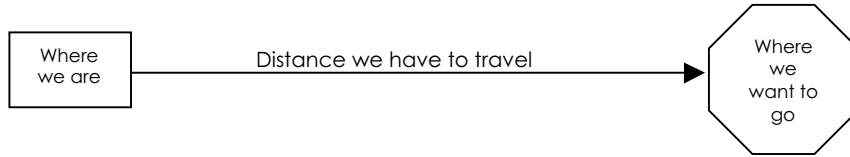


## Creating Real Movements (part 1)

Where you want to go = where you are + distance we have to travel

or

Where we are + distance we have to travel = where you want to go



As we get closer to the destination we slow down!

Distance always changing and so distance remaining is not always the same. Must continue (in flash programming) continue to assess where we are and distance traveled because object is moving (itineration).

Asking comes within an **enterFrame** script of some sort.

## The Algorithm (n)

Where we are + ((where we want to go)-where we are))/n

**If n is a number:**

- < .5 : goes past target but never stops
- = .5 : oscillates past and on target
- .5 but < 1 : goes past target but then stops on target after a few bounces
- = 1 : immediately moves to target (no iterations)
- > 1 but < 2 : slows down as get to target (1.8 is almost perfect n value)

**Target** = where we want to go

**Property of object** = where we are

Where we want to go	=where we are	+((where we want to go	-where we are)	/.8
target	get object property	target	get property	n value
Original obj. position	Original obj. position	Target position	Original obj. place	/.8

### Example One:

```
//init frame
objx = obj._x;
objy = obj._y;
targetx = 0;
targety = 0;

//this itenerates and changes because as object moves
//so does the distance change!
distx = targetx-objx;
disty=targety-objy;

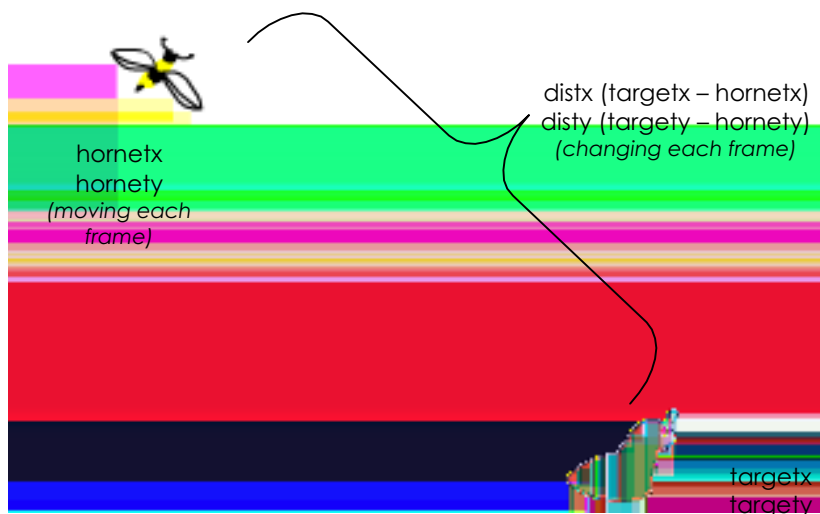
//add some inertia to slow it down while it comes closer to target
objx = Math.round(objx + (distx/5));
objy = Math.round(objy + (disty/5));

//motion for object that will be evaluated above
mc._x = objx;
mc._y = objy;
```

### Example Two: with TOAD and HORNET

```
//init ONCE!
//where the hornet is initially
hornetx = hornet._x;
hornety = hornet._y;
//where toad is (our target!)
targetx = toad._x;
targety = toad._y;

//keep evaluating over and over again for itenerations
_root.onEnterFrame = function() {
    //the changing distance between toad (target) and hornet
    distx = targetx-hornetx;
    disty = targety-hornety;
    //the inertia to slow the hornet down as it comes closer to toad
    hornetx = Math.round(hornetx+(distx)/20);
    hornety = Math.round(hornety+(disty)/20);
    //the actual animation and movement of the hornet!
    hornet._x = hornetx;
    hornet._y = hornety;
};
```



## Formulas for ... Speed and Velocity and Acceleration

### Speed

Ratio of **distance** (*units in Flash*) covered and **time** taken (*frames in Flash*) to cover this distance.

Distance = speed \* frames. (units per frame)

```
xmov = 2;  
_root.onEnterFrame= function(){  
obj._x = xmov;  
};
```

### Acceleration

Acceleration occurs whenever velocity changes. Velocity = speed and direction and so if the speed or direction changes so does the acceleration. Acceleration has a magnitude (ratio of difference in velocity and difference in time over which difference occurred as well as direction the acceleration occurred) and a direction.

```
//default acceleration  
accel = 2;  
//default speed  
xmov = 1;  
ymov = 1;  
//update in each frame  
_root.onEnterFrame = function() {  
//do this when key up is pressed  
    if (Key.isDown(Key.UP)) {  
//go up (same as xmov = xmov + accel)  
        xmov += accel;  
        ymov += accel;  
//do this when key down is pressed  
    } else if (Key.isDown(Key.DOWN)) {  
//go down (same as xmov = xmov - accel)  
        xmov -= accel;  
        ymov -= accel;  
    }  
  
//put in object (same as obj._x = obj._x + xmov)  
    obj._x += xmov;  
    obj._y += ymov;  
};
```

## Velocity and multiple forces

Acceleration of an object is inversely proportional to its mass and proportional to the net external force applied to it

Net force = mass \* acceleration

```
//default object movement
ymov = 0;
//object mass
mass = 1;
//gravity force
forcegravity = 30;
helium force
forcehelium = -31;
//creating new force based on two clashing forces
netforce = forcegravity+forcehelium;
//the new acceleration of the object based on netforce as in formula
above
yaccel = netforce/mass;
//updating in every frame the movement
_root.onEnterFrame = function() {
//same as ymov = ymov + yaccel; take acceleration -put it into ymov
variable
    ymov += yaccel;
//take number out of ymov variable and apply to object coordinate each
frame
    balloon._y += ymov;
};
```

## Friction

Objects slow down because they are losing energy; friction opposes the object's directional motion and slows the object down over time...

Choose a number between 0 and 1 (decay number)

Multiply decay by the velocity in every frame

```
//default movement of object
xmov = 10;
//decay # between 0 and 1
decay = .95
//update below values every frame
_root.onEnterFrame = function(){
//same as xmov = xmov * decay (* = multiply)
xmov*=decay;
same as obj._x = obj._x + xmov
obj._x+=xmov;
}
```

## Gravity

Gravity is a force that pulls down (I hope you know this)

```
//default movement of object
ymov = 0;
//default gravity of objects
gravity = .5;
//execute in every frame
_root.onEnterFrame = function(){
//same as ymov= ymov+gravity
ymov += gravity;
//take the ymov + gravity total and put it in the object y coordinate
obj._y+=ymov;
```