**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）

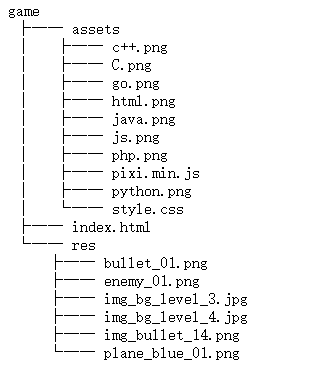
二．实现方案

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：



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



1. 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

1. <!DOCTYPE html>
2. <html lang="en">
3. <head>
4. <meta charset="UTF-8">
5. <meta http-equiv="X-UA-Compatible" content="IE=edge">
6. <meta name="viewport" content="width=device-width, initial-scale=1.0">
7. <title>Document</title>
8. <script src="./assets/pixi.min.js"></script>
9. <link rel="stylesheet" href="./assets/style.css">
10. </head>
11. <body>
12. <script>
13. *//创建应用*
14. var app = new PIXI.Application(512, 768);
15. document.body.appendChild(app.view);
16. app.view.style.height = "100%";
17. *//添加背景*
18. var bg = new PIXI.Sprite.fromImage("res/img\_bg\_level\_4.jpg");
19. app.stage.addChild(bg);
20. *//添加子弹*
21. var bulletImage = "res/bullet\_01.png";
22. var bullet = new PIXI.Sprite.fromImage(bulletImage);
23. bullet.anchor.x = 0.5;   *//设置飞机图片锚点为图片中心*
24. bullet.anchor.y = 0.5;
25. app.stage.addChild(bullet);
26. var bullets = [bullet]; *//子弹数组，用于存放子弹*
27. *//创建飞机图片*
28. var plane = new PIXI.Sprite.fromImage("res/plane\_blue\_01.png");
29. app.stage.addChild(plane);
30. plane.anchor.x = 0.5;
31. plane.anchor.y = 0.5;
32. plane.x = 200;
33. plane.y = 600;
34. plane.vx = 0;
35. plane.vy = 0;
36. *//添加敌机，从一系列不同的logo中选择，第一个固定为C++*
37. var picOfEnemy = ["assets/c++.png", "assets/java.png", "assets/python.png","assets/C.png","assets/go.png","assets/js.png","assets/php.png","assets/html.png"];
38. var enemy = new PIXI.Sprite.fromImage("assets/c++.png");
39. enemy.scale.set(0.8, 0.8);
40. app.stage.addChild(enemy);
41. enemy.anchor.x = 0.5;   *//设置飞机图片锚点为图片中心*
42. enemy.anchor.y = 0.5;
43. enemy.x = 200;
44. var enemies = [enemy]; *//存放敌机的数组*
45. *//创建得分显示文本*
46. var scoreTxt = new PIXI.Text("功德 0", { fill: "white" });
47. app.stage.addChild(scoreTxt); *//将文本添加到舞台*
48. var score = 0; *//得分变量，记录得分使用*
49. var isGameOver = false; *//是否游戏结束*
50. app.ticker.add(delta => animate(delta)); *//指定帧频函数*
51. var timer = 0; *//计时器，用于计算子弹发射间隔*
52. var interval = 50; *//子弹发射间隔*
53. var einterval = 100; *//敌机出现间隔*
54. var isUp = false; *//是否进入奖励时间*
55. var cheat = false; *//是否开启隐藏模式*
56. var crashSize = 40;
57. var times = 0; *//进入奖励时间的次数，用于动态调整难度*
58. let left = keyboard("ArrowLeft"),
59. right = keyboard("ArrowRight");
60. up = keyboard("ArrowUp");
61. down = keyboard("ArrowDown");
62. enter = keyboard("Enter");
63. *//监听键盘事件，上下左右与回车键*
64. *//Left arrow key `press` method*
65. left.press = () => {
66. plane.vx = -10;
67. plane.vy = 0;
68. };
69. *//Left arrow key `release` method*
70. left.release = () => {
71. if (!right.isDown && plane.vy === 0) {
72. plane.vx = 0;
73. }
74. };
75. *//Right*
76. right.press = () => {
77. plane.vx = 10;
78. plane.vy = 0;
79. };
80. right.release = () => {
81. if (!left.isDown && plane.vy === 0) {
82. plane.vx = 0;
83. }
84. };
85. up.press = () => {
86. plane.vy = -5;
87. plane.vx = 0;
88. };
89. up.release = () => {
90. if (!down.isDown && plane.vx === 0) {
91. plane.vy = 0;
92. }
93. };
94. down.press = () => {
95. plane.vy = 5;
96. plane.vx = 0;
97. };
98. down.release = () => {
99. if (!up.isDown && plane.vx === 0) {
100. plane.vy = 0;
101. }
102. };
103. *// enter presses记录按键次数，实现两种状态间的切换*
104. var presses = 0;
105. enter.press = () => {
106. presses++;
107. if (presses % 2 == 1) {
108. *//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*
109. interval = 15;
110. einterval = 50;
111. bulletImage = "/assets/python.png";
112. cheat = true;
113. } else {
114. interval = 50;
115. einterval = 100-10\*times;
116. bulletImage = "res/bullet\_01.png";
117. cheat = false;
118. }
119. }
120. //*主游戏动画函数*
121. function animate(delta) {
122. if (isGameOver == true) { *//如果游戏结束，则不执行下面动画*
123. return;
124. }
125. *// 如果timer满interval，发射子弹*
126. timer++;
127. if (timer % interval==0) {
128. *// 新建子弹，存入子弹数组*
129. var newBullet = new PIXI.Sprite.fromImage(bulletImage);
130. newBullet.anchor.x = 0.5;   *//设置飞机图片锚点为图片中心*
131. newBullet.anchor.y = 0.5;
132. newBullet.x = plane.x;
133. newBullet.y = plane.y;
134. app.stage.addChild(newBullet);
135. bullets.push(newBullet);
136. }
137. if (timer % einterval==0) {
138. *// 新建敌机，存入敌机数组*
139. var pic = picOfEnemy[Math.floor(Math.random() \* picOfEnemy.length)];
140. var newEnemy = new PIXI.Sprite.fromImage(pic);
141. newEnemy.anchor.x = 0.5;   *//设置飞机图片锚点为图片中心*
142. newEnemy.anchor.y = 0.5;
143. newEnemy.x = Math.random() \* 450 + 30; *//敌机水平位置随机*
144. newEnemy.y = -100;
145. newEnemy.scale.set(0.8, 0.8);
146. *// 延迟随机秒后出现*
147. setTimeout(function () {
148. app.stage.addChild(newEnemy);
149. enemies.push(newEnemy);
150. }, Math.random() \* 200);
151. }
152. plane.x += plane.vx; *// 移动自己*
153. plane.y += plane.vy;
154. *// 如果越界，复位*
155. if (plane.x < 0) {
156. plane.x = 0;
157. }
158. if (plane.x > 500) {
159. plane.x = 500;
160. }
161. if (plane.y < 0) {
162. plane.y = 0;
163. }
164. if (plane.y > 700) {
165. plane.y = 700;
166. }
167. *//背景移动*
168. bg.y += 2;
169. if (bg.y >= 0) {
170. bg.y = -768;
171. }
172. *// 敌机移动*
173. for (var i = 0; i < enemies.length; i++) {
174. var enemy = enemies[i];
175. enemy.y += 5;
176. if (enemy.y > 800) {
177. if (enemy.texture.baseTexture.imageUrl=="assets/python.png") {
178. *// Python可以正常通过*
179. app.stage.removeChild(enemy);
180. enemies.splice(i, 1);
181. continue;
182. }
183. *// 其他情况判定防御失败*
184. isGameOver = true;
185. if (confirm("敌军已突破防线，是否重开？") == true) {
186. window.location.reload();
187. }
188. continue;
189. }
190. *//玩家飞机与敌机碰撞*
191. if (hitTestRectangle(plane, enemy)) {
192. *//游戏结束标记*
193. isGameOver = true;
194. *//是否重玩*
195. if (confirm("您已坠机！") == true) {
196. window.location.reload();
197. }
198. }
199. }
200. *// 遍历bullets中的bullet*
201. for (var j = 0; j < bullets.length; j++) {
202. var bullet = bullets[j];
203. bullet.y -= 10;
204. if (bullet.y < 10) {
205. app.stage.removeChild(bullet);
206. bullets.splice(j, 1);
207. continue;
208. }
209. *// 遍历enemies中的enemy*
210. for (var i = 0; i < enemies.length; i++) {
211. var enemy = enemies[i];
212. *//子弹与敌机碰撞*
213. if(hitTestRectangle(bullet, enemy)){
214. *// 将敌机移出数组*
215. app.stage.removeChild(enemy);
216. enemies.splice(i, 1);
217. *// 将子弹移出数组*
218. app.stage.removeChild(bullet);
219. bullets.splice(j, 1);
220. *// 如果敌机图片是python*
221. if (enemy.texture.baseTexture.imageUrl=="assets/python.png") {
222. score-=2;
223. scoreTxt.text = "功德 " + score + " 误伤友军！";
224. scoreTxt.style.fill = 0xff0000;
225. setTimeout(function() {
226. scoreTxt.style.fill = 0xffffff;
227. }, 1000);
228. continue;
229. }
230. *//得分+1*
231. score++;
232. if(isUp){
233. scoreTxt.text = "功德 " + score + " 奖励时间！";
234. }
235. else{
236. scoreTxt.text = "功德 " + score;
237. }
238. if(score % 20==0 && score!=0 && !cheat) {
239. *// 奖励模式参数调整，武器更换、减少子弹发射间隔*
240. isUp = true;
241. scoreTxt.text = "功德 " + score + " 奖励时间！";
242. times++;
243. interval = 30;
244. einterval = 50;
245. bulletImage = "res/img\_bullet\_14.png";
246. crashSize = 80;
247. *// 奖励时间持续5s后复原*
248. setTimeout(function() {
249. isUp = false;
250. interval = 50;
251. einterval = 100-times\*10; *// 动态难度，提升敌机出现速度*
252. bulletImage = "res/bullet\_01.png";
253. crashSize = 40;
254. }, 5000);
255. }
256. }
257. }
258. }
259. }
260. *// copied from tutorial，定义keyboard类，监听键盘事件*
261. function keyboard(value) {
262. let key = {};
263. key.value = value;
264. key.isDown = false;
265. key.isUp = true;
266. key.press = undefined;
267. key.release = undefined;
268. *//The `downHandler`*
269. key.downHandler = event => {
270. if (event.key === key.value) {
271. if (key.isUp && key.press) key.press();
272. key.isDown = true;
273. key.isUp = false;
274. event.preventDefault();
275. }
276. };
277. *//The `upHandler`*
278. key.upHandler = event => {
279. if (event.key === key.value) {
280. if (key.isDown && key.release) key.release();
281. key.isDown = false;
282. key.isUp = true;
283. event.preventDefault();
284. }
285. };
286. *//Attach event listeners*
287. const downListener = key.downHandler.bind(key);
288. const upListener = key.upHandler.bind(key);
290. window.addEventListener(
291. "keydown", downListener, false
292. );
293. window.addEventListener(
294. "keyup", upListener, false
295. );
297. *// Detach event listeners*
298. key.unsubscribe = () => {
299. window.removeEventListener("keydown", downListener);
300. window.removeEventListener("keyup", upListener);
301. };
303. return key;
304. }
305. *// 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*
306. function hitTestRectangle(r1, r2) {
307. difficulty = 1.12;
308. *//Define the variables we'll need to calculate*
309. let hit, combinedHalfWidths, combinedHalfHeights, vx, vy;
310. *//hit will determine whether there's a collision*
311. hit = false;
312. *//Find the center points of each sprite*
313. r1.centerX = r1.x + r1.width / 2;
314. r1.centerY = r1.y + r1.height / 2;
315. r2.centerX = r2.x + r2.width / 2;
316. r2.centerY = r2.y + r2.height / 2;
317. *//Find the half-widths and half-heights of each sprite*
318. r1.halfWidth = r1.width / 2;
319. r1.halfHeight = r1.height / 2;
320. r2.halfWidth = r2.width / 2;
321. r2.halfHeight = r2.height / 2;
322. *//Calculate the distance vector between the sprites*
323. vx = r1.centerX - r2.centerX;
324. vy = r1.centerY - r2.centerY;
325. *//Figure out the combined half-widths and half-heights*
326. combinedHalfWidths = (r1.halfWidth + r2.halfWidth) / difficulty;
327. combinedHalfHeights = (r1.halfHeight + r2.halfHeight) / difficulty;
328. *//Check for a collision on the x axis*
329. if (Math.abs(vx) < combinedHalfWidths) {
330. *//A collision might be occurring. Check for a collision on the y axis*
331. if (Math.abs(vy) < combinedHalfHeights) {
332. *//There's definitely a collision happening*
333. hit = true;
334. } else {
335. *//There's no collision on the y axis*
336. hit = false;
337. }
338. } else {
339. *//There's no collision on the x axis*
340. hit = false;
341. }
342. *//`hit` will be either `true` or `false`*
343. return hit;
344. };
345. </script>
346. <div class="rule">
347. <h1>规则</h1>
348. <p>1. 消灭所有敌人，如果被敌人通过防线，即判定为负</p>
349. <p>2. 如果与敌人或者友军发生碰撞，判定为坠机</p>
350. <p>（注：唯一的友军就是Python）</p>
351. <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>
352. <p>4. 友军可以正常通过防线，不会触发判定</p>
353. <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>
354. <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>
355. <p>隐藏作弊模式：以Python为武器，横扫一切敌人和友军！</p>
356. </div>
357. </body>
358. </html>

**main.py** 基于fastapi，负责js和python通信：

1. from fastapi import FastAPI
2. from fastapi.middleware.cors import CORSMiddleware
3. headers = {
4. 'Cookie': '\_ga=GA1.2.387593542.1617107630; \_gid=GA1.2.1534665002.1617107630; Hm\_lvt\_cdb524f42f0ce19b169a8071123a4797=1617107630,1617109657; Hm\_lpvt\_cdb524f42f0ce19b169a8071123a4797=1617110185; kw\_token=QUE6LY91RKT',
5. 'csrf': 'QUE6LY91RKT',
6. 'Host': 'www.kuwo.cn',
7. 'Referer': 'http://www.kuwo.cn/search/list?key=fuck',
8. 'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/102.0.0.0 Safari/537.36',
9. }
10. app = FastAPI()
11. class status:
12. data = 1
13. changed = True
14. app.add\_middleware(
15. CORSMiddleware,
16. allow\_origins=["\*"],
17. allow\_credentials=True,
18. allow\_methods=["\*"],
19. allow\_headers=["\*"],
20. )
21. @app.get("/")
22. async def root():
23. return {"message": "Hello World. Welcome to FastAPI!"}
24. @app.get("/status")
25. async def music(id: int):
26. status.data = id
27. status.changed = True
28. return status.data
29. @app.get("/data")
30. async def data():
31. if status.changed:
32. status.changed = False
33. return status.data
34. else:
35. return 0

**server.py** 模拟键盘操作以及负责串口通信：

1. import serial
2. import serial.tools.list\_ports as list\_ports
3. from pykeyboard import PyKeyboard
4. from time import sleep
5. import requests
6. key = PyKeyboard()
7. PID\_MICROBIT = 516
8. VID\_MICROBIT = 3368
9. TIMEOUT = 0.1
10. def find\_comport(pid, vid, baud):
11. ser\_port = serial.Serial(timeout=TIMEOUT)
12. ser\_port.baudrate = baud
13. ports = list(list\_ports.comports())
14. print('scanning ports')
15. for p in ports:
16. print('port: {}'.format(p))
17. try:
18. print('pid: {} vid: {}'.format(p.pid, p.vid))
19. except AttributeError:
20. continue
21. if (p.pid == pid) and (p.vid == vid):
22. print('found target device pid: {} vid: {} port: {}'.format(
23. p.pid, p.vid, p.device))
24. ser\_port.port = str(p.device)
25. return ser\_port
26. return None
27. def process(line):
28. if "left" in line:
29. key.press\_key(key.left\_key)
30. sleep(0.010)
31. key.release\_key(key.left\_key)
32. if "right" in line:
33. key.press\_key(key.right\_key)
34. sleep(0.010)
35. key.release\_key(key.right\_key)
36. if "up" in line:
37. key.press\_key(key.up\_key)
38. sleep(0.010)
39. key.release\_key(key.up\_key)
40. if "down" in line:
41. key.press\_key(key.down\_key)
42. sleep(0.010)
43. key.release\_key(key.down\_key)
44. if "enter" in line:
45. key.tap\_key(key.enter\_key)
46. def main():
47. print('looking for microbit')
48. ser\_micro = find\_comport(PID\_MICROBIT, VID\_MICROBIT, 115200)
49. if not ser\_micro:
50. print('microbit not found')
51. return
52. print('opening and monitoring microbit port')
53. ser\_micro.open()
54. while True:
55. line = ser\_micro.readline().decode('utf-8')
56. if line:  *# If it isn't a blank line*
57. process(line)
58. now = int(requests.get("http://127.0.0.1:8000/data").text)
59. if now != 0:
60. print(now)
61. ser\_micro.write((str(now)+'\n').encode('utf-8'))
62. ser\_micro.close()
63. main()

**client.py** micro bit上跑的代码：

1. *# Imports go at the top*
2. from microbit import \*
3. status = Image.DIAMOND
4. cnt = 1
5. *# Code in a 'while True:' loop repeats forever*
6. while True:
7. n = uart.readline()
8. if n:
9. if n.decode()=="1\n":
10. status = Image.SMILE
11. elif n.decode()=="2\n":
12. status = Image.HEART
13. elif n.decode()=="3\n":
14. status = Image.SAD
15. display.show(status)
16. sleep(1000)
17. elif n.decode()=="4\n":
18. status = Image.ANGRY
19. cnt = 1
20. cnt+=1
21. if cnt % 150 ==0:
22. status = Image.DIAMOND
23. cnt = 1
24. display.show(status)
25. if accelerometer.is\_gesture('up'):
26. display.show(Image.ARROW\_S)
27. print("down")
28. if accelerometer.is\_gesture('down'):
29. print("up")
30. display.show(Image.ARROW\_N)
31. if button\_a.is\_pressed():
32. display.show(Image.ARROW\_W)
33. print("left")
34. if button\_b.is\_pressed():
35. display.show(Image.ARROW\_E)
36. print("right")
37. if pin\_logo.is\_touched() and accelerometer.was\_gesture('shake'):
38. display.show(Image.HAPPY,delay=1500)
39. print("enter")
40. sleep(20)