**Documentation for Programming Project:**

**Flappy Bird**

By Abylay Assalov w66749

Rustem Kuntuganov w66256

Aryn Aikyn w65387

**Lab Title:** Code Your Wings: Building Flappy Bird in C#

**Introduction:** Flappy Bird in C#: Build it yourself. This tutorial guides you through creating the game using Windows Forms and C#.

**Project Overview:**

Challenge yourself by creating a classic Flappy Bird game in Windows Forms with C#. This beginner-friendly project will guide you through:

* Bird control: Master the spacebar to soar through pipes.
* Obstacles galore: Dodge the ground and ever-moving pipes.
* Scorekeeping: Track your skills with points for each gap you navigate.
* Game Over: Crash or fly too high, and it's curtains!

**This is a step-by-step guide to creating a Flappy Bird game in Windows Forms with C#, based on the video you linked.**

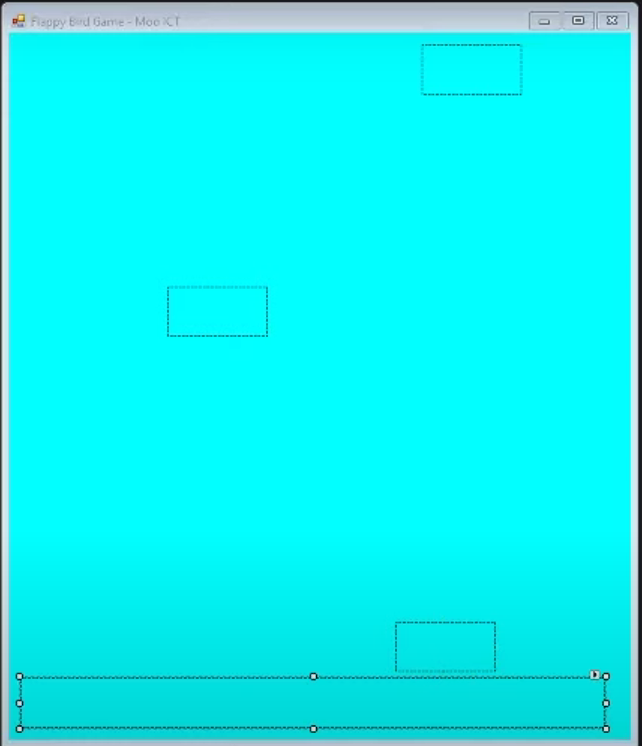
**Step 1:** Create a New Project

1. Open Visual Studio and create a new Windows Form Application project in C#.

2. Name the project and solution as desired.

3. Download the image files (bird, ground, pipe top, pipe bottom)

4. Import the image files into your project resources.



5. Set the picture boxes and label properties:

- Name the picture boxes: ground, **pipeTop**, **pipeBottom**, bird.

- Set the picture boxes' Image properties to the corresponding image files.

- Name the label: **scoreText**.

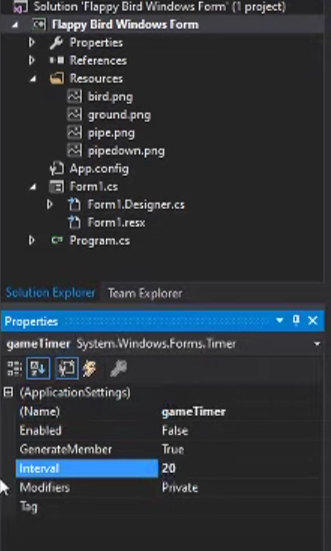
- Set the label's font and size for better visibility.

**Step 2:** Set Up Game Mechanics

1. Add a timer component to the form and name it **gameTimer**.

2. Set the timer interval to 20 milliseconds.

3. Add event handlers for **gameTimer**, **KeyDown**, and **KeyUp** events.

****

4. Define variables:

- **speed** (int) - controls the bird's movement speed.

- **gravity** (int) - controls the bird's downward pull.

- **score** (int) - keeps track of the player's score.

5. In the **gameTimer** event:

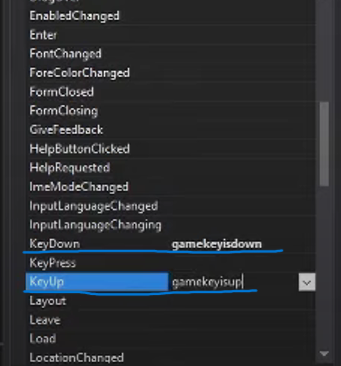
- Move the ground picture box downwards using the gravity variable.

- Move the pipe picture boxes to the left using the **pipeSpeed** variable (not defined yet).

- Update the **scoreText** label with the current score value.

6. Set the timer's Enabled property to false initially.

**Step 3:** Implement Bird Movement and Scoring



1. In the **KeyDown** event:

- Check if the space key is pressed.

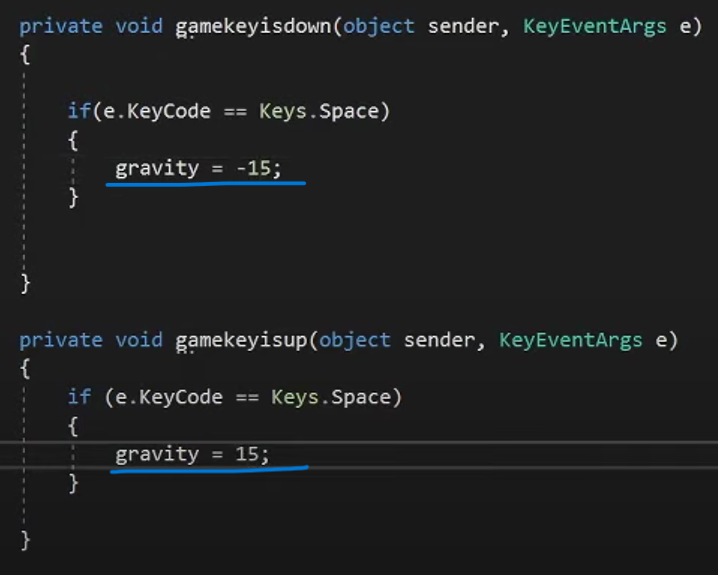
- If pressed, change the **gravity** value to a negative number

(e.g., **-15**) to make the bird move upwards.

2. In the **KeyUp** event:

- Change the **gravity** value back to its original value (e.g., **15**) to make the bird fall again.

3. Test the bird movement by running the application. Adjust the **speed** and **gravity** values for desired behavior.



**Step 4:** Add Pipe Movement and Respawn

1. Define the **pipeSpeed** variable (e.g., -5) to move the pipes leftwards.

2. In the **gameTimer** event:

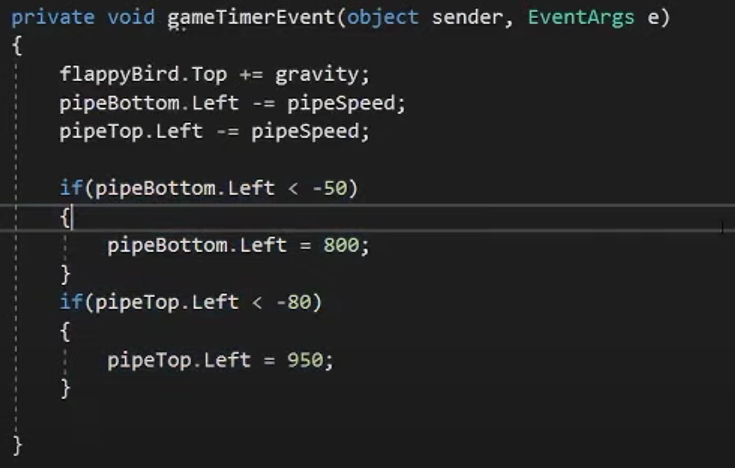
- Update the left positions of the pipe picture boxes using the **pipeSpeed** value.

3. Add an **if** statement to check if a pipe has left the screen (position < **-50**):

- If true, reset its position to **800** (off-screen right) and increment the score.

4. Add another **if** statement to check if a pipe has completely gone off-screen (position < **-630**):

- If true, reset its position to a random value between **800** and **1100** (off-screen right with varied gap).



**Step 5**: Implement Collision Detection and Game Over

1. Define an **endGame** function:

- Stop the timer.

- Display "Game Over" on the screen.

2. In the **gameTimer** event:

- Check for collisions between the bird and ground, pipe top, pipe bottom using their bounding boxes.

- If collision detected, call the **endGame** function.

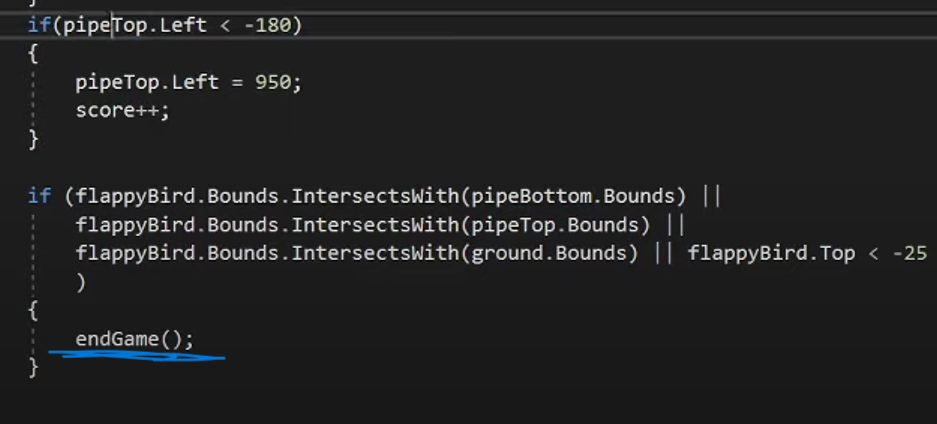
3. Additionally, check if the bird goes above the top of the screen and call **endGame** if so.

**Step 6:** Enhancements and Game Over Text

1. Include an **if** statement in the **gameTimer** event to increase the pipe speed based on the player's score (optional).

2. Display the actual score text next to "Game Over" using string concatenation.

3. Refine the collision detection code for better accuracy.



**Conclusion:**

This step-by-step guide provides a comprehensive walkthrough of creating a Flappy Bird game in C#. Remember, this is a starting point, and you can personalize it further with your own creativity!