# 第3章 写出你的第一个游戏

接下来你即将拥有自己的第一款游戏“看水果学单词”。想想就令人兴奋不已。游戏要好玩，能给人带来快乐或挑战才有意义，就像我们小时候玩过的各种游戏，如老鹰捉小鸡、捉迷藏、跳方块等，无论长多大，都令人难忘。

## 3.1 创意

“看水果学单词”这款游戏是为英语学习者设计的，游戏主旨是帮助玩家记单词。基本想法是把图片和单词放到舞台上，由玩家进行识别和匹配。游戏设 4 关，每关给出 3 个水果和 3 个单词，全部匹配成功则过一关。

最初创意这款游戏时，我有两种设计思路。一种是，每一关水果图片是固定的(题目固定)，这样便于玩家强化记忆和复习。另一种是，每一关水果图片是随机的，这样玩家更有新鲜感，会更有乐趣，当然挑战性也高。新东方出了一本记单词的书，有两种版本，一种是按字母顺序的正序版，一种是乱序版。据我所知，许多人愿意用乱序版，因为乱序版更活泼，不死板。受此启发，游戏每一关内容采用随机抽取和随机排列的方法，以增强游戏的活泼性。

## 3.2 准备游戏素材

为本游戏创建一个项目文件夹：Chapter3。将所有图片素材放到 images 子文件夹中，声音素材放到 sound 子文件夹中。将单词做成一个记事本文件，放到 words 文件夹中。

(1) 游戏中使用的 12 种水果图片如图 3.1 所示。

(2) 声音素材如图 3.2 所示。

对 4 种声音文件应用场合做如下规定。

① 明亮提示音：匹配成功时播放。

② 我看错了：匹配错误时播放。

③ 太棒了：过关时播放。

④ 成功了：全部过关时播放。


图 3.1　游戏用到的 12 种水果素材


图 3.2　声音素材

(3) 单词表。与 12 种水果对应的单词如表 3.1 所示。

**表 3.1　单词与关卡设定表**

| 序　号 | 单　词 | 水果名称 | 关　卡 |
|---|---|---|---|
| 1 | Apple | 苹果 | 随机出现在各关卡中 |
| 2 | Pomegranate | 石榴 | |
| 3 | Grape | 葡萄 | |
| 4 | Strawberry | 草莓 | |
| 5 | Cherry | 樱桃 | |
| 6 | Banana | 香蕉 | |
| 7 | Peach | 桃子 | |
| 8 | Mango | 芒果 | |
| 9 | Watermelon | 西瓜 | |
| 10 | Orange | 橘子 | |
| 11 | Tomato | 西红柿 | |
| 12 | Pineapple | 菠萝 | |

## 3.3　导入素材到库

启动 Flash Professional CC 2015，创建一个 FLA 新文档，将新建 FLA 文档保存到 Chapter3 文件夹，命名为 MatchingGame. fla。

### 1. 图片导入

选择“文件”→“导入”→“导入到库”命令，打开“导入资源”对话框，定位到 images 文件夹，选中所有图片，导入到库中。

### 2. 声音导入

选择“文件”→“导入”→“导入到库”命令，打开“导入资源”对话框，定位到 sound 文件夹，选中所有声音文件，导入到库中。

在库中创建两个文件夹，分别命名为 images 和 sound。将图片归集到 images 文件夹，将声音归集到 sound 文件夹。整理完成后的“库”面板如图 3.3 所示。


图 3.3　将素材导入库中

## 3.4　创建游戏元件

### 1. 创建图片元件

单击“库”面板左下角“新建元件”按钮，创建水果卡片影片剪辑，如图 3.4 所示。设定

“名称”为“水果卡片”,“类型”为“影片剪辑”,导出“类”名称为 Fruit。单击“确定”按钮,进入元件创作舞台。


图 3.4 创建水果卡片影片剪辑元件

转到水果卡片元件创作舞台,完成一个 12 帧元件的创作。每一帧对应一幅水果卡片图。这里让卡片顺序与表 3.1 的单词顺序保持一致,如图 3.5 所示。

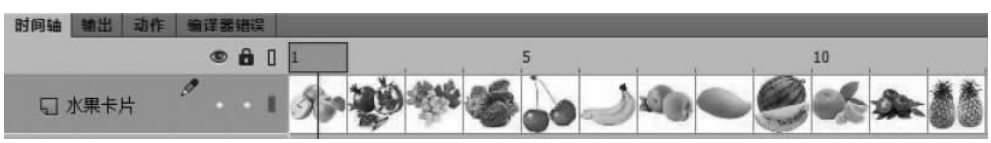


图 3.5 水果卡片时间轴排列

**小贴士**:快速创建水果卡片剪辑技巧

将第一张苹果图片从库中拖放到剪辑舞台,可用舞台对齐工具使其放置到舞台正中央。选择时间轴第 2 帧,插入一个关键帧。此时相当于对第 1 关键帧进行了复制操作。右击舞台中央的苹果图片,在快捷菜单中选择“交换位图”命令,在打开的对话框中选择石榴图片,确定之后第 2 帧图片即换成石榴,而且已经放到舞台正中央。依次类推,可以迅速完成 3～12 帧的创作,效果如图 3.5 所示。

### 2. 创建声音类型对象

选择库中的“成功了.mp3”声音对象,在右键菜单中选择“属性”,打开“属性”面板,切换到 ActionScript 子面板,勾选“为 ActionScript 导出”复选框,将声音的类名设为 Success。单击“确定”按钮,完成 Success 声音类定义。

按照类似办法,分别完成余下 3 个声音类的创建。完成效果如图 3.6 所示。


图 3.6 水果卡片剪辑和声音导出类型

### 3. 创建单词表剪辑

创建单词表影片剪辑，导出类名为 Words。参照表 3.1 顺序，对应每一帧，在单词表舞台上放置一个文本框，字体为 Arial Rounded MT Bold，字号为 48 磅，颜色为深灰色 #666666。用舞台对齐工具将每一帧文本放置到舞台正中央。完成后单词表剪辑时间轴效果如图 3.7 所示。

“库”面板如图 3.8 所示。请记住以下类名组合：

(1) 水果卡片——Fruit；

(2) 单词表——Words；

(3) 成功了.mp3——Success；

(4) 明亮提示音.mp3——Hint；

(5) 我看错了.mp3——Sorry；

(6) 太棒了.mp3——Best。


图 3.7　单词表剪辑时间轴(每一帧对应一个单词)


图 3.8　“库”面板状态

下一阶段的工作是为游戏开始、进行和结束分别创建 3 个影片剪辑，分别用来展示游戏的开始界面、进行界面和结束界面。

## 3.5　创建游戏封面剪辑 StartGame

游戏起始页面，也称游戏封面。就像一本书的封面一样，游戏标题和“开始”按钮是两个基本元素，其他可根据游戏需要，添加作者信息、帮助信息、游戏模式选择信息等。这里先完成一个最小化设计，效果如图 3.9 所示。

创建封面影片剪辑 StartGame 参考步骤如下。

(1) 创建“开始”按钮。单击“库”面板左下角的“新建元件”按钮，创建一个新元件，名称为“开始按钮”，类型为“按钮”。单击“确定”按钮进入元件创作舞台。在舞台正中央画一个椭圆作为按钮形状，上面写上文字“开始”，完成按钮创作。

(2) 创建影片剪辑 StartGame。单击“库”面板左下角的“新建元件”按钮，创建一个新元件，名称为 StartGame，类型为“影片剪辑”。勾选“为 ActionScript 导出”复选框，类名仍为 StartGame。单击“确定”按钮进入元件创作舞台。


图 3.9　游戏封面

(3) 绘制一个矩形背景框。在舞台上绘制一个宽高为 540×390 的矩形框,无填充颜色。让矩形框对齐舞台正中央。

(4) 添加标题。在舞台正上方添加游戏标题"看水果学单词"。

(5) 添加按钮。从库中将"开始"按钮拖放到标题正下方。

(6) 为按钮设定实例名称 btnStart。选择"开始"按钮,在"属性"面板中将其实例命名为 btnStart。

游戏封面剪辑创建完成,效果如图 3.9 所示。

## 3.6 创建游戏进行剪辑 PlayGame

游戏进行剪辑创建步骤如下。

(1) 在库中选择 StartGame 剪辑,右击,弹出快捷菜单,选择"直接复制"命令,弹出对话框,将剪辑名称改为 PlayGame,勾选"为 ActionScipt 导出"复选框,导出类名为 PlayGame。


图 3.10 游戏进行状态

(2) 双击库中的 PlayGame 影片剪辑,进入影片编辑舞台,删除标题和按钮。拖放 3 个单词实例放到左边,拖放 3 个水果实例放到右边,使各行对象上下对齐,效果如图 3.10 所示。

(3) 通过"属性"面板仔细地记下 3 个单词和 3 个水果的出现位置。这里采集到的单词位置如下:(−120,−135)、(−120,0)、(−120,135)。水果位置如下:(170,−135)、(170,0)、(170,135)。这些坐标数据将用于控制每一关单词和水果的合理摆放位置。

## 3.7 创建游戏结束剪辑 EndGame

创建 EndGame 剪辑,步骤如下。

(1) 首先在库中选择"开始按钮",在右键菜单中选择"直接复制"命令,在弹出的对话框中将元件名称改为"再来一遍"。单击"确定"按钮。双击打开"再来一遍"按钮,把按钮上的文字由"开始"改为"再来一遍"。对齐按钮形状,完成新按钮的创建。

(2) 选择库中的 StartGame 剪辑,在右键菜单中选择"直接复制"命令,在弹出的对话框中将剪辑名称改为 EndGame,勾选"为 ActionScipt 导出"复选框,类的导出名称为 EndGame,单击"确定"按钮完成新剪辑的创建。在库中双击 EndGame 剪辑,进入 EndGame 舞台。将标题文字改为"恭喜闯关成功",选择标题下方的"开始"按钮,在右键菜单中选择"交换元件"命令,与"再来一遍"按钮进行交换。至此,游戏结


图 3.11 元件创作完成

束界面 EndGame 剪辑创作完成。将库中元件用文件夹的方式进行分类整理。库中内容如图 3.11 所示。

## 3.8　游戏逻辑设计

游戏逻辑设计如下。

(1) 启动游戏：游戏初始运行时展示给玩家的界面是游戏封面。玩家单击封面上的“开始”按钮后，转入步骤(2)，进入游戏第一关。

(2) 首先完成初始化工作，包括抽取水果和单词卡片，生成游戏画面，呈现游戏画面，效果类似图 3.10 所示的布局，3 个单词卡片和 3 个水果卡片随机排列，等待玩家操作游戏。

(3) 玩家操作和玩法：玩家只可以拖动单词卡片，不能拖动水果卡片。将单词拖放到水果上后，程序判断其是否匹配。如果不匹配，则单词自动回到原位置；如果匹配，则单词与卡片一起消失。单词与水果卡片全部匹配成功后，如果还有下一关，重回步骤(2)，进入下一关。否则转入步骤(4)。

(4) 进入游戏结束界面：单击“再来一遍”按钮，重回步骤(1)。

为游戏设计两个类，一个为 WordCard 类，定义单词卡片拖放行为。另一个类是 GameMain 类，作为游戏主类，与 MatchingGame.fla 主文档关联。

## 3.9　数据结构设计

(1) 定义单词卡片和水果卡片摆放位置，用两个数组实现。

① 单词卡片位置定义：

```
private var aWordPos:Array = [{x: - 120,y: - 135},{x: - 120,y:0},{x: - 120,y:135}];
```

② 水果卡片位置定义：

```
private var aFruitPos:Array = [{x:170,y: - 135},{x:170,y:0},{x:170,y:135}];
```

(2) 将所有单词卡片对象和水果卡片对象分别用两个数组存放。

```
① private var aWords:Array;
② private var aFruits:Array;
```

(3) 将每一关单词卡片和水果卡片也用数组表示。

```
① private var aLevelWords:Array;
② private var aLevelFruits:Array;
```

## 3.10　WordCard 类设计

选择“文件”→“新建”命令，类型选择“ActionScript 3.0 类”，类名称指定为 WordCard，单击“确定”按钮，生成类初始框架。这里为简化起见，不指定包名称。选择“文件”→“保存”

命令，保存位置选择 Chapter3 文件夹，文件名称保持和类名 WordCard 一致。

完成 WordCard 类编码，如程序 3.1 所示。

程序 3.1：WordCard.as

```
01  package  {
02    import flash.display.MovieClip;
03    import flash.events.MouseEvent;
04    import flash.media.Sound;
05    /*
06        功能：WordCard 类定义了单词卡片的拖放行为处理函数，是库中 Word 剪辑的父类
07        设计：董相志
08        日期：2015.8
09    */
10    public class WordCard extends MovieClip {
11        var origX:Number;                              //单词卡片初始位置
12        var origY:Number;
13        var target:MovieClip;                          //与单词卡片匹配的目标对象
14
15        public function WordCard() {
16            this.addEventListener(MouseEvent.MOUSE_DOWN,Drag); //侦听拖动
17        } //end WordCard
18
19        //拖动卡片函数
20        function Drag(evt:MouseEvent) : void {
21            this.addEventListener(MouseEvent.MOUSE_UP,Drop);   //侦听放下
22            this.parent.addChild(this);            //目的是让当前卡片出现在容器的最上层
23            startDrag();
24        } //end Drag
25
26        //放下卡片函数
27        function Drop(evt:MouseEvent) : void {
28            this.removeEventListener(MouseEvent.MOUSE_UP,Drop); //移除侦听
29            stopDrag();
30            if (this.hitTestObject(target))                    //单词与目标匹配
31            {
32                var match:Sound = new Hint();                  //匹配提示音
33                match.play();
34                this.parent.removeChild(target);               //先移除目标对象
35                this.parent.removeChild(this);                 //再移除自身
36            } else {                                           //不匹配回归原位
37                var sorry:Sound = new Sorry();
38                sorry.play();
39                x = origX;
40                y = origY;
41            } //end if
42        } //end Drop
43    } //end class
44  } //end package
```

**小贴士**：WordCard 类完成之后，需要回到 MatchingGame.fla 主文件。打开“库”面板，

选择单词表剪辑，在右键菜单中选择“属性”命令，打开“属性”对话框，把单词表剪辑的基类改为 WordCard。这样，单词表剪辑就具备了 WordCard 中定义的拖放行为能力。

## 3.11　GameMain 类设计

选择“文件”→“新建”命令，类型选择“ActionScript 3.0 类”，类名称指定为 GameMain，单击“确定”按钮，生成类的初始框架。这里为简化起见，不指定包的名称。选择“文件”→“保存”命令，保存位置选择 Chapter3 文件夹，文件名称保持和类名 GameMain 一致。

完成的 GameMain 类编码如程序 3.2 所示。

程序 3.2：GameMain.as

```
01  package  {
02    import flash.display.MovieClip;
03    import flash.events.MouseEvent;
04    import flash.events.Event;
05    import flash.media.Sound;
06    /*
07        功能：GameMian 类实现游戏主逻辑，是与 MatchingGame.fla 关联的文档主类
08        设计：董相志
09        日期：2015.8
10    */
11    public class GameMain extends MovieClip {
12        //单词卡片位置定义
13        private var aWordPos:Array = [{x: -120,y: -135},{x: -120,y:0},{x: -120,y:135}];
14        //水果卡片的位置定义
15        private var aFruitPos:Array = [{x:170,y: -135},{x:170,y:0},{x:170,y:135}];
16        //所有的单词卡片对象和水果卡片对象数组
17        private var aWords:Array;
18        private var aFruits:Array;
19        //每一关的单词卡片和水果卡片数组
20        private var aLevelWords:Array;
21        private var aLevelFruits:Array;
22        //游戏状态影片剪辑定义
23        private var startGame:MovieClip;
24        private var playGame:MovieClip;
25        private var endGame:MovieClip;
26        private var bNewLevel:Boolean = false;                    //新的一关是否开始
27
28        public function GameMain() {
29            start();                                              //启动游戏
30        } //end GameMain
31
32        //游戏封面函数
33        function start() : void {
34            startGame = new StartGame();                          //创建游戏封面
35            addChild(startGame);
```

```
36          startGame.x = stage.stageWidth/2;                    //开始页定位到舞台中央
37          startGame.y = stage.stageHeight/2;
38          initGame();                                          //游戏全局初始化
39          startGame.btnStart.addEventListener(MouseEvent.CLICK,playing);
                                                                 //为开始按钮注册侦听器
40      } //end start
41
42      //游戏进行状态函数
43      function playing(evt:MouseEvent) : void {
44          startGame.btnStart.removeEventListener(MouseEvent.CLICK,playing);
                                                                 //移除“开始”按钮侦听器
45          removeChild(startGame);                              //从舞台移除首页
46          startGame = null;
47          playGame = new PlayGame();                           //创建开始页
48          addChild(playGame);
49          playGame.x = stage.stageWidth/2;                     //进行页定位到舞台中央
50          playGame.y = stage.stageHeight/2;
51          initLevel();                                         //本关初始化
52      } //end playing
53
54      //游戏全局初始化
55      function initGame() :void {
56          aFruits = [];                                        //数组初始化
57          aWords = [];
58          for(var i = 0;i < 12;i++) {
59              var fruitCard:Fruit = new Fruit();               //创建新水果卡片
60              fruitCard.gotoAndStop(i + 1);
61              var wordCard:WordCard = new Words();             //创建新单词卡片
62              wordCard.gotoAndStop(i + 1);
63              wordCard.target = fruitCard;//将单词卡片的匹配目标设定为对应的水果卡片
64              //新卡片统一放到数组保存
65              aFruits.push(fruitCard);
66              aWords.push(wordCard);
67          } //end for
68      } //end initGame
69
70      //本关游戏初始化
71      function initLevel() :void {
72          var randj:int;
73          aLevelFruits = [];              //初始化数组
74          aLevelWords = [];
75          //随机选取 3 张水果卡片和与之对应的单词卡片
76          for(var i:int = 0;i < 3;i++) {
77              //生成一个总卡片数组下标范围内的随机数
78              randj = Math.floor(Math.random() * aFruits.length);
79              aLevelFruits[i] = aFruits[randj];
80              aLevelWords[i] = aWords[randj];
81              aFruits.splice(randj,1);//从总卡片数组删除已被抽取的元素,避免重复抽取
82              aWords.splice(randj,1);
```

```
83              } //end for
84              for (i = 0;i < 3;i++) {                                  //随机摆放
85                  randj = Math.floor(Math.random() * aLevelFruits.length);
                                                                         //抽取一个水果卡片位置
86                  aLevelFruits[randj].x = aFruitPos[i].x;
                                                                         //确定水果卡片摆放位置
87                  aLevelFruits[randj].y = aFruitPos[i].y;
88                  playGame.addChild(aLevelFruits[randj]);              //添加到进行页面显示
89                  aLevelFruits.splice(randj,1);
90                  randj = Math.floor(Math.random() * aLevelWords.length);
                                                                         //抽取一个单词卡片位置
91                  aLevelWords[randj].x = aWordPos[i].x;                //确定单词卡片摆放位置
92                  aLevelWords[randj].y = aWordPos[i].y;
93                  aLevelWords[randj].origX = aWordPos[i].x;            //确定回归位置
94                  aLevelWords[randj].origY = aWordPos[i].y;
95                  playGame.addChild(aLevelWords[randj]);               //添加到进行页面显示
96                  aLevelWords.splice(randj,1);
97              } //end for
98              addEventListener(Event.ENTER_FRAME,checkingState); //检查游戏状态
99          } //end initLevel
100
101         //检查游戏状态
102         function checkingState(evt:Event) : void {
103             if (playGame.numChildren == 1 ) {          //说明只剩下背景框,可以开始下一关
104                 bNewLevel = true;
105             } //end if
106             if (bNewLevel && aFruits.length > 0) {     //进入下一关
107                 var best:Sound = new Best();           //“太棒了”声音
108                 best.play();
109                 initLevel();                           //本关初始化
110                 bNewLevel = false;
111             } //end if
112             if (bNewLevel && aFruits.length == 0)  { //进入游戏结束页面
113                 bNewLevel = false;
114                 removeEventListener(Event.ENTER_FRAME,checkingState);     //停止检查
115                 removeChild(playGame);                                    //移除游戏进行页面
116                 playGame = null;
117                 var success:Sound = new Success();                        //“成功了”声音
118                 success.play();
119                 endGame = new EndGame();                                  //创建结束页面
120                 addChild(endGame);
121                 endGame.x = stage.stageWidth/2;                           //定位到舞台中央
122                 endGame.y = stage.stageHeight/2;
123                 endGame.btnRePlay.addEventListener(MouseEvent.CLICK,rePlay);
                                                                          //注册重玩按钮
124             } //end if
125         } //end checkingState
126
127         //再来一遍
```

```
        function rePlay(evt:MouseEvent) : void {
            endGame.btnRePlay.removeEventListener(MouseEvent.CLICK,rePlay);
            removeChild(endGame);
            endGame = null;
            start();
        } //end rePlay
    } //end class
} //end package
```

## 3.12 游戏发布与测试

“看水果学单词”游戏项目发布后，整个项目文件夹内容如图 3.12 所示。MatchingGame.fla 文件、GameMain.as 文件和 WordCard.as 文件是原始的创作成果，MatchingGame.swf 文件是最终表现形式。

(1) 游戏开始界面如图 3.13 所示。测试要点：能否正确转到进行页面。

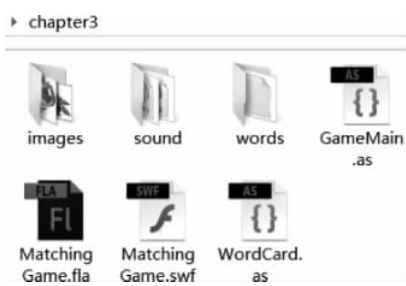


图 3.12 项目完成文件夹


图 3.13 游戏开始页面

(2) 游戏进行界面如图 3.14 所示。测试要点：对每一关进行测试。检查单词拖放功能是否正常，能否顺利进入下一关。检查单词匹配正确和错误时的处理逻辑。

(3) 下一关界面测试要点：一关通过后，能否顺利出现下一关界面。图 3.15 所示为一个可能出现的随机画面。

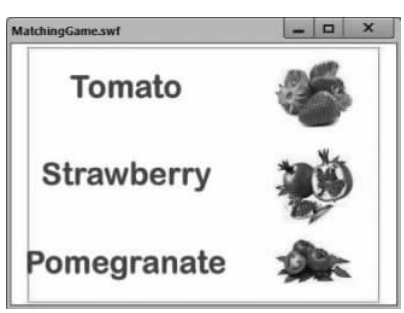


图 3.14 游戏进行页面

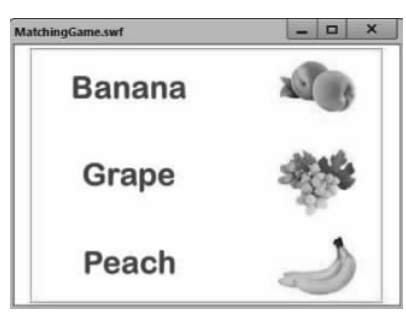


图 3.15 下一关界面

（4）游戏结束界面测试。所有关卡通过后，应该出现图 3.16 所示的结束页面。单击“再来一遍”按钮，又回到图 3.13 所示的开始页面。


图 3.16　闯关成功页面

## 3.13　小结

本章用短短 150 行程序，完成了“看水果学单词”游戏的创作。可以总结的东西太多了。麻雀虽小，五脏俱全。本章包括了游戏界面设计和逻辑设计。界面设计部分包括素材导入、影片剪辑元件和按钮创作、游戏状态定义、导出类应用、文档类与外部普通类、游戏中的事件逻辑、随机抽取单词与随机抽取水果卡片、灵活运用编程技巧等。

本章对游戏的讲解是粗线条的，没有对众多新语句和编程技巧做出详细解释，因为本章目的是让读者以速成方式看到一个完整的游戏全貌，并进行试玩体验，在此基础上进行反刍式学习。实际上，本章内容和方法会被后续游戏设计反复用到，许多问题会随着新的学习进程一一化解。

## 3.14　习题

1. 如何导入游戏中用到的图片素材和声音素材？

2. 为什么要将 12 张水果图片和 12 个单词分别做成影片剪辑？简述创作步骤。

3. 为什么要为单词卡片和水果卡片影片剪辑及声音文件设置导出类？简述实现步骤。

4. 对照程序 3.1，探究单词卡片拖放操作是如何实现的？

5. 对照程序 3.2 中 initGame 函数和 initLevel 函数，探究游戏初始化与关卡初始化的不同。

6. 简要描述本章游戏的整体创作步骤。

7. 简述游戏中两个按钮的设计步骤。

8. 简述开始页面的创作步骤。

9. 简述进行页面的创作步骤。

10. 简述结束页面的创作步骤。

11. 在程序 3.1 中，你可以从第 16 行语句开始，探究 drag 和 drop 这两个函数的事件逻辑，描述拖、放这两个操作行为之间的逻辑关系。

12. 程序 3.2 第 98 行，Event.ENTER_FRAME 是个什么事件？

13. 程序 3.1 和程序 3.2 中多次出现了 addEventListener 这个语句，尝试对这个语句的用法做个探究式学习。

14. 探究程序 3.2 中 initGame 函数，第 55～68 行，了解创建单词卡片与水果卡片的方法与逻辑。

15. 探究程序 3.2 中 initLevel 函数，第 71～99 行，了解单词卡片与水果卡片的随机抽取与排列方法。

16. 探究程序 3.2 中 checkingState 函数的内部逻辑。

17. 试玩游戏过程中会遇到各种声音提示，对照程序 3.1 和 3.2，找出所有发声的编程语句。

18. 上述问题可能对于刚起步的你有些难。不要紧，把所有令你困惑的问题列出来，这就是学习的起点。要知道，本章并不急于立即搞懂游戏的来龙去脉，一步步来吧。