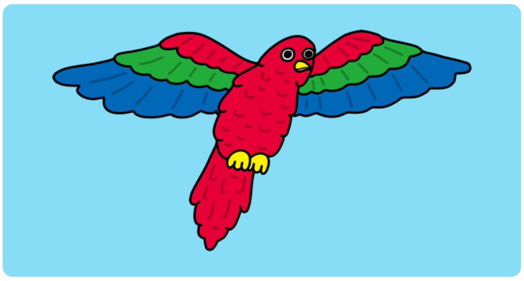
Projects

Flappy parrot

Create a game in which you guide a parrot past moving obstacles

Scratch



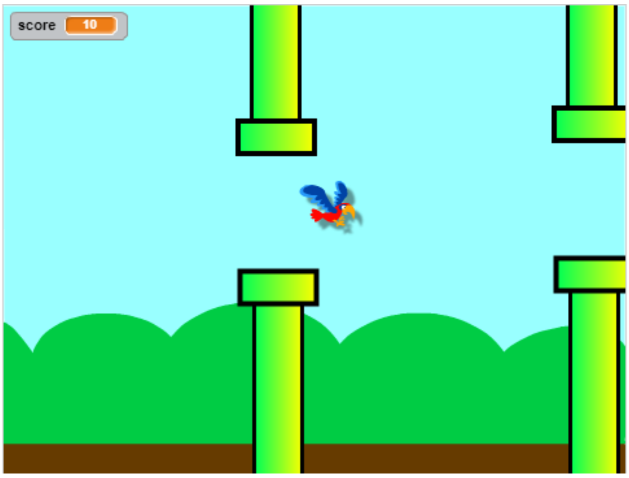
Step 1 Introduction

Create a game in which you have to guide a parrot past scrolling pipes to score points.

What you will make

Click the green flag to start the game. Press the space bar to make the parrot flap its wings, and try to get it to fly through the gaps in the pipes! You score one point for every pipe that you manage to get the parrot past.

You will press the space bar to make the parrot flap its wings, and score one point for every pipe that you manage to get the parrot past.



What you will need

Hardware

- A computer capable of running Scratch 2.0

Software

- Scratch 2.0 **offline** (https://rpf.io/scratchoff)

What you will learn

- How to create sprites using Vector mode
- How to use sounds
- How to detect collisions
- How to control a sprite using the keyboard

Additional notes for educators

You can **find the solution to this project here** (https://rpf.io/p/en/flappy-parrot-scratch2-get).

Step 2 Add the pipes

First, create the pipes.

Open a new empty Scratch project. ☑

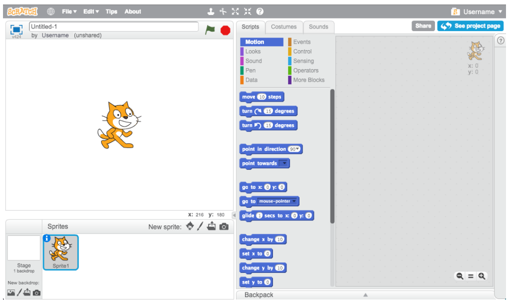
Creating a new Scratch project

You can use Scratch online or offline.

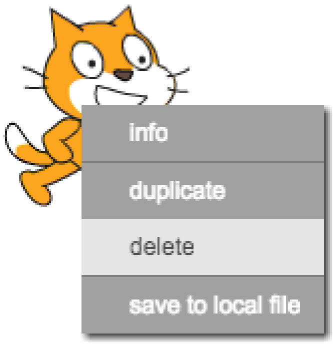
- To create a new Scratch project using the online editor, go to **jumpto.cc/scratch-new** (http://jumpto.cc/scratch-new).

- If you prefer to work offline and have not installed the editor yet, you can download it from **jumpto.cc/scratch-off** (http://jumpto.cc/scratch-off).

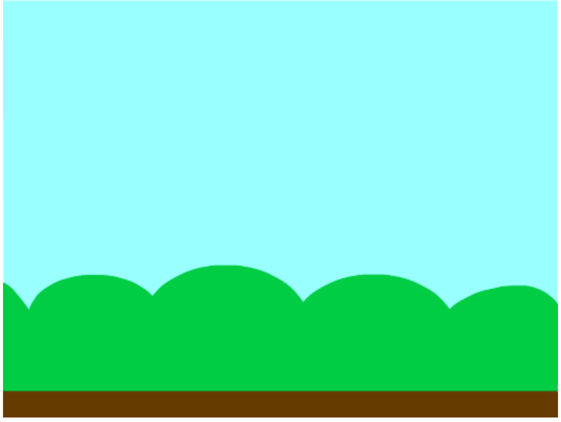
  The Scratch editor looks like this:



- The cat sprite that you can see is the Scratch mascot. If you need an empty Scratch project, you can delete the cat by right-clicking it and then clicking **delete**.

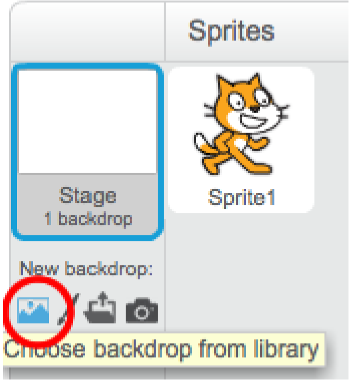


Add a background with an outdoor landscape. 'blue sky' is a good choice.

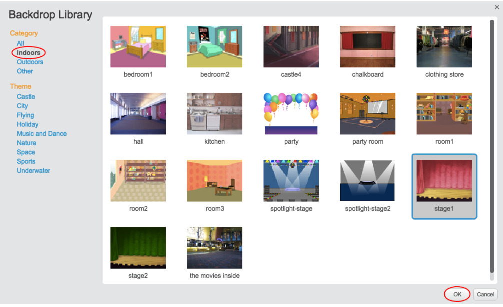


Adding a backdrop from the Scratch library

- Click **Choose backdrop from library**.



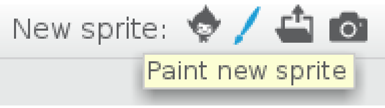
- You can browse backdrops by category or theme. Click on a backdrop and click **OK**.



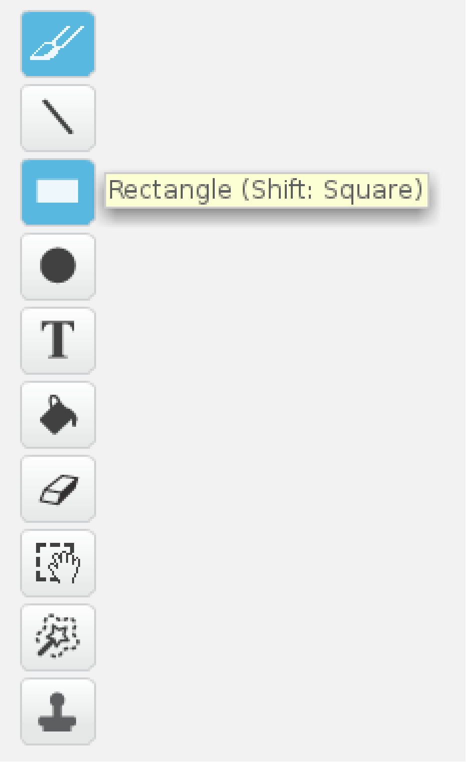
Click on the **Paint new sprite** button and name your sprite ‘Pipes’.

Drawing sprites

- In the Sprites panel, click on the paintbrush icon to **Paint new sprite**.



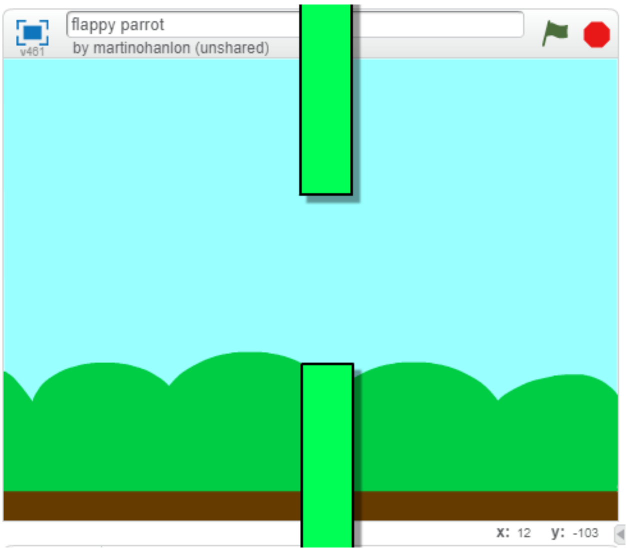
- Use the drawing tool in the **Costumes** tab to paint your new sprite.



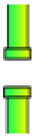
- When you are finished, don’t forget to give your new sprite a sensible name.

The ‘Pipes’ sprite should be a pair of pipes with a gap in the middle. By moving the sprite up or down, you can put the gap in a different place.

This picture shows an example of how the pipes could be positioned. The parts of the sprite outside the Stage are normally hidden, you only see then when you drag the sprite:

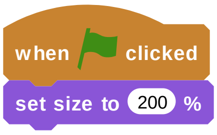


You can’t draw a sprite as big as the pipes need to be, but you can increase the size at which the sprite shows on the Stage.



☑

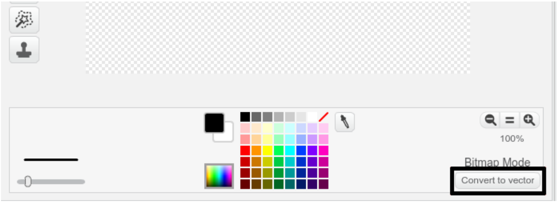
Add code to make the sprite bigger.



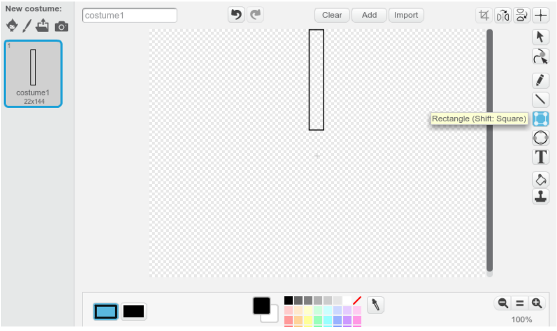
```
when green flag clicked
set size to (200) %
```

This makes it’s easier to see how big the pipes should be.

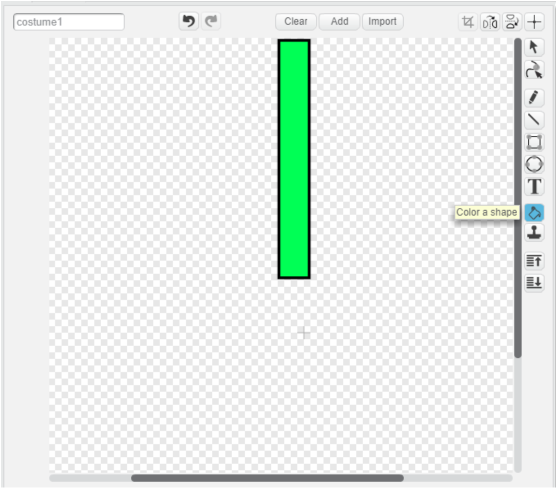
Switch the Paint editor to Vector mode by clicking on **Convert to vector** in the bottom right-hand corner. ☑



Draw a black outline rectangle for the top pipe as shown here: ☑

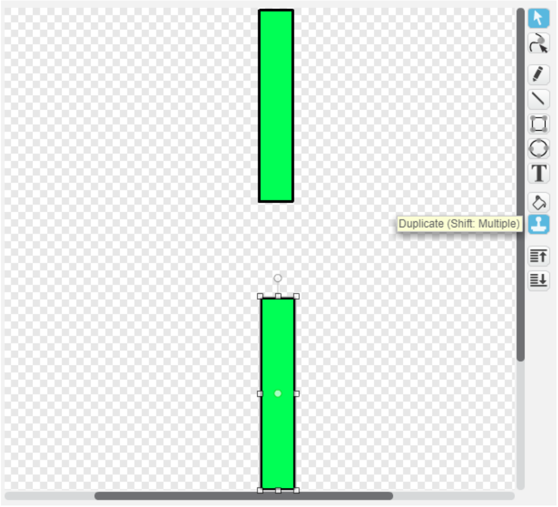


Fill the rectangle with a colour you like. ☑



Click on the **duplicate** (stamper) tool and then on your pipe to create a copy of the pipe.

Drag the copy of the pipe to the bottom of the screen so that the copy is in line with the other pipe and there’s a gap between the two pipes.

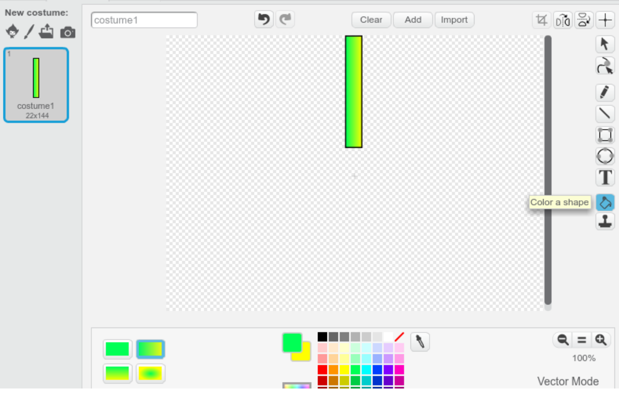


Challenge!

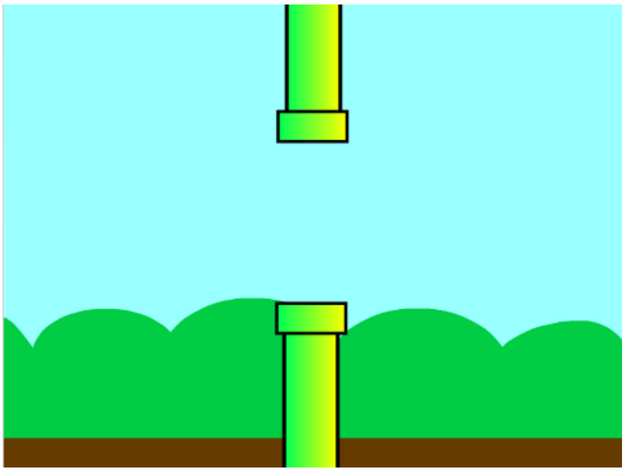
Challenge: make better pipes

Can you make your pipes look better?

- Shade your pipes with a left-to-right gradient.

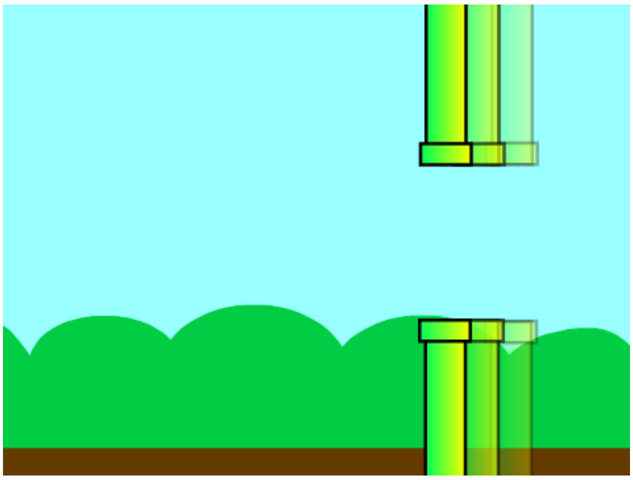


- Add extra rectangles to the ends of the pipes:

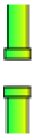


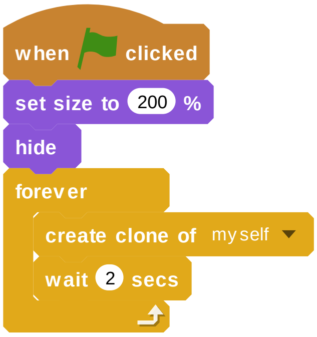
Step 3 Make the pipes move

Next you’re going to make the pipes move across the screen to create an obstacle course.



☑ First make the pipes appear by adding code to the Pipes sprite so that, `when the green flag is clicked`, the sprite `forever` `creates a clone of itself` every two seconds.

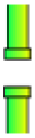


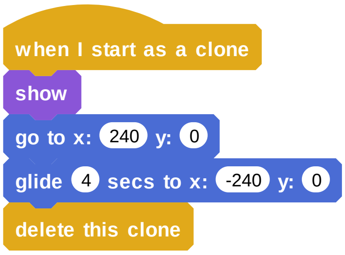


```
when [flag] clicked
set size to (200) %
hide
forever
    create clone of [myself ▾]
    wait (2) secs
```

Tip: clones are just copies of a sprite, and they are really useful for creating games.

☑ Next make the pipes move by adding code so that, `when a clone starts`, the clone appears on the right side of the Stage and `glides` across to the left.



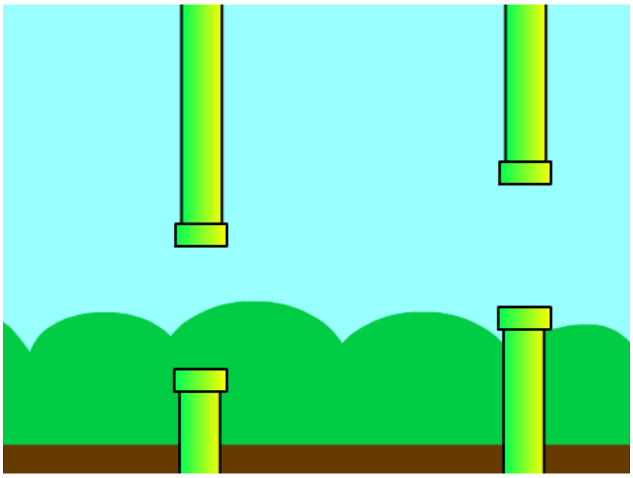


```
when I start as a clone
show
go to x: (240) y: (0)
glide (4) secs to x: (-240) y: (0)
delete this clone
```

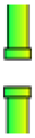
Tip: you can stop the pipes scrolling by clicking the red **stop** button next to the green flag.

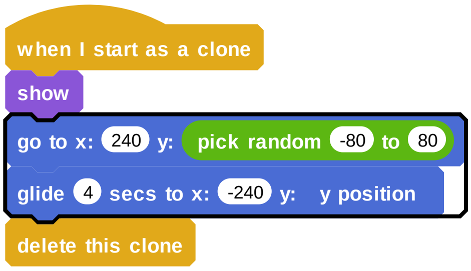
Now you should have lots of pipes, but their gaps are always in the same place.

You can add some variety by using a `random` number for the Pipes sprite's `y position`.



Modify your sprite's code so that each sprite clone `picks a random number from -80 to 80` and `glides` to that `y position`:





```
when I start as a clone
show
go to x: 240 y: pick random -80 to 80
glide 4 secs to x: -240 y: y position
delete this clone
```

Step 4 Make Flappy fall

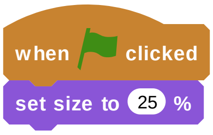
Now add a sprite called Flappy and create code it so Flappy falls down the Stage. In the next step, you will add the code to make Flappy fly when you press a key.

Add a new sprite that has two costumes, for 'wings up' and 'wings down', and name it `Flappy`.

The parrot sprite is a good choice.

Flappy needs to be smaller.

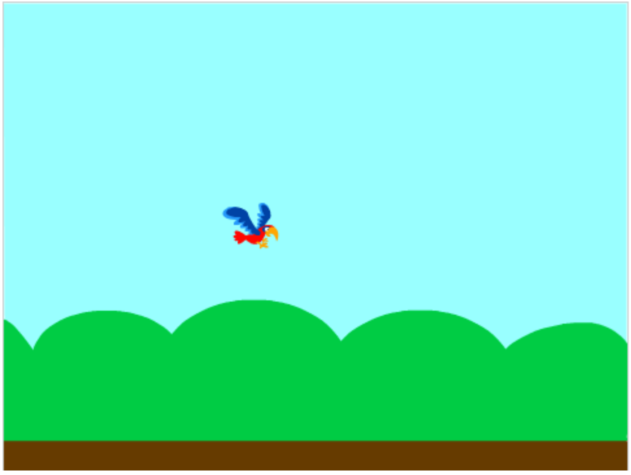
Add code to `set Flappy's size to 25%` `when the green flag is clicked`.



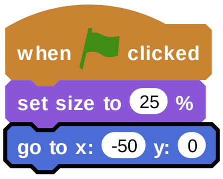
```
when [flag] clicked
set size to (25) %
```

Tip: you could also use the **Shrink** tool to make your sprite smaller.

When the game starts, Flappy needs to be just left of the centre of the Stage, at coordinates `-50, 0`.



Add code to make Flappy `go to the x and y` starting position of `x: -50` and `y: 0`.

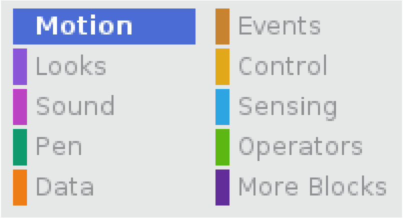


```
when green flag clicked
set size to 25 %
go to x: -50 y: 0
```

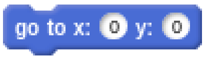
Set a sprite's coordinates

To set a sprite’s coordinates so that it appears at a certain location on the Stage, follow the steps below.

- Click on the **Motion** menu in the **Scripts** palette.

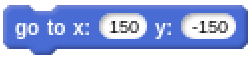


- Find the `go to x: ( ) y: ( )` block.



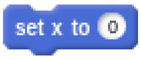
```
go to x: 0 y: 0
```

- Type in the `x` position and `y` position that you want your sprite to go to.

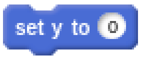


```
go to x: 150 y: -150
```

- If you only want to set the `x` or `y` position, you can use either of the following two blocks instead.

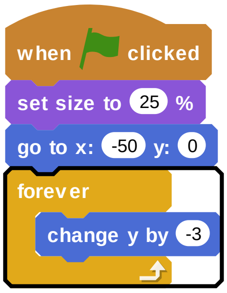


```
set x to 0
```



```
set y to 0
```

Now make Flappy keep falling down the Stage by `forever` `changing the sprite's y position by -3`.



```
when green flag clicked
set size to (25) %
go to x: (-50) y: (0)
forever
    change y by (-3)
end
```

☑ Test your code to make sure Flappy starts in the middle of the screen and falls to the bottom. When you drag Flappy to the top of the Stage, the sprite should fall again.

Step 5 Make Flappy fly

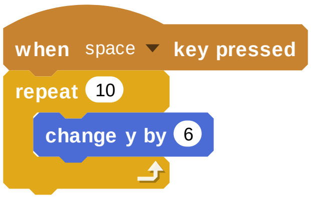
Now you will make Flappy flap upwards when you press the space bar. When you play the game, you have to time your taps to get Flappy through the gaps in the pipes.

Make Flappy fly upwards when you tap the space bar.

☑ When the `space key is pressed`, Flappy should move upwards by `changing its y coordinate` by a small amount, for example `6`.

Flappy flies upwards by `repeating` this movement `10 times`.

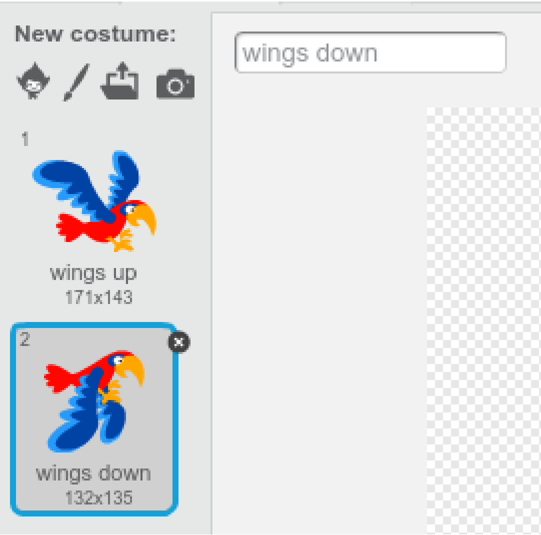
Add this code to your `Flappy` sprite:



```
when [space v] key pressed
repeat (10)
    change y by (6)
end
```

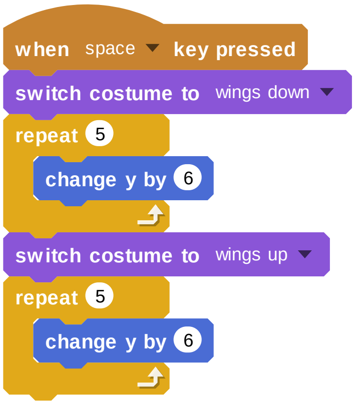
Now you need to get Flappy's wings flapping!

Click on the **Costumes** tab, and name Flappy's costumes 'wings up' and 'wings down'.



Can you make Flappy's costume change to `wings down` when you press `space`, and then change it back to `wings up` halfway through the upward movement?

Your code should look like this:



```
when space key pressed
switch costume to wings down
repeat 5
    change y by 6
switch costume to wings up
repeat 5
    change y by 6
```

Test your code. As you see, at the moment nothing happens if you let Flappy hit a pipe.

Step 6 Detect collisions

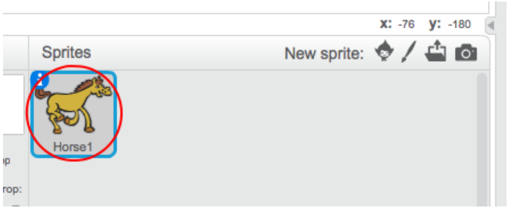
To make the game a challenge, the player needs to guide Flappy through the gaps without letting the parrot touch the pipes or the edges of the Stage. You need to add some blocks to detect when Flappy hits something.

This is called **collision detection**.

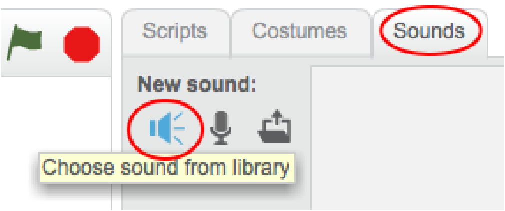
☑ Import a sound from the library that you want to play when Flappy collides with something. The ‘screech’ sound is a good choice.

Adding a sound from the library

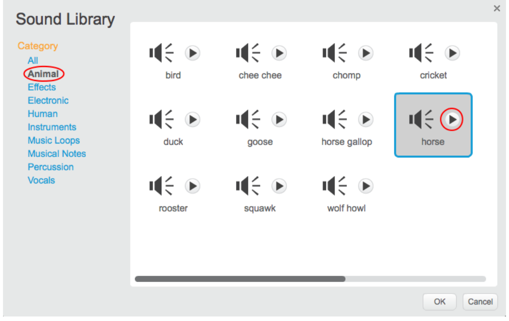
- Select the sprite you want to add the sound to.



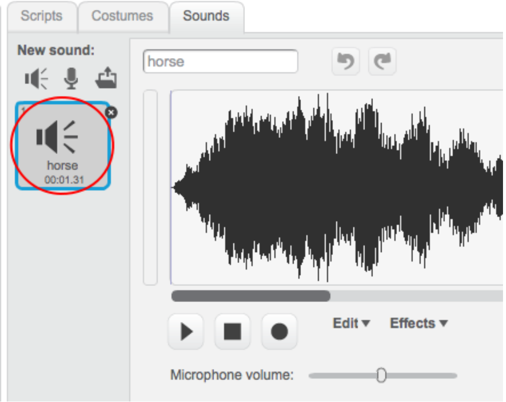
- Click the **Sounds** tab, and click **Choose sound from library**:



- Sounds are organised by category, and you can click the **Play** button to hear a sound. Choose a suitable sound and click **OK**.

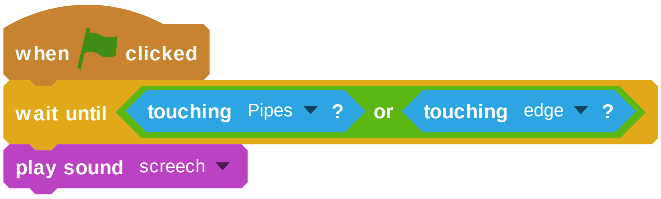


- You should then see that your sprite has your chosen sound.



A `wait until` block is necessart to check whether Flappy is `touching the pipes` `or` `touching the edge`.

Add a new `when green flag clicked` block to the ‘Flappy’ sprite, and also add the following code:

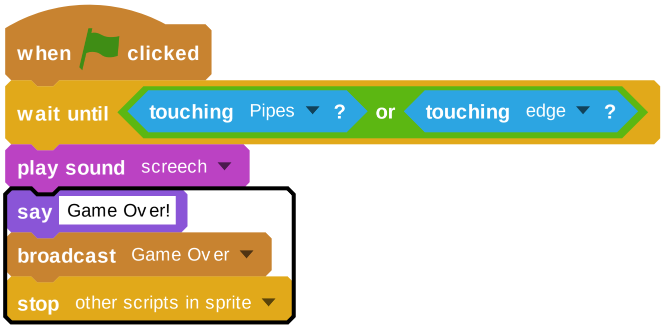


```
when [flag] clicked
wait until <<touching [Pipes v] ?> or <touching [edge v] ?>>
play sound [screech v]
```

Test your code. If Flappy touches a pipe, the ‘screech’ sound should play.

Next, update the code so that the game stops when Flappy hits a pipe.

Add the following code to stop the game after a collision is detected: ☑

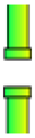


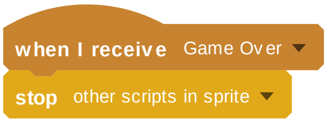
```
when ⚑ clicked
wait until <touching Pipes ? or touching edge ?>
play sound screech
say Game Over!
broadcast Game Over
stop other scripts in sprite
```

The `broadcast` block tells other sprites that the game is over.

The `stop` block stops other Flappy scripts that are running so that Flappy stops falling after a collision.

Finally, add the following code to the `Pipes` sprite so that pipes `stop` appearing `when the sprite receives Game Over`. ☑





```
when I receive Game Over
stop other scripts in sprite
```

Test your game and see how long you can play before it’s ‘Game over’! ☑

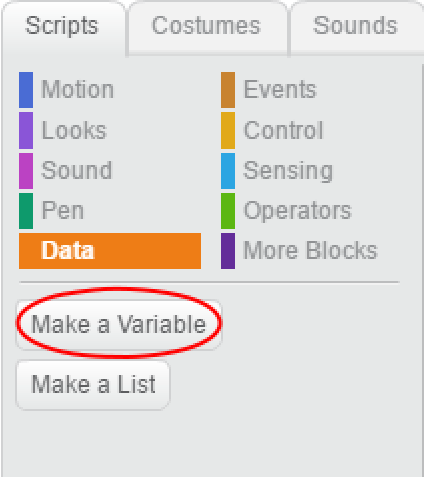
Step 7 Add a score

The player should score a point every time Flappy makes it through a gap between pipes.

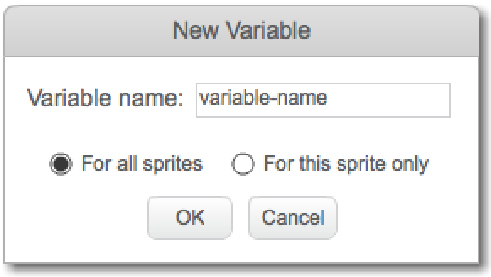
Make a new variable **for all sprites** and call it `score`. ☑

Add a variable in Scratch

- Click on **Data** in the Scripts tab, then click on **Make a Variable**.



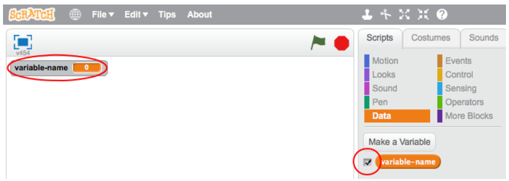
- Type in the name of your variable. You can choose whether you would like your variable to be available to all sprites, or to only this sprite. Press **OK**.


New Variable

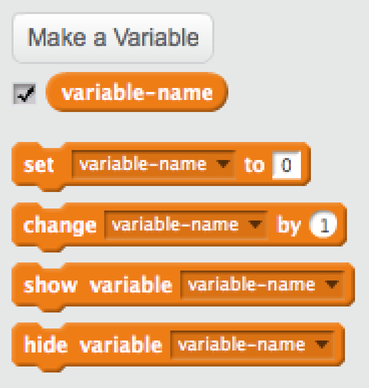
Variable name: variable-name

● For all sprites ○ For this sprite only

- Once you have created the variable, it will be displayed on the Stage, or you can untick the variable in the Scripts tab to hide it.



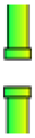
- New blocks will appear and allow you to change the value of the variable.

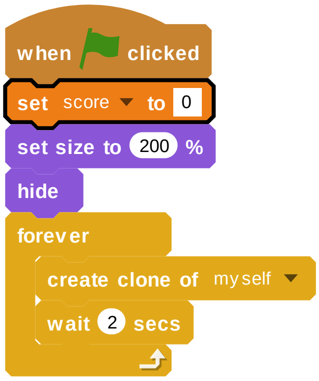


```
Make a Variable
☑ variable-name
set variable-name ▼ to 0
change variable-name ▼ by 1
show variable variable-name ▼
hide variable variable-name ▼
```

Each ‘Pipes’ sprite clone should `wait until` Flappy has flown past and then increase the `score`.

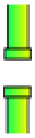
First, `set score to 0` when the game begins:

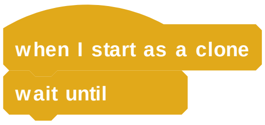




```
when 🏴 clicked
set score ▼ to 0
set size to 200 %
hide
forever
    create clone of myself ▼
    wait 2 secs
```

Then add the following code to the `Pipes` sprite:



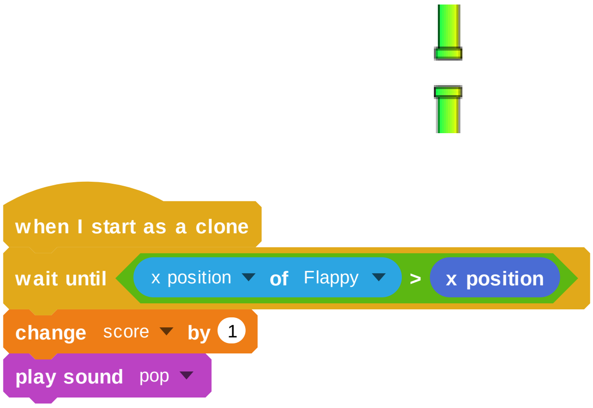


```
when I start as a clone
wait until
```

☑ Add more code so that, when Flappy's `x` position is greater than the pipe clone's `x` position, the `score` increases by `1` and a sound of your choice plays.

You could use the 'pop' sound if you want, or add a sound from the library, for example 'bird'.

Your code should look like this:



```
when I start as a clone
wait until <([x position v] of [Flappy v]) > (x position)>
change [score v] by (1)
play sound [pop v]
```

☑ Test your code and make sure you score a point every time Flappy gets through a gap between pipes. Check whether the `score` is set to `0` when you start a new game.

🏆 Challenge!

Challenge: adjust the difficulty

Is the game too hard or too easy for you? How many ways can you find to change the difficulty?

Adjust the game until you are happy with its difficulty!

🏆 Challenge!

Challenge: add a high score

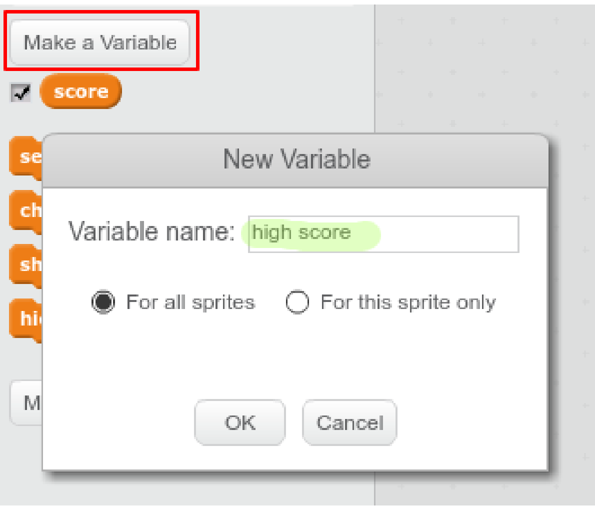
Can you add a high score to the game so that, in addition to keeping track of score for the current round, it keeps track of the highest score you've ever reached?

Create a high score

It's fun to keep track of a high score in a game.

Let's say you have a variable called `score`, which gets set to zero at the beginning of each game.

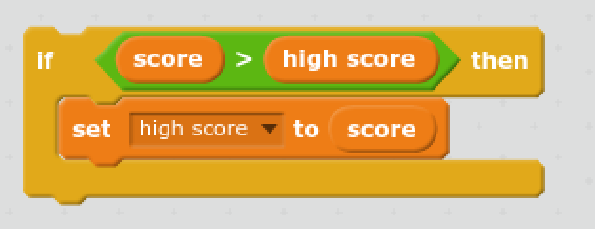
Add another variable called `high score`.



Variable name: high score

● For all sprites ○ For this sprite only

At the end of the game (or whenever you want to update the high score), you'll need to check whether you have a new `high score`.



```
if score > high score then
    set high score to score
```

Step 8 What next?

Try the **Binary hero** (https://projects.raspberrypi.org/en/projects/binary-hero) project, where you will make a game in which you play the notes of a song as they scroll down the Stage.