoire? Utilise les costumes pour animer les lutins.
2. Le joueur fait-il quelque chose ou tout se fait tout seul et il regarde?
3. Musique? Une musique différente par lutin? Utilise « *envoyer à* » et « *quand je reçois* » pour qu'un lutin « dise » à un autre d'agir (par exemple quand le loup souffle la maison de paille, il envoie un message à tous et la maison s'envole alors que les cochons se sauvent).
4. Enregistre toutes les répliques de ton histoire...! Compose ta musique avec Scratch.

Jeu style Pac Man (qui attrapera quoi? Laisse aller ton imagination): *****

1. Crée un lutin qu'on peut faire bouger avec les flèches
2. Crée de la « nourriture » que le lutin mangera. Il faut que la nourriture disparaisse si le lutin la touche. Il doit y avoir plusieurs parts de nourriture.
3. Crée un parcours (une sorte de labyrinthe) où le lutin rebondit quand il le touche. (Indice : crée ce labyrinthe comme un nouveau lutin.)

Tu veux faire plus?

4. Crée une variable qui compte les points; crée une variable qui garde le plus haut score (donc il faut aussi que tu fasses arrêter la partie à un moment donné!).
5. Crée un deuxième personnage qui aide ou est en compétition avec le tien (compte ses points à lui aussi) et affiche le gagnant à la fin du jeu.
6. Crée un personnage (ou plus) qui fait diminuer la « vie » de ton lutin. Tu peux lui dire de toujours avancer en direction de ton lutin.

Jeu de danse *****

1. Crée des lutins qui dansent quand une condition est vraie (ex. : si variable *danse* =1).
2. Crée deux boutons : un qui arrête la danse si on appuie dessus avec la souris (voir *contrôle*). Modifie l'apparence du bouton quand il est actif.
3. Faut faire jouer de la musique quand on danse!

Tu veux faire plus?

4. Utilise des photos de toi pour les lutins dansant.
5. Pense à un éclairage intéressant, peut-être changeant? Et le décor? Les décors?

Jeu de cibles *****

1. Crée un canon en bas de l'écran dont le joueur peut changer la direction (avec les flèches? Avec la souris?).
2. Crée un projectile qui est lancé par le canon : il part du canon, est lancé dans la même direction. (quand? Si on appuie sur « espace »? sur le bouton de la souris?) Indice : le canon peut mettre sa position et direction dans des **variables** que le projectile utilise pour les connaître.
3. Qu'arrive-t-il quand la balle touche le bord? Quand elle touche la cible?
4. Crée des cibles qui disparaissent quand elles sont touchées (et réapparaissent pour se réutiliser). Elles bougent comment, ces cibles? Lancent-elles des projectiles aussi?

Quiz *****

1. Crée des lutins qui représentent chacune des réponses (A, B, C, D, E)
2. Crée des fonds d'écran où tu écris des questions et des choix de réponses; assure-toi que les réponses sont bien disposées par rapport aux lettres (le choix A est près de la lettre A, etc.)
3. Crée une variable *question* qui « sait » à quelle question on est rendu et assure-toi que les fonds d'écran sont les bons (à la question 1, c'est le fond no 1).
4. Organise ton code : à la question 1, on doit attendre quelle réponse? Quand on appuie sur la bonne lettre, que se passe-t-il? Augmente-t-on les points? Félicite-t-on le joueur? On attend un peu puis on passe à la question suivante.

Tu veux faire plus?

5. Fais un fond d'introduction, un fond de fin de jeu. Musique? « top score »?
6. As-tu organisé ton code pour pouvoir ajouter des questions sans avoir à trop travailler dans le code? Par exemple, on utilise « ajouter 1 à la variable » plutôt que d'avoir plusieurs lignes où on écrit « attribuer 5 à la variable » et ensuite « attribuer 6 à la variable » et « attribuer 7 à la variable », etc.

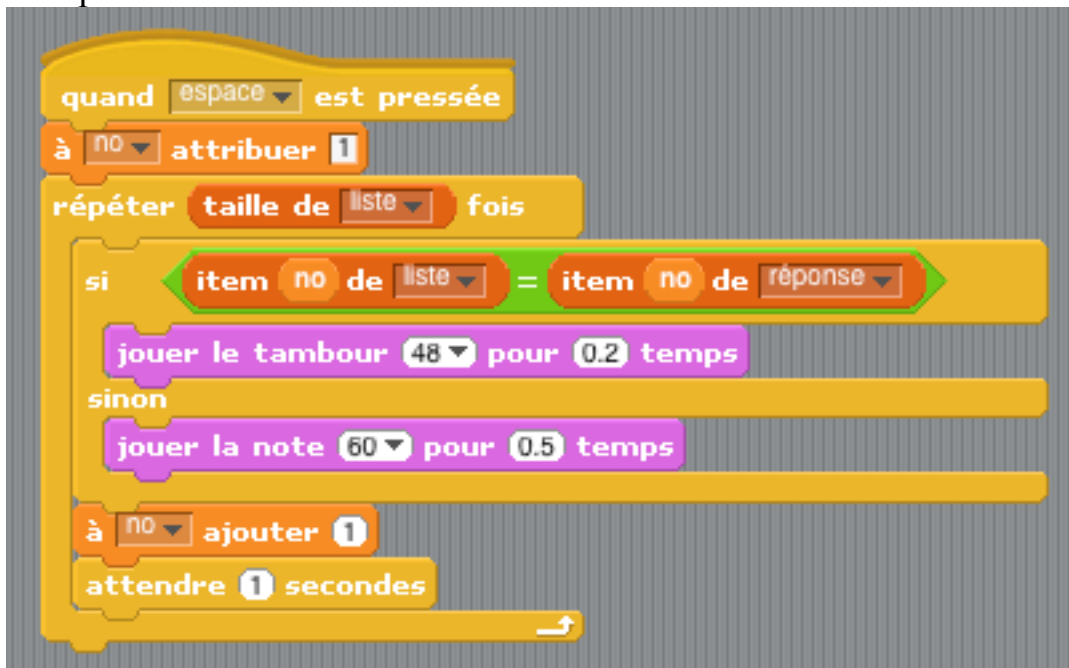
Quiz de multiplication *****

1. Crée trois variables : les 2 nombres à multiplier et la réponse du joueur.
2. Sur le fond d'écran écris la question et place les variables au bon endroit.
3. Donne des valeurs aléatoires (au hasard) aux 2 nombres; attend que la *réponse* soit bonne; félicite, attend un peu et recommence.
4. Il faut savoir attraper la réponse du joueur. Utilise une liste, sinon voici une autre façon :
quand le joueur entre un nombre (disons 4), il faut prendre l'ancienne réponse (disons 5) la transformer en dizaine et ajouter le nombre (ça donne 54). S'il veut effacer sa réponse, il appuie sur espace?
5. Crée une variable qui contient la bonne réponse et un lutin qui sera un bouton pour afficher la bonne réponse.
6. Fais avancer un petit lutin à chaque bonne réponse pour encourager le joueur.

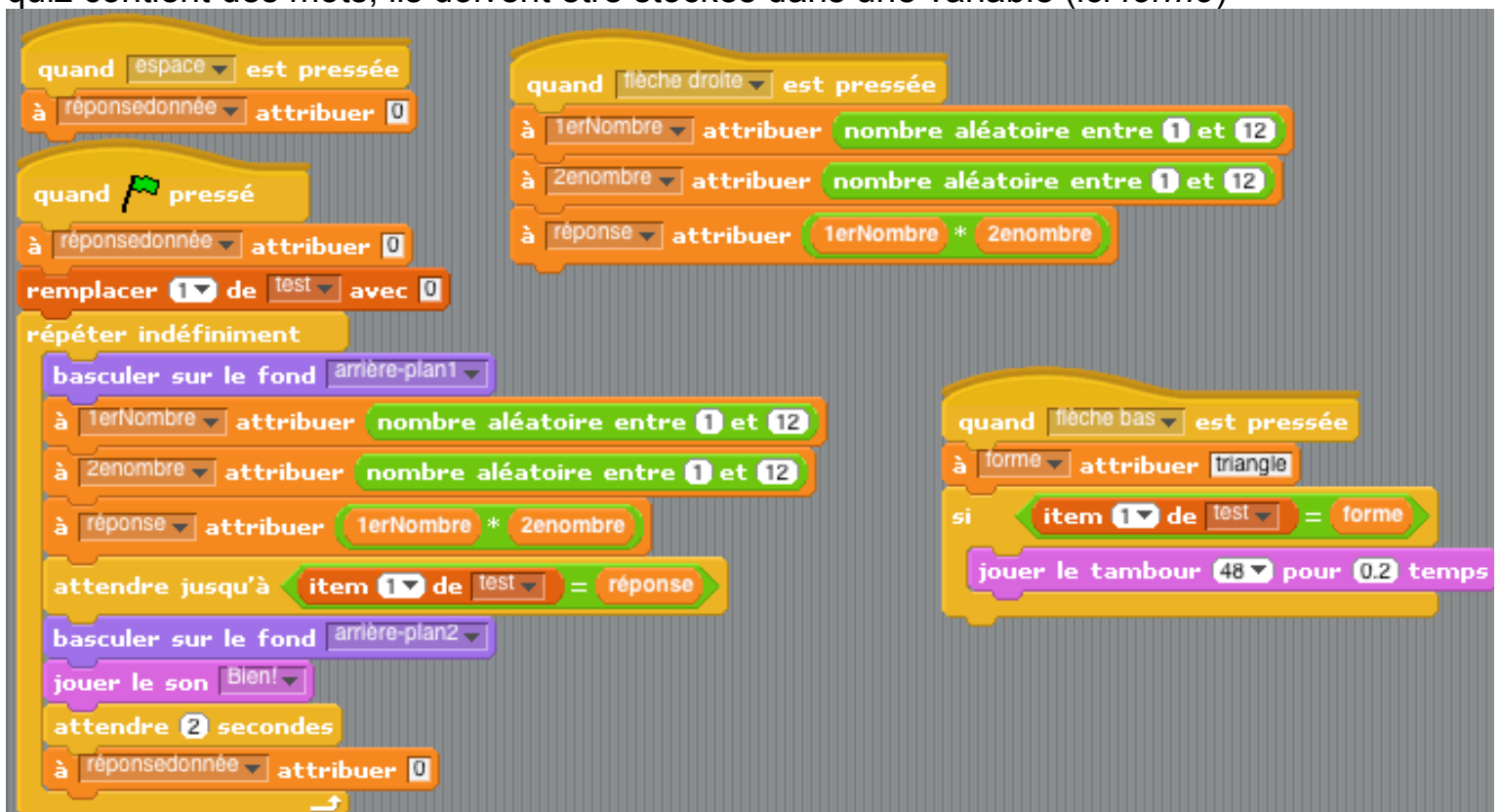
Jeu de « tag » (idée de Marie L.) *****

1. Crée 2 lutins contrôlés par les joueurs; de plus, crée un lutin qui représente la tag.
2. La tag doit toujours *aller vers la position* du lutin qui a la tag. Comment savoir qui a la tag? La tag doit changer quand les lutins se touchent. Tu peux créer une variable *x* qui augmente chaque fois qu'ils se touchent, et si *x* est pair la tag se colle à un lutin, si *x* est impair, la tag se colle à l'autre (un nombre *x* est pair si $x \bmod 2 = 0$, il est impair si $x \bmod 2 = 1$; « mod 2 » veut dire « le reste de la division par 2 »).
3. Compte les points, fais un « top score ».
4. Pour pouvoir jouer seul, donne un code à un des lutins qui le fait rebondir tout seul (ou même essayer de te suivre en *glissant* vers toi).

Comparer 2 listes :



Une réponse possible pour le quiz de multiplication : le script « flèche bas » à droite ne sert à rien mais montre comment comparer un élément de liste en mot (en nombre c'est plus facile) avec ... un mot! Il faut créer une variable qui contient le mot parce qu'on ne peut pas écrire « triangle » dans le « trou » du losange vert qui compare (avec =). Donc si notre quiz contient des mots, ils doivent être stockés dans une variable (ici *forme*)



Page suivante, vous trouverez des solutions à quelques problèmes ci-haut, pris sur le site de nebo music.



[PTA Performances](#)

[Fine Arts ASP Weekly Schedule](#)

[Music Links](#)

[Exercises](#)

[Sub Plans](#)

[Computer Help](#)

[Nebo Music Curriculum Overview](#)

[Recorder](#)

[Nebo Specials Schedule](#)

[Nebo Music Calendar](#)

[Nebo Fine Arts Home](#)

[Rhythm Composing Project](#)

[4th Grade Instrument Project Template](#)

[5th Grade Composer Project Template](#)

[5th Grade Listening Links for Composer Project](#)

Chasing/Eating: (Pac Man Type Game)

Move Right:

```
when right arrow key pressed
  point in direction 90
  move 10 steps
```

Move Left:

```
when left arrow key pressed
  point in direction -90
  move 10 steps
```

Move Up and Down:

```
when up arrow key pressed
  point in direction 0
  move 10 steps
```

```
when down arrow key pressed
  point in direction 180
  move 10 steps
```

Bouncing of wall or Maze:

```
when clicked
  forever if touching maze
    turn 180 degrees
    move 10 steps
```

Combined Commands:

```
when right arrow key pressed
  point in direction 90
  move 10 steps
```

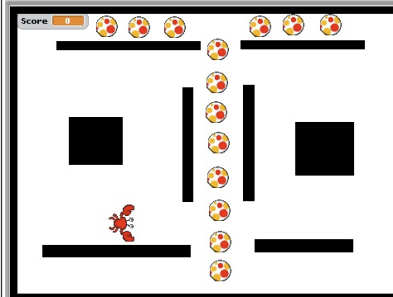
```
when left arrow key pressed
  point in direction -90
  move 10 steps
```

```
when up arrow key pressed
  point in direction 0
  move 10 steps
```

```
when down arrow key pressed
  point in direction 180
  move 10 steps
```

```
when clicked
  forever if touching maze
    turn 180 degrees
    move 10 steps
```

Basic Pac Man Stage:



"Food Scripts" Scoring and Hiding:

```
when clicked
  show
  set Score to 0
  forever if touching crab
    hide
    change Score by 1
```

[Email Mr. Michaud](#)

[Chorus](#)

[ASP Dance](#)

[ASP Drama](#)

[ASP Orchestra](#)

[ASP Broadcast](#)

[Nebo Fine Arts Home](#)

[Nebo Elementary Home](#)

[Paulding County Public Schools](#)





**Teamwork
Creativity
Excellence**

**Nebo Elementary School
Department of Fine Arts**

[PTA
Performances](#)

[Fine Arts ASP
Weekly
Schedule](#)

[Music Links](#)

[Exercises](#)

[Sub Plans](#)

[Computer
Help](#)

[Nebo Music
Curriculum
Overview](#)

[Recorder](#)

[Nebo
Specials
Schedule](#)

[Nebo Music
Calendar](#)

[Nebo Fine
Arts Home](#)

[Rhythm
Composing
Project](#)

[4th Grade
Instrument
Project
Template](#)

[5th Grade
Composer
Project
Template](#)

[5th Grade
Listening
Links for
Composer
Project](#)

[Email Mr. Michaud](#)

Red Light Green Light:

Stage:



Dancer:



Green Light:



Red Light:



Variables:



[Chorus](#)

[ASP Dance](#)

[ASP Drama](#)

[ASP Orchestra](#)

[ASP Broadcast](#)

[Nebo Fine Arts Home](#)

[Nebo Elementary Home](#)

[Paulding County Public Schools](#)





[PTA Performances](#)

[Fine Arts ASP Weekly Schedule](#)

[Music Links](#)

[Exercises](#)

[Sub Plans](#)

[Computer Help](#)

[Nebo Music Curriculum Overview](#)

[Recorder](#)

[Nebo Specials Schedule](#)

[Nebo Music Calendar](#)

[Nebo Fine Arts Home](#)

[Rhythm Composing Project](#)

[4th Grade Instrument Project Template](#)

[5th Grade Composer Project Template](#)

[5th Grade Listening Links for Composer Project](#)

Pong Game

Pong Stage:

The Pong Stage interface includes a 'ball' sprite with coordinates x: -39, y: -27, and direction: -89. It features a 'Score' display showing 0. The scripts area contains two event-driven loops: one for when the green flag is clicked, setting a random direction (0 to 360) and moving the ball 5 steps; and another for when the ball touches the paddle, calculating a new direction based on the touch point and moving the ball 5 steps.

Pong Paddle:

The Pong Paddle interface shows a 'paddle' sprite with coordinates x: -190, y: 5, and direction: 180. The scripts area contains a single event-driven loop: when the green flag is clicked, a forever loop that sets the paddle's y-position to the mouse's y-position.

Pong Ball:

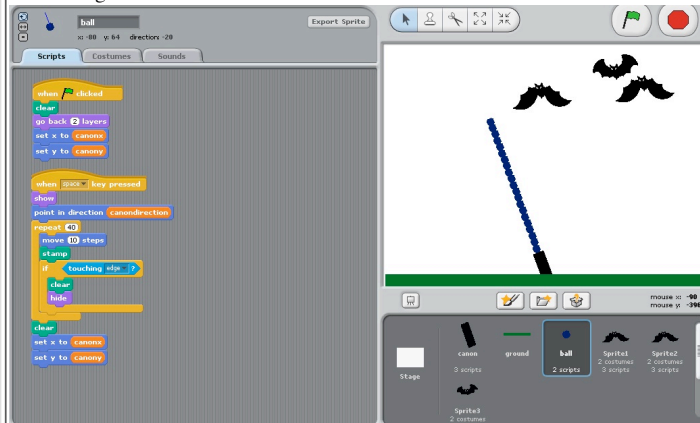
The Pong Ball interface shows a 'ball' sprite with coordinates x: -18, y: -9, and direction: 58. The scripts area contains two event-driven loops: one for when the green flag is clicked, setting a random direction (0 to 360) and moving the ball 5 steps; and another for when the ball touches the paddle, calculating a new direction based on the touch point and moving the ball 5 steps.

[Email Mr. Michaud](#)

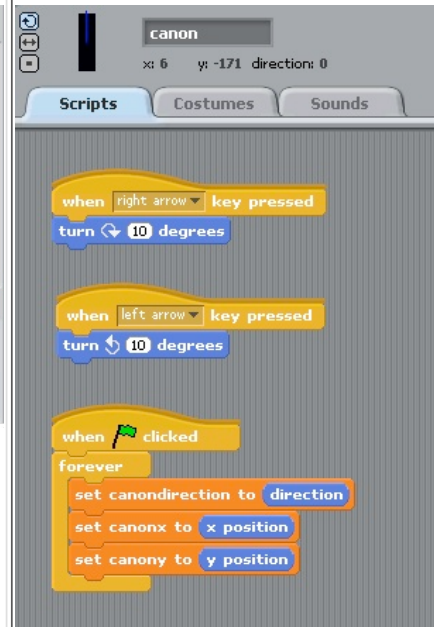


Target Game

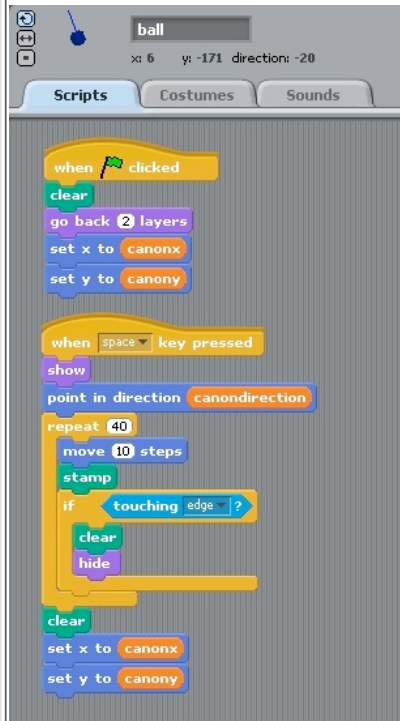
Canon Stage:



Basic Canon:



Basic Canon Ball:



Canon Variables:



Canon Target:



Sprite1

Export Sprite

x: -134 y: 141 direction: 90

Scripts

Costumes

Sounds

when clicked

forever

glide 1 secs to x: pick random -250 to 250 y: pick random 100 to 150

when clicked

forever

next costume

wait pick random 0.2 to 1 secs

when clicked

forever

if touching ball ?

hide

wait 5 secs

show

[Email Mr. Michaud](#)

[Chorus](#)

[ASP Dance](#)

[ASP Drama](#)

[ASP Orchestra](#)

[ASP Broadcast](#)

[Nebo Fine Arts Home](#)

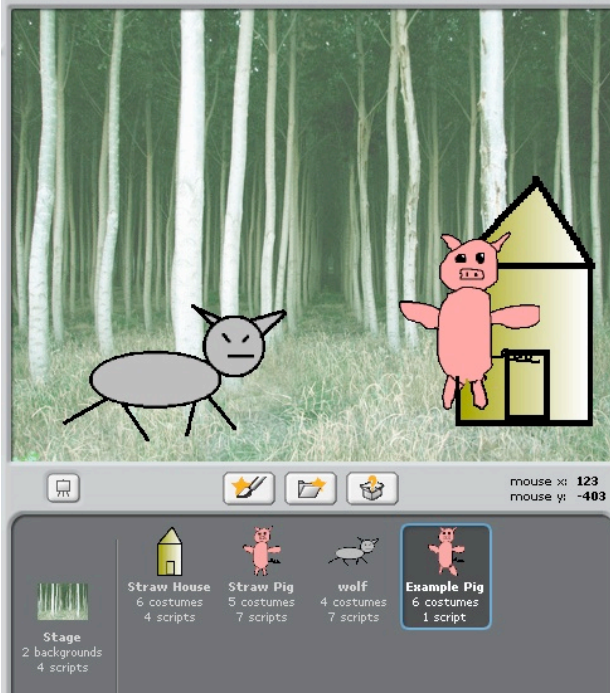
[Nebo Elementary Home](#)

[Paulding County Public Schools](#)



Basic Animation Project

Stage and Sprites:



Go Home Command:



Basic Walk Right:



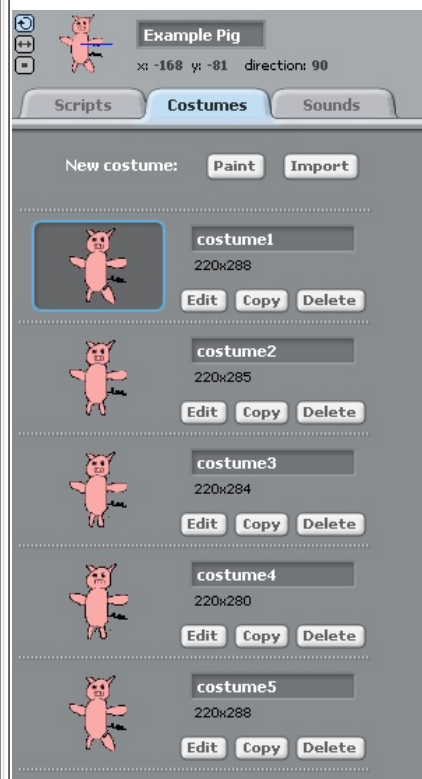
Repeated Walking Right:



Play Sound and Wait:

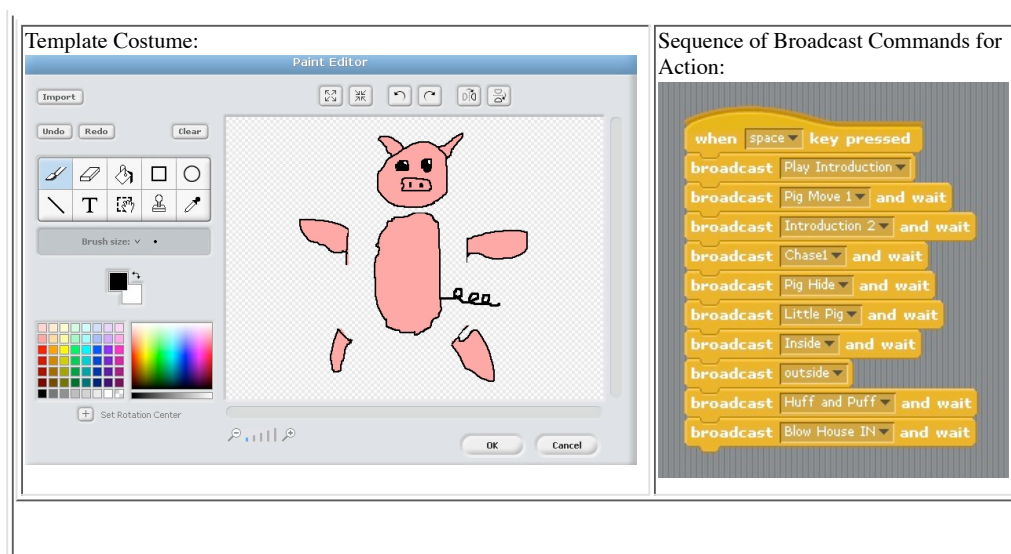


Costumes for Walking:



Combined Action and Sound:





[Email Mr. Michaud](#)

[Chorus](#)

[ASP Dance](#)

[ASP Drama](#)

[ASP Orchestra](#)

[ASP Broadcast](#)

[Nebo Fine Arts Home](#)

[Nebo Elementary Home](#)

[Paulding County Public Schools](#)

