```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Flabby Bird</title>
  <style>
    body {
      background: #aeeeee;
      margin: 0;
      overflow: hidden;
      font-family: Arial, sans-serif;
    }
    #gameCanvas {
      display: block;
      margin: 40px auto;
      background: #87ceeb;
      border: 4px solid #333;
      border-radius: 20px;
    }
    #score {
      text-align: center;
      font-size: 2em;
      color: #333;
      margin-top: 10px;
      font-weight: bold;
    }
  </style>
</head>
<body>
  <div id="score">Score: 0</div>
  <canvas id="gameCanvas" width="400" height="600"></canvas>
  <script>
    // Get canvas and context
    const canvas = document.getElementById('gameCanvas');
    const ctx = canvas.getContext('2d');
    const scoreDiv = document.getElementById('score');

    // Game variables
    let birdY = 300;
    let birdX = 80;
    let birdRadius = 25;
    let birdVelocity = 0;
    let gravity = 0.5;
    let flapStrength = -8;
```

```
let pipes = [];
let pipeWidth = 60;
let pipeGap = 160;
let frame = 0;
let score = 0;
let gameOver = false;

// Bird image (optional, using a simple circle for now)
function drawBird() {
  ctx.save();
  ctx.beginPath();
  ctx.arc(birdX, birdY, birdRadius, 0, Math.PI * 2);
  ctx.fillStyle = "#ffcc66";
  ctx.fill();
  ctx.strokeStyle = "#e59400";
  ctx.lineWidth = 4;
  ctx.stroke();
  // Eye
  ctx.beginPath();
  ctx.arc(birdX + 10, birdY - 8, 5, 0, Math.PI * 2);
  ctx.fillStyle = "#333";
  ctx.fill();
  ctx.restore();
}

// Draw pipes
function drawPipes() {
  ctx.fillStyle = "#228B22";
  pipes.forEach(pipe => {
    // Top pipe
    ctx.fillRect(pipe.x, 0, pipeWidth, pipe.top);
    // Bottom pipe
    ctx.fillRect(pipe.x, pipe.top + pipeGap, pipeWidth, canvas.height - pipe.top - pipeGap);
  });
}

// Update pipes
function updatePipes() {
  if (frame % 90 === 0) {
    // Random gap position
    let top = Math.floor(Math.random() * (canvas.height - pipeGap - 100)) + 50;
    pipes.push({ x: canvas.width, top });
  }
  pipes.forEach(pipe => pipe.x -= 2);
```

```
  // Remove off-screen pipes
  if (pipes.length && pipes[0].x < -pipeWidth) {
    pipes.shift();
    score++;
    scoreDiv.textContent = "Score: " + score;
  }
}

// Collision detection
function checkCollision() {
  // Ground or ceiling
  if (birdY + birdRadius > canvas.height || birdY - birdRadius < 0) {
    return true;
  }
  // Pipes
  for (let pipe of pipes) {
    if (
      birdX + birdRadius > pipe.x &&
      birdX - birdRadius < pipe.x + pipeWidth &&
      (birdY - birdRadius < pipe.top || birdY + birdRadius > pipe.top + pipeGap)
    ) {
      return true;
    }
  }
  return false;
}

// Main game loop
function gameLoop() {
  ctx.clearRect(0, 0, canvas.width, canvas.height);

  // Draw and update
  drawBird();
  drawPipes();

  if (!gameOver) {
    birdVelocity += gravity;
    birdY += birdVelocity;
    updatePipes();

    if (checkCollision()) {
      gameOver = true;
      scoreDiv.textContent = "Game Over! Final Score: " + score + " (Tap or Space to Restart)";
```

```
      } else {
        frame++;
        requestAnimationFrame(gameLoop);
      }
    }
  }

  // Flap event
  function flap() {
    if (!gameOver) {
      birdVelocity = flapStrength;
    } else {
      // Restart game
      birdY = 300;
      birdVelocity = 0;
      pipes = [];
      frame = 0;
      score = 0;
      gameOver = false;
      scoreDiv.textContent = "Score: 0";
      requestAnimationFrame(gameLoop);
    }
  }

  // Controls
  document.addEventListener('keydown', e => {
    if (e.code === 'Space') flap();
  });
  canvas.addEventListener('mousedown', flap);
  canvas.addEventListener('touchstart', flap);

  // Start game
  requestAnimationFrame(gameLoop);
 </script>
</body>
</html>
```