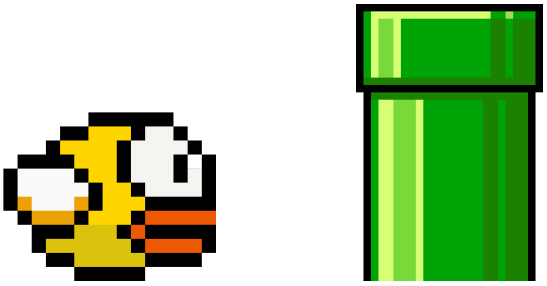
Code Flappy Bird in JavaScript

Beginner Level • 13-16 y/o • 1h30

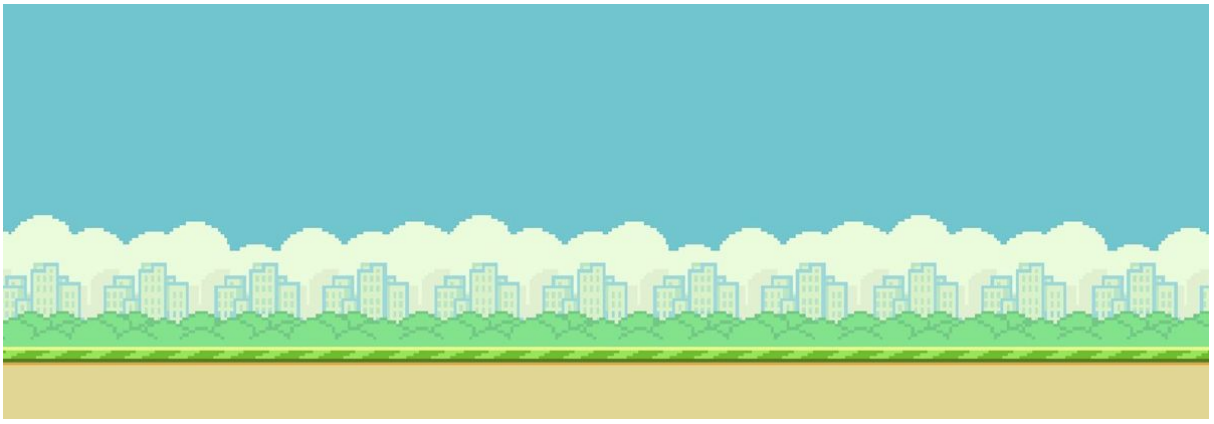
1. Presentation of the game



Flappy Bird is a scrolling mobile game where the player controls a bird and tries to make it fly between the green pipes without touching them.

Some parts of **Flappy Bird**'s code have been lost . . . It's up to you to complete the game code to get it working again!

In this exercise, you will learn how to program loops, conditions in **JavaScript**, and a web game's overall functioning.



2. Tools & Resources

2.1 Resources

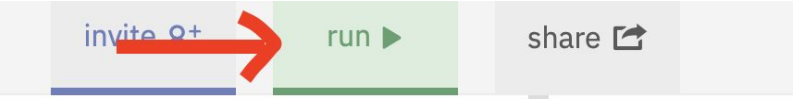
To get started, go to https://repl.it/@EmmaEpitech/FlappyBird.

The screen is split into several parts: the file part, on the left: this is where you will code the **Flappy Bird** game. And the rendered part, on the right: this is where you will see your progress and can test your game.

2.2 Getting started with the platform

On the left side of the screen, you can see a list of files.
HTML files, images, and **JavaScript** files.
Today, you will only need to edit the "**game.js**" file. However, you can go and see what happens in the other files if you wish.

To run your code, press the green arrow "**RUN**" at the top of the screen.



2.3 Game.js

Several functions are present in the **game.js** file. Your role will be to complete them.

```
function setConfiguration()
```

The **setConfiguration** function is one of the very first called in the game. Anything you put in it will therefore be executed at the very start of the game.

```
function preparePipes(bird)
```

The **preparePipes** function will allow you to configure the green pipes and the collisions with Flappy Bird.

```
function onLoop(input, bird)
```

The **onLoop** function is called on each turn of the game loop (several times per second!)

3. Discovery of the game

3.1 Make Flappy Bird jump

If you launch the game, you can see that **Flappy Bird** can't jump!
The first step will therefore be to make it jump when the user clicks on their mouse.

To do this, you must first check in the game loop if the mouse is clicked, then call the **jump** function.

This line allows you to know if the mouse is clicked. Add it in the **onLoop** function

```
if (input.activePointer.isDown) {

}
```

This line allows **Flappy Bird** to jump:

```
jump(-250);
```

Try changing the value to see what happens!

3.2 Add the pipes

Now that **Flappy Bird** knows how to jump, it's time to add the green pipes to the game.

To do this, you must call the two functions that allow you to add the pipes at the top and at the bottom, inside the **preparePipes** function:

```
addTopPipes();
addBottomPipes();
```

Launch the game again to see what happens!

3.3 Space the pipes

You probably noticed that the pipes were superimposed on each other. Now is the time to change the game settings so that they display correctly!

In the **setConfiguration** function, add the following line:

```
setSpaceBetweenPipe(400);
```

Change the value 400 to see what happens!

Now, the pipes are more spaced, but they are all at the same level!
It's up to you to change the height of the pipe by adding the line below after the **setConfiguration** function.

```
setPipesRandomGap(10);
```

3.4 Detect the collisions

Now that the pipes are displayed, and you can jump, it's time to detect the collisions between **Flappy Bird** and the pipes.

At the end of the **preparePipes** function, add the following line:

```
pipesCollidesWith(bird);
```

Launch the game to see the result!

In **Flappy Bird**, if you touch the ground, the game is lost too.
To add this functionality, you need to detect in the game loop if **Flappy Bird** touches the ground via its position on the **Y axis**.

*The ground is located at **Y = 350***
*If you return **false** in the game loop, the game ends.*

Add the following lines after the **onLoop** function:

```
if (bird.y >= 350) {
   return false;
}
```

3.5 Count the points

 Now that your **Flappy Bird** is working, all you have to do is count the points!

For that, you must add 1 point when **Flappy Bird** passes between two pipes.

*This involves comparing the position on the **X axis** of **Flappy Bird** with the next pipe's position.*

```
if (bird.x == ???) {

}
```

The following function allows you to know the position on the **X axis** of the next pipe:

```
nextPipePosition()
```

And the following function allows you to add points:

```
setScore(score + 1);
```

*Press **RUN** and try to pass the pipes to see the points add up!*

Good luck to you!

4. Some useful links

To learn JavaScript:

➔ https://www.w3schools.com/js

To redo the exercise:

➔ https://repl.it/@EmmaEpitech/FlappyBird

For more information on our activities:

➔ https://www.e-mma.org