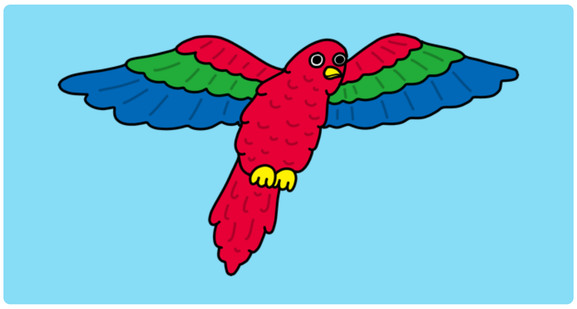
Projects

Flappy Parrot

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

Scratch

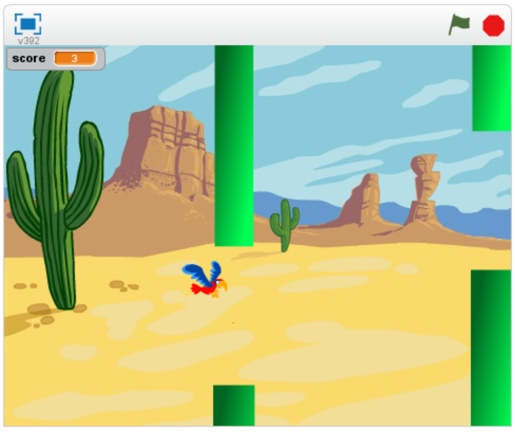


Step 1 Introduction

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

What you will make

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



What you will learn

This project covers elements from the following strands of the **Raspberry Pi Digital Making Curriculum** (http://rpf.io/curriculum):

- **Apply basic programming constructs to solve a problem** (https://www.raspberrypi.org/curriculum/programming/builder)

Step 2 What you will need

Hardware

- Computer capable of running Scratch 2.0

Software

- Scratch 2.0 (either **online** (https://scratch.mit.edu/projects/editor/) or **offline** (https://scratch.mit.edu/scratch2download/))

Step 3 Add the backdrop and pipes

First let's create the pipes.

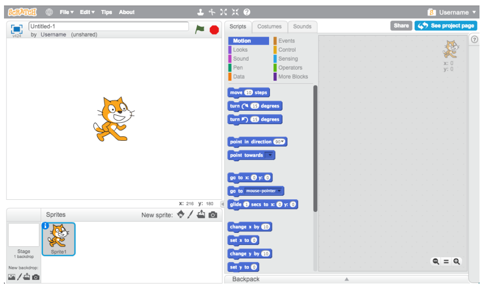
- Open a new empty Scratch project.

Creating a new Scratch project

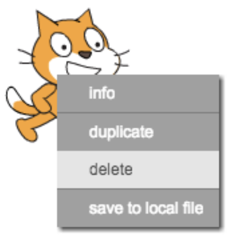
You can use Scratch online or offline.

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

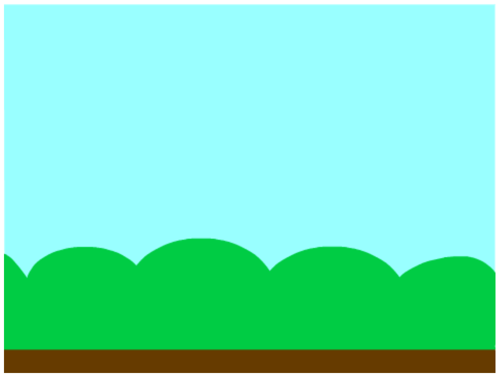
  The Scratch editor looks like this:



- 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**.

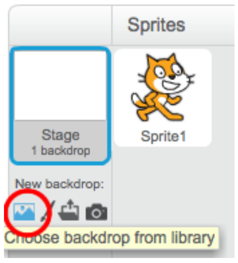


- Add a background with an outdoor landscape. 'blue sky' is a good choice.

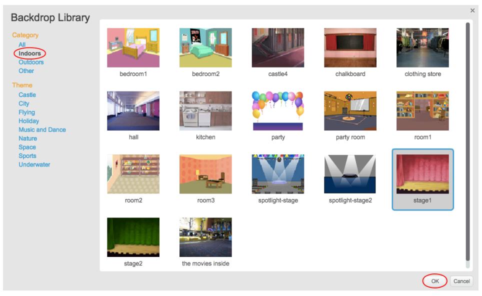


Adding a backdrop from the Scratch library

- Click **Choose backdrop from library**.



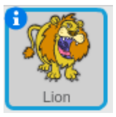
- You can browse backdrops by category or theme. Click on a backdrop and click **OK**.



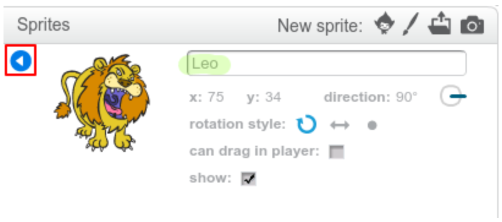
- Click on the `Paint new sprite` button.
- Name your sprite **Pipes**.

Rename a sprite in Scratch

To rename a sprite in Scratch, click on the **i** on the sprite:



This opens the **info** panel. You can also right-click on a sprite and choose `info`.

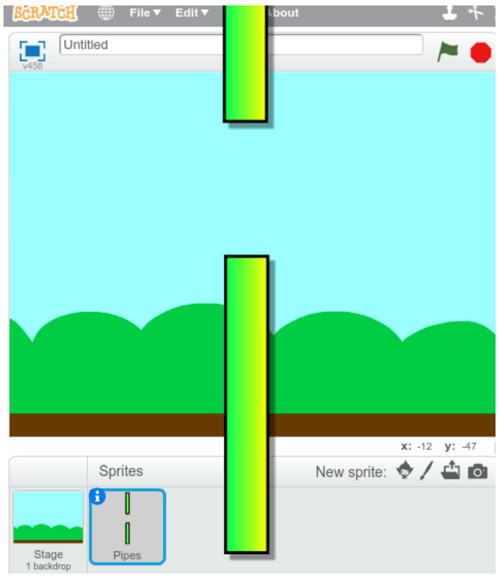


You can edit the name of the sprite and then click the **triangle** to close the **info** panel.

- The `Pipes` sprite will have a pair of pipes with a gap in the middle. You'll be able to move the sprite up and down to get the gap in a different place.

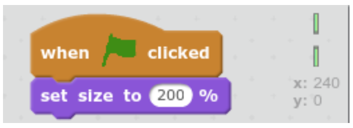
  Let's take a look at how this is going to work.

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



- You can't draw a pipe as big as the pipes need to be, but you can increase the size of a sprite when it's used.

  Add code to make the sprite bigger.



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

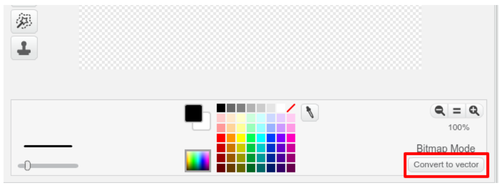
  This will make it easier to see how big you need to make the pipes.

- Switch the Paint editor to Vector mode.

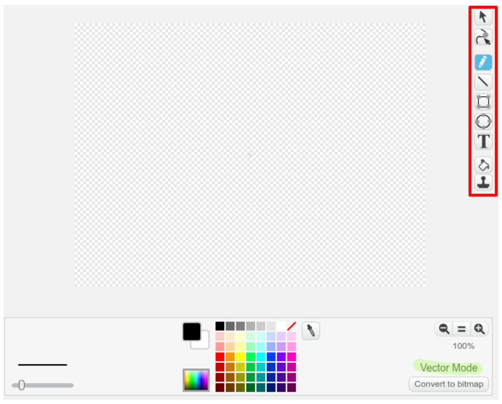
Scratch paint tool vector mode

The Scratch paint tool has two modes: bitmap and vector. Vector mode stores your picture as shapes and lines that you can move around individually. This is really useful!

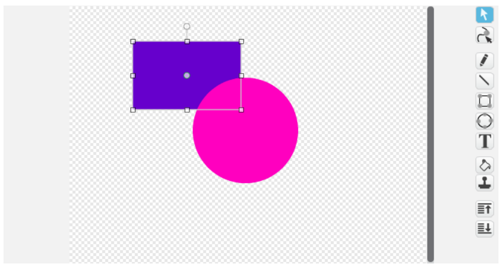
When you are in bitmap mode you can click **Convert to vector** to switch to vector mode.



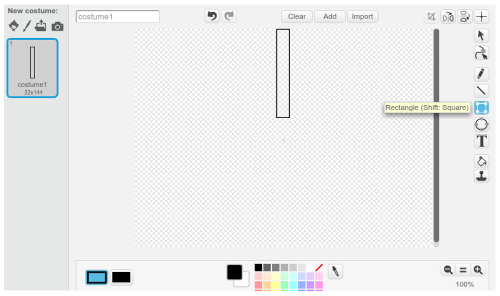
In vector mode, the tools appear on the right side of the drawing area.



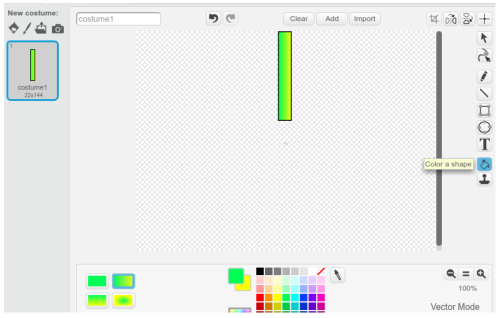
Choose the select tool (the arrow) to move, resize, and rotate shapes and lines.



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

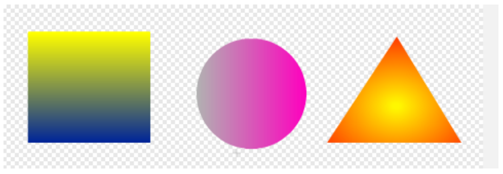


- Shade your pipe with a left-to-right gradient.

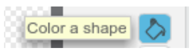


Using gradients in Scratch paint

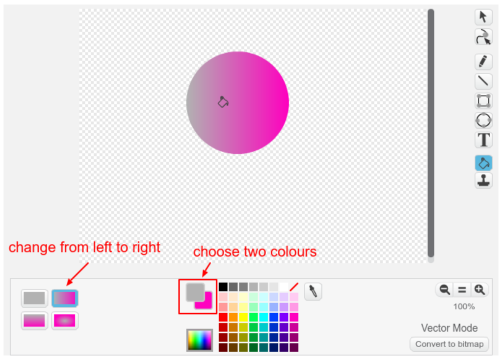
When you are using the Scratch paint tool in vector mode, you can fill a shape with a colour gradient to get interesting shading effects.



First create a shape, then choose the **Color a shape** tool:

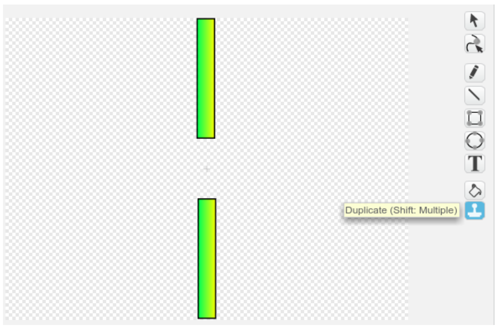


Select the kind of gradient you want, and the two colours to which you want to apply the gradient. Then click on the shape you want to fill.

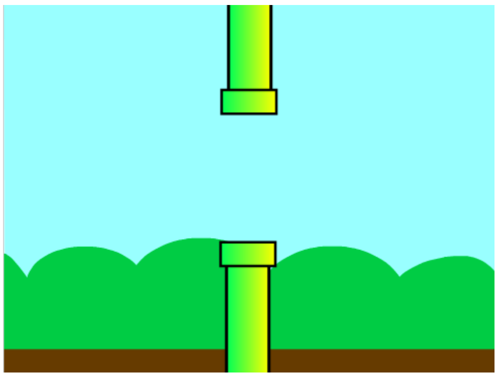


- Click on the duplicate (stamper) tool and then on your pipe to create a copy.

  Drag the copy of the sprite to the bottom of the screen, in line with the top sprite.



- If you like, you can add extra shaded rectangles to the ends of the pipes:

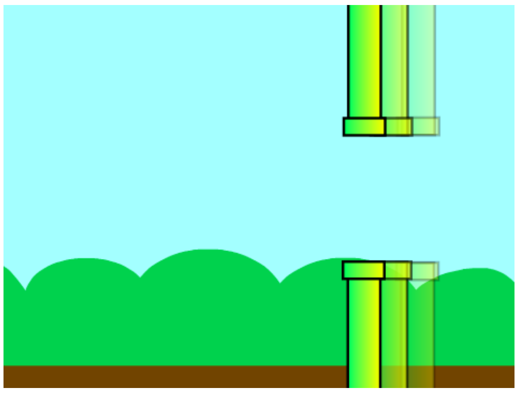


Step 4

Make the pipes move

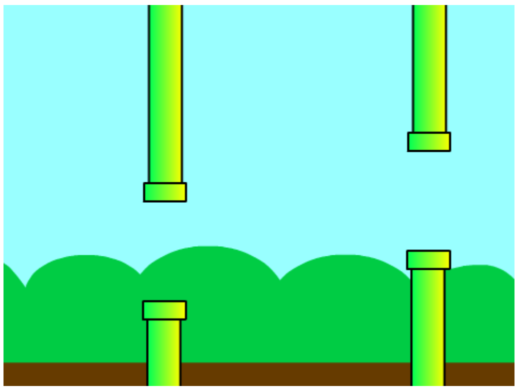
Next you'll get the pipes moving across the screen to create an obstacle course.

- Create a clone of your `Pipes` sprite every two seconds. Each clone should scroll across the stage from right to left (towards the parrot).



Tip: you can stop the pipes scrolling by clicking the red stop button.

- Now you should have lots of pipes, but the gap is always in the same place. Add some variety by putting the gap between each set of pipes at a different height.

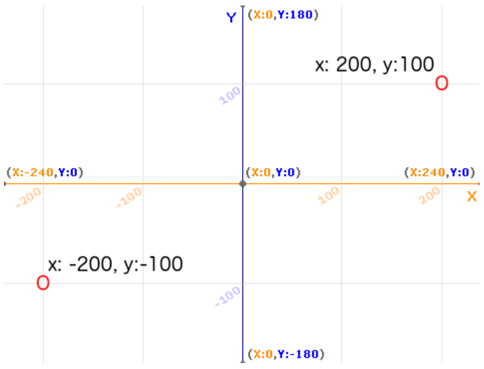


Scratch coordinates

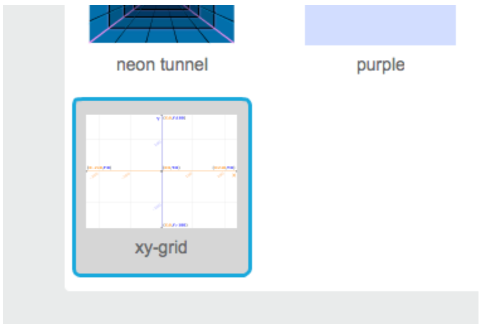
Scratch coordinates

- In Scratch, the coordinates `x:0, y:0` mark the central position on the Stage.

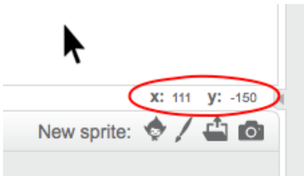
  A position like `x:-200, y:-100` is towards the bottom left on the Stage, and a position like `x:200, y:100` is near the top right.



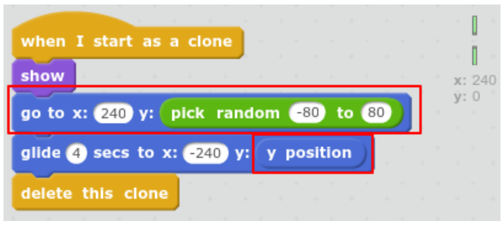
- You can see this for yourself by adding the **xy-grid** backdrop to your project.



- To find out coordinates of a specific position, move your mouse pointer to it and check the readings below the bottom right corner of the Stage.



This is what your code should look like:



```
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 5 Make Flappy fall

Now you can add a sprite called Flappy. If you don't press any keys, then Flappy should just fall down the screen.

- Add a sprite with two costumes, for 'wings up' and 'wings down'. The parrot sprite is a good choice. Name your sprite 'Flappy'.

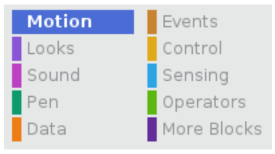
- Flappy needs to be smaller — reducing the sprite's size to about 25% should do it. You can either use the **Shrink** tool or a `set size to ( )` block.

- When the game starts, Flappy should be just to the left of the centre of the screen, at coordinates `-50, 0`. Code Flappy to go to the starting position at the start of the game.

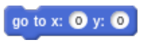
Set a sprite's coordinates

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

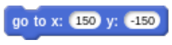
- Click on the **Motion** menu in the **Scripts** palette.



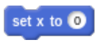
- Find the `go to x: ( ) y: ( )` block.

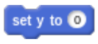


- Type in the `x` position and `y` position that you want your sprite to go to.



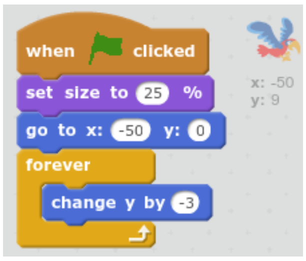
- If you only want to set the `x` or `y` position, you can use either of the following two blocks instead.





- Now you need to make Flappy keep falling down the stage.

Your code should look like this:


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

- 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 screen, the sprite should fall again.

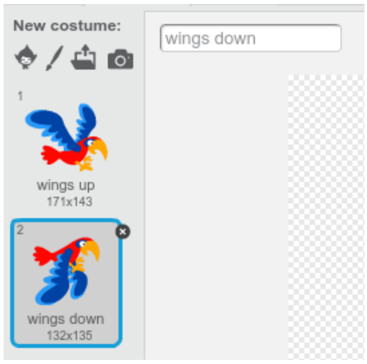
Step 6 Make Flappy fly

Next we want Flappy to flap upwards when you press the `space` bar. You'll need to time your taps to get through the gaps in the pipes.

- Can you add code to make Flappy fly upwards when you tap the `space` bar?

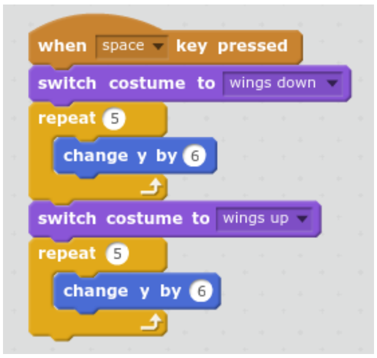
Now you need to get Flappy's wings flapping!

- Click on the **Costumes** tab, and name the costumes `wings up` and `wings down`.



- Can you make Flappy's costume change to `wings down` when you press `space`, and then 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
```

- Now you can test your code. As you'll see, nothing bad happens if you hit a pipe. In the next step, you'll change that.

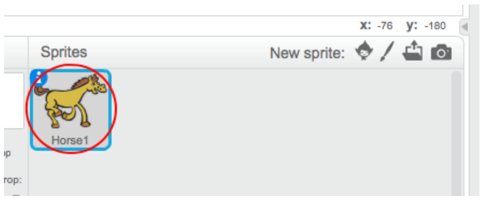
Step 7 Detect collision with the pipes

To make the game a challenge, the player needs to guide Flappy through the gaps without touching the pipes or the edges of the screen. To set this up, we'll add some blocks to detect when Flappy hits something. This is called **collision detection**.

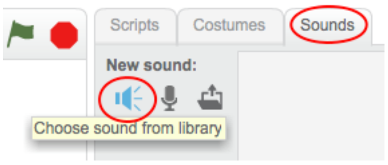
- Import a sound from the library that will play when Flappy collides with something. The 'screech' sound is a good choice.

Adding a sound from the library

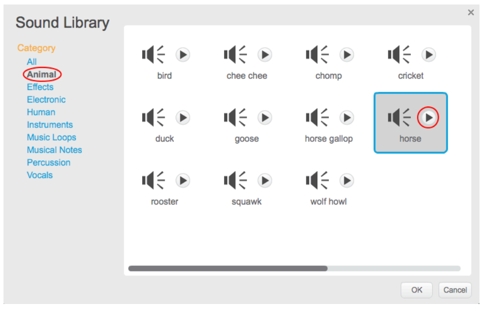
- Select the sprite you want to add the sound to.



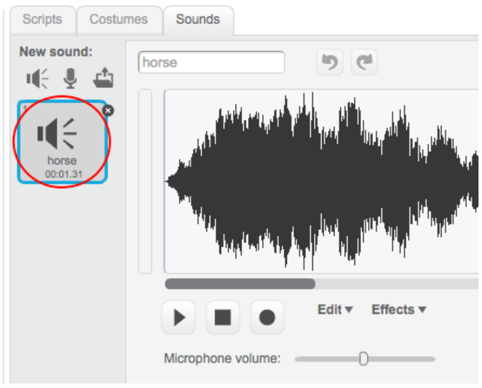
- Click the **Sounds** tab, and click **Choose sound from library**:



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

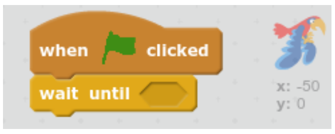


- You should then see that your sprite has your chosen sound.



- You're going to use a `wait until` block to check for whether Flappy is touching the pipes.

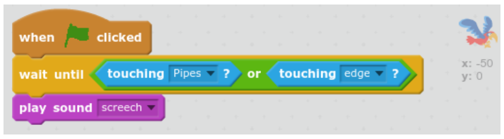
  Use a new `on green flag clicked` block:



  Any code you place after a `wait until` block will only run after the condition is met.

- Can you add to the code so Flappy screeches if she touches a pipe **or** the edge of the stage.

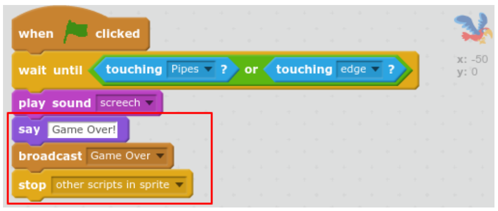
Your code should look like this:



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

- Test your code. You might notice that you only hear the screech the first time you have a collision. That's okay, because the game ends if you have a collision.

- Add the highlighted code to stop the game after a collision is detected:

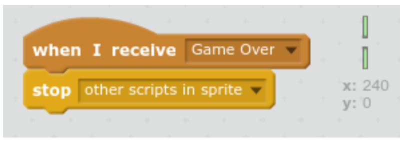


```
when green flag clicked
wait until <<touching [Pipes v] ?> or <touching [edge v] ?>>
play sound [screech v]
say [Game Over!]
broadcast [Game Over v]
stop [other scripts in sprite v]
```

  The `stop` block stops other Flappy scripts that are running. Flappy won't fall after a collision.

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

- Add the following code to the `Pipes` sprite so that the pipes stop when a `Game over` message is received.



```
when I receive [Game Over v]
stop [other scripts in sprite v]
```

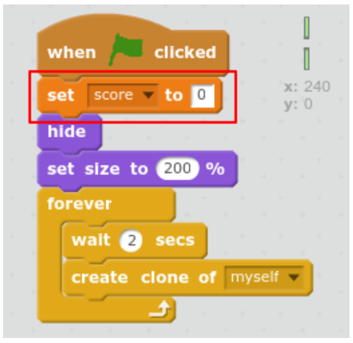
- Now test your game and see how long you can last!

Step 8 Add scoring

The player should score a point every time Flappy makes it through a gap between pipes. Let's add that code for that next!

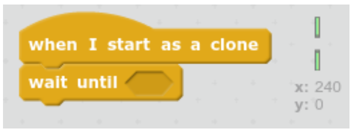
- Make a new variable `For all sprites` and call it `score`.
- Each `Pipes` clone is going to `wait until` Flappy has flown past, and then increase the score.

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



```
when green flag clicked
set [score v] to 0
hide
set size to 200 %
forever
    wait 2 secs
    create clone of [myself v]
```

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

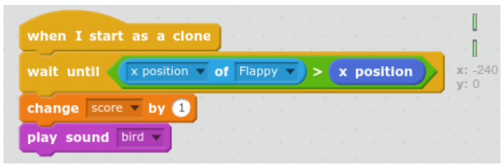


```
when I start as a clone
wait until < >
```

- Complete the code so that a point is scored, and a sound of your choice is played, when Flappy's `x` position is greater than (`>`) the pipe's `x` position.

Test your code and make sure you score a point every time Flappy gets past an obstacle. Make sure the score goes back to `0` when you start a new game.

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 [bird v]
```

Step 9 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 it!

Step 10 Challenge: add a high score

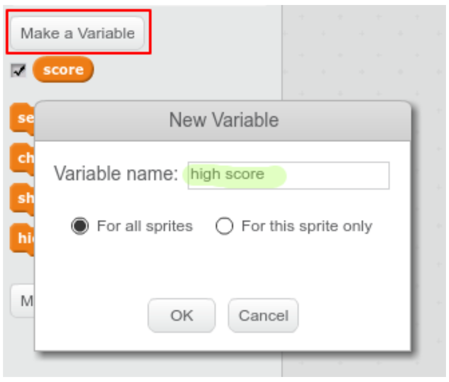
Can you add a high score to the game so that, in addition to the score for the current round, it keeps track of the highest score you've ever reached?

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.

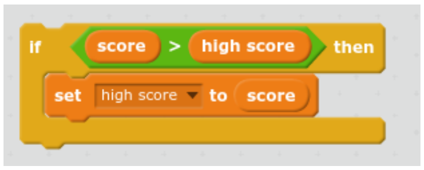
Add another variable called `high score`.



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)
end
```