**Scratch - Flappy Bird Clone**

**Tutorial**

1. Open scratch or create a new scratch project if already open.
2. Delete the existing **Felix** sprite from the stage.  
     
   **ADDING THE BACKGROUND**
3. Create a new sprite from a file.
4. Select either **fb-background-day.**png or **fb-background-night.png**
5. Scale and size the image so it fills the stage completely.
6. Name the sprite “**Background**”.
7. Create the following script:  
     
   
8. Save the Scratch Project to your own folder. Give it a name like “**RamsdenBird**”.  
     
   **SETUP A FEW VARIABLES**
9. Create three **variables** for all sprites:  
     
   score  
   playing  
   gravity
10. Click on the **stage** and add the following script to set all of the variables to start at 0.  
      
      
      
    **ADDING THE PLAYER BIRD CHARACTER**
11. Create a new sprite from a file.
12. Select either **fb-blue-middle.png** or **fb-green-middle.png** depending on your choice of colour for your player character.
13. Rename the sprite to be “**Flappy**”.  
      
    **LETS ANIMATE OUR PLAYER**
14. Import costumes into the **Flappy** sprite. Add them in the following order:  
      
    fb-green-up.png  
    fb-green-middle.png  
    fb-green-down.png  
      
    Remember to change the name of the files you import to green or blue depending on the colour bird you have chosen, or go with a multi-colour player if you want to!
15. To animate the character we move between the available costumes after a small delay. Add the following script to the **Flappy** sprite:  
      
    
16. **Test** your project and see if it works? **Save** it!  
      
    You can change the value of the wait command if you want the character to animate faster or slower. It is your choice!
17. Let’s make sure that the Flappy sprite always starts in the middle of the screen. Add the following script:  
      
      
      
    **WHAT GOES UP, MUST COME DOWN! THE EFFECT OF GRAVITY**
18. Add another script to **Flappy** for him to fall down the screen depending on the value of **gravity**:  
      
    
19. Then add another script to **Flappy** to decrease the value of **gravity** all of the time, up to a limit:  
      
      
      
    **Test** your project and **save** it! You can alter the values in the previous script if you want the character to fall faster or slower. Experiment!  
      
    **RESPOND TO A PLAYER INTERACTION**
20. We want the player to be able to click the mouse or press a key to keep Flappy flapping. Let’s add another script to Flappy:  
      
    
21. **Test** and **save** your project! Does Flappy now respond to a mouse-click or a key press? Is the flying movement as you’d expect? Again you can change the value of the code to get the effect you want.
22. We now want **Flappy** to animate a little more naturally. We need to change the code we added in point 18 to make Flappy fall with gravity, to look like the following:  
      
      
      
    By default Flappy points straight left – this is 90 degrees, 0 is straight up. So, to give the player a nice animation touch we take away gravity from 90 degrees. When Flappy is falling, gravity is negative, taking away a negative adds this value – thus the angle increases and Flappy points downwards more. When we tap the mouse or space-bar gravity is a positive, and thus the angle decreases from 90 and Flappy points upwards more. Try it!
23. **Test** and **save** your project.  
      
    **ADDING THE GAME TITLE**
24. OK, so we now have our player character animating nicely. We need to add more of the game structure, including a title screen. **Add a new sprite from a file**, and select:  
      
    fb-game-title.png
25. Rename the sprite “**Title**” and add a new script to it:  
      
      
      
    We want the title sprite to be shown when the game starts, wait for a mouse click or key pressed and then hide. We set the **playing** variable to be 1 to indicate that the game has started.  
      
    **CONTROLLING WHEN THE GAME IS PLAYED**
26. We now only want **Flappy** to respond and move when the game is started. We know the game is started when playing is set to 1 – so we need to change our existing script that moves **Flappy** depending on **gravity** to check for this **condition**:  
      
    
27. As usual, **test** it and **save** it!  
      
    **LOOK OUT, IT’S A PIPE!**
28. We now need to add the famous pipes, something for Flappy to avoid. **Add a new sprite from a file**, and select:  
      
    fb-green-pipe.png
29. **Name** the sprite, “**Pipe1**”.
30. **Scale** the pipe, making it larger so that it looks something like this:  
      
      
      
    You can change and alter the size of the pipe if you want to make the game easier or harder.
31. Create a new **variable** “**scrollx**” for this sprite only.
32. Now **add** a script to “**Pipe1**” to hide it when the game starts, and set the new **variable** to its initial value:  
      
      
      
    The variable **scrollx** will control the position of the pipe on the screen.
33. We now want the pipe to display and move when the game starts. Add another script to “Pipe1”  
      
      
      
    We reduce **scrollx** by 5 on every loop, and then make the X position of the pipe equal to the variable value.
34. **Test** it and **save** it! The pipe should start on the right-hand side of the screen and move smoothly over to the left-hand side.  
      
    You can speed up or slow down the pipe by adjusting the number that **changes** **scrollx**.
35. In our game, avoiding the pipes is **endless** – it is the type of game we are making an “endless runner”. Add the following script to the pipe.  
      
      
    Let’s look at the script to understand it. We are adding one to the **score** – if the pipe reaches the left-hand side of the screen then the player will have successfully avoided it. We set the **Y** (vertically up or down) position of the pipe to be something **random**. The gameplay element in our game means the player must flap up or down to avoid the pipes. Finally we set scrollx back to be **250** to move the pipe to the right-hand side – remember we update the **X** position (left or right) of the pipe by using the variable **scrollx** in the other piece of script.
36. Test it and save it! Does your pipe move from right to left then re-start at the right-hand side but in a random vertical position?
37. Tidy up your screen: Make sure that all of the **variables** aren’t displayed except for the score. Click the **score** so it shows it in a **number-only format** and not with the label:  
      
      
      
    **PIPES, PIPES AND MORE PIPES!**
38. It’s now easy to add another pipe to increase the difficulty of the game. Duplicate the “Pipe1” sprite to a new sprite. Rename the new sprite to be “Pipe2”.
39. The only thing we need to do to the new pipe is to change its starting position. Alter the bit of code that hides the pipe and sets the first value of scrollx:  
      
    
40. Test it! Save It! Do you have two pipes working as they should?  
      
    **AVOID THE PIPES!**
41. Back in the Flappy sprite we need to test whether the player touches the pipe. If they do then it is game over, the object of the game is to avoid them:  
      
    Be careful adding this script!  
      
    We need to check whether the bird touches the colour at the edge of the pipe. Take care when creating this piece of code. If you want to make the game slightly easier you can test for a touching colour that is one of the shades of green in the pipe.   
      
      
    Setting the **variable** **playing** to 0 means that the bird movement and pipe movement should stop.
42. We have **broadcast** “**gameover**”, so now need a sprite that will show a game over message. **Add a new sprite from a file**. Select:  
      
    fb-game-over!.png
43. **Name** the sprite “**GameOver**”.
44. We just need two final scripts. One to hide the game over graphic when the game starts:  
      
    
45. The next to respond to the “**gameover**” **broadcast** and display the score:  
      
      
      
    Be careful adding this script.  
      
    Make sure that you have a space at the end of “You scored “ otherwise the message might not make sense!
46. Test it! Save it!  
      
    **WELL DONE! NOW EXTEND OR CUSTOMISE YOUR GAME IF YOU HAVE TIME!**

**EXTENSION 1 – ADD A SCROLLING FLOOR**

1. Can you add a scrolling floor?

**EXTENSION 2 – KILL FLAPPY WHEN THE CHARACTER HITS THE FLOOR**

1. Can you add the code for killing the player when they touch the floor?

**CHALLENGES OR CUSTOMISATIONS**

1. Change the “Title” sprite to be whatever you want it to be. Change the name of the game, use different fonts, or draw your own title logo.
2. Change the “GameOver” sprite to be whatever you want it to be. Change the message shown or add your own end of game customisation.
3. Change the bird character – use a different character altogether or customise the look and colours of the provided animations.
4. Several sounds are available in the resource folders. Can you add a sound when a point is scored or when the player dies, or even when the player flaps their wings?
5. Can you get the game to restart without having to press the green flag?
6. Can you record your own sound effects and perhaps even record some music to go with the game?