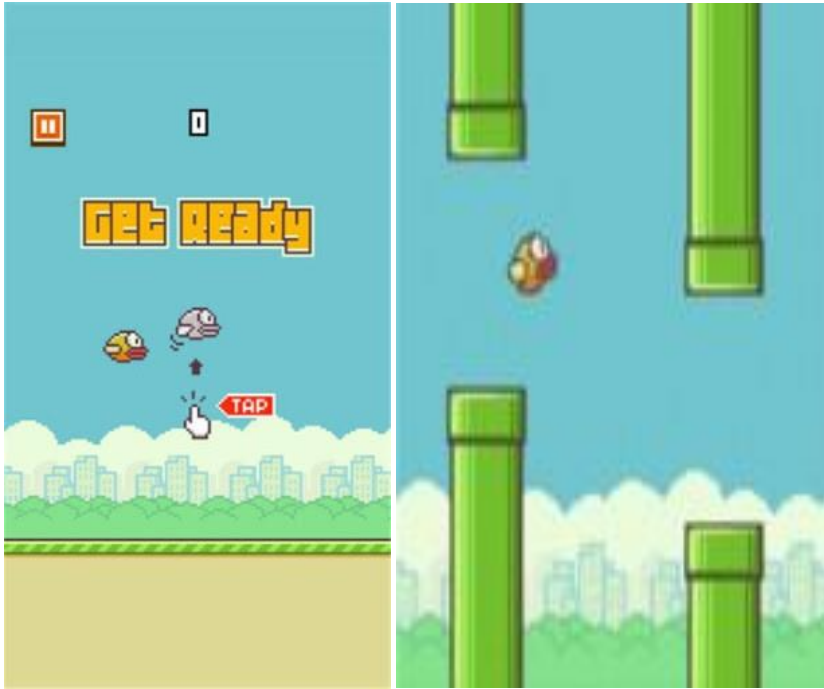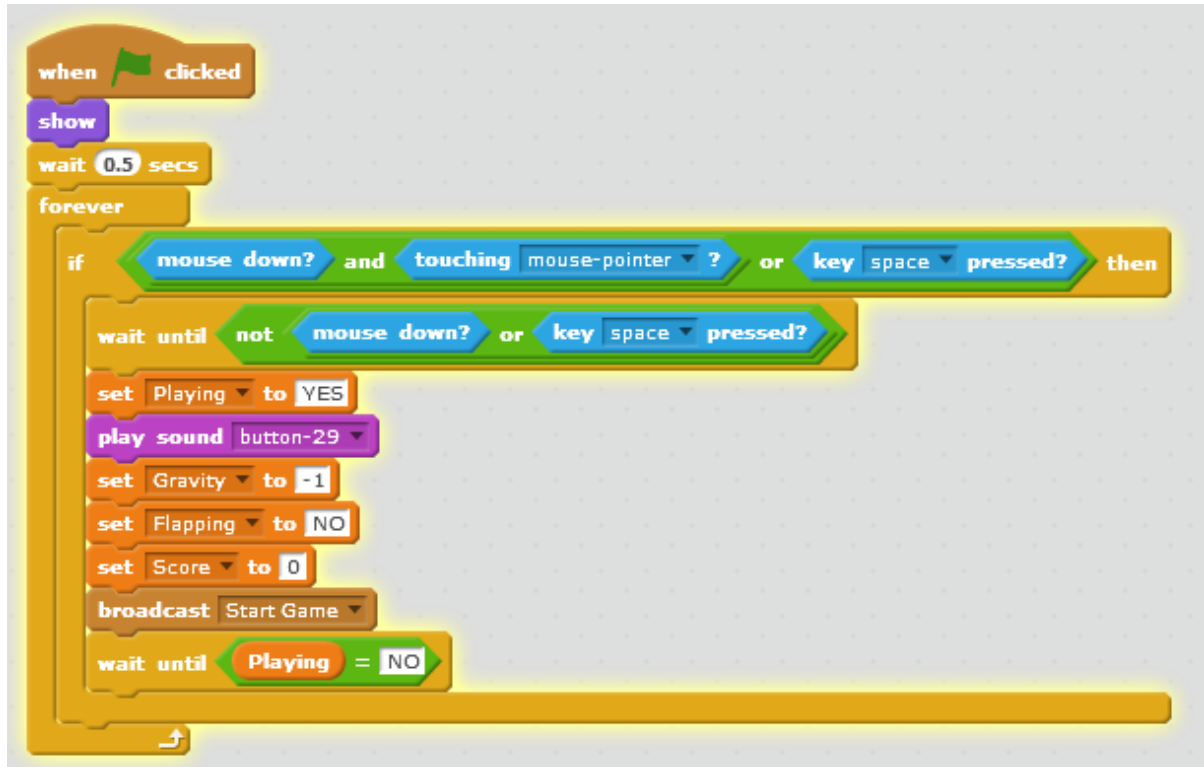# Flappy Bird

Flappy Bird is a great example of a wonderfully simple platform based game that's both simple but challenging, repetitive and with only one aim – beat your own high score. Many of the original video games had these simple attributes and are still being played today!

So how difficult will it be to create this game in Scratch? Well quite a few people have attempted it but @RememberNovember has done a great job and we will use his project to examine how this great platform games can be pieced together!
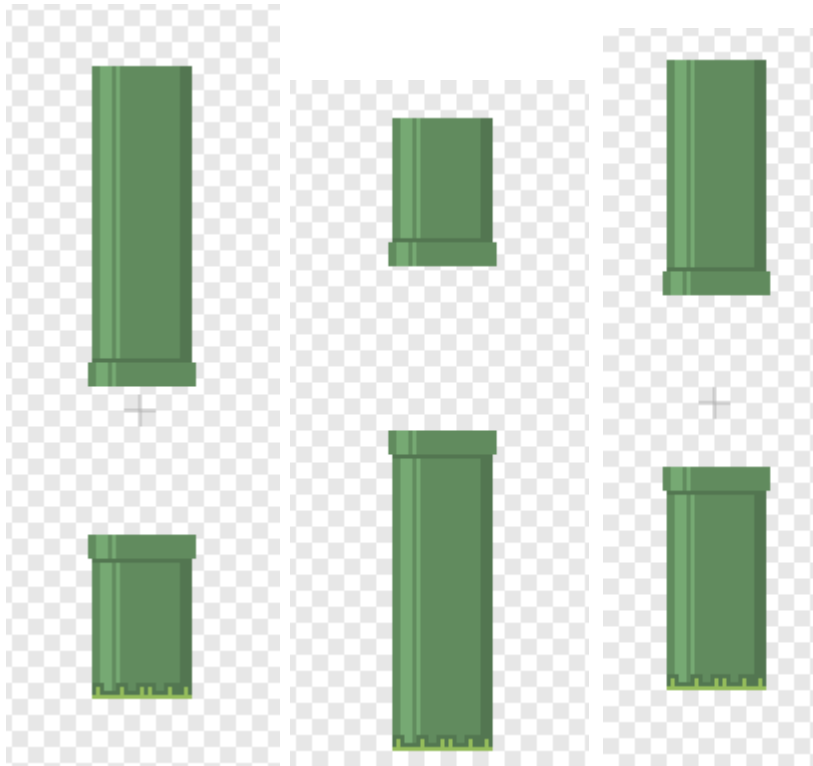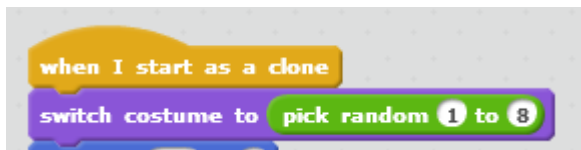
Starting the Game

Here's some interesting code, it tests for you clicking on the sprite or pressing the space key to start, then waits for you to release the mouse or you are still pressing the space bar. But then also waits until the game ends before starting again, so the forever loop is paused between cycles...

```
when [flag] clicked
show
wait 0.5 secs
forever
    if < <mouse down?> and <touching [mouse-pointer v] ?> > or <key [space v] pressed?> then
        wait until < not <mouse down?> or <key [space v] pressed?> >
        set [Playing v] to [YES]
        play sound [button-29 v]
        set [Gravity v] to [-1]
        set [Flapping v] to [NO]
        set [Score v] to [0]
        broadcast [Start Game v]
        wait until < (Playing) = [NO] >
```
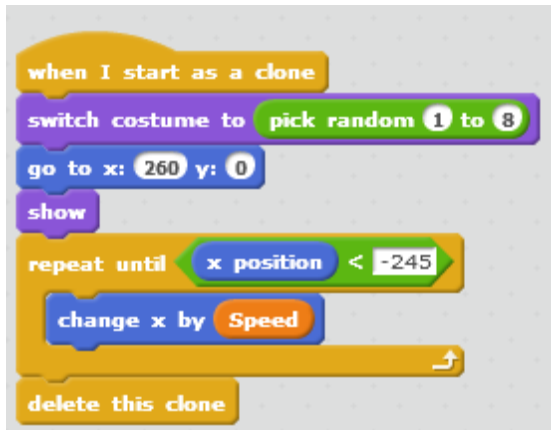
Creating the Course

There's a few options here but the simplest is to just create a few costumes for the Pipe sprites and then select those to appear in a random basis.

Making things Scroll

The whole premise of the game is to make it look like main sprite is moving along at a constant rate and the player controls when the bird flies up or drops down. There is a trick here, although the sprite looks like it's moving in fact it's entirely stationary – it simply moves up and down! That means of course that everything else is moving!

Pipe Sprites need to move from right to left (as do other sprites that are part of the game)

Speed is everything! Here it's set to a variable so we can increase the speed the higher the score – making it more difficult as the game progresses.
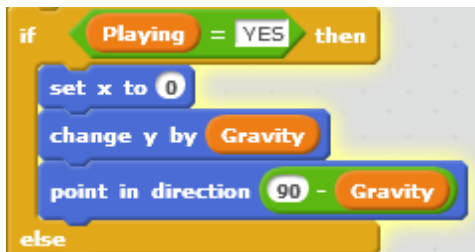
Creating many pipes

And of course we need to keep those pipes coming!

Experiment with this code, rather than making it equal lengths could you randomize this a bit between 2 and 4 seconds?
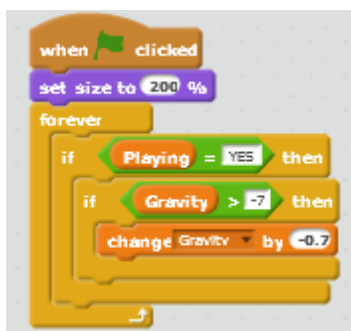
Now for Flappy

Simulating Gravity makes the game more realistic and challenging. Here we control rate of change of the y co-ordinate of the sprite with a variable and also control which way the sprite is pointing using the same variable. The use of a variable means we can effect control of how flappy moves using a single variable.

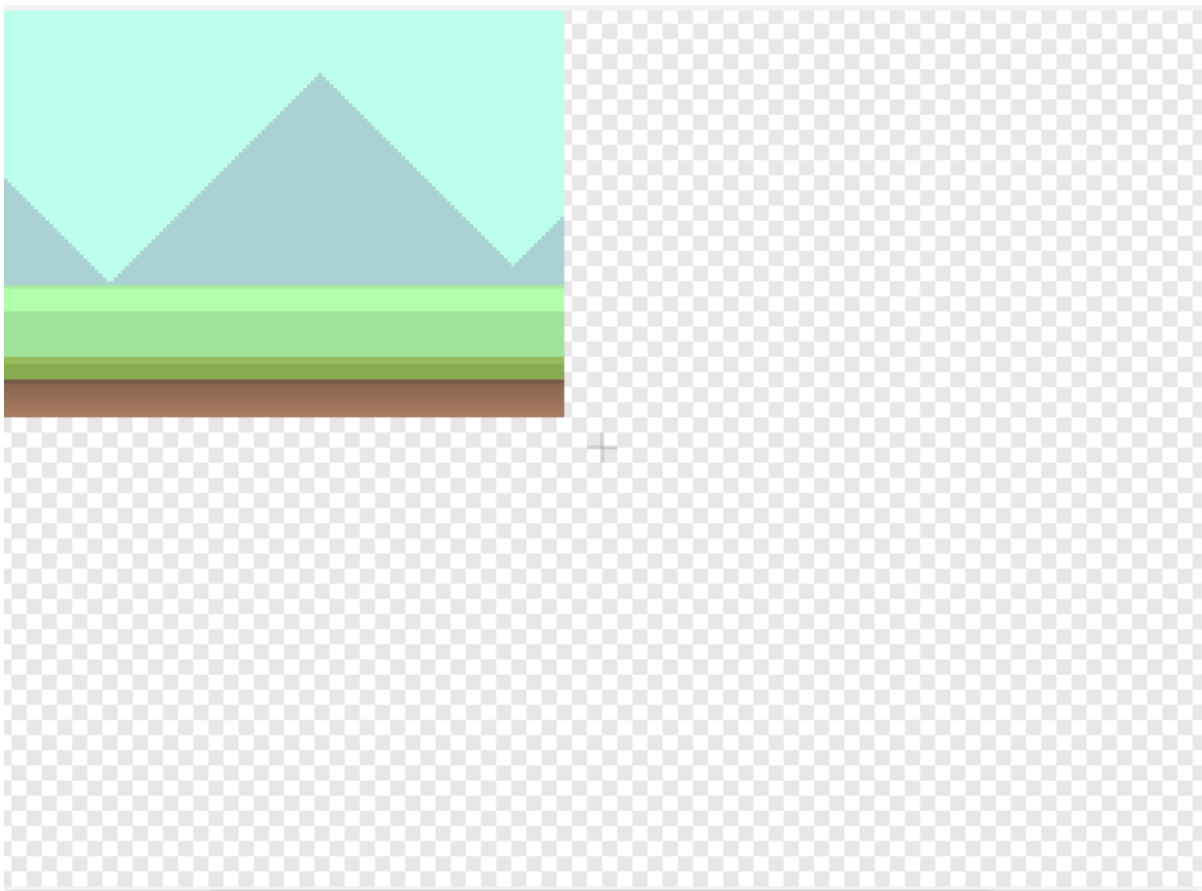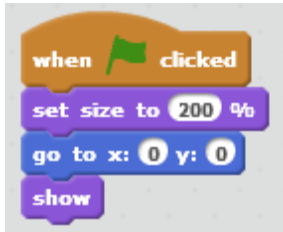So the more gravity the larger the angle of Flappy

Gravity starts at 0 and then is constantly decremented

Until we press a key (e.g. Space Bar) to make Flappy jump/climb by setting Gravity to 8
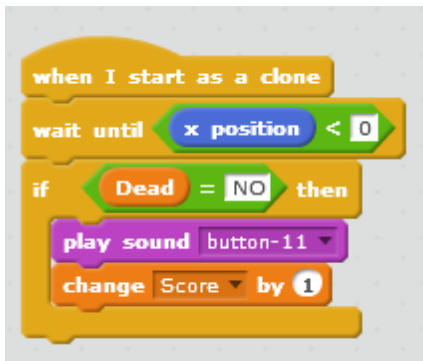
Resizing Sprites

This project uses the "Set Size" block to expand the sprites – could save some time when drawing out your sprites

How to Score

A simple technique – once the pipe reaches a certain position then you can safely increase the score

Figuring out if the Game has ended

Again potentially simple, if you touch a pipe or you hit the bottom (reach a negative y position) then you be dead!

Improvements to the Game

- Add Scoring
- Add a backdrop
- Collecting Bonus Tokens as you go
- Add Health/Lives/Powerups