

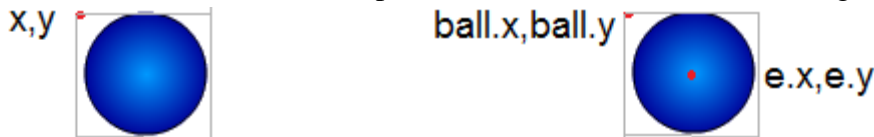
## The Jumping Ball

We can use our math to create a movie where a ball moves to the location clicked.

1. Start a new Flash ActionScript3 file. Create a Movie Clip named ball that is a circle with a gradient fill.
2. Write the code to move the ball to the location clicked. The argument e, a mouse event has properties stageX and stageY that tell us the point that was clicked.

```
function moveBall(e:MouseEvent)
{
    ball.x=e.stageX;
    ball.y=e.stageY;
} //moveBall
stage.addEventListener(MouseEvent.CLICK, moveBall);
```

3. Run the movie and click a few places. The ball does not move gradually from one place to another, it just jumps to that location immediately. Furthermore, the top left corner of the ball is at the location we clicked as shown on the left. What we really want is for the center of the ball to be at the point we clicked, as shown on the right:



4. Alter the code as shown below:

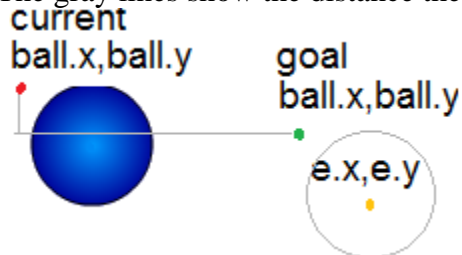
```
function moveBall(e:MouseEvent)
{
    ball.x=e.stageX-(ball.width/2);
    ball.y=e.stageY-(ball.height/2);
} //moveBall
stage.addEventListener(MouseEvent.CLICK, moveBall);
```

5. Run the movie and click. Now the ball is centered on the point clicked. It still just jumps to that location.

## The Moving Ball

We can use our math to create a movie where a ball moves to the location clicked. In the illustration the ball is near the top left corner, at the point shown by the red dot. We click at the point shown by the yellow dot. This will be e.x,e.y. We want to move the ball so that it is centered on the point clicked. Our goal is to move the ball to the point indicated by the green dot.

The gray lines show the distance the dot must travel.



We will declare variables outside the function called goalx and goalY. Inside the click event we will give them values as shown:

```
goalX=e.stageX-Math.round(ball.width/2);
goalY=e.stageY-Math.round(ball.height/2);
```

Let's suppose that we would like the ball to move from the current location to the goal point in 10 frames. The complete code is shown below. Study the math so that you understand exactly how the program works.

```

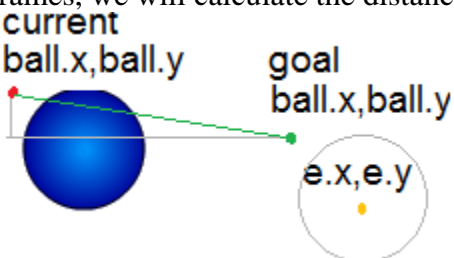
var dx:Number=0;
var dy:Number=0;
var goalX=ball.x; //not moving because we have not clicked yet
var goalY=ball.y; //not moving because we have not clicked yet

function frame(e:Event):void
{ //if the ball isn't at the goal, move it
  if(Math.round(ball.x)!=goalX) ball.x=ball.x+dx;
  if(Math.round(ball.y)!=goalY) ball.y=ball.y+dy;
} //frame
this.addEventListener(Event.ENTER_FRAME, frame);

function moveBall(e:MouseEvent) //mouse clicked on stage
{ goalX=e.stageX-Math.round(ball.width/2);
  goalY=e.stageY-Math.round(ball.height/2);
  var distanceX:Number=goalX-ball.x;
  var distanceY:Number=goalY-ball.y;
  dx=distanceX/10; //to reach goal in 10 frames
  dy=distanceY/10;
} //moveBall
stage.addEventListener(MouseEvent.CLICK, moveBall);

```

When you test the movie, you will find that the ball moves very fast if you click further away from the ball, and it moves slowly if you click close to the ball. In the illustration bellow, the green line indicates the total distance the ball must travel. Instead of moving the ball in ten frames, we will calculate the distance and move the ball in that many frames.



The green line is the hypotenuse of the right triangle formed by the two gray lines. We can find the length of the green line using the formula:  $c = \sqrt{a^2 + b^2}$  In this case we will substitute distance for a, and distance for b:

```
var distance=Math.sqrt(Math.pow(distanceX,2)+Math.pow(distanceY,2));
```

The complete code is on the next page:

```

var dx:Number=0;
var dy:Number=0;
var goalX=ball.x; //not moving because we have not clicked yet
var goalY=ball.y; //not moving because we have not clicked yet

function frame(e:Event):void //move ball if not at goal
{
    if(Math.round(ball.x)!=goalX) ball.x=ball.x+dx;
    if(Math.round(ball.y)!=goalY) ball.y=ball.y+dy;
} //frame
this.addEventListener(Event.ENTER_FRAME, frame);

function moveBall(e:MouseEvent) //set dx, dy to move gradually
{
    goalX=e.stageX-Math.round(ball.width/2);
    goalY=e.stageY-Math.round(ball.height/2);
    var distanceX:Number=goalX-ball.x;
    var distanceY:Number=goalY-ball.y;
    var distance=Math.sqrt(Math.pow(distanceX,2)+
        Math.pow(distanceY,2));
    dx=distanceX/Math.round(distance);
    dy=distanceY/Math.round(distance);
} //moveBall
stage.addEventListener(MouseEvent.CLICK, moveBall);

```

The ball now moves at the same painfully slowwww pace whether you click near it, or farther away. You can speed the ball up by multiplying dx and dy by a speed factor. Add a variable speed with an initial value of 5 above the functions, making it global.

```
var speed:Number=5;
```

At the end of the moveBall function multiple both dx and dy by speed:

```
dx=dx*speed;
dy=dy*speed;
```

Now we have a new problem. Suppose that GoalX is 45. If we are counting by 5 and start at 13 we will move ball.x to 18, 23, 28, 33, 38, 43, 48, 53, 58 and never actually reach the goal. Instead of looking for the ball to hit an exact goal, we must look for it to fall within a range of values. In other words, we want the ball to stop when it gets near the goal: when it is within dx of the goal.

This is complicated a little by the fact that the ball can be moving either left (dx is negative) or to the right (dx is positive.) Fortunately, Math.abs can make a negative number positive. It does not change a positive value. We will calculate a minimum value for ball.x to be considered at goal:

```
var minx=goalX-Math.abs(dx);
```

The maximum value is:

```
var maxx=goalX+Math.abs(dx);
```

We will do the same for y and then test to see if the ball is between those two values.

```
var dx:Number=0;
var dy:Number=0;
var speed=5;
var goalX=ball.x; //not moving because we have not clicked yet
var goalY=ball.y; //not moving because we have not clicked yet

function frame(e:Event):void //if the ball isn't at the goal,
move it
{
    var minx=goalX-Math.abs(dx);
    var maxx=goalX+Math.abs(dx);
    var miny=goalY-Math.abs(dy);
    var maxy=goalY+Math.abs(dy);
    var cx=ball.x;
    var cy=ball.y;
    if(cx<minx||cx>maxx) ball.x=ball.x+dx;
    if(cy<miny||cy>maxy) ball.y=ball.y+dy;
} //frame
this.addEventListener(Event.ENTER_FRAME, frame);

function moveBall(e:MouseEvent) //based on point clicked calc.
goal point and dx,dy
{
    goalX=e.stageX-Math.round(ball.width/2);
    goalY=e.stageY-Math.round(ball.height/2);
    var distanceX:Number=goalX-ball.x;
    var distanceY:Number=goalY-ball.y;
    var
distance=Math.sqrt(Math.pow(distanceX,2)+Math.pow(distanceY,2))
;
    dx=distanceX/Math.round(distance);
    dy=distanceY/Math.round(distance);
    dx=dx*speed;
    dy=dy*speed;
} //moveBall
stage.addEventListener(MouseEvent.CLICK, moveBall);
```

**Challenge:** Add buttons for faster and slower like you did for the car. This time the buttons will be used to change the value of speed. (Hint: Don't let the speed become negative.)