

Processing

互动编程

任远◎著



科学出版社

Processing互动编程

任 远 著

科 学 出 版 社

北 京

内 容 简 介

本书共分3部分：第1部分主要介绍 Processing 编程语言，为避免枯燥，使用了大量的与现实生活类比的方法，让你一层一层地深入，培养抽象和从整体到部分再到整体的思维方式；第2部分主要介绍图形绘制和图形动画，让你学会用程序描述的方法来绘制各种各样的图形，并且在图形的基础上，利用数学知识来实现程序动画，该部分会涉及坐标、颜色、文本、位图、3D 等概念；第3部分主要介绍在程序动画的基础上添加交互因素，构成完整的交互结构，这些交互因素包括鼠标、键盘、时间等。本书可以让读者学会互动编程的思维模式，同时激发个人的互动艺术理念。

本书适合电子专业、交互设计专业、新媒体技术专业学生阅读，也可供电子爱好者开展电子制作项目参考。

图书在版编目(CIP)数据

Processing 互动编程 / 任远著. — 北京：科学出版社，2014.10

ISBN 978-7-03-042136-4

I. P… II. ①任… III. ①程序设计 IV. ①TP311.1

中国版本图书馆CIP数据核字(2014)第234456号

责任编辑：杨 凯 / 责任制作：魏 谨

责任印制：赵德静 / 封面制作：杨安安

北京东方科龙图文有限公司 制作

<http://www.okbook.com.cn>

科学出版社出版

北京东黄城根北街16号

邮政编码：100717

<http://www.sciencep.com>

安泰印刷厂 印刷

科学出版社发行 各地新华书店经销

*

2015年1月第 一 版 开本：787×960 1/16

2015年1月第一次印刷 印张：18

印数：1—3 000 字数：372 000

定价：46.00元

(如有印装质量问题，我社负责调换)

前言

古往今来，哲学家在不断地认识这个世界，艺术家用他们的艺术方式表达他们对这个世界的看法，自然科学家也在不断地通过实验和观察，寻求和探索世界的规律。而数学是可以精确描述这些规律的一种语言，利用数学可以精确描述出物质（哲学上的物质概念）的运动、形态、结构、变化等。我热爱科学和艺术，一直在寻求他们的完美结合。在学习和研究过程中发现编程就是一个可以很完美地同时表达两者的一种方式。所以编程是互动艺术的一门必修课。

早期的艺术家通过绘画来表达他们的世界观和时空观。后来随着数字科技的发展，艺术家可以通过时间的延续来表达他们的艺术思想。而如今在人机交互的信息时代，非线性的数字影像和互动装置成为表达艺术的主要手段，我通过不断探索和实践总结出了大量关于互动编程的经验，希望更多的人来了解互动艺术，进入互动艺术的世界。我之所以选用 Processing 来给大家打开互动艺术的大门，是因为以下几点原因：

（1）国内互动艺术逐渐开始发展，各大院校也开始设立相应的课程。对于想从事这方面的设计师和艺术类的同学而言，Processing 是一个比较容易入手的互动编程软件，它已经在国外的许多院校被教授或使用。它的操作界面和 API（Application Programming Interface，应用程序编程接口）非常清晰、简约。但是它简约而不简单，通过它可以创造出非常出色的互动作品。

（2）人们每天都接收和创造着大量的信息和数据，电影《钢铁侠 3》中事故现场的数据重现、人脑的可视化分析、微博的人际关系拓扑图、科学研究中微观粒子的结构模拟，这些可视化的图形让人们从不同的角度了解了数据，得到了许多有用的信息。如今，数据可视化已经成为一门热门学科。Processing 提供了大量的图形绘制和数据处理函数，在数据可视化方面得到了广泛的应用。

（3）随着数字科技的发展和人们的审美提高，许多博物馆、世博会展馆、政府规划馆等也都采用了各种各样的多媒体互动形式来设计展项，由于 Processing 和硬件接口（Kinect、Arduino 等）的完美结合，再加上其高分辨率的渲染效率，使其在展览展示方面也取得了举足轻重的地位。

（4）HTML5 带来了网页革命性的改变，目前各大浏览器都支持它，Processing 也发布了 Javascript 版本，基于 HTML5 的 canvas 标签进行渲染，不需要在浏览器安装任何插件就可以运行。通过它可以做出很多网络互动作品，更多的网页设计师和开发人员都开始关注并学习它。

由于上面的种种原因，再加上国内关于 Processing 的中文资源比较缺乏，所以我写了这本《Processing 互动编程》。虽然本书主要是通过 Processing 教你互动编程，但一些互动的概念和编程技巧是完全可以应用到其他领域。在这里我需要和每一个读者强调一下，任何软件和程序语言都只是工具而已。所以我希望通过本书，更多的是让读者学会互动编程

的思维模式，同时激发个人的互动艺术理念。

本书共分 3 部分：

第 1 部分语言基础：主要介绍 Processing 编程语言，编程语言是人与计算机沟通的一种方式，所以你想要和它很好地沟通，请认真学习这部分内容，为后面的内容打好基础。在讲编程语法的时候避免枯燥，我用了大量的与现实生活类比的方法，让你一层一层地深入，培养抽象和从整体到部分再到整体的思维方式。

第 2 部分图形、动画：主要介绍的是有关图形绘制和图形动画，让你学会用程序描述的方法来绘制各种各样的图形，并且在图形的基础上，利用数学知识来实现程序动画。这一部分会涉及坐标、颜色、文本、位图、3D 等概念。

第 3 部分交互：主要介绍的是在程序动画的基础上添加交互的因素，构成完整的交互结构。这些交互因素包括鼠标、键盘、时间。

以上 3 部分内容都不是互相独立的，每一部分内容都会穿插前面学到的知识。整体的学习过程是一个迭代的过程，在语言的基础上绘制图形，在图形的基础上增加交互动画。

任 远

2014 年 6 月 6 日

目 录

第 1 部分 语言基础

第 1 章 Processing

1.1 Hello Processing.....	3
1.2 Hello world.....	4
1.3 第一个图形.....	5
1.4 第一个动画.....	5
1.5 第一个交互.....	6

第 2 章 变 量

2.1 声明变量.....	7
2.2 数据类型.....	7
2.3 变量名.....	8
2.4 变量赋值.....	8
2.5 常 量.....	11

第 3 章 运算符

3.1 基本算数运算符.....	12
3.2 赋值算数运算符.....	14
3.3 自增、自减运算符.....	14
3.4 关系运算符.....	16
3.5 逻辑运算符.....	17
3.6 条件运算符.....	18

第 4 章 条件语句

4.1 if 条件语句.....	19
4.2 if else 语句.....	19
4.3 else if 语句.....	20
4.4 swith 分支语句.....	22

第 5 章 循环语句

5.1 while 循环语句.....	25
---------------------	----

5.2 for 循环语句	25
5.3 循环嵌套	26
5.4 跳出循环	27
第 6 章 函 数	
6.1 定义函数	28
6.2 函数的参数	29
6.3 函数返回结果	31
6.4 函数重载	32
6.5 函数递归	32
第 7 章 类	
7.1 定义类	34
7.2 类的继承	36
7.3 覆盖父类方法	37
7.4 重写 toString() 方法	39
7.5 标签类	39
第 8 章 字符串	
8.1 数组基本概念	41
8.2 字符串的初始化	42
8.3 字符串的方法	43
8.4 字符串处理函数	45
8.5 实现字符串大小写转换	51
第 9 章 数 组	
9.1 二维数组	53
9.2 数组处理函数	55
9.3 实现连接两个数组	60
9.4 值数据类型与引用数据类型	62
第 10 章 列 表	
10.1 添加和删除列表元素	64
10.2 设置和返回列表元素	66
10.3 列表的排序	67
10.4 列表的运算	68

第 11 章 字典

11.1 字典的键和值	71
11.2 遍历键和值	73
11.3 排序键和值	74
11.4 字典的运算	75

第 2 部分 图形、动画

第 12 章 运行环境

12.1 坐标系统	79
12.2 程序主结构	79
12.3 帧速率	81
12.4 窗口大小	82

第 13 章 数学基础

13.1 数学计算	84
13.2 约束、映射	87
13.3 角度制和弧度制	90
13.4 三角函数	91
13.5 反正切函数	94

第 14 章 2D 基础图形

14.1 基础几何图形	97
14.2 描边属性	103
14.3 扇形、弓形、环形、扇环	106
14.4 虚线、网格	110

第 15 章 颜 色

15.1 色彩模式	114
15.2 创建和设置颜色	115
15.3 颜色分量	119
15.4 颜色渐变	123

第 16 章 变 换

16.1 基本变换	125
-----------------	-----

16.2 变换堆栈	129
16.3 使用变换绘制图形	131
16.4 变换动画	132

第 17 章 曲 线

17.1 Bezier 曲线	136
17.2 Bezier 曲线路径动画	139
17.3 Curve 曲线	142
17.4 Curve 多条连续路径动画	145

第 18 章 复杂图形

18.1 vertex 绘制	148
18.2 正多边形和星形	156
18.3 vertex 环形、扇环	159
18.4 PShape 图形对象	166

第 19 章 3D 图形

19.1 3D 坐标系统	174
19.2 三维灯光	179
19.3 三维材质	183
19.4 三维透视	186

第 20 章 随机、噪波、向量

20.1 随机函数	191
20.2 噪波函数	194
20.3 PVector 向量对象	195
20.4 粒子动画	201

第 21 章 位 图

21.1 加载位图	207
21.2 显示位图	209
21.3 PImage 位图对象	212
21.4 位图遮罩动画	217
21.5 滤 镜	219
21.6 混 合	223
21.7 贴 图	231

21.8 位图随机动画	233
第 22 章 文 本	
22.1 文本术语	235
22.2 文本显示	235
22.3 字体属性	239
22.4 创建和设置字体	242
22.5 环形文本	246
22.6 文本随机动画	247

第 3 部分 交 互

第 23 章 鼠标交互

23.1 鼠标事件	251
23.2 鼠标事件函数	251
23.3 鼠标状态变量	252
23.4 draw() 函数里实现所有鼠标事件	253
23.5 鼠标拖拽	257
23.6 鼠标光标	258
23.7 鼠标绘制	260

第 24 章 键盘交互

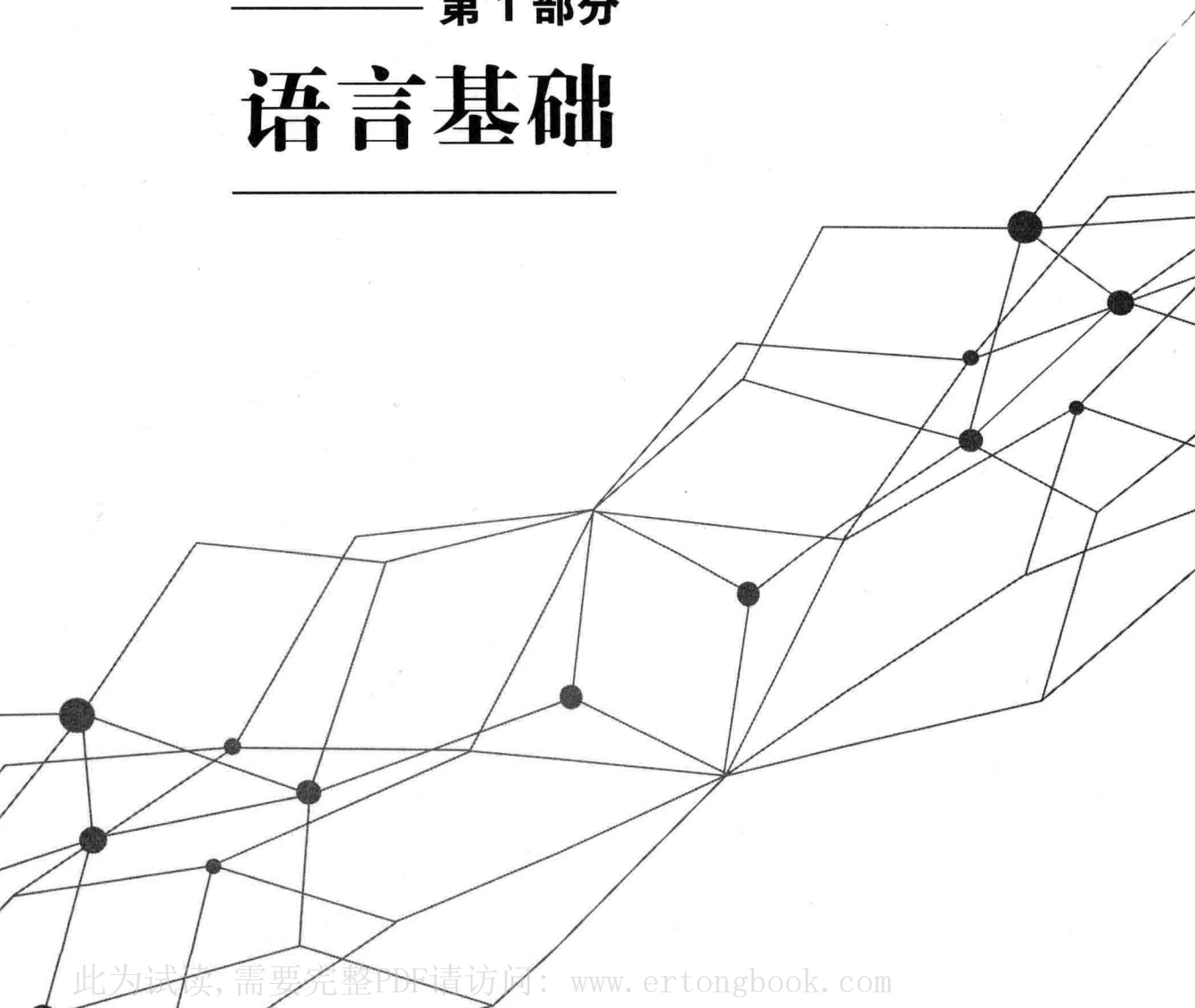
24.1 键盘事件	262
24.2 键盘控制	264
24.3 键盘交互	264
24.4 窗口焦点	265

第 25 章 日期、时间

25.1 日期、时间函数	268
25.2 设置程序使用期限	268
25.3 制作简单计时器	270
25.4 模拟时钟	271

—— 第 1 部分

语言基础



第 1 章 Processing

1.1 Hello Processing

Processing 开发环境 (Processing Development Environment), 简称 PDE, 可以从它的官方网站 (<https://www.processing.org/download/>) 下载安装。PDE 主要有运行窗口和编辑窗口两个窗口。运行窗口用来运行你的程序效果, 编辑窗口主要用来编辑代码和发布程序。编辑窗口包括工具栏、标签栏、代码编辑区、消息区、控制台。

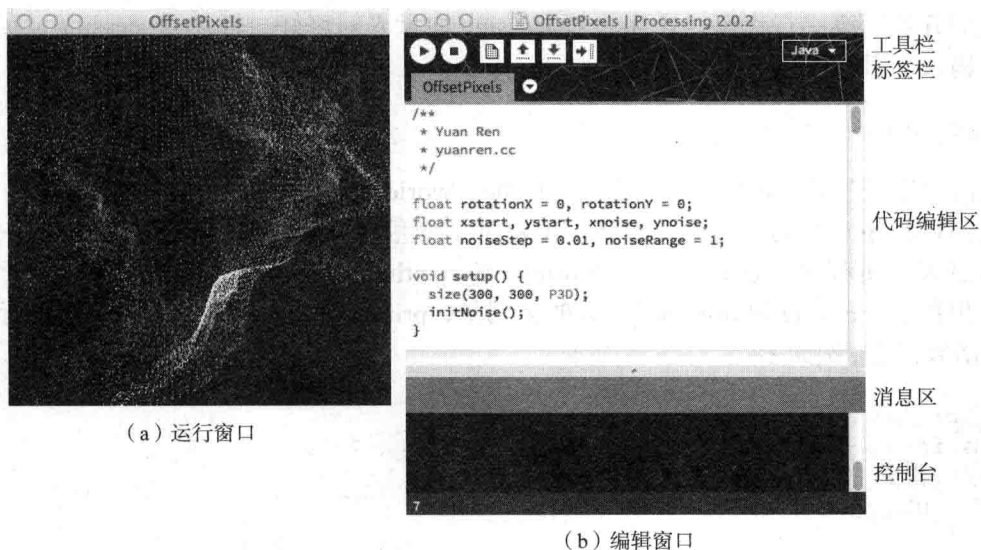


图 1.1 Processing Development Environment (PDE)

工具栏: 包括运行、停止运行、新建、打开、保存、发布等按钮。

☉ **运行:** 点击“运行”按钮可以运行你的程序。运行时会弹出“运行窗口”, 你可以通过“运行窗口”来观察程序的视觉效果和交互效果。

⊙ **停止运行:** 点击“停止运行”按钮可以关闭“运行窗口”。

▢ **新建:** 新建一个 Processing 工程文件, 会弹出一个新的未保存的代码编辑窗口。在开发过程中, 可以利用它来新建代码编辑窗口, 测试某一段程序代码单独运行的效果。

▣ **打开:** 打开一个指定路径的 Processing 工程文件, 还可以打开最近的工程或者是 Processing 事例代码。

▤ **保存:** 保存目前的工程文件。

▥ **开发模式:** 点击右上角“Java”按钮可以切换开发模式。Processing 支持很多种开发模式, 可以通过点击“Add Mode”来添加新的开发模式 (JavaScript、Android 等), 添加后需要重新运行 Processing 才能显示。

标签栏: 打开标签栏向下箭头的弹出菜单, 可以点击“New Tab”新建标签。如果代

码有很多行的话可以用它来扩展程序。不过一般我们会用它来定义一个单独的类，在后面的第7章我们会详细讲到。

代码编辑区：所有的程序代码都在这里编辑。会用不同颜色来区分数据类型、系统变量、系统常量、系统函数、语句等。

消息区：显示程序编译时的错误（语法错误），还有一些提示消息。

控制台：显示程序运行时的错误，还可以用 `print()` 或 `println()` 函数在这里输出信息。

1.2 Hello world

我们开始写第一行代码，让你的 Processing 向世界问声好。打开 Processing，在代码编辑区输入下面的代码：

```
println("Hello, world!");
```

点击“运行”后，在控制台会输出“Hello, world!”（不包含双引号）。

我们在程序开发的过程中，可能会追踪某个变量在某个时刻的值，或者输出一些有用的信息来分析程序。这时可以使用 `print()` 和 `println()` 函数，在程序运行的时候通过控制台输出信息。`print()` 和 `println()` 函数的区别是，`println()` 函数输出完信息后会换行，而 `print()` 函数不会，例如：

```
println("Hello,");  
println("world!");  
/* 输出：  
    Hello,  
    world!  
*/  
  
print("Hello,");  
print("world!");  
// 输出：Hello,world!
```

有时候我们在写代码时为了便于阅读和理解会给代码添加一些标注，但是我们知道在代码编辑器里写的任何非程序语法的文字，程序都会报错。在 Processing 里可以使用注释，注释是不会被当作执行代码的。用“//”来添加单行注释，用“/*”开始符号和“*/”结束符号来添加多行注释：

```
// 单行注释  
  
/*  
多行注释  
多行注释  
*/
```

1.3 第一个图形

以画面中心为原点画一个直径为 100 像素的圆，如图 1.2 所示。

```
ellipse(50,50,100,100); // 画一个椭圆，x、y 坐标都为 50，宽和高都为 100
```

Processing 提供了大量有关绘图的函数，这些函数就好比是各种各样的绘图工具。而且还可以添加图片，支持叠加和滤镜功能，如果你之前学习过图形编辑软件的话，比如 Adobe Photoshop，那么在学习 Processing 的时候，会发现有很多概念都似曾相识。没错！图形编辑软件其实就是把绘图操作可视化。而程序绘图是使用程序语言来描述绘图。所谓描述绘图就是，概括出图形的所有特征属性，用相应的属性值来描述它。比如画一个圆，圆的特征属性是半径和位置，那么我们只要告诉 Processing 这个圆的半径和位置，Processing 就会在屏幕帮我们画出这个圆。是不是很简单，就好像你在告诉一个使用绘图软件的人，让他帮你画出你想要的东西一样。只不过你和他沟通的时候，需要和他说他能听得懂的语言——Processing 程序语言。本书将教你如何使用 Processing 程序语言，让你学会和他沟通，绘制出各种各样的图形。

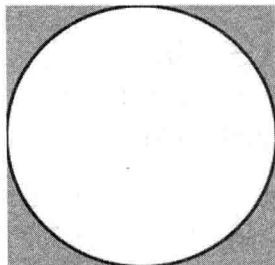


图 1.2

1.4 第一个动画

以画面中心为原点画一个圆，让圆的直径从 0 开始逐渐变大，如图 1.3 所示。

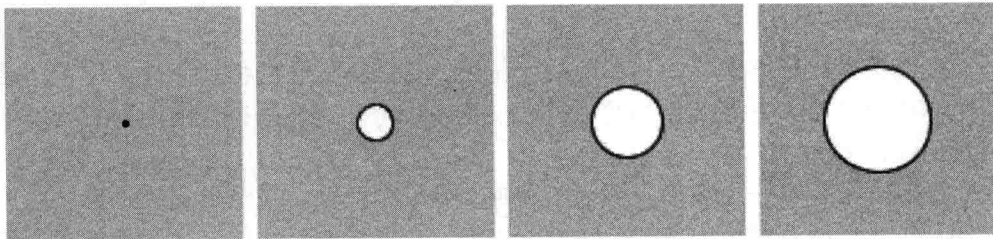


图 1.3

```
int diameter = 0; // 声明一个直径变量，初始值为 0

//draw() 函数会不断地循环执行
void draw() {
  // 画一个圆，直径为 diameter 变量
  ellipse(50, 50, diameter, diameter);
  diameter++; // 每次循环，让直径变量增加 1
}
```

上面的圆形可以随着时间逐渐变大，是因为把它的特征属性“直径”设置成了随着时间变化的量，这样就形成了动画。这种把特征属性设置为随着时间而变化的方法，是程序动画的基本实现机制。和我们传统的逐帧动画不同，传统逐帧动画需要在每一帧都重画，而程序动画只需要改变它的特征属性，计算机就会帮我们重绘，这也是为什么我们在绘图的时候用描述图形特征属性的方法来绘制。

1.5 第一个交互

不断以鼠标 x、y 坐标为原点画直径为 100 像素的圆，你可以移动鼠标来进行交互，如图 1.4 所示。

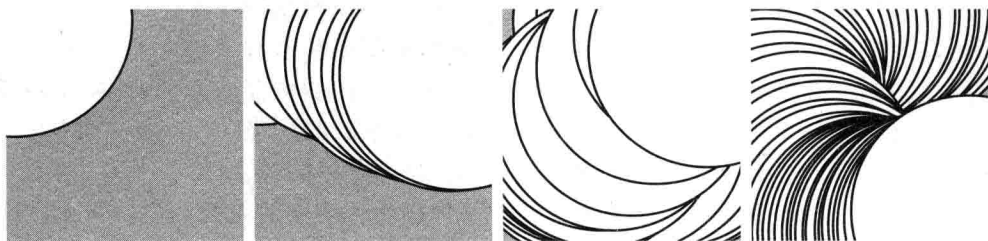


图 1.4

```
//draw() 函数会不断循环执行
void draw() {
    // 画一个圆，位置为鼠标的 x、y 坐标
    ellipse(mouseX, mouseY, 100, 100);
}
```

上面的圆形可以随着鼠标的移动而改变位置，是因为把它的特征属性“位置”设置成了随着时间变化的量，并且变化的量由鼠标位置来决定，这就是在程序动画的基础上添加交互因素的设计方法。

我们已经学习了第一个图形、第一个动画、第一个交互。在这里我想告诉读者，本章不要求你理解上面所有例子中每一行代码的原理，主要目的是让你对交互动画有个初步的理解，并且能够带着交互动画的概念来完成本书的学习。下面是构成交互动画的三个过程：

- (1) 图形：用特征描述绘图。
- (2) 动画：把特征属性定义为随时间变化的量。
- (3) 交互：设置变化量的决定因素。

第2章 变 量

2.1 声明变量

变量是存储指定数据类型空间。可以理解为存放同种类型东西的箱子，你可以把东西放进去，用的时候再拿出来，还可以拿出来再放进去同种类型的其他东西。这个箱子不像我们日常生活中用的箱子，能放不同类型的东西，在程序中它只能存放事先声明好的固定类型的东西。因为变量可以声明不止一个，就如你要买很多的箱子存放不同的东西一样，为了分辨它们，你会在外面给它们贴上标签，写上相应的名字，在需要用某个箱子的时候能通过它的名字找到它。所以当你想用变量储存数据时，你需要这样声明它：

```
// 数据类型 变量名字  
int numOfBooks;
```

“int”是数据类型，代表整型数据，“numOfBooks”是变量名字，最后用“;”结尾。这样我们就声明了一个变量用来储存书的个数。这里要注意“numOfBooks”变量里存储的不是书，而是书的个数。因为书的个数是整数，所以我们用“int”来定义它的数据类型。

2.2 数据类型

数据类型是指你要在变量里储存什么类型的数据。用箱子来比喻就是你要放什么类型的东西，是放零食，还是玩具，或者别的类型的东西，不管什么你都需要确定。当你定义了变量的数据类型以后，就意味着你之后只能往这个变量里存放这种类型的数据，你定义了箱子里放玩具，那你就只能往里面放玩具，不能把零食放进去。

Processing 的数据类型分两类，基础数据类型和复杂数据类型。我们在这一章先学习基础数据类型，在后面的章节会讲有关复杂数据类型的内容。

表 2.1 所示是 Processing 常用的基础数据类型。

表 2.1

数据类型	占用空间	取值范围
boolean	1 位	true 或 false
byte	8 位 (1 字节)	-128 ~ 127
char	16 位 (2 字节)	0 ~ 65535
color	32 位 (4 字节)	0 ~ 4,294,967,295
int	32 位 (4 字节)	-2,147,483,648 ~ 2,147,483,647
float	32 位 (4 字节)	-3.4E+38 ~ 3.4E+38

位的意思类似十进制数字的个位、十位、百位。例如：十进制数字 168 有 3 位，从右到左分别是个位、十位、百位，每 1 位都是满 10 进 1，用 0 ~ 9 来表示每 1 位的数值。