飞机大战·编程语言版

一. 创意介绍

作为《数据结构与算法》Python 班中的一员，在我们的心里 Python 就是世界上最好的语言~~（才不是 PHP）。~~为了体现 Python 在一众编程语言中的领先地位，我特地设计了飞机大战之编程语言版，将各类编程语言化身为敌机，而英勇的你将守护好 Python 老巢的防线，绝不让除 Python 外的任何一种编程语言通过防线。更有隐藏开挂模式，化 Python 为武器，对敌军进行疯狂扫射攻击（当然也很可能误伤友军）。体验刺激飞行体验，尽在飞机大战·编程语言特别版！（手动狗头）

二. 实现方案

其实一开始想用 pygame，但作为《数据结构与算法》Python 班的学生，不应把自己局限在一门编程语言中，而要敢于走出去探索新世界，使用新语言。最终经过广泛调研（看 CSDN 上的小广告）和慎重考虑（pygame 学不会……），在多方权衡（看哪个教程最好找）之后，选择了 PixiJS 项目作为游戏界面使用的框架。主要原因如下：

1. 高速：快速利用 WebGL 渲染高质量游戏画面，
2. 简单：无需了解渲染引擎的具体知识
3. 丰富的文档和示例~~（方便白嫖）~~：有非常炫酷的官网（https://pixijs.com/） 和比较落后的中文翻译（还在用 4.x 的版本）（http://pixijs.huashengweilai.com/）

整个项目共分为四个部分：

第一部分：游戏界面与简单操控逻辑的实现，基于 pixi.js 项目，使用 html+css+javascript 三件套实现（主要靠 js），在具体的贴图上使用了网上下载好的现成的贴图，自己制作了敌机（各大编程语言 logo）的贴图，文件结构图如下：

```
game
├── assets
│   ├── c++.png
│   ├── C.png
│   ├── go.png
│   ├── html.png
│   ├── java.png
│   ├── js.png
│   ├── php.png
│   ├── pixi.min.js
│   ├── python.png
│   └── style.css
├── index.html
└── res
    ├── bullet_01.png
    ├── enemy_01.png
    ├── img_bg_level_3.jpg
    ├── img_bg_level_4.jpg
    ├── img_bullet_14.png
    └── plane_blue_01.png
```

其中 res 文件夹包含了网上下载下来的贴图，assets 文件夹包括自己制作的编程语言 logo 贴图、pixi.min.js 项目库（由于目前中文教程较为滞后，而我是照着教程写的，所以使用的是较旧的 v4.3.4 版本）、style.css 文件（没啥用，主要是简单排版一下右边的游戏规则）；最后需要自己写的内容基本都集中在 index.html 中，详细的代码分析和展示见附录。

第二部分：数据通信，使用 fastapi 框架构建服务器，解决 js 和 python 之间的状态通信问题，代码见 main.py（其实因为一直在本地调试，没必要加入 CORS 跨域策略的调整，但是写习惯了，就直接套了个模板）

第三部分：服务端。利用 python usb 通信库 PySerial 监听 usb 端口，利用 micro bit 的串口通信功能获取按钮的状态信息，实现 micro bit 和电脑的交互。再通过 pykeyboard 库模拟出键盘操作，实现对游戏的操控，代码见 server.py：

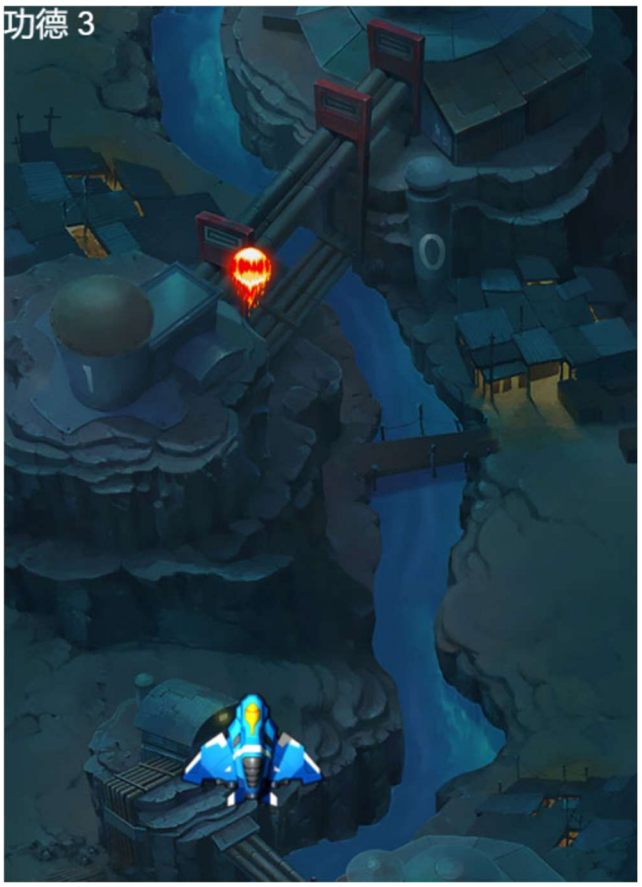
第四部分：客户端。即跑在 micro bit 上的代码，这一部分主要实现对按键状态的捕获并通过内置的 print 方法传输至串口，以及一些较简单的状态显示功能。代码见 client.py：

三．目前的不足

1. 过度的性能浪费。Pixi.js 就需要一个 JavaScript 运行时，我又开了个循环监听的服务端，后来觉得 python 和 js 联系不到一起，于是又开了个 fastapi 服务。简直是不知死活，浪费性能的典范

2. 极其简陋的界面设计

全靠素材撑着，除掉素材一无所有



3. 硬件监听有时间间隔，使用体验不够丝滑。考虑到你也不能让 microbit 完全不休息，键盘模拟的时候 press 与 release 也必须要有间隔，所以就这样了吧

四. 小组分工

孤家寡人，实在惭愧

五. 附录

index.html 详细注释版

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <script src="./assets/pixi.min.js"></script>
  <link rel="stylesheet" href="./assets/style.css">
</head>

<body>
  <script>
    //创建应用
    var app = new PIXI.Application(512, 768);
    document.body.appendChild(app.view);
    app.view.style.height = "100%";

    //添加背景
    var bg = new PIXI.Sprite.fromImage("res/img_bg_level_4.jpg");
    app.stage.addChild(bg);

    //添加子弹
    var bulletImage = "res/bullet_01.png";
    var bullet = new PIXI.Sprite.fromImage(bulletImage);
    bullet.anchor.x = 0.5;   //设置飞机图片锚点为图片中心
    bullet.anchor.y = 0.5;
    app.stage.addChild(bullet);
    var bullets = [bullet]; //子弹数组，用于存放子弹

    //创建飞机图片
    var plane = new PIXI.Sprite.fromImage("res/plane_blue_01.png");
    app.stage.addChild(plane);
    plane.anchor.x = 0.5;
    plane.anchor.y = 0.5;
    plane.x = 200;
    plane.y = 600;
    plane.vx = 0;
    plane.vy = 0;

    //添加敌机，从一系列不同的 Logo 中选择，第一个固定为 C++
    var picOfEnemy = ["assets/c++.png", "assets/java.png", "assets/pytho
            n.png","assets/C.png","assets/go.png","assets/js.png","asse
            ts/php.png","assets/html.png"];
    var enemy = new PIXI.Sprite.fromImage("assets/c++.png");
    enemy.scale.set(0.8, 0.8);
    app.stage.addChild(enemy);
    enemy.anchor.x = 0.5;   //设置飞机图片锚点为图片中心
    enemy.anchor.y = 0.5;
```

```
    enemy.x = 200;
    var enemies = [enemy];  //存放敌机的数组

    //创建得分显示文本
    var scoreTxt = new PIXI.Text("功德 0", { fill: "white" });
    app.stage.addChild(scoreTxt); //将文本添加到舞台
    var score = 0; //得分变量，记录得分使用

    var isGameOver = false; //是否游戏结束
    app.ticker.add(delta => animate(delta)); //指定帧频函数
    var timer = 0; //计时器，用于计算子弹发射间隔
    var interval = 50; //子弹发射间隔
    var einterval = 100; //敌机出现间隔
    var isUp = false; //是否进入奖励时间
    var cheat = false; //是否开启隐藏模式
    var crashSize = 40;
    var times = 0; //进入奖励时间的次数，用于动态调整难度

    let left = keyboard("ArrowLeft"),
      right = keyboard("ArrowRight");
      up = keyboard("ArrowUp");
      down = keyboard("ArrowDown");
      enter = keyboard("Enter");

    //监听键盘事件，上下左右与回车键
    //Left arrow key `press` method
    left.press = () => {
      plane.vx = -10;
      plane.vy = 0;
    };
    //Left arrow key `release` method
    left.release = () => {
      if (!right.isDown && plane.vy === 0) {
        plane.vx = 0;
      }
    };

    //Right
    right.press = () => {
      plane.vx = 10;
      plane.vy = 0;
    };
    right.release = () => {
      if (!left.isDown && plane.vy === 0) {
```

```
93.         plane.vx = 0;
94.       }
95.     };
96.
97.     up.press = () => {
98.       plane.vy = -5;
99.       plane.vx = 0;
100.    };
101.    up.release = () => {
102.      if (!down.isDown && plane.vx === 0) {
103.        plane.vy = 0;
104.      }
105.    };
106.
107.    down.press = () => {
108.      plane.vy = 5;
109.      plane.vx = 0;
110.    };
111.    down.release = () => {
112.      if (!up.isDown && plane.vx === 0) {
113.        plane.vy = 0;
114.      }
115.    };
116.
117.    // enter presses 记录按键次数，实现两种状态间的切换
118.    var presses = 0;
119.    enter.press = () => {
120.      presses++;
121.      if (presses % 2 == 1) {
122.        //进入作弊模式，子弹发射间隔和敌人生成间隔同时缩短，子弹外形变为
              python
123.        interval = 15;
124.        einterval = 50;
125.        bulletImage = "/assets/python.png";
126.        cheat = true;
127.      } else {
128.        interval = 50;
129.        einterval = 100-10*times;
130.        bulletImage = "res/bullet_01.png";
131.        cheat = false;
132.      }
133.    }
134.
135.    //主游戏动画函数
```

```
136.    function animate(delta) {
137.      if (isGameOver == true) { //如果游戏结束，则不执行下面动画
138.        return;
139.      }
140.
141.      // 如果timer满interval，发射子弹
142.      timer++;
143.      if (timer % interval==0) {
144.        // 新建子弹，存入子弹数组
145.        var newBullet = new PIXI.Sprite.fromImage(bulletImage);
146.        newBullet.anchor.x = 0.5;   //设置飞机图片锚点为图片中心
147.        newBullet.anchor.y = 0.5;
148.        newBullet.x = plane.x;
149.        newBullet.y = plane.y;
150.        app.stage.addChild(newBullet);
151.        bullets.push(newBullet);
152.      }
153.      if (timer % einterval==0) {
154.        // 新建敌机，存入敌机数组
155.        var pic = picOfEnemy[Math.floor(Math.random() * picOfEnemy.lengt
              h)];
156.        var newEnemy = new PIXI.Sprite.fromImage(pic);
157.        newEnemy.anchor.x = 0.5;   //设置飞机图片锚点为图片中心
158.        newEnemy.anchor.y = 0.5;
159.        newEnemy.x = Math.random() * 450 + 30; //敌机水平位置随机
160.        newEnemy.y = -100;
161.        newEnemy.scale.set(0.8, 0.8);
162.        // 延迟随机秒后出现
163.        setTimeout(function () {
164.          app.stage.addChild(newEnemy);
165.          enemies.push(newEnemy);
166.        }, Math.random() * 200);
167.      }
168.
169.      plane.x += plane.vx; // 移动自己
170.      plane.y += plane.vy;
171.      // 如果越界，复位
172.      if (plane.x < 0) {
173.        plane.x = 0;
174.      }
175.      if (plane.x > 500) {
176.        plane.x = 500;
177.      }
178.      if (plane.y < 0) {
```

```
        plane.y = 0;
      }
      if (plane.y > 700) {
        plane.y = 700;
      }

      //背景移动
      bg.y += 2;
      if (bg.y >= 0) {
        bg.y = -768;
      }

      // 敌机移动
      for (var i = 0; i < enemies.length; i++) {
        var enemy = enemies[i];
        enemy.y += 5;
        if (enemy.y > 800) {
          if (enemy.texture.baseTexture.imageUrl=="assets/python.png") {
           // Python 可以正常通过
            app.stage.removeChild(enemy);
            enemies.splice(i, 1);
            continue;
          }
         // 其他情况判定防御失败
          isGameOver = true;
          if (confirm("敌军已突破防线，是否重开？") == true) {
            window.location.reload();
          }
          continue;
        }
        //玩家飞机与敌机碰撞
        if (hitTestRectangle(plane, enemy)) {
          //游戏结束标记
          isGameOver = true;
          //是否重玩
          if (confirm("您已坠机！") == true) {
            window.location.reload();
          }
        }
      }
      // 遍历bullets 中的bullet
      for (var j = 0; j < bullets.length; j++) {
        var bullet = bullets[j];
        bullet.y -= 10;
```

```
223.         if (bullet.y < 10) {
224.           app.stage.removeChild(bullet);
225.           bullets.splice(j, 1);
226.           continue;
227.         }
228.         // 遍历enemies 中的enemy
229.         for (var i = 0; i < enemies.length; i++) {
230.           var enemy = enemies[i];
231.           //子弹与敌机碰撞
232.           if(hitTestRectangle(bullet, enemy)){
233.             // 将敌机移出数组
234.             app.stage.removeChild(enemy);
235.             enemies.splice(i, 1);
236.             // 将子弹移出数组
237.             app.stage.removeChild(bullet);
238.             bullets.splice(j, 1);
239.             // 如果敌机图片是python
240.             if (enemy.texture.baseTexture.imageUrl=="assets/python.png")
                  {
241.               score-=2;
242.               scoreTxt.text = "功德 " + score + " 误伤友军！";
243.               scoreTxt.style.fill = 0xff0000;
244.               setTimeout(function() {
245.                 scoreTxt.style.fill = 0xffffff;
246.               }, 1000);
247.               continue;
248.             }
249.             //得分+1
250.             score++;
251.             if(isUp){
252.               scoreTxt.text = "功德 " + score + " 奖励时间！";
253.             }
254.             else{
255.               scoreTxt.text = "功德 " + score;
256.             }
257.             if(score % 20==0 && score!=0 && !cheat) {
258.               // 奖励模式参数调整，武器更换、减少子弹发射间隔
259.               isUp = true;
260.               scoreTxt.text = "功德 " + score + " 奖励时间！";
261.               times++;
262.               interval = 30;
263.               einterval = 50;
264.               bulletImage = "res/img_bullet_14.png";
265.               crashSize = 80;
```

```
                // 奖励时间持续5s后复原
                setTimeout(function() {
                  isUp = false;
                  interval = 50;
                  einterval = 100-times*10; // 动态难度，提升敌机出现速度
                  bulletImage = "res/bullet_01.png";
                  crashSize = 40;
                }, 5000);
              }
            }
          }
        }

        // copied from tutorial，定义keyboard类，监听键盘事件
        function keyboard(value) {
          let key = {};
          key.value = value;
          key.isDown = false;
          key.isUp = true;
          key.press = undefined;
          key.release = undefined;
          //The `downHandler`
          key.downHandler = event => {
            if (event.key === key.value) {
              if (key.isUp && key.press) key.press();
              key.isDown = true;
              key.isUp = false;
              event.preventDefault();
            }
          };

          //The `upHandler`
          key.upHandler = event => {
            if (event.key === key.value) {
              if (key.isDown && key.release) key.release();
              key.isDown = false;
              key.isUp = true;
              event.preventDefault();
            }
          };

          //Attach event listeners
          const downListener = key.downHandler.bind(key);
```

```
310.      const upListener = key.upHandler.bind(key);
311.
312.      window.addEventListener(
313.        "keydown", downListener, false
314.      );
315.      window.addEventListener(
316.        "keyup", upListener, false
317.      );
318.
319.      // Detach event listeners
320.      key.unsubscribe = () => {
321.        window.removeEventListener("keydown", downListener);
322.        window.removeEventListener("keyup", upListener);
323.      };
324.
325.      return key;
326.    }
327.
328.    // copied from tutorial, 碰撞判断，引入难度系数机制，可以通过修改
           difficulty 的值改变碰撞的判定难度
329.    function hitTestRectangle(r1, r2) {
330.      difficulty = 1.12;
331.      //Define the variables we'll need to calculate
332.      let hit, combinedHalfWidths, combinedHalfHeights, vx, vy;
333.      //hit will determine whether there's a collision
334.      hit = false;
335.      //Find the center points of each sprite
336.      r1.centerX = r1.x + r1.width / 2;
337.      r1.centerY = r1.y + r1.height / 2;
338.      r2.centerX = r2.x + r2.width / 2;
339.      r2.centerY = r2.y + r2.height / 2;
340.      //Find the half-widths and half-heights of each sprite
341.      r1.halfWidth = r1.width / 2;
342.      r1.halfHeight = r1.height / 2;
343.      r2.halfWidth = r2.width / 2;
344.      r2.halfHeight = r2.height / 2;
345.      //Calculate the distance vector between the sprites
346.      vx = r1.centerX - r2.centerX;
347.      vy = r1.centerY - r2.centerY;
348.      //Figure out the combined half-widths and half-heights
349.      combinedHalfWidths = (r1.halfWidth + r2.halfWidth) / difficulty;
350.      combinedHalfHeights = (r1.halfHeight + r2.halfHeight) / difficulty
             ;
351.      //Check for a collision on the x axis
```

```
      if (Math.abs(vx) < combinedHalfWidths) {
        //A collision might be occurring. Check for a collision on the y
            axis
        if (Math.abs(vy) < combinedHalfHeights) {
          //There's definitely a collision happening
          hit = true;
        } else {
          //There's no collision on the y axis
          hit = false;
        }
      } else {
        //There's no collision on the x axis
        hit = false;
      }
      //`hit` will be either `true` or `false`
      return hit;
    };
  </script>

  <div class="rule">
    <h1>规则</h1>
    <p>1. 消灭所有敌人，如果被敌人通过防线，即判定为负</p>
    <p>2. 如果与敌人或者友军发生碰撞，判定为坠机</p>
    <p>（注：唯一的友军就是 Python）</p>
    <p>3. 每消灭一个敌人，会获得 1 分，如果误伤友军，会扣除 2 分</p>
    <p>4. 友军可以正常通过防线，不会触发判定</p>
    <p>5. 每消灭 20 个敌人，会获得 5s 奖励时间，此段时间内武器升级，但敌人的生成
        速度也会加快</p>
    <p>6. 为了防止游戏时间过长，每进入一次奖励时间，敌人生成速度就会加快一次
        </p>
    <p>隐藏作弊模式：以 Python 为武器，横扫一切敌人和友军！</p>
  </div>

</body>
</html>
```

main.py 基于 fastapi，负责 js 和 python 通信：

```
from fastapi import FastAPI
from fastapi.middleware.cors import CORSMiddleware
headers = {
    'Cookie': '_ga=GA1.2.387593542.1617107630; _gid=GA1.2.1534665002.161
        7107630; Hm_lvt_cdb524f42f0ce19b169a8071123a4797=1617107630
```

```
            ,1617109657; Hm_lpvt_cdb524f42f0ce19b169a8071123a4797=16171
            10185; kw_token=QUE6LY91RKT',
    'csrf': 'QUE6LY91RKT',
    'Host': 'www.kuwo.cn',
    'Referer': 'http://www.kuwo.cn/search/list?key=fuck',
    'User-
            Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/5
            37.36 (KHTML, like Gecko) Chrome/102.0.0.0 Safari/537.36',
}
app = FastAPI()

class status:
    data = 1
    changed = True

app.add_middleware(
    CORSMiddleware,
    allow_origins=["*"],
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"],
)

@app.get("/")
async def root():
    return {"message": "Hello World. Welcome to FastAPI!"}

@app.get("/status")
async def music(id: int):
    status.data = id
    status.changed = True
    return status.data

@app.get("/data")
async def data():
    if status.changed:
        status.changed = False
        return status.data
    else:
        return 0
```

server.py 模拟键盘操作以及负责串口通信：

```
import serial
```

```
import serial.tools.list_ports as list_ports
from pykeyboard import PyKeyboard
from time import sleep
import requests

key = PyKeyboard()

PID_MICROBIT = 516
VID_MICROBIT = 3368
TIMEOUT = 0.1

def find_comport(pid, vid, baud):
    ser_port = serial.Serial(timeout=TIMEOUT)
    ser_port.baudrate = baud
    ports = list(list_ports.comports())
    print('scanning ports')
    for p in ports:
        print('port: {}'.format(p))
        try:
            print('pid: {} vid: {}'.format(p.pid, p.vid))
        except AttributeError:
            continue
        if (p.pid == pid) and (p.vid == vid):
            print('found target device pid: {} vid: {} port: {}'.format(
                p.pid, p.vid, p.device))
            ser_port.port = str(p.device)
            return ser_port
    return None

def process(line):
    if "left" in line:
        key.press_key(key.left_key)
        sleep(0.010)
        key.release_key(key.left_key)
    if "right" in line:
        key.press_key(key.right_key)
        sleep(0.010)
        key.release_key(key.right_key)
    if "up" in line:
        key.press_key(key.up_key)
        sleep(0.010)
        key.release_key(key.up_key)
    if "down" in line:
        key.press_key(key.down_key)
```

```
        sleep(0.010)
        key.release_key(key.down_key)
    if "enter" in line:
        key.tap_key(key.enter_key)

def main():
    print('looking for microbit')
    ser_micro = find_comport(PID_MICROBIT, VID_MICROBIT, 115200)
    if not ser_micro:
        print('microbit not found')
        return
    print('opening and monitoring microbit port')
    ser_micro.open()
    while True:
        line = ser_micro.readline().decode('utf-8')
        if line:  # If it isn't a blank line
            process(line)
        now = int(requests.get("http://127.0.0.1:8000/data").text)
        if now != 0:
            print(now)
            ser_micro.write((str(now)+'\n').encode('utf-8'))
    ser_micro.close()

main()
```

client.py micro bit 上跑的代码：

```
# Imports go at the top
from microbit import *
status = Image.DIAMOND
cnt = 1
# Code in a 'while True:' loop repeats forever
while True:
    n = uart.readline()
    if n:
        if n.decode()=="1\n":
            status = Image.SMILE
        elif n.decode()=="2\n":
            status = Image.HEART
        elif n.decode()=="3\n":
            status = Image.SAD
            display.show(status)
            sleep(1000)
        elif n.decode()=="4\n":
            status = Image.ANGRY
```

```
        cnt = 1
    cnt+=1
    if cnt % 150 ==0:
        status = Image.DIAMOND
        cnt = 1
    display.show(status)
    if accelerometer.is_gesture('up'):
        display.show(Image.ARROW_S)
        print("down")
    if accelerometer.is_gesture('down'):
        print("up")
        display.show(Image.ARROW_N)
    if button_a.is_pressed():
        display.show(Image.ARROW_W)
        print("left")
    if button_b.is_pressed():
        display.show(Image.ARROW_E)
        print("right")
    if pin_logo.is_touched() and accelerometer.was_gesture('shake'):
        display.show(Image.HAPPY,delay=1500)
        print("enter")
    sleep(20)
```