

HTML5实验室 ——Canvas世界

张磊 编著

这不仅仅是一本介绍HTML5的书，
也不仅仅是一本介绍Canvas API的书，这
是开发者必须经历的算法洗礼，让聪明的
你去打造另一个Canvas世界。



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
<http://www.phei.com.cn>

HTML5 实验室 ——Canvas 世界

张磊 编著

电子工业出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书通过多个实验,由点到面地对 HTML5 相关的技术进行详细的介绍和剖析,涵盖了 HTML5 标准中描述的新特性。本书以简洁的文字,结合生动的实验介绍 HTML5 的特性,并深入剖析其内部原理,讲授如何使用 Canvas 集成特定的算法去实现绚丽的效果、应用和游戏,其中涵盖了数学和物理方面的知识,让读者不仅知其然,而且知其所以然;最后通过几个综合实验和经典游戏的重现,将各种新特性综合,实现酷炫的网页效果。

阅读本书不需要预先具备特定编程语言的知识,任何渴望投入 HTML5 世界的新手,想要巩固和复习数学和物理知识的编程老手,以及具备其他语言的编程经验(如 JavaScript、C#),想要了解 HTML5 新特性或加深对编程语言理解的程序员都适合阅读本书。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,侵权必究。

图书在版编目(CIP)数据

HTML5 实验室:Canvas 世界/张磊编著. —北京:电子工业出版社,2012.6

ISBN 978-7-121-17157-4

I. ①H… II. ①张… III. ①超文本标记语言 ②网页制作工具 IV. ①TP312 ②TP393.092

中国版本图书馆 CIP 数据核字(2012)第 106624 号

责任编辑:竇 昊

印 刷:北京丰源印刷厂

装 订:三河市鹏成印业有限公司

出版发行:电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本:787×980 1/16 印张:16 字数:358 千字

印 次:2012 年 6 月第 1 次印刷

印 数:4000 册 定价:49.00 元

凡所购买电子工业出版社图书有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系,联系及邮购电话:(010) 88254888。

质量投诉请发邮件至 zltz@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线:(010) 88258888。

/ 前 言 /

2005 年，许多网站相继加入 AJAX 应用。AJAX 是可以创建更好、更快和交互性更强的 Web 应用程序的技术，基于此技术，互联网应用程序可以变得更完善、更友好。AJAX 最开始代表异步请求和局部刷新，后来演变成为网页里一切好玩的东西，它拉近了 Web 程序与桌面程序的距离。

到了 2007 年，W3C 针对是否接纳 HTML5 进入标准审核程序的提议进行了投票，多数人表示赞成。后来，W3C 承认 HTML5 是正式标准。HTML5 是近 10 年来 Web 标准最巨大的飞跃，和以前的版本不同，HTML5 并非仅仅用来表示 Web 内容，它还提供了 Canvas、WebGL、Audio、Notifications、LBS、WebSocket、SVG、Storage、Web Workers、IndexedDB、FileAPI 等重要元素，每个标签都有其相应的开源项目，且可以独立成书。HTML5 的使命是将 Web 带入一个成熟的应用平台，在这个平台上，视频、音频、图像、动画、存储，以及与计算机的交互等都被标准化。如果说 AJAX 是在拉近 Web 程序与桌面程序的距离，那么 HTML5 即将抹去 Web 程序与桌面程序的差距，没人能阻止开发者将桌面程序搬到网页里。Web APP 的时代已经来临，HTML5 正在重塑一个崭新、绚烂的 Web 世界。

本书使用 HTML5 的 Canvas 作为实验平台，JavaScript 为编程语言，进行了大量的粒子实验、物理实验、3D 实验、像素实验和文字实验，然后带着读者从实验走向实战，带领读者一步步制作一个物理引擎，最后带领读者制作一款完整的 HTML5 游戏。

本书介绍的每个实验都可以移植为 Java、Objective-C、C++ 或者 ActionScript 版本，也可以移植到 XNA 或者 Silverlight 上，等等。所以本书面向的读者，不仅仅是前端开发工程师，也适合游戏开发人员、大学生或者高级美工阅读。本书同样适合对视觉艺术、计算机图形学、物理、数学（特别是线性代数）感兴趣的读者。本书所有代码都经过严格的测试和反复的使用，读者可以放心使用，如有任何问题及疑问，请邮件联系 mHTML5@qq.com。

关于本书代码及演示

本书是一本以实践为目标的图书，包含大量的物理、数学和计算机图形学的集成实验，每个实验的设计思想以及核心算法均辅以对应的代码示例。本书不包含任何伪代码算法描

述，全部代码均采用 JavaScript 语言加以实现。

本书代码提供了理解算法问题所必需的细节，并辅以大量的分析图片，展示了解决问题的核心推导步骤，读者可以在支持 HTML5 的浏览器（如 IE9 及以上版本、火狐浏览器、谷歌浏览器、Opera 等）中运行相关代码并查看呈现结果。读者也可以改变代码中的一些核心参数或者变量的值，然后执行代码，查看其结果的变化，这对于不熟悉算法、数学思想和调试的读者尤为重要。

虽然本书的代码示例采用的是 JavaScript 语言，但是需要强调的是，其他计算机语言都可以完成本书介绍的所有实验。需要注意的是，本书中的示例代码并非最终版本，虽然所有代码都经过反复使用和严格测试，但是不能保证其没有重构的空间，读者在理解其核心思想和架构的基础上可以自己进行相应的重构。

本书中的所有代码按照章节依次分类，可在电子工业出版社官网（www.phei.com.cn）下载，也可以向编辑索要（yangbo2@phei.com.cn）。书中介绍的每个实验都包含一个或者多个演示文件，详细展示了整个实验的制作过程，让读者循序渐进地理解其算法和思想。读者可以用各种文本编辑器或者 IDE（如 notepad、notepad++、Visual Studio、sublime text、aptana studio、Web Developer Express 或 Expression Web 等）打开查看。

作 者
2012 年 4 月

目 录

CONTENTS

上篇 实 验

第 1 章 基础实验	2
实验 1 奥运五环	2
实验 2 台球	8
实验 3 绘制动画	14
实验 4 超越祖冲之	18
实验 5 立体文字	21
实验 6 鸟巢	22
实验 7 贪吃蛇	31
第 2 章 物理实验	36
实验 8 质点运动与反射	36
实验 9 万有引力	40
实验 10 疯狂的大炮	43
实验 11 动能守恒不守恒你说了算	49
实验 12 密闭球	54
实验 13 不规则的密室	60
实验 14 大球欺负小球	67
第 3 章 3D 实验	76
实验 15 立方体	76
实验 16 星星点灯	88

实验 17 矩阵变换	92
实验 18 3D 变形金刚蝙蝠侠	101
实验 19 世界上最简单的 3D 场景渲染	107
第 4 章 综合实验	113
实验 20 正 N 边形变换	113
实验 21 动态加载文字	115
实验 22 Loading 图片	122
实验 23 繁花之上, 又生繁花	124
实验 24 心	127
实验 25 烟花易冷	131
实验 26 波	143
实验 27 粒子计数器	145
实验 28 时间之沙	149
实验 29 心碎	151
实验 30 Canvas 类库	154

下篇 游戏开发

第 5 章 一步一步搭建物理引擎	162
概述	162
第 1 步 面向对象编程	162
第 2 步 建立基本对象	165
第 3 步 集成单元测试框架	175
第 4 步 集成图形化输出接口	183
第 5 步 碰撞检测	188
第 6 步 方向包围盒——OBB	193
第 7 步 碰撞反应	198
第 8 步 重叠处理	204
第 9 步 贴图	206
物理引擎作品展示一	209
物理引擎作品展示二	211
物理引擎作品展示三	213

第 6 章 游戏开发全程实录	217
6.1 概述	217
6.2 框架搭建	217
6.3 资源加载	220
6.4 菜单制作	224
6.5 对象建立	228
6.6 碰撞检测	235
6.7 游戏音效	237
6.8 键盘控制	240
6.9 可玩性增强——积分、技能	244
6.10 总结	248

上 篇

实 验

第1章 基础实验

实验1 奥运五环

简介

奥林匹克五环标志是由皮埃尔·德·顾拜旦先生于1913年构思设计的，是由《奥林匹克宪章》确定的，也被称为奥运五环标志，它是世界范围内最为人们广泛认知的奥林匹克运动会标志。它由5个奥林匹克环套接组成，有蓝、黄、黑、绿、红5种颜色。环从左到右互相套接，上面是蓝、黑、红环，下面是黄、绿环。整个造形为一个上方大底部小的规则梯形。五个不同颜色的圆环代表了参加现代奥林匹克运动会的五大洲——欧洲、亚洲、非洲、澳洲和美洲，如图1-1所示。

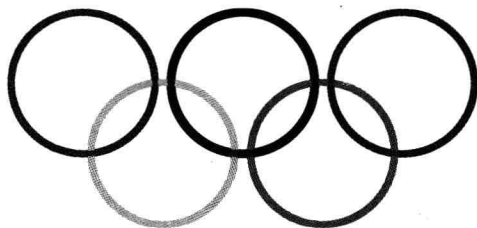


图 1-1 奥林匹克五环

如何在 Canvas 上绘制出一个一模一样的奥林匹克五环呢？首先从画圆开始。

画圆

在一切实验之前，要先理解 Canvas 中的坐标系统，假设一个宽是 300 高是 150 的 Canvas 标签，它内部的坐标变化如图 1-2 所示。

理解了坐标，开始画圆。CanvasRenderingContext2D 对象提供了一组用来在画布上绘制的图形函数。使用 CanvasRenderingContext2D.arc 来绘制一个圆形。



图 1-2 Canvas 坐标

语法:

```
arc(x, y, radius, startAngle, endAngle, counterclockwise)
```

利用该方法，直接在 Canvas 中画 5 个圆环。

```
<canvasid="myCanvas"width="400"height="200"style="border: 1px solid #c3c3c3;">
Your browser does not support the canvas element.
</canvas>
<scripttype="text/javascript">
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
    ctx.beginPath();
    ctx.arc(70, 70, 40, 0, Math.PI * 2, false);
    ctx.stroke();
    ctx.beginPath();
    ctx.arc(160, 70, 40, 0, Math.PI * 2, false);
    ctx.stroke();
    ctx.beginPath();
    ctx.arc(250, 70, 40, 0, Math.PI * 2, false);
    ctx.stroke();
    ctx.beginPath();
    ctx.arc(110, 110, 40, 0, Math.PI * 2, false);
    ctx.stroke();
    ctx.beginPath();
    ctx.arc(200, 110, 40, 0, Math.PI * 2, false);
    ctx.stroke();
</script>
```

效果如图 1-3 所示。

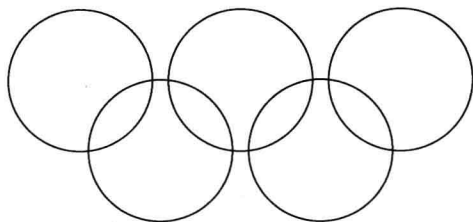


图 1-3 简单的黑白五环

可以看到和标准的奥林匹克标志相比，圆环宽度不够，而且不是彩色的，所以要设置 `CanvasRenderingContext2D` 的属性来绘制更加逼真的圆环。

```
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
    ctx.lineWidth = 5;
    ctx.strokeStyle = "#163B62";
    ctx.beginPath();
    ctx.arc(70, 70, 40, 0, Math.PI * 2, false);
    ctx.stroke();
    ctx.strokeStyle = "#000000";
    ctx.beginPath();
    ctx.arc(160, 70, 40, 0, Math.PI * 2, false);
    ctx.stroke();
    ctx.strokeStyle = "#BF0628";
    ctx.beginPath();
    ctx.arc(250, 70, 40, 0, Math.PI * 2, false);
    ctx.stroke();
    ctx.strokeStyle = "#EBC41F";
    ctx.beginPath();
    ctx.arc(110, 110, 40, 0, Math.PI * 2, false);
    ctx.stroke();
    ctx.strokeStyle = "#198E4A";
    ctx.beginPath();
    ctx.arc(200, 110, 40, 0, Math.PI * 2, false);
    ctx.stroke();
```

效果如图 1-4 所示。

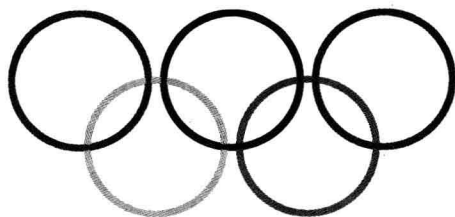


图 1-4 不嵌套的彩色五环

现在基本上已经和奥林匹克标志差不多了。可是细心的读者会发现，后画的圆形会覆盖在先画的圆形上面，这样就失去了环环相扣的感觉，那么如何环环相扣呢？现在就要学习画弧。

画弧

可以通过画弧来重新覆盖应被覆盖的部分，以实现环环相扣的视觉效果。

在画弧之前，先回到 CanvasRenderingContext2D 的 arc 方法。

```
arc(x, y, radius, startAngle, endAngle, counterclockwise)
```

以上代码中各参数的意义如表 1-1 所示。

表 1-1 各参数的意义

参 数	描 述
x, y	描述弧的圆心的坐标
radius	描述弧的半径
startAngle, endAngle	沿着圆指定弧的开始点和结束点的一个角度，这个角度用弧度来衡量，沿着 X 轴正半轴的三点钟方向的角度为 0，角度沿逆时针方向增加
counterclockwise	弧沿着圆周的逆时针方向（true）还是顺时针方向（false）遍历

如图 1-5 所示，更加形象化地表示了每个参数的意义。

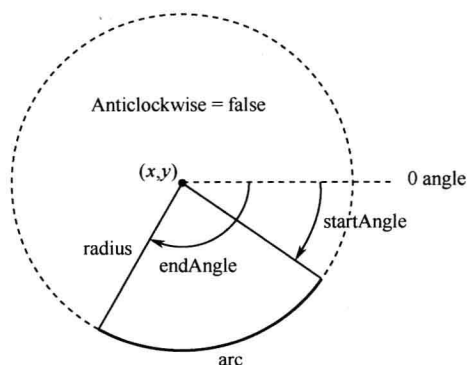


图 1-5 弧的各项参数

那么，需要怎样的一张图像盖在上面才能实现环环相扣的效果呢？如图 1-6 所示。



图 1-6 画弧

通过下面的代码实现上面 3 段弧形：

```
ctx.strokeStyle = "#163B62";
ctx.beginPath();
ctx.arc(70, 70, 40, Math.PI * 1.9, Math.PI * 2.1, false);
ctx.stroke();
ctx.strokeStyle = "#000000";
ctx.beginPath();
ctx.arc(160, 70, 40, Math.PI * 0.9, Math.PI * 2.1, false);
ctx.stroke();
ctx.strokeStyle = "#BF0628";
ctx.beginPath();
ctx.arc(250, 70, 40, Math.PI * 0.9, Math.PI * 1.1, false);
ctx.stroke();
```

把上面画圆的代码和画弧的代码合起来，顺序是先画圆，后画弧，弧就会盖在圆的上面。
完整的代码如下：

```
<canvas id="myCanvas" width="400" height="200" style="border: 1px solid #c3c3c3;">
Your browser does not support the canvas element.
</canvas>
<script type="text/javascript">
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
    ctx.lineWidth = 5;
    ctx.strokeStyle = "#163B62";
    ctx.beginPath();
    ctx.arc(70, 70, 40, 0, Math.PI * 2, false);
    ctx.stroke();
    ctx.strokeStyle = "#000000";
    ctx.beginPath();
    ctx.arc(160, 70, 40, 0, Math.PI * 2, false);
```

设置圆环的宽度

设置圆环的颜色

```

ctx.stroke();
ctx.strokeStyle = "#BF0628";
ctx.beginPath();
ctx.arc(250, 70, 40, 0, Math.PI * 2, false);
ctx.stroke();
ctx.strokeStyle = "#EBC41F";
ctx.beginPath();
ctx.arc(110, 110, 40, 0, Math.PI * 2, false);
ctx.stroke();
ctx.strokeStyle = "#198E4A";
ctx.beginPath();
ctx.arc(200, 110, 40, 0, Math.PI * 2, false);
ctx.stroke();

ctx.strokeStyle = "#163B62";
ctx.beginPath();
ctx.arc(70, 70, 40, Math.PI * 1.9, Math.PI * 2.1, false);
ctx.stroke();
ctx.strokeStyle = "#000000";
ctx.beginPath();
ctx.arc(160, 70, 40, Math.PI * 0.9, Math.PI * 2.1, false);
ctx.stroke();
ctx.strokeStyle = "#BF0628";
ctx.beginPath();
ctx.arc(250, 70, 40, Math.PI * 0.9, Math.PI * 1.1, false);
ctx.stroke();
</script>

```

从这里开始，
下面是画弧

运行效果如图 1-7 所示。

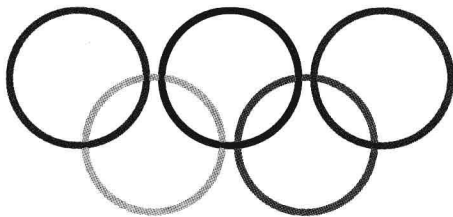


图 1-7 奥林匹克五环

就这样，实现了奥林匹克五环的环环相扣。其实 CanvasRenderingContext2D 不仅可以画空心圆，也可以画实心圆，下一个实验就会用到它相关的 API。

实验 2 台 球

简介

台球源于英国，它是一项在国际上广泛流行的高雅的室内体育运动，是一种用球杆在台上击球，依靠计算得分确定比赛胜负的室内娱乐体育项目。台球也称为桌球（中国港、澳地区的称法）、撞球（中国台湾地区的称法）。在没有开球之前，每个球的摆放位置如图 1-8 所示。



图 1-8 桌球

这次，要绘制的就是类似图 1-8 的效果。在绘制之前，先来绘制实心圆。

实心圆

CanvasRenderingContext2D 提供了 fill 函数，为一个封闭的区域内填充颜色，如圆形、矩形等。实心圆和空心圆的区别在于：

- 设置实心圆颜色：CanvasRenderingContext2D.fillStyle;
- 设置空心圆路径颜色：CanvasRenderingContext2D.strokeStyle;
- 填充实心圆：CanvasRenderingContext2D.fill;
- 连接空心圆路径：CanvasRenderingContext2D.stroke。

掌握了这些，可以轻松地在 Canvas 中画一个实心圆：

```
<canvasid="myCanvas"width="200"height="100"style="border:1px solid #c3c3c3;">
Your browser does not support the canvas element.
</canvas>
```

```

<script type="text/javascript">
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
    ctx.fillStyle = "#FF7F00";
    ctx.beginPath();
    ctx.arc(70, 18, 15, 0, Math.PI * 2, true);
    ctx.closePath();
    ctx.fill();
</script>

```

效果如图 1-9 所示。

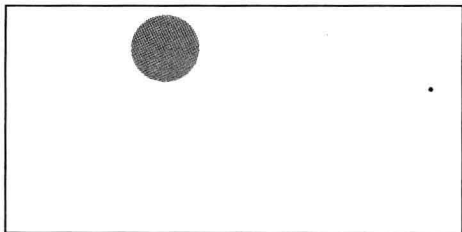


图 1-9 画圆

实心圆 ● /

掌握了画实心圆，就可以开始绘制台球桌面上的 16 个小球（包括母球）了。在绘制之前，先复习一下初等几何知识，一个直角三角形，如果有一个角为 30° ，那么三角形的三个边的比例如图 1-10 所示。

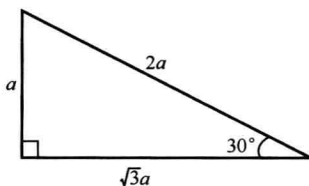


图 1-10 三角形各边比例

再回到要绘制的台球，如图 1-11 所示。最外层的圆心连成的三角形是等边三角形，然后被分成两个直角三角形，等边三角形的 3 个角都是 60° ，所以被分成的直角三角形的三个边的比例是 $1:\sqrt{3}:2$ 。