Beginner Scripting Tutorial 1

This is an easy tutorial for beginners. It is not intended to teach you new techniques but more to show you all the neat stuff you can do with the functions and the HPL2 engine.

In this tutorial you'll learn how to make entities 'float' or rather make them seem to be very very light. Possible uses could be for some strange sanity effects or even simulating an underwater environment!

Getting started

Start by creating a new level in the level editor and then saving the map. There are no real limits, but keep it simple and include some movable entities you can use the script on. In my little room it's the skull and two books. You should include another entity you don't use the effect on, so you can compare their characteristics!

My room looks like this:

Link

After you have saved your map, create a .hps file with the same name. This is where we'll do all the scripting, add the default script there. I have removed the if-clause in my example, it's not needed.

```
////////////////////////////
// Run first time starting map
void OnStart()
{
   GiveItemFromFile("lantern", "lantern.ent");

   for(int i=;i<10;i++) GiveItemFromFile("tinderbox_"+i, "tinderbox.ent");
}

////////////////////////////
// Run when entering map
void OnEnter()
{

}

////////////////////////////
// Run when leaving map
void OnLeave()
{

}
```

The scripting

Now we'll do the actual scripting! The floating effect is based on timer functions, and one of these we'll add in *OnStart()* so it is immediately called when the level loads! The syntax for timer function looks like this:

```
//Function syntax: Func(string &in asTimer)
void  AddTimer(string& asName, float afTime, string& asFunction);
```

The first string *asName* is the name of our timer, *afTime* is the time when the timer will call the function in *asFunction* after the timer is executed. Adding our timer in *OnStart()* will make our code look like this:

```
////////////////////////////
// Run first time starting map
void OnStart()
{
    GiveItemFromFile("lantern", "lantern.ent");
    for(int i=;i< 10;i++) GiveItemFromFile("tinderbox_"+i, "tinderbox.ent");

    AddTimer("FloatTime", 0.01, "FloatObjects");
}
```

As you can see, **FloatTime** is the name of our timer which calls the function **FloatObjects** after a very short time, exactly 0.01 seconds. The purpose of this will be explained after we created **FloatObjects**!

For actually making the objects float, we'll add this function in **FloatObjects**:

```
AddPropImpulse(string& asName, float afX, float afY, float afZ, string&
asCoordSystem);
```

AddPropImpulse gives a specified entity an impulse, which means pushing it in a certain way with certain force. *asName* is the name of the entitity, The floats *afX*, *afY* and *afZ* are the direction coordinates the push has to follow and *asCoordSystem* specifies the coordinate system the game refers to. You can leave it empty, but valid strings would be “World” for using the XYZ planes you can see in the editor or “local” if it should use the XYZ axis of the object you are adressing.
AddPropImpulse has to be created for **every** object you want to float, which makes three of them in my case! The last thing we add in **FloatObjects** is a timer function that practically calls itself so we can create a loop that will continually execute **FloatObjects** and give our entities a little push every 0.01 seconds! Our function should look like this:

```
void FloatObjects(string &in asTimer)
{
    AddPropImpulse("human_skull_1", , 0.15, , "");
    AddPropImpulse("book01_1", , 0.15, , "");
    AddPropImpulse("book01_2", , 0.15, , "");
    AddTimer("FloatTime", 0.01, "FloatObjects");
```

```
}
```

So now I've added an impulse for every object I want to float (the skull and the two books) and a timer that calls itself, repeating **FloatObjects** over and over.

You may have noticed that I've only put in a value for the *afY* float. That means the impulse only goes in the Y direction or UP! 0.15 is a very weak impulse, but it is applied every 0.01 seconds so all the little pushes can't be noticed by the player which creates the effect of the objects floating or falling very slowly. You can now save the .hps file and load your level. Have fun and thanks for reading!

Additional information

Here you can put the little things you've noticed while playing around and testing my script. Stuff I've already found out:

- Different entities need different *afY* values! For small objects like books 0.15 is enough, but everything bigger than 2 will make them float upwards like a helium balloon.
- Play around with *afTime*! Bigger values like 0.5 combined with larger impulses like 5 can create some crazy bouncing effect!