##

EBrowser v 0.2 - by Masklin Jul-2001

##

Description:

EBrowser is an utility that allows to manage entities easily from the own Blade with a LED style.

This tool is for Levels designers and maybe it's not intuitive for users without experience in LED editing. However, all the habitual LEDx user will find into EBrowser an invaluable toy.

EBrowser is not limited to placement of objects, it can to assign object properties as breaking or change the parameters of light and fire for the luminous objects.

It can also create and move enemies. With an similar interface you can create an enemy with a weapon, a shield and a potion.

Finally, EBrowser can export the data of the current object in python and dol format.

Installation:

EBrowser uses Tcl-Tk widgets through the Pmw library implementation. The installation includes a simplified version of both libraries. EBrowser also needs update the Py-Tk library included with Blade.

If you have a 8.0.5 compatible version of TCL installed into your computer, don't install the version supplied with EBrowser.

To install EBrowser run the EBSetup program. Make sure that installation directory is the same where Blade was installed. EBrowser make a EBrowser subfolder in this directory with this Readme.txt file.

Usage:

* Startup.

Start Blade with the console enabled. Go, with your player to the selected area and imports EBrowser:

```
import EBrowser
```

The EBrowser main window will appear and you can select any entity from the combo box. You can increase the Area of Scan and push the Scan button to get additional entities. On screen you will have properties as the position, orientation or scale of the entity. You can change them from here, although it will be easier with the keyboard.

* Entity control.

If you has selected an entity then you can close the EBrowser window. From Blade you will control the current entity with the keypad:

Movement:	Up - Down	KP8 - KP2
	X	KP4 - KP6
	Y	KP1 - KP3
Rotate:	Left - Right	KP7 - KP9
	Axis change	KPMultiply (*)
Scale	bigger - smaller	KP+ - KP-
Drop	Drop to the floor	KPEnter
Freeze	Freeze enemies	KPDivide (/)
Change	Next/Previous Entity	Nxt. Pag. / Prv. Pag.
Drag	Drag to player	KP5

You can drag the selected entity until the player position with KP5. The entity is located above the player. It is possible that, if the entity is very big, the player be blocked. You can use the movement keys to move the entity.

With the Prev/Next entity keys you can select any entities in the current area of scan. On screen you can see the type and the name of the selected entity.

The Drop key allows to fall the object or the enemy to the floor. Some objects have physical properties and they will fall according to their principles.

The Freeze key freezes all the near enemies to the player. If an enemy doesn't stop you can come closer and test again. In the EBrowser windows you can change the range for this command. It is also possible to freeze/unfreeze the enemies from the EBrowser window.

When a Enemy is selected from the EBrowser window, the enemy is frozen automatically.

* Properties tabs.

Press KP0 to activate again the EBrowser main window.

In this window you can change the increments used in the movement, rotation and scale so that you can adjust comfortably the placement of any entity.

The More... button shows several windows with additional information that depend on the type of entity selected. So, it is possible to make a static or breakable object. If the object is luminous you will be able to change all the associate properties: intensity, radio, color,...

For the enemies you can set the Level or the life and it is also possible to assign them a weapon, shields and even a potion. Although the list of weapons and shields are quite complete, it is necessary to keep in mind that the creation of some weapons demand specialized procedures. This programming task it's a exercise for you xDD.

Finally, EBrowser makes public a pointer to the entity selected:

```
EBrowser.CurrEnt
```

so, you can change any attribute of the selected entity and call to any method. For example,

```
EBrowser.CurrEnt.Position
```

is the current position of the entity. Any change made from the console will be reflected in EBrowser.

* Entities creation.

  From EBrowser it's possible to create objects and enemies. You should select the wanted object/enemy and then, use the Create button. It is also possible to create entities from Blade using the key Insert.

  The entity is created at target camera position: this is approximately in the player's head. You can use the movement keys to move the entity to the required position.

  If you have created an enemy, EBrowser assigns him the same orientation that your player, a Level=1 and the maximum of life for that level/chartype.

  Once created the entity uses the properties sheet to change all the required characteristics.

  You will also be able to erase an entity selecting it and using the Delete key.

  With the Clone key (End) you make a copy of the current entity. EBrowser will be copied many attributes of the original entity.

* Exporting entities.

  From EBrowser you can export the definition of any selected entity as python script or dol format. Obviously the dol is only available for objects.

  In python scripts, exports all the attributes set by EBrowser, including calls to Blade routines.

  The changes are added at the end of the selected file without to erase their current content. For that reason, are not included the required imports so that the routines work. You should include them at the beginning of the script.

  For the new entities, EBrowser generates names with the format Kind_NN, where Kind is the object/enemy class (table, ork,..) and NN is a sequential number. This number is saved in he configuration file:

  ```
  BOD\Config\EBrowser.py.
  ```

  So, it is difficult that the name of the entities repeats although you should always make sure that there are not two entities with the same name in your map.

Notes:

This utility helps in the level design process, but you need some knowledge of python and the entities and functions used by Blade.

When you enable EBrowser you are not in game mode: maybe, it don't work as before activating it.

While you are in the EBrowser window Blade is not active. Don't play with your luck: don't try to go to the Blade or console windows.

Sometimes, the EBrowser window hang. Close and enable it again.

The parameters of light are not saved in dol files

You have to pre-load all the entities in the lvl file before using them with EBrowser.

I have simplified some aspects of EBrowser design. The entity's handling in Blade is complex and it would be difficult to implement a full graphical interface. The objective has been to facilitate the basic work. The designer will have to customize the generated code.

Legal Comments (or something like that):

EBrowser is a script that adds new functions to Rebelact Blade/ Severance game; all rights reserved to Ras for doing this great game.
EBrowser is distributed in Ras forums, Blade Mods,... just for pure fun and thanks to Rebelact´s idea of letting the scripts source code available. If this is a problem for Ras or there are legal conflicts, we stop doing immediately.

CREDITS:

RAS left the code of EntityList and gave me the graphical interface concept.

BirdMadBoy for their quick and useful answers.

Goosebrush by beta testing.

Lerfox13 by their design contributions.

Power Translator for the english version of Leeme.txt. You can send to me a humanized version of this file, if you like.

Thanks to forum members by your patience and comments received: Raziel, KeoH, fremen,...

Masklin

Revisions:

0.1 - Original version.
0.2 - Fixed syntax errors in generated scripts.

```
          Fixed a initialization problem when not entities are
          found at default scan area.
```

Comments by Prospero:

The EntityBrowser is the way to go to accurately place objects. You start it as the game is running, then select an object from the list, create it, use keys to move/orientate and when it is in the right place you can save the script to a .py file. I think the d/l link on BladeResources site is not working at the mo.

Adding objects manually is OK, but you have to guestimate the Position and working out the Orientation is mind-boggling.

Basically, create a file objs.py (the name is not critical, but don't name a file Objects.py or objects.py as this is the name of a source file.) The basic code for obj/enemy creation is:

```
import Bladex # you need this once only at top of file
```

```
o=Bladex.CreateEntity("MyObject1","Espada",x,z,y,"Weapon")
```

This will create a Heavy Sword at coordinates x,z,y. You can get the coords from your *.mp file, but remember to reverse the y axis reading. A positive y reading on the .mp file will need to be positive in the scripts and vice-versa. Also the z (up/down) axis is negative up from zero and positive down. You also need to multiply LED units by 1000

-30,1,45 in LED = -30000,-1000,-45000 in scripts.(you get used to it!)

The first argument "MyObject1" is the individual name of the object and must be unique. The last arg, "Weapon" is the entity class. Most objects are classed "Physic". All characters are Classed "Person"
If you are uncertain, use

```
Reference.ObjType("TypeOfObject") # remember to import Reference.
```

in place of "Physic" or whatever.

This function returns the correct enity class for the TypeOfObject.

An easier way of getting coords is to eneble the Debug mode.

Open file Lib/Reference.py and edit

```
DEBUG_INFO=1
PYTHON_DEBUG=2
```

(Normally these are set to 0)

Then when in the game, press T key to get a readout of player position. If you make a note of the coords you can enter them into your script and the obj/enemy will appear next time you load the map. The Debug mode also enables other keys. Pressing T repeatedly will cycle though other stuff which may be useful later. The 'cheats' keys will be activated (K,P,F10). Also the numpad keys will teleport you if you are playing a RAS map.
(If you are playing a map that uses the Talk System, The T key will not work this way in a 'live' game. It will work in a saved game, but the Talk System will not.)

btw. The objs.py must be executed in cfg.py

```
execfile("objs.py)
```

All the correct entity names are in the Objects Catalogue. (Same place as EBrowser, maybe link not working .)

Most of the names are in Spanish, but this is not really a great problem.

Espada = Sword
Escudo = Sheild
Arco = Bow
Flecha = Arrow
Carcaj = Quiver
Lanza = Spear
Martillo = Hammer
Maza = Mace
etc

Other attributes can be set after the creation code:

```
o.Scale=1.0
o.Orientation= (0,0,0,0)
```

If you omit these the objevt is assigned default values.

Some objects need some extra code to work correctly. E.g potions, breakable objects, etc. Check the game files.

Enemies are slightly different. They don't have an Orientation value.
Instead they have Angle, which just sets the direction they are looking at the start. 0 is North on the map, going counter-clockwise to 6, which is back to North. You can work out the rest.

To enable their AI add:

```
import EnemyTypes
```

```
EnemyTypes.EnemyDefaultFuncs(o.Name)
```

..after creation code.

Python is very unforgiving on syntax errors. If it finds an error when loading a map it will stop there and not exec any other code past that point. The map will still load, but things will be missing.
If you have the console enabled,(inst on BladeMods site) it will print out the error, which sometimes tells you exctly what is wrong and sometimes just gives you a vague clue.

Check some of the game files to see how things work. They were written by a number of programmers so styles will vary, but you can learn a lot.