LESSON 8 ROBOT PROJECT-FLY ELEPHANT GAME!

Lesson Overview

Students will practice using ultrasonic module to make a new game - fly elephant game.

Lesson Target

1. Practice the use of variable data and ultrasonic sensor.
2. Learn the knowledge of coding block broadcast, costume, random number, wait until.
3. Practice the ability of finding out problem and solving problem during coding.

Lesson Tag

GRADE LEVEL	SUBJECTS	DIFFICULTY	DURATION	GROUP
Elementary, middle	STEAM, computer science Math	Beginner	180 mins	1-2 students

Supplies

Robot	Accessories	Other Material	Tools Used
WeeeBot Kit	USB cable	PC with WeeeCode software USB port required;	

Lesson Outline

INTRO: Show the picture of flappy bird game, or let students play the flappy bird game. Lead students summarize objects in this game and functions. Introduce students about coding block broadcast, costume, random number, wait until. (60 mins)

CREATE: Create a fly elephant game. (70 mins)

PLAY: Each group tests, then records learnings from their invention. Students explore how their invention works, plus the coding concepts behind it. (30 mins)

REMIX: Students will customize and enhance their inventions to create fly elephant game. (20 mins)

Routine

1. INTRODUCE FLY ELEPHANT GAME AND SUMMARIZE FOR EACH OBJECT.

Objects in fly elephant game

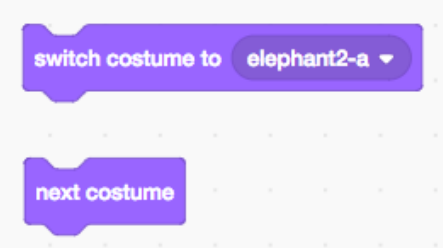
Show picture of flappy bird game to students, or let students plan flappy bird game, and then share and summarize the function of each object.

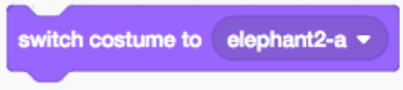
Button	1. Will be enlarged when touched by mouse pointer
	2. Once be clicked, game begins.
Waterpipe	1. Appear from right of screen and moving to left.
Bird	1. Be controlled moving up and down to avoid obstacle
	2. Score when avoided tube, game over when touched tube.

2. HARDWARE AND SOFTWARE INTRODUCTION

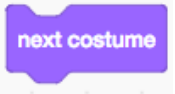
Coding block costume

In flappy bird game, the bird will flap wings during fly. This animation is formed by several movements, combine those movements we can see an action animation. One sprite's different movements are called costume, in WeeeCode, there are two coding blocks to control the change of costume in category "Looks".





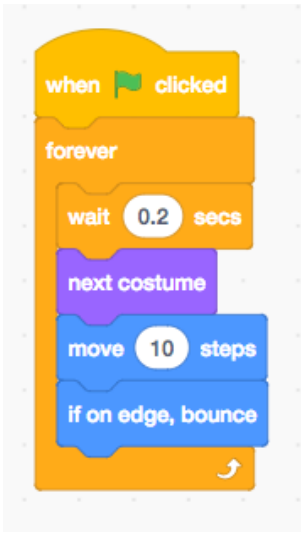
: every costume has its name, please check in costume list. This coding block will switch costume to the assigned movement.



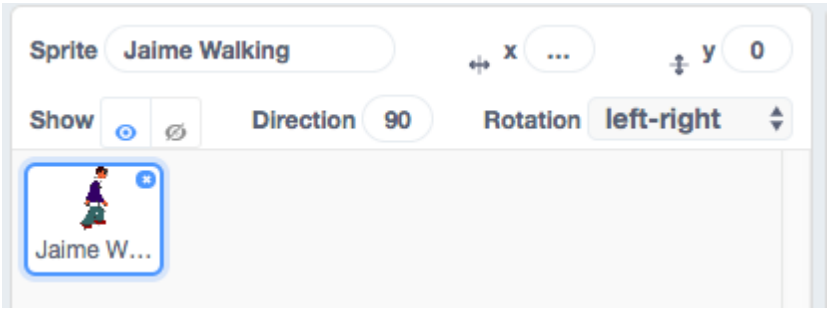
: sprite will switch costume in turns of costume list.

Exercise:

1. Click "add sprite", open sprite library and put mouse pointer on different sprite, we can see all costume of the pointed sprite.
2. Select "Jaime Walking", click "Costumes" and we can find all 5 costume of this sprite.
3. Code as below, observe.



Result: sprite will walk on stage, when touch the edge of stage the sprite will turn upside down. The reason why it happened is because the rotation mode of sprite is “all around” instead of “left-right”. Change the setting and sprite will walk back and forth.



Coding block broadcast

In WeeeCode, broadcast is mainly used as a command between sprites. Generally, we use broadcast command in below three ways:

1. One to one broadcast, means sprite 1 use broadcast to control sprite 2.
2. One to a group broadcast, means sprite 1 use broadcast to control multiple other sprites at the same time.
3. A group to one broadcast, means multiple sprites can send broadcast to control sprite 1.

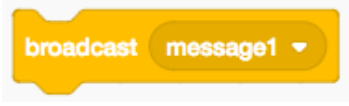
Coding block broadcast is very important to coordinate multiple sprites.

In category “Events”, we can see below three coding block related to broadcast:

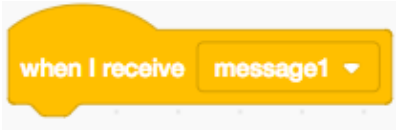
1. 

: broadcast a message to other sprite as a trigger, wait until other sprites finish the preset task, and then continue.

WEEEEMAKE

2. 

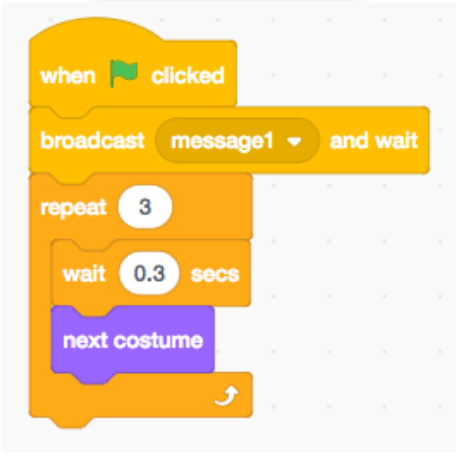
: broadcast a message to other sprite, and then continue.

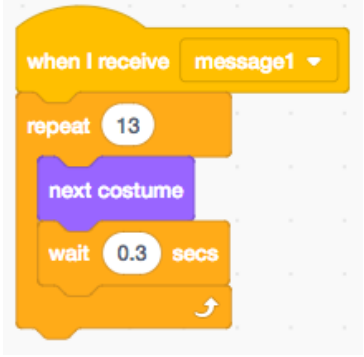
3. 

: work with broadcast command, once receive message it will be a start event.

Exercise:

1. Add sprites “AZ Hip-Hop” and “Breakdancer1”.
2. Code as below.





Result: Once running program, “Breakdancer1” will broadcast message 1 and wait, “AZ Hip-Hop” will start switch costume, once all 13 costumes finished, “Breakdancer1” will switch its costumes.

Exercise:

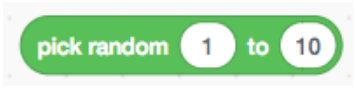
What if change the code of “Breakdancer1” from “broadcast messageX and wait” to “broadcast messageX”?

Result: on the dropbox of coding block “broadcast” and “broadcast…and wait”, we can create “New Message”. Please pay attention to mark the same name on coding block “When I receive…”.

WEEEEMAKE

Coding block random number

In category “Operators”, we can find the coding block random number. In this coding block, two blanks can be modified, we should fill in numbers in those two blanks to get a range. This coding block will pick random number in the range we set.



Exercise:

Write below codes and observe the result.

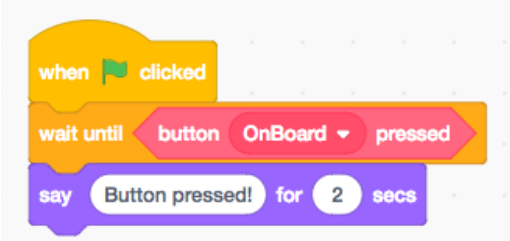
Result: It will keep generating number between 1 to 10 (including 1 and 10).

Coding block wait until…

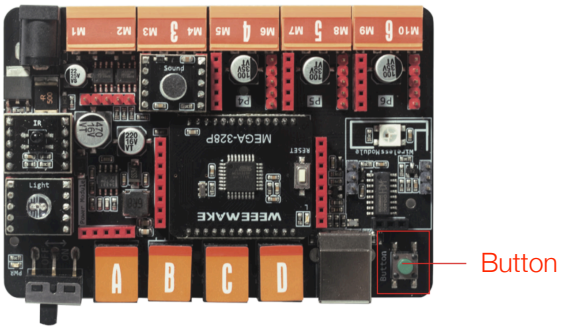
In category “Control”, we can find a coding block “wait until …”. The function of this coding block is: wait until the condition in blank is satisfied (the blank here is a decision box), running codes after this command; otherwise, program will be pending. Generally, this coding block is used when a sprite must wait for a certain event.

Exercise:

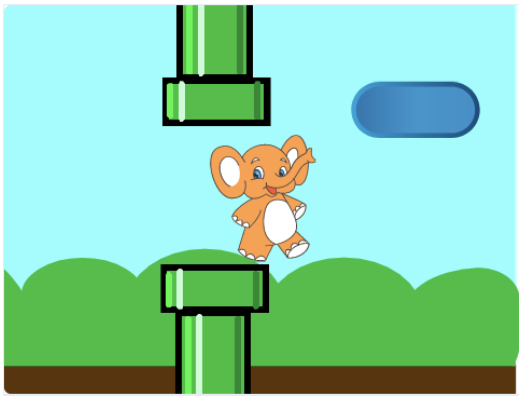
1. Connect USB cable, restore online firmware, robot enter online control mode.
2. Write codes below and observe result.



Result: Running codes, press button on mainboard ELF, and you will see below result on stage.



3. WRITE CODE FOR FLY ELEPHANT GAME

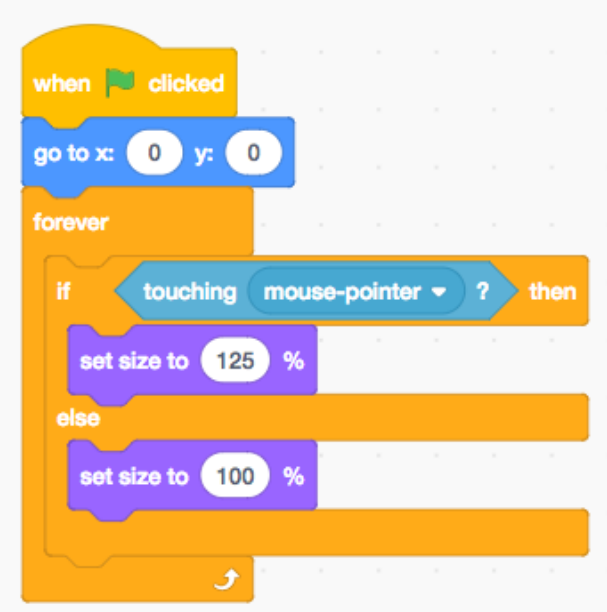


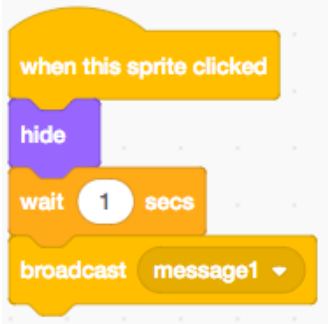
Set backdrop and sprite

1. Click “Add backdrop” and select “blue sky” as background. (You can choose other backdrop.)
2. We already have default sprite “Elephant2”, add sprite “Waterpipe1” and “Button2” as below. (Sprite position may be different.)

Code for sprite “Button2”

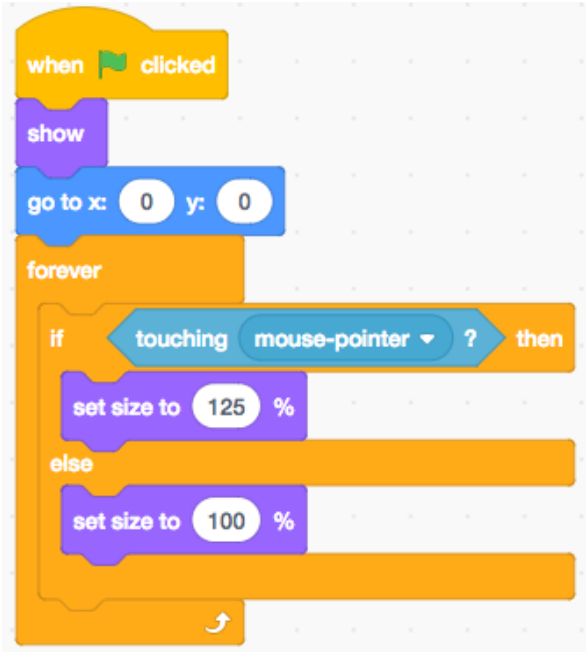
1. Set the initial position of “Button2”, and then write codes for its animation. The result should be: when it’s touched by mouse pointer, it will be enlarged; when mouse pointer left, it will convert back into normal size.
2. Write codes for “Button2”, the result should be: when it’s clicked, it will vanished and game starts.

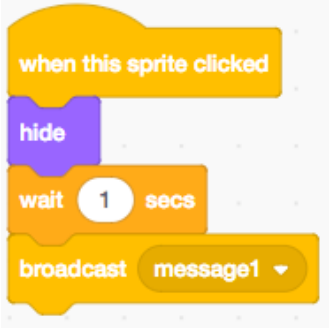




WEEEEMAKE

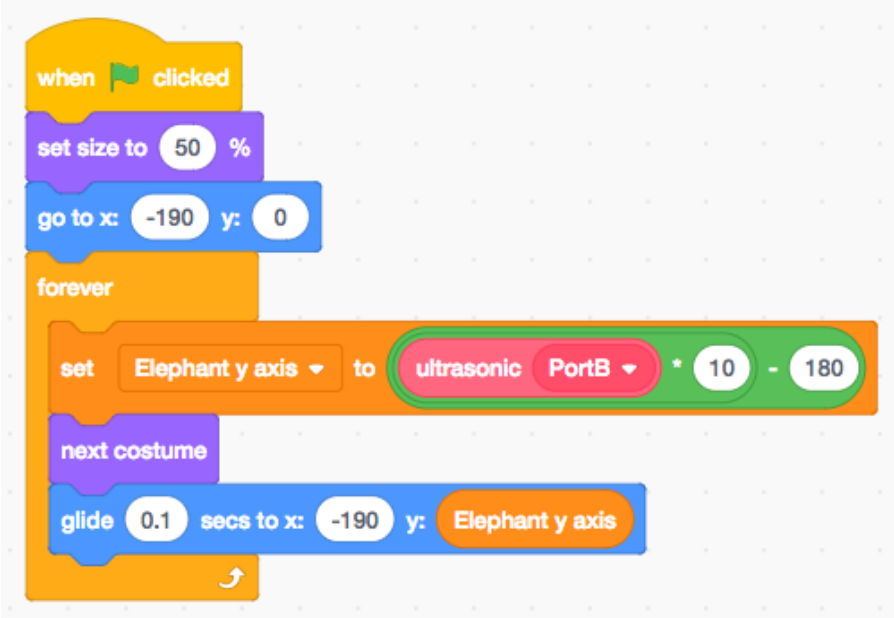
Result: Running codes, when we click “Button2”, the button will be hided and broadcast a message to other sprites, game starts. However, when we click “Button2” and then running codes all over again, “Button2” is disappeared. It is because that we don’t show this sprite again after hiding it. The optimized program as below.





Code for sprite “Elephant2”

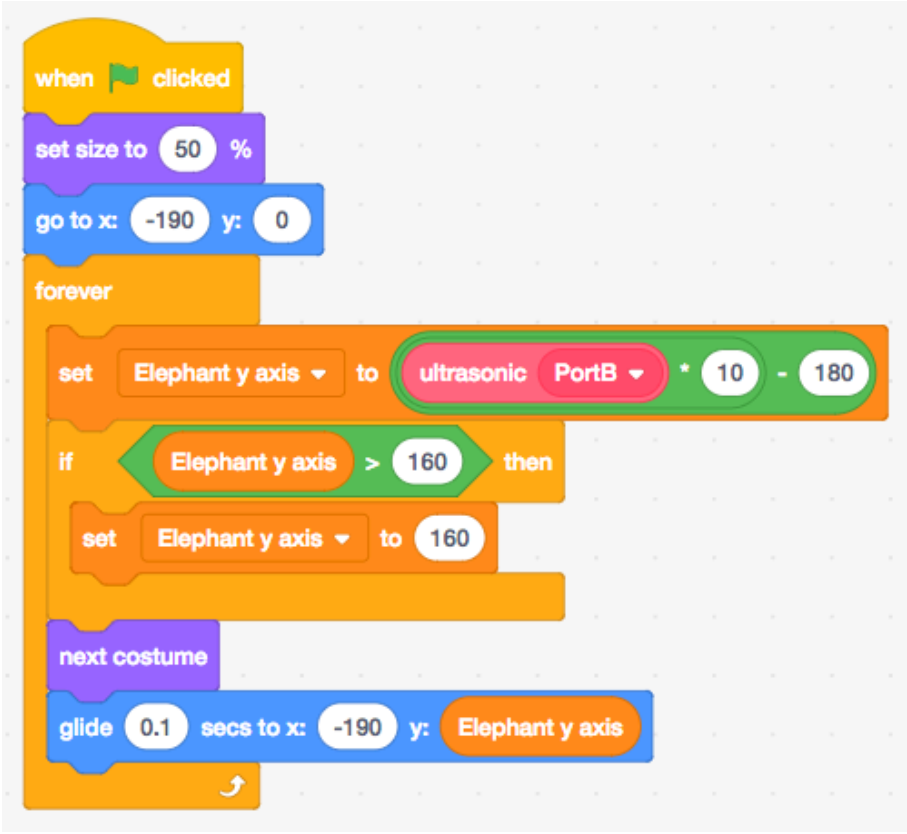
1. Set the initial position and size of sprite “Elephant2”, write codes for it as below.



Result: Running codes, “Elephant2” move to the left of stage. Students use hand or other object to control the distance between ultrasonic sensor, and control “Elephant2” move up and down. The distance measured by ultrasonic is longer, the “Elephant2” fly higher.

WEEEEMAKE

We find a problem, that when the distance value of ultrasonic is over 160, “Elephant2” will exceed the top of stage. So we need to set a limit of the maximum value of variable data Elephant y axis as below.

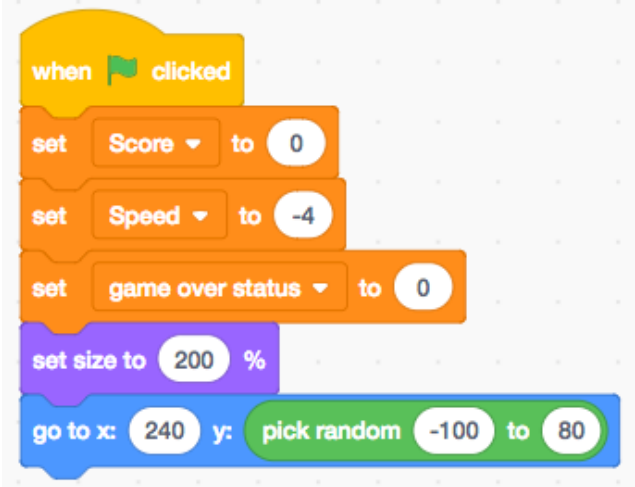


Tips:

1. The size of “Elephant2” should be set to 50%, or it’s too large to avoid water pipe.
2. We set the variable data “Elephant y axis” multiple 10 times, to transfer the distance value from ultrasonic sensor to millimeter. This is enlarge the variable range; Then we minus 180 to convert the detected value to coordinate system value, means when ultrasonic sensor defects distance as 0, elephant should be in the bottom of stage, y axis should be -180.

Code for sprite “Waterpipe1”.

1. Set initial value of variable data “score”, “speed” of water pipe, and “game over status”. Set the initial position and size of “Waterpipe1”.



Tips:

a. The initial value of variable data can be write into any sprite.
b. “Waterpipe1” is moving from right to left, means x axis is decreasing, so speed value is negative number.
c. Use variable data “game over status” to

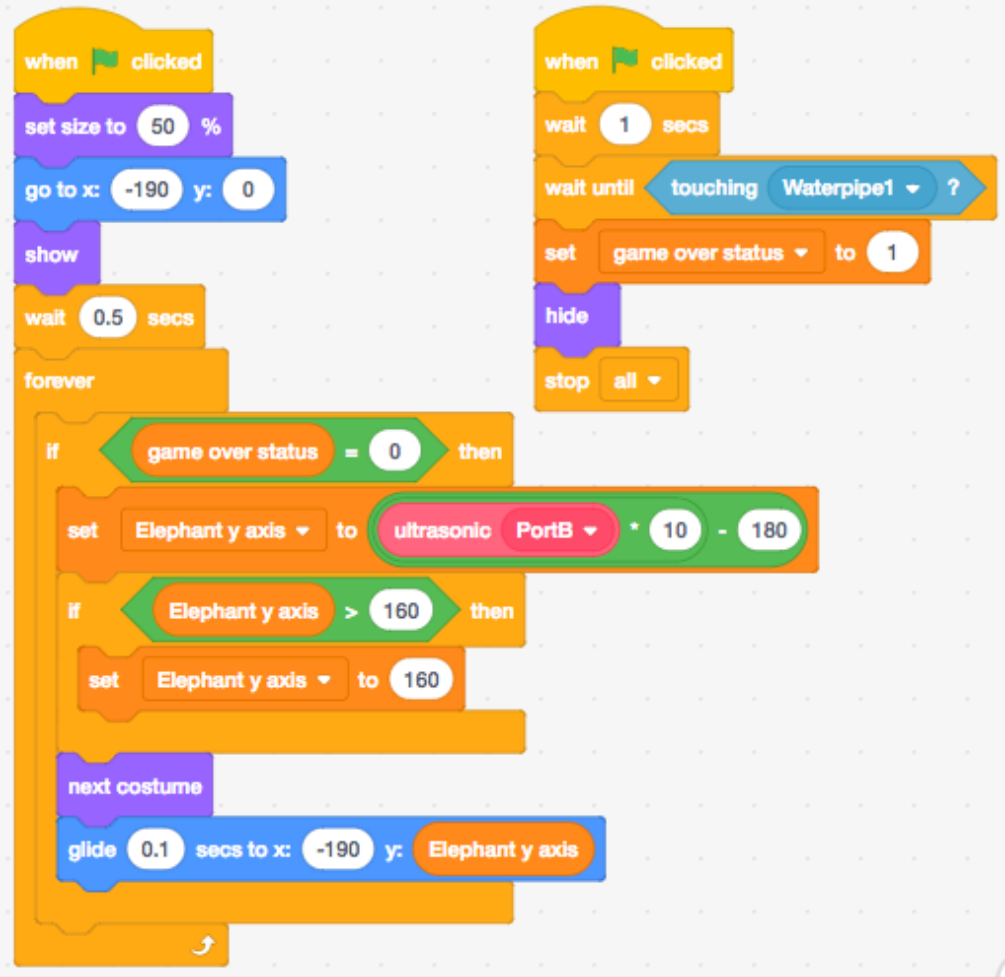
stands for game status. 0 means game running, 1 means game over.

d. Initial y axis -180~80 is the highest and lowest position of enlarged “Waterpipe1”.

2. Code for “Waterpipe1” movement: when received broadcast message1, “Waterpipe1” starts moving. Before game over, “Waterpipe1” will move to left side at speed -4. Once “Waterpipe1” reaches to the left wall of stage, it will appear on the right side and start moving to the left again, player will get 1 score.

Tips: We know the coordinate system in WeeeCode is -240~240 for x axis, but here we cannot set -240 as the decision value of x axis. The axis of a sprite is the center point, however, “Waterpipe1” has width, it should stop when the edge of sprite touch the wall of stage. In this circumstance, “Waterpipe1” cannot move to -240.

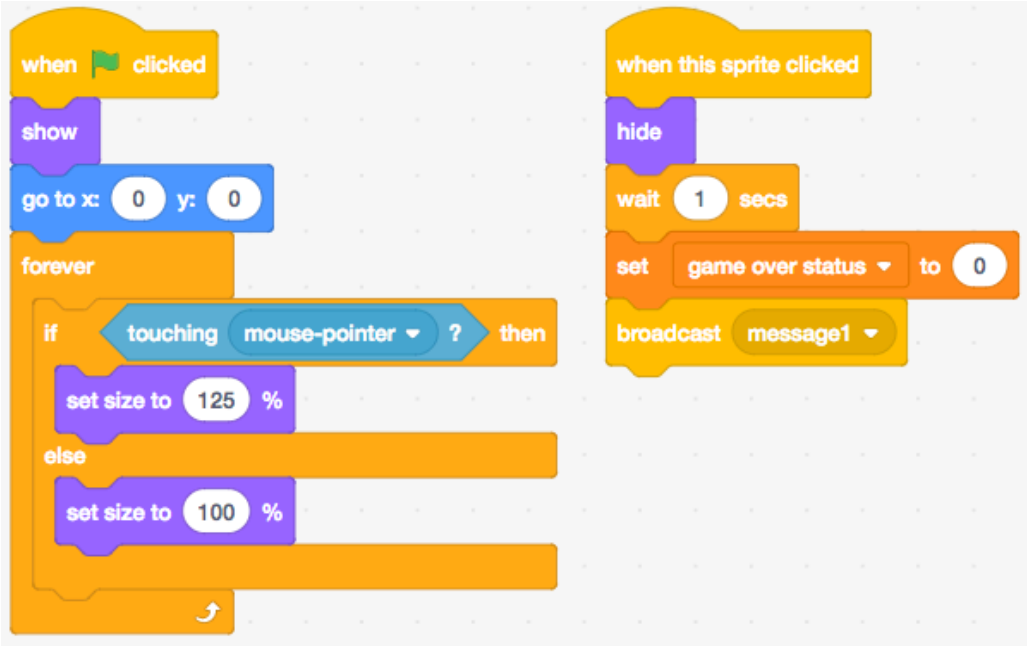
3. Code for variable data “game over status”. If “Elephant2” touched “Waterpipe1”, game should stop and be game over. Revise program for sprite “Elephant2” as below:



WEEEEMAKE

Result: Running codes, “Elephant2” will follow the ultrasonic sensor value moving up and down. Each time when “Elephant2” avoided “Waterpipe1”, player get 1 score; Once “Elephant2” touched “Waterpipe1”, “Elephant2” will be hidden, “Waterpipe1” will stop moving, program stops.

However, we find one problem. “Elephant2” can move up and down before we click “Button2”, because initial value of “game over status” is set when running codes instead of clicked button. To solve this problem, we should remove the initial value setting of “game over status” in sprite “Waterpipe1” and add into “Button2” as below.

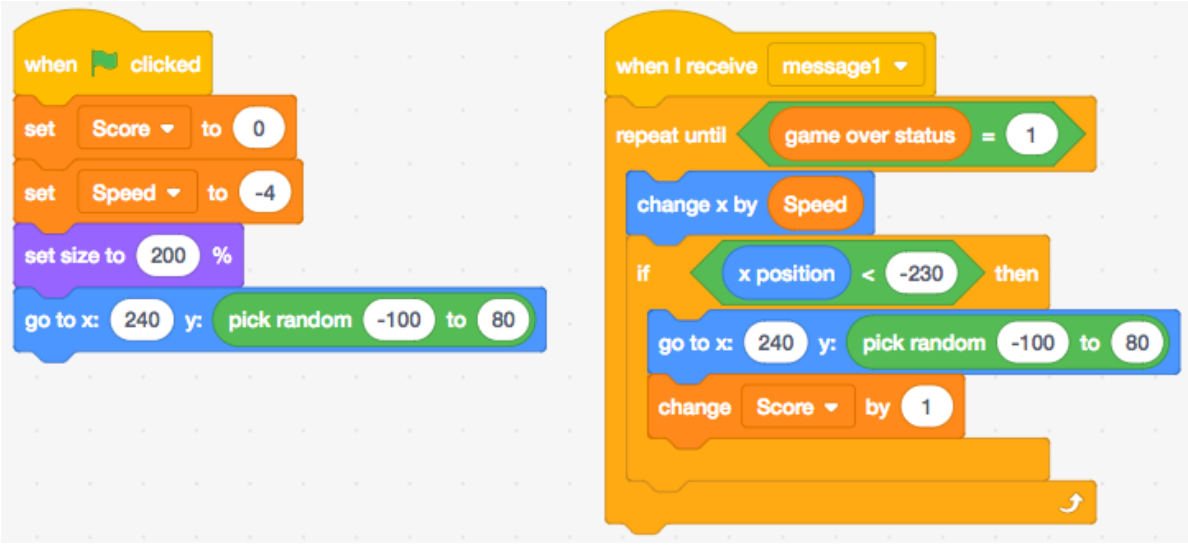


Tips:

a. This program is to add a condition before “Elephant2” moving. If the variable data “game over status” is 0, “Elephant2” should move according to ultrasonic sensor value; otherwise “Elephant2” shouldn’t move.

b. In “Elephant2” program, we should add “Wait…secs” before “forever” and “wait until…”, or program will start checking the condition in the beginning, when condition can never be satisfied.

Here you go the final program of “Waterpipe1”.

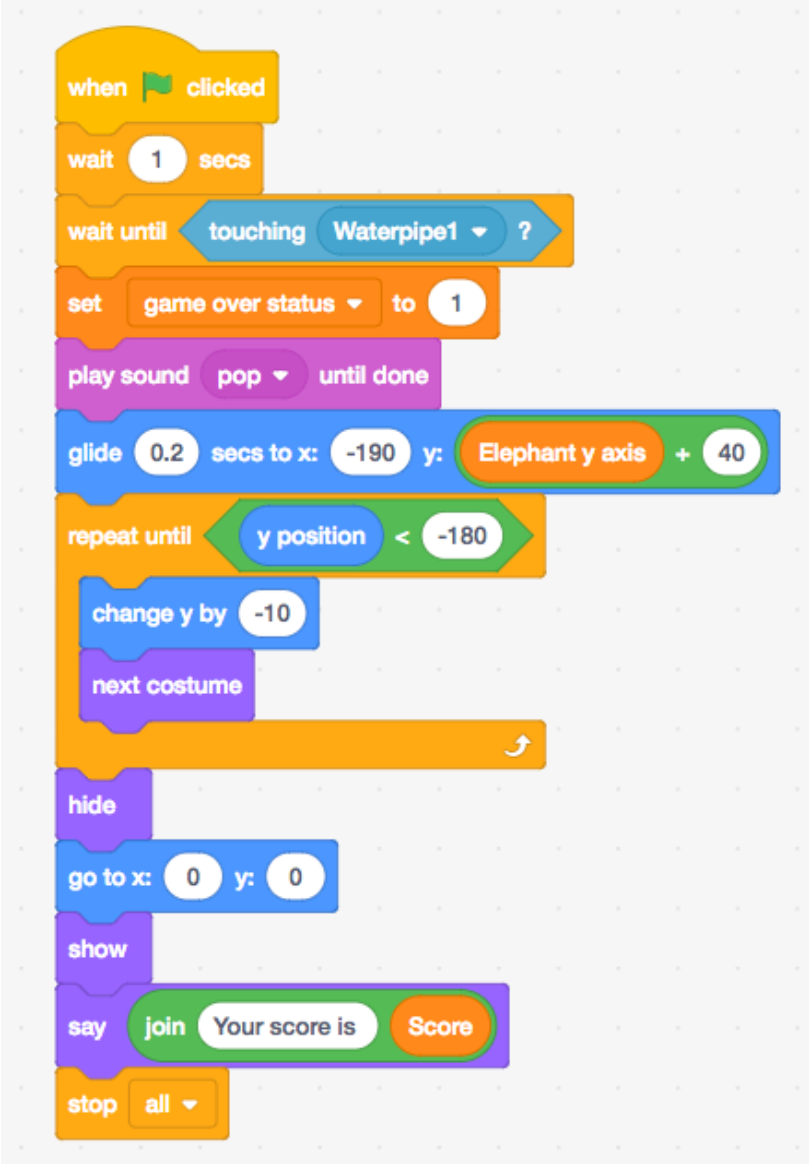


4. REMIX

Make animation for game over, add acceleration for “Waterpipe1”

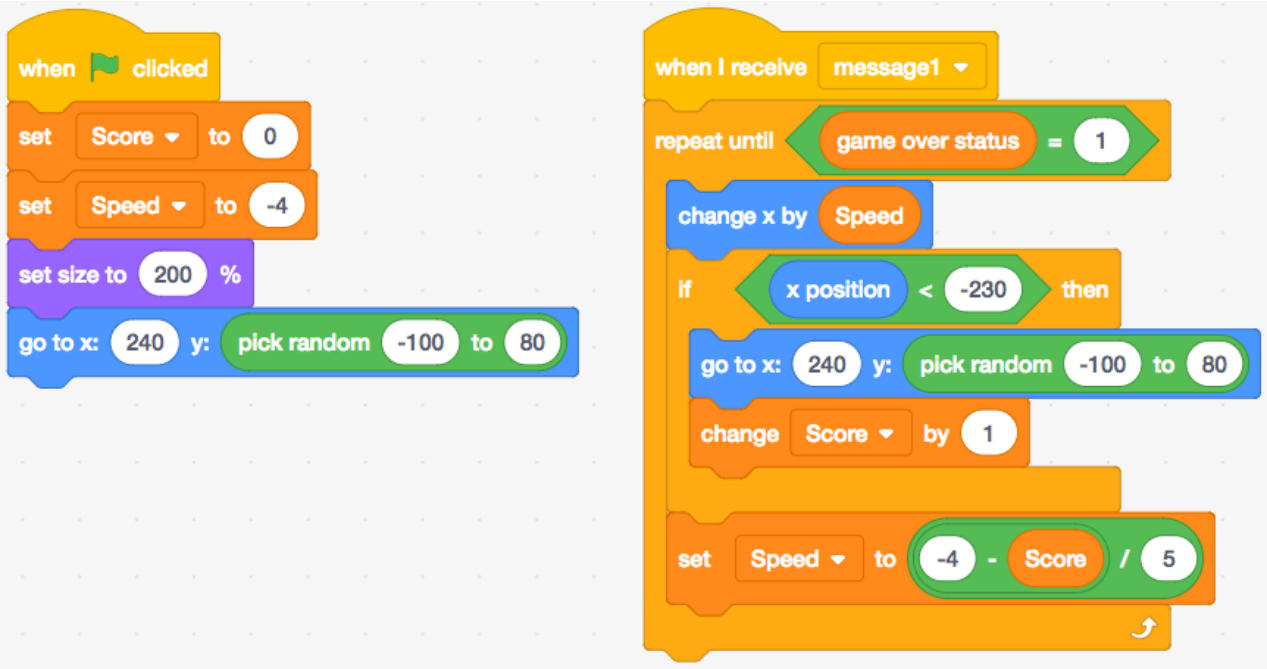
1. Game over animation: when “Elephant2” touch “Waterpipe1”, it will make sounds, move up, drop down to the bottom of stage, hide, show again in the center of stage, and then say score.

To achieve this animation, we should revise program in sprite “Elephant2” as below.



2. “Waterpipe1” acceleration animation: when score is higher, the speed of “Waterpipe1” will be faster. Revise “Waterpipe” program as below:

Tips: Encourage students use different way to achieve game result.


when clicked
set Score to 0
set Speed to -4
set size to 200 %
go to x: 240 y: pick random -100 to 80
when I receive message1
repeat until game over status = 1
change x by Speed
if x position < -230 then
go to x: 240 y: pick random -100 to 80
change Score by 1
set Speed to -4 - Score / 5