Pigeon Preventer

by

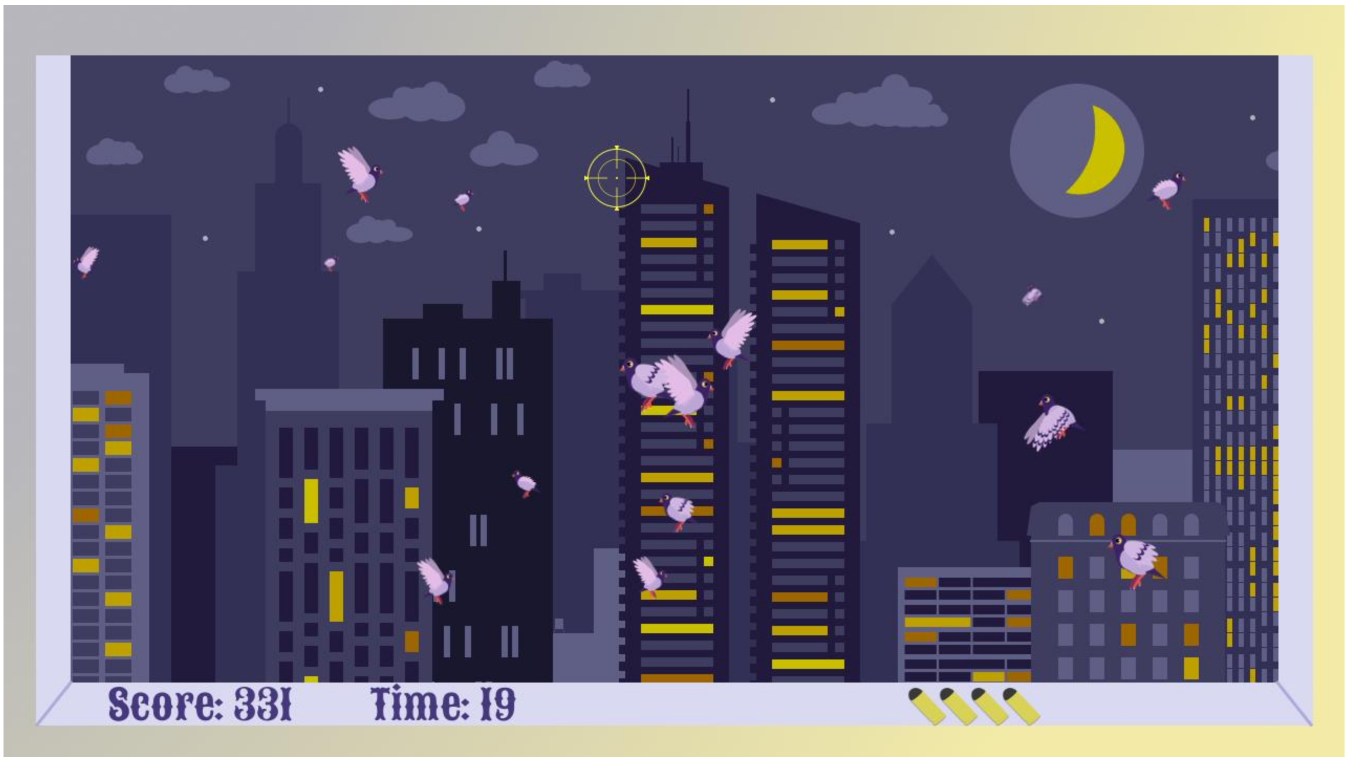
Tiziano Schacht

&

Manuel Wüst

Pigeon Preventer

(By Tiziano Schacht and Manuel Wüst)



What we have done & how

Pigeon Preventer is a Moorhuhn inspired shooter where the player must shoot down the pigeons that want to s(h)it on your balcony. You have to collect as many points as possible in the given time.

We decided to use a window size of 1280x720 pixel for the implementation. The scene got inspired by a internet image which shows a city at night from your window in a big tower building.

The game's background music is a classic. It is a free version of the popular "Cantina Band" music from Star Wars.

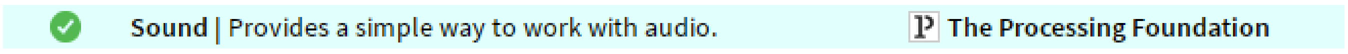
Results

We created all the scenes and assets, ourselves with the Software Figma. An exception is the pigeon, which we found online and made some adjustments. We have also downloaded all sounds (free music) online and edited them in Audacity to fit our project.

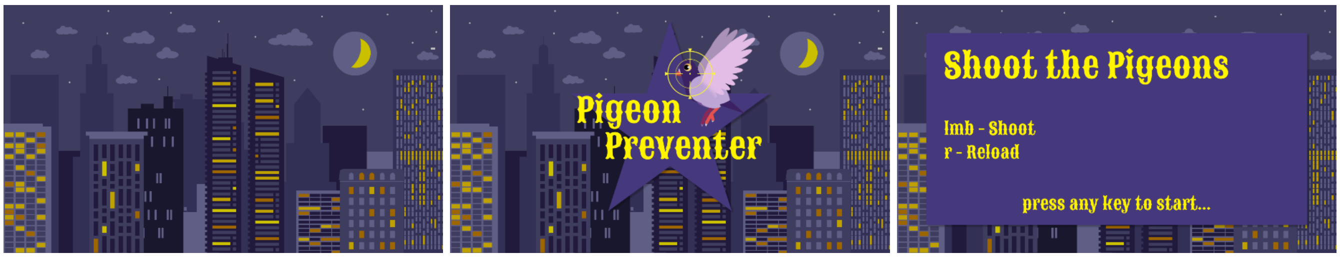
Library and Versions

The game requires **Processing 4.0**.

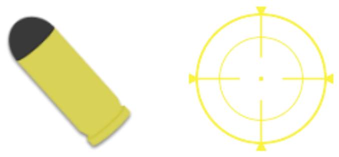
We used the official **Sound library** from The Processing Foundation. To install it click “Sketch” -> “Library importieren” and then search for “Sound”.


✓ Sound | Provides a simple way to work with audio. The Processing Foundation

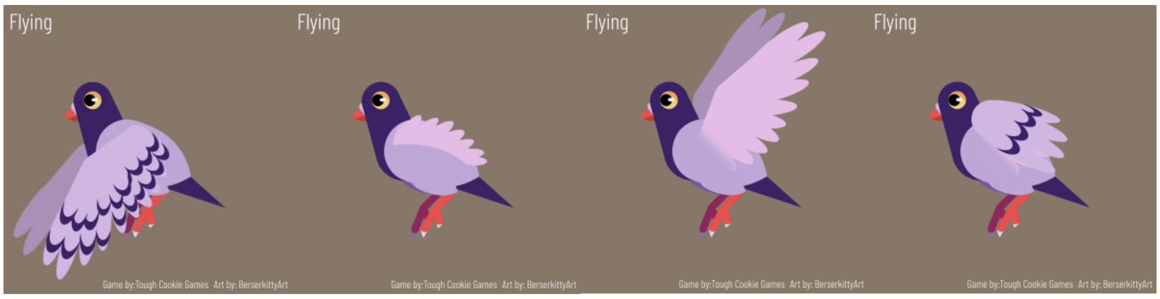
Art and screen creation



Asset creation



Flying Animation


Flying
Flying
Flying
Flying
Game by:Tough Cookie Games Art by: BerserkittyArt
Game by:Tough Cookie Games Art by: BerserkittyArt
Game by:Tough Cookie Games Art by: BerserkittyArt
Game by:Tough Cookie Games Art by: BerserkittyArt

Feather Animation

When a pigeon is shot, it disappears and is replaced by falling feathers.

Algorithms & Game Mechanics

Game state

The game includes different game states:

0. the loading screen until a button is pressed
1. the info screen, showing how to play the game
2. the game itself
3. the highscore screen when the game is over

From the game state 3, you can only go back to 2 by clicking the mouse button on the highscore screen.

Animation Controller

While instantiating the animation object all the frames for the animation will be loaded into an array. Each time the displayLoop() method is called, the next frame in the array will be displayed at a certain position. The frame will be flipped according to the pigeons flight direction.

```
void displayLoop(PVector pos) {
  frame = (frame + speed) % count;
  push();
  translate(pos.x, pos.y);
  if (flip) {
    scale(-1, 1);
  }
  image(images[(int)frame], 0, 0);
  pop();
}
```

Flying position

While doing an animation, the pigeons also fly slightly up and down when going into a direction:

```
void calcFlyingPos() {
  pos.x = pos.x + velocity;
  pos.y = pos.y + (float)Math.cos((flyAnimation.getFrame()+1)*90);

  if (pos.x < -50 && velocity < 0) {
    pos.x = width + 50;
    pos.y = random(100, height - 100);
  } else if (pos.x > width + 50 && velocity > 0) {
    pos.x = -50;
    pos.y = random(100, height - 100);
  }
}
```

Shooting detection and score calculation

The gun is shot when the left mouse button is clicked:

```
if (gun.shot()) {
  for (int i = 0; i < pigeonCount; i++) {
    score += pigeons[i].gotShot(new PVector(mouseX, mouseY));
  }
}
```

The gotShot() method in the pigeon class calculates if the pigeon got shot by checking the coordinates where the click was made and checks if the current position of the pigeon instance is at the same position. It then sets the “dead” Boolean to true and plays two sounds which signal the hit. It then returns the score based on speed and size of the pigeon:

```
int gotShot(PVector mousePos) {
  if (pos.dist(mousePos) < size / 2) {
    dead = true;
    scream.play();
    wingFlutter.play();
    return (int)(50 - Math.abs(this.size * 10 * (1 / this.velocity * 10)*2)/400);
  }
  return 0;
}
```

Gun mechanics

The gun has 6 pieces of ammunition in the chamber. After each shot, the ammo is reduced. When it is empty, the “no Ammo”- sound is played. When typing “r”, the gun reloads with a reload-sound and the ammunition sound is set back to 6.

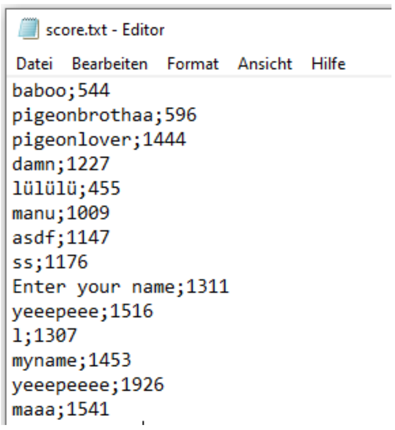
```
if (key == 'r') {
  gun.reload();
}
```

```
boolean shot() {
  if (!reloadSound.isPlaying()) {
    if (ammo > 0) {
      ammo--;
      shotSound.play();
      return true;
    } else {
      noAmmoSound.play();
    }
  }
  return false;
}

void reload() {
  reloadSound.play();
  ammo = 6;
}
```

Highscore

The score of each game is saved in the score.txt file when clicking the mouse button after typing in the name.



```
baboo;544
pigeonbrothaa;596
pigeonlover;1444
damn;1227
lülülü;455
manu;1009
asdf;1147
ss;1176
Enter your name;1311
yeeepeee;1516
1;1307
myname;1453
yeeepeeee;1926
maaa;1541
```

Small discussion

What works well

We think the animations in the game looks very nice and the sound-feedback when playing is very satisfying. Also, the loading and info-screen with a short tutorial is useful and well-done.

Oh, and we like the background music a lot! 😊

Issues / Problems

Relatively long loading time in the beginning, mostly because of the sound-files. The file-size has been reduced and the scene loading optimized, but it still takes a while.

The name-entry at the end of a game-round is something to get used to and is not fully tested. This is something that could be improved in the future.

Links and References

Sound library - https://processing.org/reference/libraries/sound/index.html

Pigeon - https://www.artstation.com/artwork/lVXXXG

Cantina Rag by Jackson F. Smith - https://freemusicarchive.org/music/Jackson_F_Smith/Jackson_Frederick_Smith/Cantina_Rag