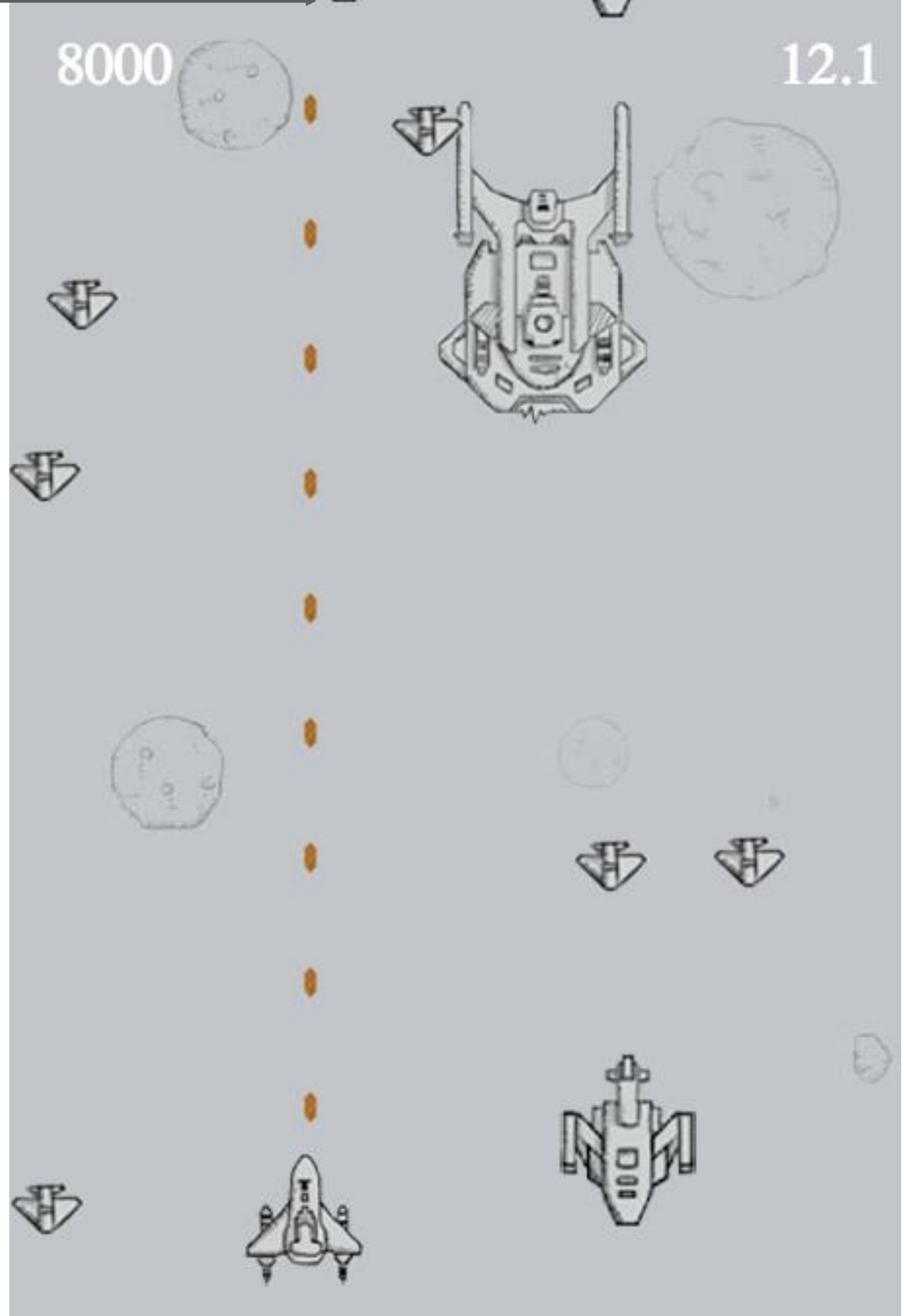
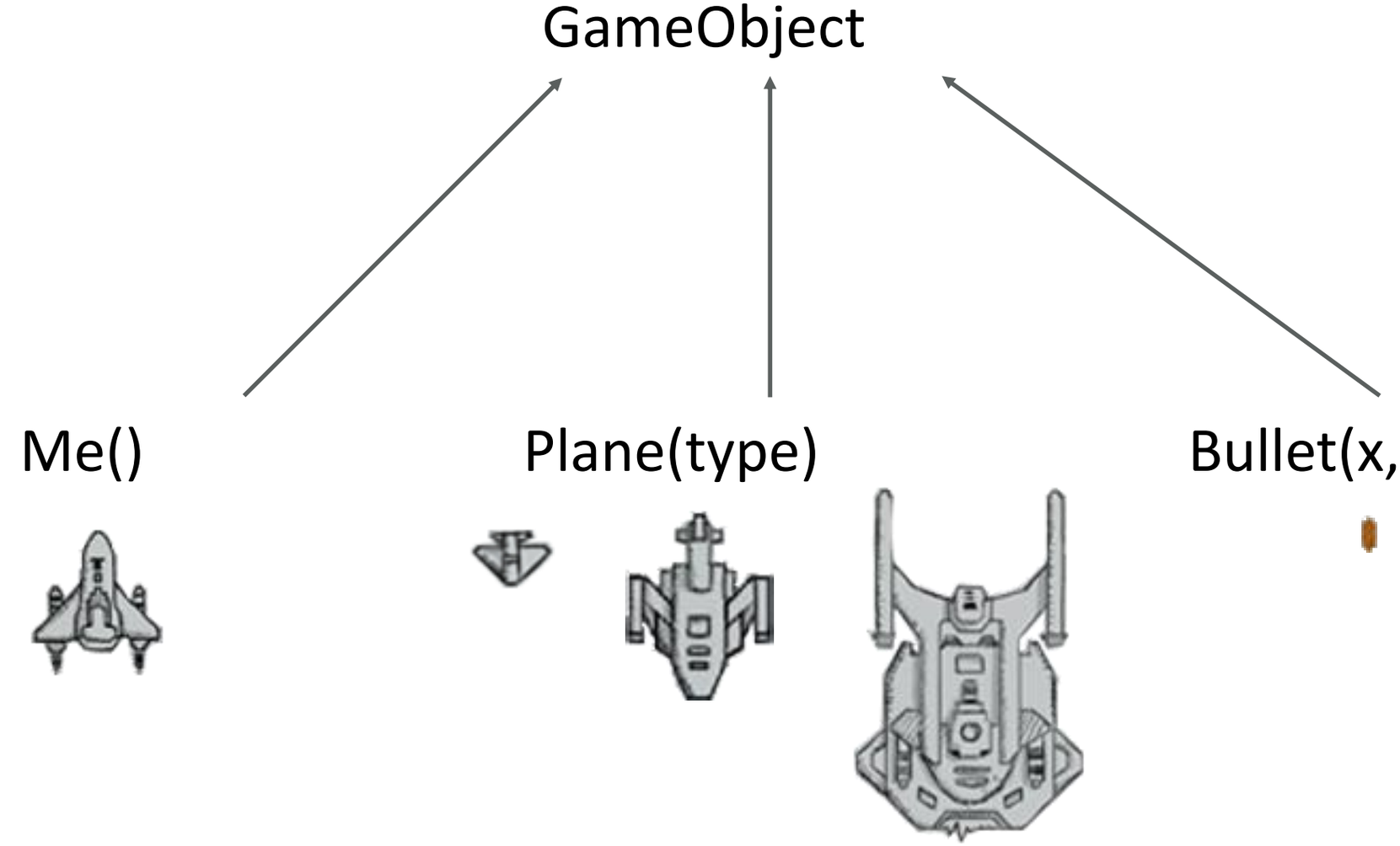

8000
12.1

BackGround()

GameTime()

Score()



整个游戏共有这些类，GameObject是一个游戏中物体的抽象类，
一些类继承自它。

init();

stop = setInterval("run()",1000/100);

每秒调用100次run()函数

run()

各物体运行到下一帧

backGround.run()
me.run()
planes[i].run()
bullets[i].run()
score.run()
time.run()

各物体重绘

backGround.draw()
me.draw()
planes[i].draw()
bullets[i].draw()
score.draw()
time.draw()

gameover()

clearInterval(stop);

停止每秒调用

每种物体是一个类，有点类似于 C 语言里的 struct。
每种物体具有 move, draw, run 等不同的方法。
run() 函数里会操纵该物体的 x,y 坐标值。并调用该物体的 draw() 函数。
draw() 函数会根据 x,y 值在 canvas 上指定位置画出图像。

run() 函数举例

```
this.run = function() {this.y -= 2;}
```

draw() 函数
画图

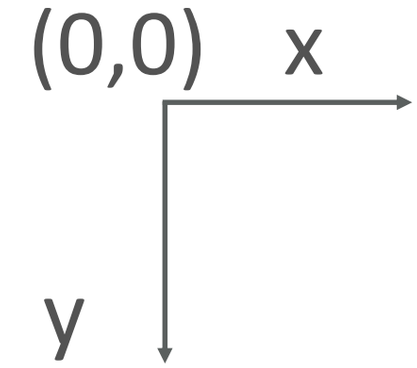
```
context.drawImage(this.img, this.x, this.y);
```

画圆

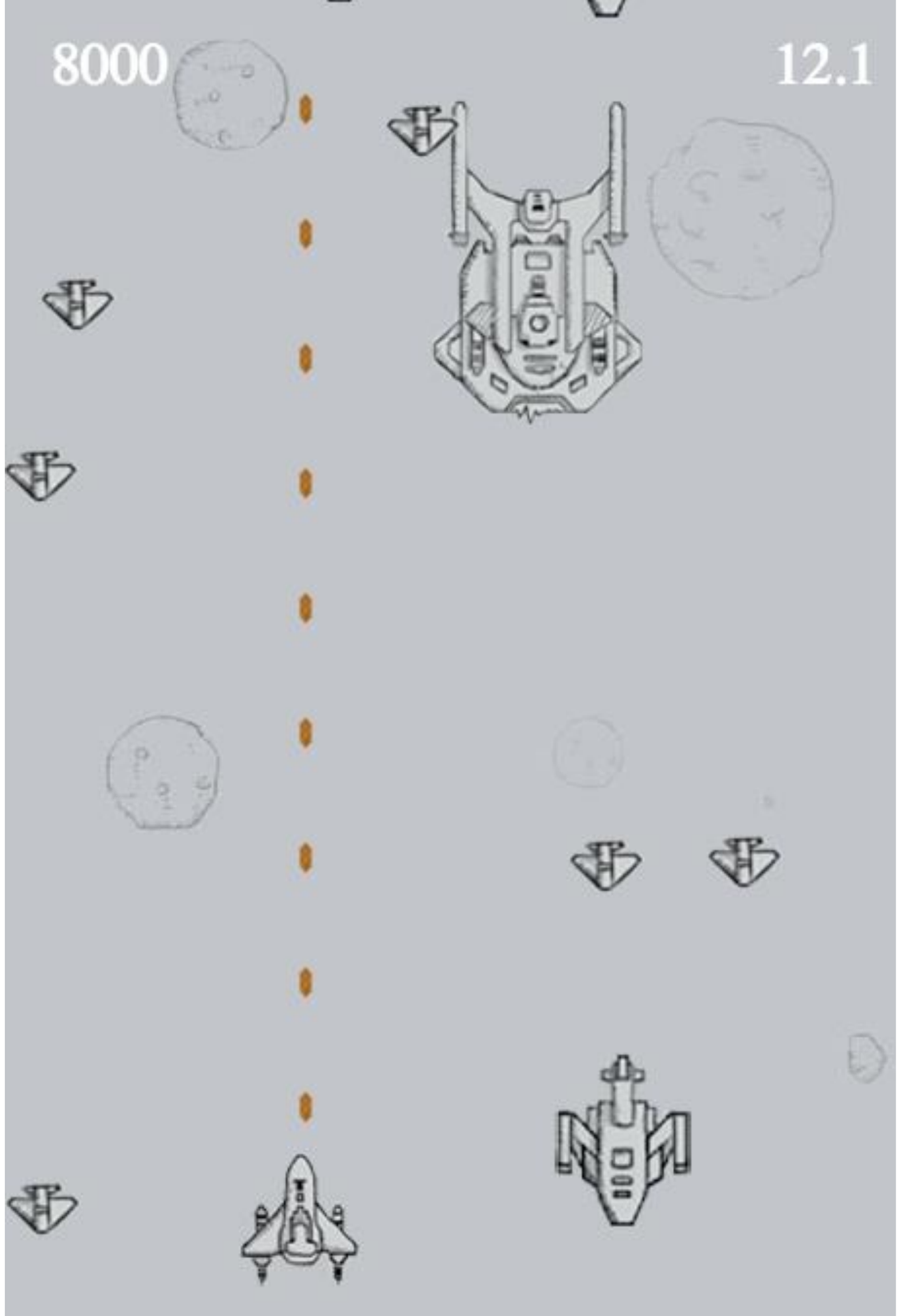
```
context.arc(this.x, this.y, 10, 0, Math.PI * 2, false);
```

画方

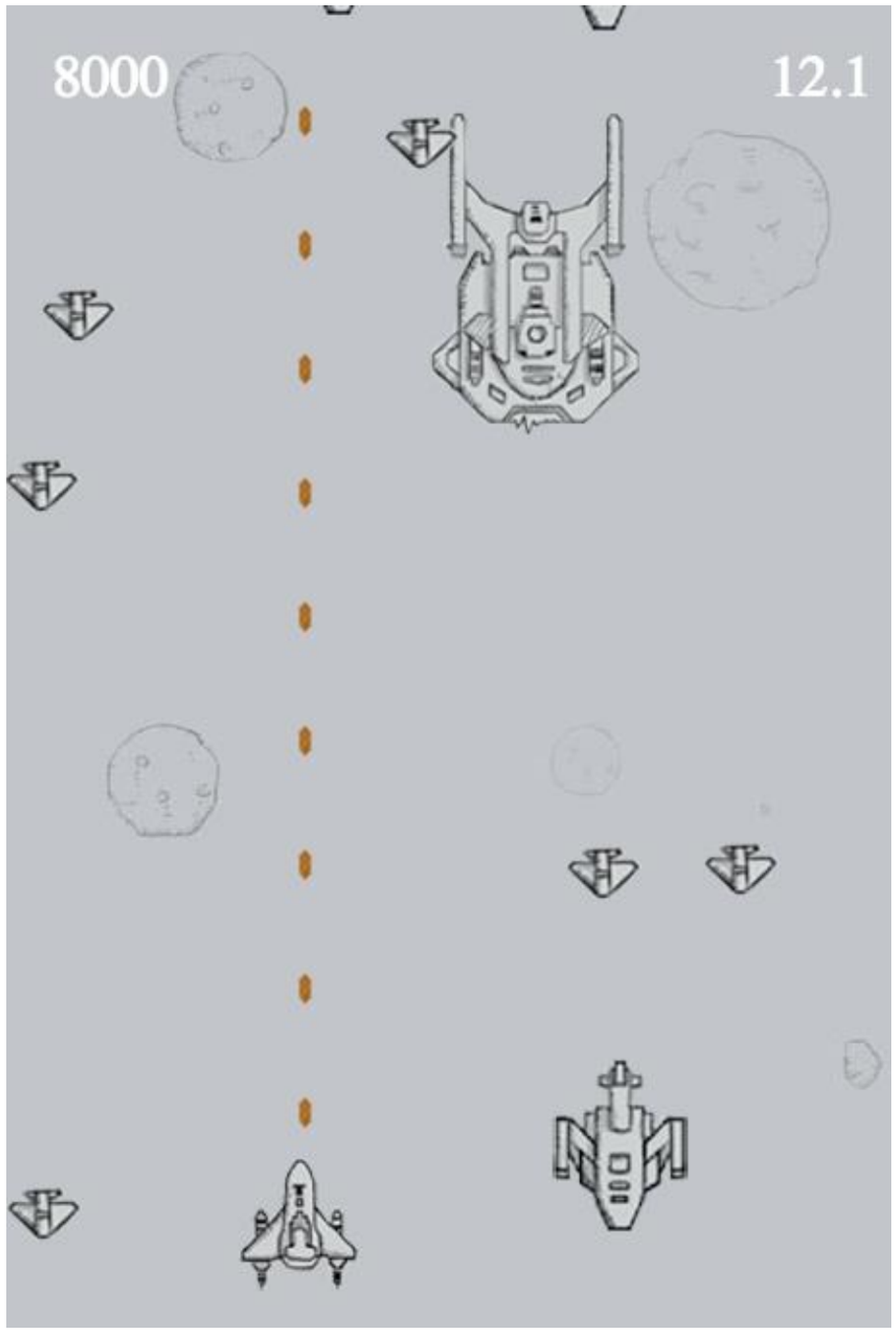
```
context.rect(this.x, this.y, 10, 10);
```



canvas 的 坐标体系



```
217     init();
218     function init () {
219       score = new Score();
220       time = new GameTime();
221       backGround = new BackGround();
222       me = new Me();
223       planes = new Array();
224       bullets = new Array();
225       bulletCreateCnt = 0;
226     }
```



变量初始化过程， 初始化分数、时间、背景、我的飞机、敌方飞机们、子弹们。
planes 和 bullets 是存有所有飞机的数组。

```
273     window.onload = function() {
274       mouse = captureMouse(canvas);
275       stop = setInterval("run()",1000/100);
276       me = new Me();
277     }
```

window.onload = function() {}
表示这一段程序是在整个窗口加载完成后执行的。

setInterval(“run()”,1000/100);
表示告诉电脑每隔10毫秒调用一次run()函数。

```
232     function run() {
233       var type = randInt(0, 3);
234       if (!randInt(0, (type + 1) * (type + 1)
                                        * 20 / time.factor) )
235         planes.push(new Plane(type));
236       if (bulletCreateCnt == 15) {
237         bullets.push(new Bullet(me.centerX(), me.y));
238         bulletCreateCnt = 0;
239       } else bulletCreateCnt++;
```

这里的主要逻辑是判断什么时候生成飞机和子弹。
run函数每帧调用一次，先随机一种敌方飞机类型type，
然后再随机一个数来判断该时刻是否要随机生成飞机。
子弹是有一个计数器 bulletCreateCnt 每15帧生成一颗子弹。

(续)

```
241       backGround.run();
242
243       for (var i = 0; i < bullets.length; i++) {
244         if (!bullets[i].run()) {
245           bullets.splice(i, 1);
246           i--;
247         }
248       }
249
```

接下来run()函数会分别调用各种东西的run()函数，
如果一个物体应该被删掉，他的run函数会返回 false
然后调用 Array.splice(i, 1), 删掉数组第i个元素，
删掉后第i+1个元素会变成第i个元素，
而第i+1个元素还没有run()过，因此应该 i—-

(续)

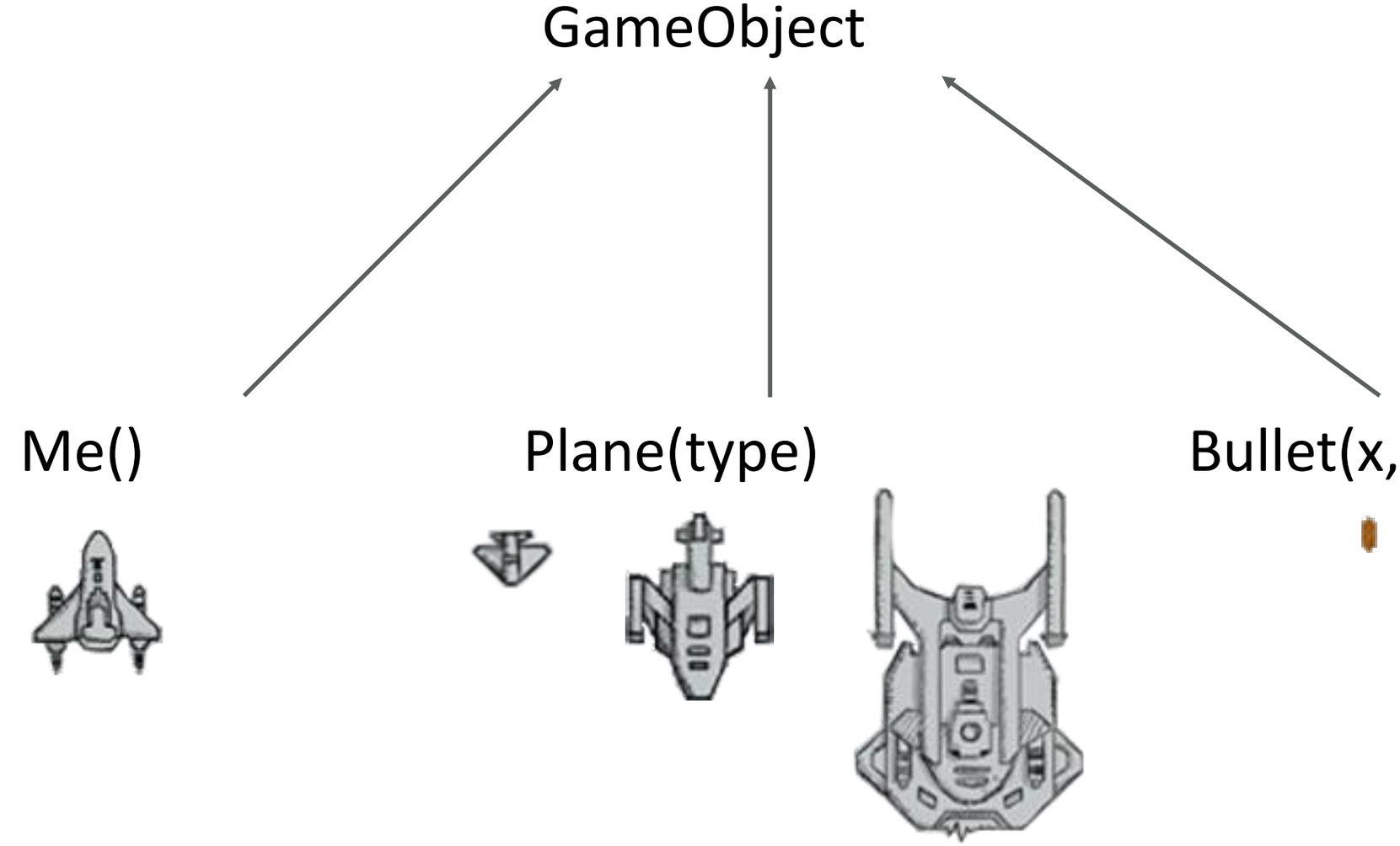
```
250       for (var i = 0; i < planes.length; i++) {
251         if (!planes[i].run()) {
252           planes.splice(i, 1);
253           i--;
254         }
255       }
256
257       if (!me.run()) {
258         gameover();
259       }
260       time.run();
261         score.run();
262       }
```

继续run()别的各种东西，注意这里run()的顺序是按照canvas覆盖关系从下往上的。
因为在每种东西的run()的函数里会将每种物体移动一小步，并且在canvas上画出它。
即每帧都会画一次。

BackGround()

GameTime()

Score()



整个游戏共有这些类，GameObject是一个游戏中物体的抽象类，
一些类继承自它。

```
149     Bullet.prototype = new GameObject();
150     function Bullet(x, y) {
151       this.x = x;
152       this.y = y;
153       this.img = imgBullet;
154       this.speed = 4;
155       this.run = function() {
156         this.move(0, -this.speed);
157         if (this.isOut()) return false;
158         this.draw();
159         return true;
160       }
161     }
```

```
237     bullets.push(new Bullet(me.centerX(), me.y));
```

子弹类的写法及如何生成一个新子弹，其中 this 指向这颗子弹本身。
this.move() this.isOut() this.draw() 均继承自GameObject。
但是子弹本身有自己的run()函数。

```
28    var imgPlane = [ newImg("./image/plane0.png"),
29                     newImg("./image/plane1.png"),
30                     newImg("./image/plane2.png") ];
31    var imgMe = newImg("./image/me.png");
32    var imgBullet = newImg("./image/bullet.png");
33    var imgOver = newImg("./image/over.png");
34    var imgBg = [newImg("./image/bg1.jpg"),
                                        newImg("./image/bg2.jpg")];
35
36    function newImg(src){
37     var obj = new Image();
38      obj.src = src;
39      return obj;
40    }
```

所有图片变量的初始化过程。
方括号可用来直接初始化数组。

```
68    function GameObject() {
69
70      this.move = function (dx, dy) {
71        this.x += dx;
72        this.y += dy;
73      }
74
75      this.draw = function() {
76        context.drawImage(this.img, this.x, this.y);
77      }
78
79      this.isOut = function(onlydown) {
80        return (this.y > height || (!onlydown && this.y < 0));
81      }
```

GameObject 里定义了 飞机，敌方飞机，子弹所共有的一些方法。
移动，画到画布上，判断是否出界。

(续)

```
83      this.centerX = function() {
84        return this.img.width / 2 + this.x;
85      }
86
87      this.collide = function(another) {
88        return !(this.x + this.img.width < another.x
                                          || this.x > another.x + another.img.width
89              || this.y + this.img.height < another.y
                                          || this.y > another.y + another.img.height);
90      }
91
92    }
```

GameObject 里定义了 飞机，敌方飞机，子弹所共有的一些方法。
所有物体的定位坐标是左上角。
物体中部的 x 坐标， 以及是否与另一物体碰撞。

```
94    Me.prototype = new GameObject();
95    function Me() {
96      this.x = this.y = 0;
97      this.img = imgMe;
98      this.run = function() {
99        this.x = mouse.x;
100       this.y = mouse.y;
101       for (var i = 0; i < planes.length; i++) {
102         if (this.collide(planes[i])) {
103           this.img = imgOver;
104           this.draw();
105           return false;
106         }
107       }
108       this.draw();
109       return true;
110     }
111   }
```

玩家所控制的飞机的写法，与敌方飞机碰撞后就挂了。
挂了之后会把飞机的图片替换成 imgOver。
需注意的是其中 mouse 坐标的获取方式。

```
273     window.onload = function() {
274       mouse = captureMouse(canvas);
275       stop = setInterval("run()",1000/100);
276       me = new Me();
277     }
```

```
42    function captureMouse(element) {
43      var mouse={x:width/2, y:height/2};
44
45      element.addEventListener('mousemove',function(event){
46        var x,y;
47        if ( event.pageX || event.pageY ) {
48          x = event.pageX;
49          y = event.pageY;
```

addEventListener 添加对 mousemove 事件的监听。
当鼠标移动是更新mouse变量的x的值和y的值。

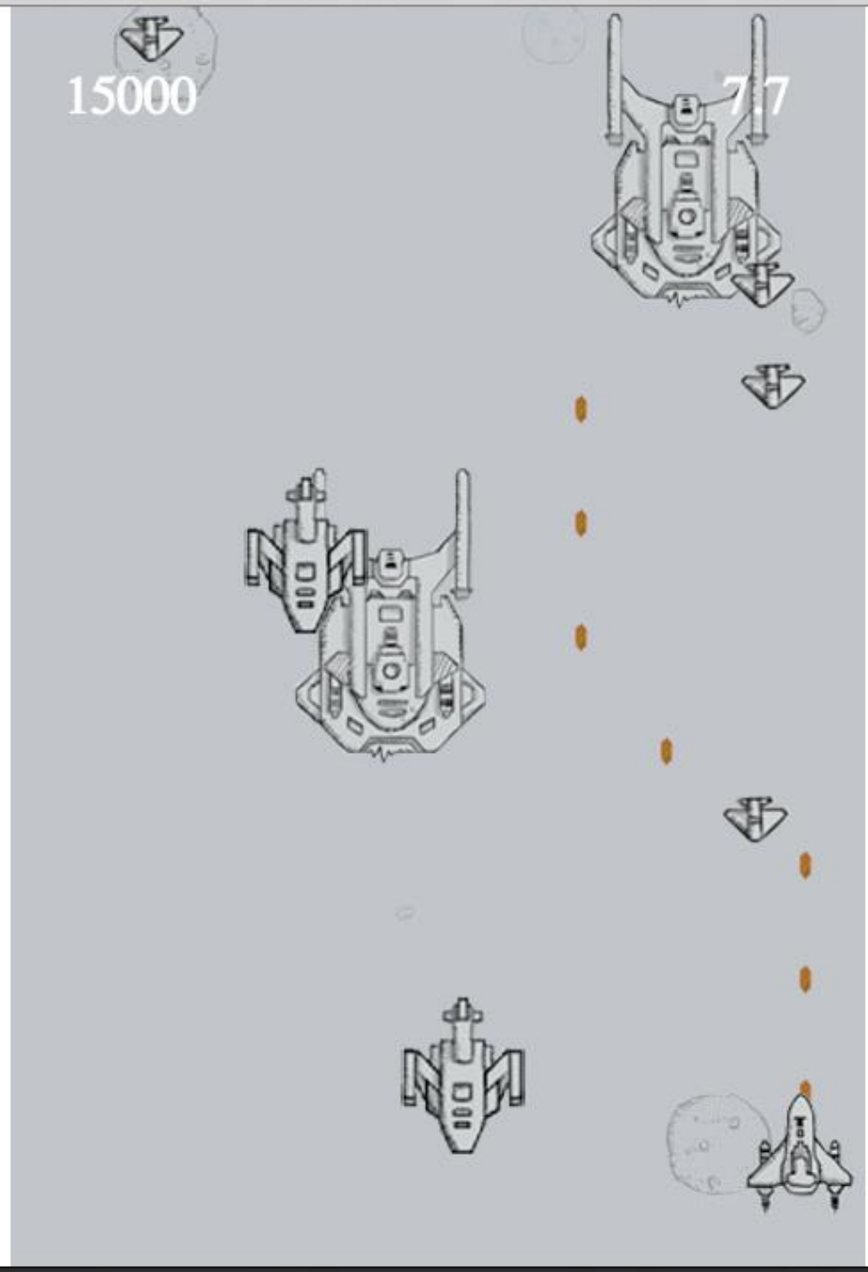
(续)

```
47        if ( event.pageX || event.pageY ) {
48          x = event.pageX;
49          y = event.pageY;
50        }

56
57        x -= element.offsetLeft;
58        y -= element.offsetTop;
60        mouse.x = x;
61          mouse.y = y;
62
63        }, false);
64
65        return mouse;
66      };
```

```
canvas.addEventListener( 'click', createBullet);

document.onkeypress = function(event) {
      if (event.keyCode == 37) {  //left
      me.move(-2, 0);
  } else if (event.keyCode == 38) { //up
        ……
  }
     ……
}
```

除了可以监听鼠标位置，还可以用这些方法监听鼠标事件，按键事件等。

x -= canvas.offsetLeft;
y -= canvas.offsetTop;



event.pageX,
event.pageY

```
113     Plane.prototype = new GameObject();
114     function Plane(type) {
115       this.img = imgPlane[type];
116       this.x = randInt(0, width - this.img.width);
117       this.y = - this.img.height;
118       if (type == 0) {
119         this.lives = 1;
120         this.speed = randInt(3 * time.factor, 4 *
                    time.factor);
121       } else if (type == 1) {
122         this.lives = 6;
123         this.speed = randInt(2 * time.factor, 3 *
            time.factor);
124       }
                        ……
```

生成飞机时在 canvas 的上面随机一个位置生成，
并根据飞机类型设置其速度和生命（能被子弹打几次）。

```
128       this.run = function() {
129         this.move(0, this.speed);
130         if (this.isOut(true)) return false;
131         for (var i = 0; i < bullets.length; i++) {
132           if (this.collide(bullets[i])) {
133             bullets.splice(i, 1);
134             i--;
135             this.lives--;
136           }
137           if (this.lives == 0) {
138             if (type == 0) score.add(1000);
139             if (type == 1) score.add(6000);
140             if (type == 2) score.add(20000);
141             return false;
142           }
143         }
144         this.draw();
145         return true;
146       }
```

每当和子弹碰撞，生命数减一，当子弹为 0 的时候，加分。

```
163     function BackGround() {
164       this.x = this.y = 0;
165       this.cur = 0;
166       this.speed = 1;
167       this.run = function() {
168         this.y += this.speed;
169         if (this.y > height) {
170           this.y = 0;
171           this.cur = 1 - this.cur;
172         }
173         context.drawImage(imgBg[this.cur], this.x,
                                                    this.y - height);
174         context.drawImage(imgBg[1 - this.cur],
                                                    this.x, this.y);
175       }
176     }
```

为了让背景有滚动效果，它是由两张图拼出来的。

```
178     function GameTime() {
179       this.startTime = (new Date()).getTime();
180       this.cur = 0;
181       this.ms = 0;
182       this.factor = 1;
183
184       this.run = function () {
185         var s = parseInt((new Date()).getTime()
                                                - this.startTime) / 100;
186         this.ms = parseInt(s % 10);
187         this.cur = parseInt(s / 10);
188         this.factor = 1 + this.cur * 0.01;
189         context.font="30px Times";
190         context.fillStyle = "#fff";
191         context.fillText(this.cur + "." + this.ms, 400, 60);
192       }
193     }
```

(new Date()).getTime() 是系统当前时间精确到毫秒的浮点数类型。
this.factor 是为了让游戏的飞机速度越来越快，生成飞机的间隔越来越小。
所以让他随着时间递增。

```
264     function gameover() {
265       clearInterval(stop);
266       var res = window.confirm( + score.num +
                                                    "\n是否重新开始？");
267       if (res) {
268         init();
269         stop = setInterval("run()",1000/100);
270       }
271     }
272
273     window.onload = function() {
274       mouse = captureMouse(canvas);
275       stop = setInterval("run()",1000/100);
276       me = new Me();
277     }
```

游戏结束时的控制逻辑， clearInterval用来停止每帧调用函数。

徐明昭

kilnyy@gmail.com