Physics Introduction Axe Tutorial

One of the things that I could never understand in Axe was acceleration/velocity and physics. And although everybody kept saying Builderboy's tutorials were the best for the job, I thought they were too advanced for me when I read them. When I read Axe Physics tutorials I was confused. Of course the tutorials were great, but just not for a true beginner.

All of my games were always non-realistic, without acceleration and using simple code and lots of dirty hacks. The other I finally understood acceleration and physics, and decided to write this very simple tutorial for the job.

Also, you may note most of the code in this tutorial is not optimized so you can understand it better.

Let's start with displaying an image and gravity:

```
.Y Position of the block
0→Y
.X Position of the block
44→X

Repeat getKey(15)
  ClrDraw

  .Draw a line at the bottom
  Line(0,63,95,63)

  .Draw our main image, a square of width=8
  Rect(X,Y,8,8

  !If (pxl-Test(X+3,Y+8))
    Y+1→Y
  End

  DispGraph
End
```

An image can be found here. This is an irrealistic gravity example.

This code draws a square and it will fall down vertically (gravity) until it hits the line. However, this code is not realistic. If you're surprised, don't worry, it's easy to get.

When an object falls down from, let's say, the top of a building, it will fall down vertically and it's speed will increase. So, the highest speed reached is higher if you throw it from a higher building. Newton calls this speed 'gravitational acceleration'.

So how to make it realistic? Weneed a variable for the acceleration and a variable for the Y position of the sprite. We don't change the Y position of the sprite directly, but add to its speed.

Now the next code uses the Fixed Point or as I prefer the x256 mode. We multiply the X and Y positions of stuff by 256 to get more precision. So if in the code I write `X+1→X` I'm adding 1/256 of a pixel to X. If you ever had the need to move a sprite/block to the right but not so fast as `X+1→X` then you can use the x256 mode. It's very important in Axe Physics.

To display sprites we write lines such as `Pt-On(X/256,Y/256,PTR)`. So when we display them, we divide their positions by 256 to fit on the graphscreen. Functions like pxl-Test also require us to divide variables by 256 (if we're using the x256 mode).

Here's how it works to check if a pixel is black:

```
.X And Y Positions as pxl-Test( arguments
If (pxl-Test(X/256,Y/256))
  .CODE
End
```

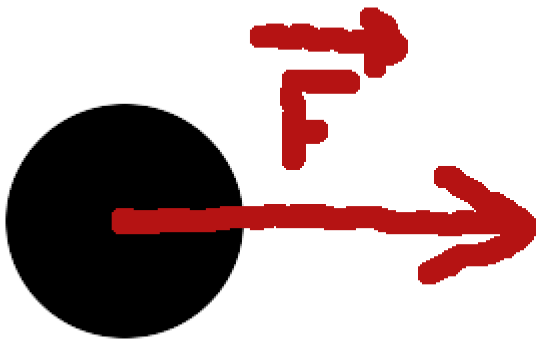
Now here's some code that you can use as an example of acceleration and the x256 mode.

```
.Y Position of the block
0→Y
.Y Acceleration of the block
0→B
.X Position of the block
0→X

Repeat getKey(15)
  ClrDraw
  Line(0,63,95,63)

  Rect(X/256,Y/256,8,8)

  .If nothing beneath block, make block go down
  !If (pxl-Test(X/256+3,Y/256+8))
    B+4→B
  End

  .If something beneath block, make block stop
  If (pxl-Test(X/256+3,Y/256+8))
    0→B
  End

  Y+B→Y

  DispGraph
End
```

An image can be found here. This is a more realistic gravity example.

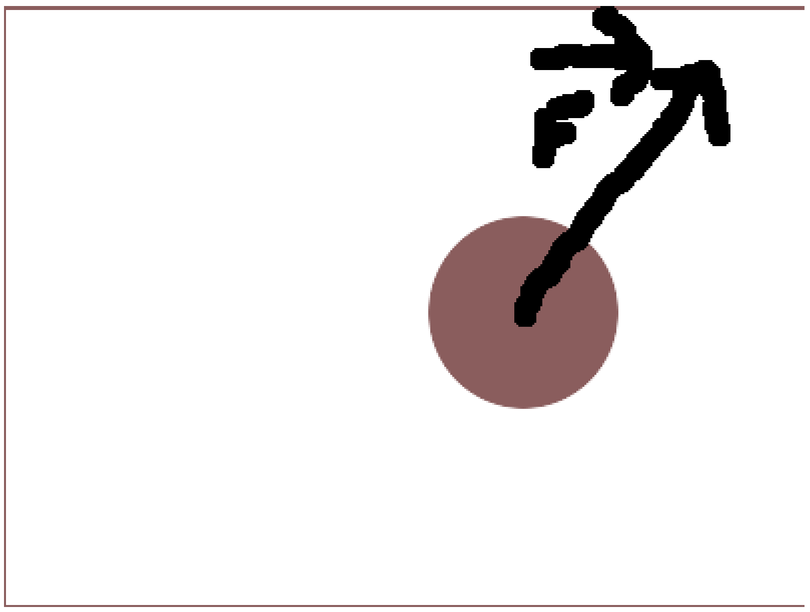
As you can see, the speed of the block in this code gets higher and higher. However, if it hits the line, it immediately stops.

Now there's a very important concept you need to understand. Vectors and Components of Vectors. A movement can be represented by a vector, here is an example:



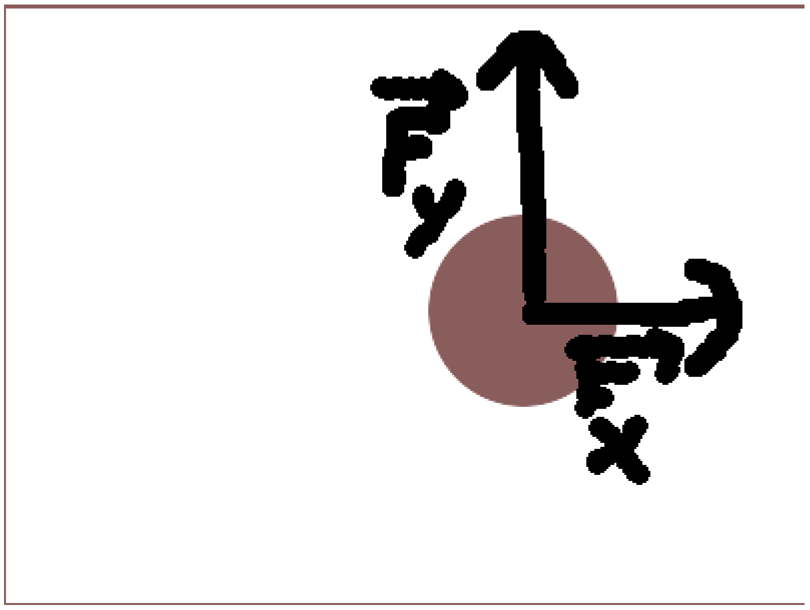
The ball is moving right because there's a force in the direction where it is going.

Now let's imagine a ball being thrown in the air:



The ball is not moving to a single direction, but both up and right. The vector has a force, but it can be split in two forces, the X force and Y force. The Y force is the vertical component of the force and the X component is the horizontal component of the force.

So, the last image could be actually represented this way:



When we add Fx to Fy we have the F force in the last image.

In Axe, to make acceleration work like in real lifes, we have 4 variables:

```
.X Position
0→X
.Y Position
0→Y
.X Acceleration
0→A
.Y Acceleration
0→B
Repeat getKey(15)
  ClrDraw

  .CODE

  .Sum acceleration to the positions of the sprites
  X+A→X
  Y+B→Y

  .MORE CODE

End
```

This will make gravity much more realistic.

Yet again, note this tutorial is just to let you more comfortable with some concepts and acceleration and Physics in Axe. I recommend you to move on to more advanced tutorials such as Builderboy's and others which can be found in axe.omnimaga.org.

Thanks
I'd like to thank Builderboy for inspiring me to learn Physics by making awesome games.

Thanks squidgetx for a tutorial you wrote on Omnimaga.

Thanks all Omnimaga members for support!