**第17章 用户交互**

在Canvas中，鼠标事件可以用来实现以下3种用户交互的效果：捕获物体、拖拽物体、抛掷物体。

拖拽物体是建立在捕获物体的基础上，而抛掷物体是建立在拖拽物体基础上的。

**17.5.1 捕获物体**

**1、矩形的捕获**

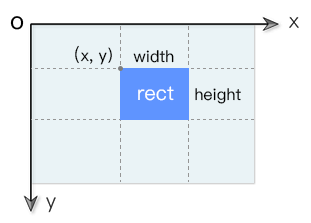


图17-16 矩形的捕获

判断鼠标点击坐标是否落在矩形区域上，我们只要判断mouse.x和mouse.y的范围。

**语法：**

if (mouse.x > rect.x &&

mouse.x < rect.x + rect.width &&

mouse.y > rect.y &&

mouse.y < rect.y + rect.height) {

……

}

**说明：**

当mouse.x和mouse.y同时满足以上条件，也就是4个条件必须都为true（缺一不可）时，则表示鼠标点击坐标就落在了矩形区域上，也就是我们捕获了该矩形。

**2、圆的捕获**

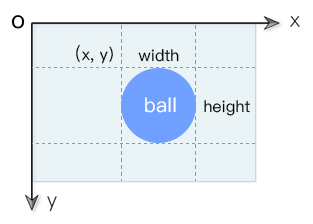


图17-17 圆的捕获

判断鼠标点击坐标是否落在矩形区域上，我们只要判断鼠标与圆心之间的距离。

**语法：**

dx = mouse.x - ball.x;

dy = mouse.y - ball.y;

distance = Math.sqrt(dx\*dx + dy\*dy);

if(distance < ball.radius){

……

}

**17.5.2 拖拽物体**

在Canvas中，想要拖拽一个物体，我们需要以下3步：

（1）**捕获物体**：在鼠标按下（mousedown）时，判断鼠标坐标是否落在物体上面，如果落在，就添加2个事件：mousemove和moveup；

（2）**移动物体**：在鼠标移动（mousemove）中，更新物体坐标为鼠标坐标；

（3）**松开物体**：在鼠标松开（mouseup）时，移除mouseup事件（自身事件也得移除）和mousemove事件；

**语法：**

cnv.addEventListener("mousedown", function () {

cnv.addEventListener("mousemove", onMouseMove, false);

cnv.addEventListener("mouseup", onMouseUp, false);

}, false);

**说明：**

上面是拖拽功能的简化语法。想要实现功能，我们一般都是在mousedown事件内部来处理mousemove和mouseup这个事件的。

**17.5.3 抛掷物体**

抛掷物体的实现思路：我们用鼠标选中一个物体，拖拽着它向某个方向移动，松开鼠标后物体会沿着拖拽的方向继续前进。在抛掷物体时，必须在拖拽物体的过程中计算物体的速度向量，并且在释放物体时将这个速度向量赋给物体。

举个例子，如果你以每帧10px的速度向左拖拽小球，那么在你释放小球时，它的速度向量应该是vx=-10。如果你以每帧10px的速度向下拖拽小球，那么在你释放小球时，它的速度向量应该为vy=10，以此类推。

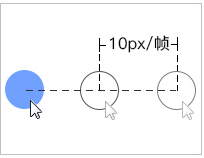


图17-18 抛掷原理

走到这里，相信大家也是越学越有激情，因为我们终于可以自行做出曾经想也不敢想的炫丽动画效果了。再努力一把，说不定哪天就当上总经理，出任CEO，迎娶白富美，想想内心还有点小激动呢。