(32)WSAG – ClientMod - Player Selection (BV 2.44 PV 2.5.1) (LU 21.3.08)
(33)WSAG – ClientMod –Player Name/Dynamic text input Display (BV 2.44 PV 2.5.1) (LU 21.1.08)
(34)WSAG – ClientMod –IP Input /Connect to Server / Dynamic Info Text(BV 2.44 PV 2.5.1) (LU 20.2.08)
(35)WSAG – ClientMod –Player Name already in use on Server(BV 2.44 PV 2.5.1) (LU 20.2.08)
(36)WSAG – ClientMod/ServerMod –Lists and Dictionaries(BV 2.44 PV 2.5.1) (LU 26.1.08)
(37)WSAG – ServerMod/ClientMod–Register to Player list (BV 2.44 PV 2.5.1) (LU 20.2.08)
(38)WSAG –ClientMod– Send Orientation/Position (BV 2.44 PV 2.5.1) (LU 26.2.08)
(39)WSAG –ServerMod– Receive/Send Orientation/Position of Client' (BV 2.44 PV 2.5.1) (LU 26.2.08)
(40)WSAG –ClientMod– Receive Client's Orientation/Position from Server (BV 2.44 PV 2.5.1) (LU 26.2.08)
(41)WSAG –Restart Gameplay (BV 2.44 PV 2.5.1) (LU 26.2.08)
(42)WSAG –Transmitting Shoot Signal between Clients (BV 2.44 PV 2.5.1) (LU 27.3.08)
(43)WSAG –Counting GamePoint(GP) (BV 2.44 PV 2.5.1) (LU 27.3.08)
(44)WSAG –Transmitting GamePoints(GP)between Clients (BV 2.44 PV 2.5.1) (LU 30.3.08)
(45)WSAG – Disconnect Clients(BV 2.44 PV 2.5.1) (LU 18.4.08)
(46)WSAG – Disconnect (Cosmetic lift up) (BV 2.46 PV 2.5.1) (LU 3.6.08)
(47)WSAG – And here comes the Mouse(BV 2.46 PV 2.5.1) (LU 1.7.08)

Possibilities how you can support this Project can be found here:

Notes (LU 6.4.08)

LU = **L**ast **U**pdate
BV = **B**lender **V**ersion (Make sure your using the same Blender Version…)
PV = **P**ython **V**ersion
Make Sure your Document View is set to Weblayout to get the original Layout for this Page.
XXXX = Description will be inserted here later.

Link List (LU 3.2.08)

Blender download:
http://www.blender.org/download/get-blender/
Blender Manual:
http://wiki.blender.org/
Python download:
http://www.python.org/download/
Python Manual:
http://www.python.org/doc/
Basic Network by Ash :
http://www.ash.webstranka.info/multiplayer_eng.htm
Text Input:
http://www.blending-online.co.uk/8501/17001.html
http://www.blendenzo.com/tutInputText.html

Tutorials and Game building Guide:
http://bgetutorials.wordpress.com/

Introduction (LU 23.10.07)
This Tutorial will give you a Step by Step Guide on how to make an Online Multiplayer Game. Even if this is the first time you use Blender, this Tutorial will guide you to a solid Foundation in building a Game. If you already are an Advanced Blender Guru but would like to get some Information on how to make an Online Multiplayer Game then you are at the right place here to…. And maybe you can help to improve this Project in the sections that you are more advanced then this WSAGbasicSetup currently is…
For we are continually moving on in development of the WSAGbasicSetup, This Guide will be updated on a regular base. In the Title of every Section you can see when the last update was and with what Version of Blender it was made.
This Project depends on the Help of everybody… So if you have an Idea on how to improve either the .blend file ore this Tutorial, please let me know. Also let me know if there are any mistakes in this Tutorial or if something just won't work…
Hope the WSAGbasicSetup will help a lot of People

Copyright/Credits (LU 6.3.08)
Copyright
You can use all of the Information in this Tutorial and the .Blend Files on my Homepage for what ever you want to use it, even commercial use!
But it would be nice if you would send me an Email with you're Work and add the following Credits to your Program:

Credits:

Old Jim - Project Leader - (www.wsag.ch.vu)

Fox64 - Help and Inspiration with Python Programming
Littlebob - Inspiration on Python programming
Ash - Inspiration on Network setup - http://ash.webpark.sk/Default_eng.htm
IzE Design - Inspiration on Network setup - http://www.izedesign.it/
The Mighty Owl - Tips on Programming - http://www.themightyowl.com/
Blendenzo - Tutorial Text input - http://www.blendenzo.com/tutInputText.html
JiBe - Help with Network programming/Setups - www.bzooworld.org
Richard Perkins – Grammatical and spelling help on the Tutorial - http://bgetutorials.wordpress.com
whitetiger – IP Script -http://snippets.dzone.com

How to Support This Project (LU 12.3.08)
You can Support this Project by:

-Sending me Models:
As a Character I will need a Mobile Office... Like a Small Go-Cart or a Water-Scooter with a build in Office... It should look like a Person sitting on in an Office chair in front of his Desktop and on that Desktop there is a Computer.
And all this is inside an open Scooter that can drive around, not on Wheels but on a Nitro Engine just floating about 1cm above the surface … Can you image what I'm talking about if

not please ask?
The Measurements have to be:

- 1Blender Unit = 1 Meter
- The Mobile Office should have the dimensions of
- high 1.34 meters
- length 0.95 meters
- with 0.75 meters

-Sending me Scenarios:
I would like Scenarios form places that really exist… My first scenario will be the Workshop of my Grandfather were I used to play as a Child (read more about it on my Webpage Games Gerlin and WSAG) So, if you have a scenario that I can add to my Game send It to me together with the (Google Earth) Coordinates of the Upper right and the lower left Corner.
Ideas of Scenarios:

- National Parks
- City Parts
- Your Home Town
- High Ways
- Stations
- Airports
- Racing Areas
- Swimming Pools
- What ever you can think of….

Tutorial
Read the Tutorial and let me know what I do well and what I could do better…
If you find: Mistakes, wrong spelling or a Way how to write something better, rewrite that part of the Tutorial and send it to me.
If you have pictures that are better then mine Place them in to the Tutorial and send it to me.
Please mark all the Changes you made in the Tutorial Red and underlined this way I can see the changes faster.

Basic Development:
If you know a way to improve ore future develop my WSAGdev.blend file, make the Changes and send the changed .blend File to me together with a description on what, why and how you change it…Something like a mini Tutorial.

Donations:
On bottom of the www.wsag.ch.vu Page you can find a Donations Button where you can donate Money to this Project. The more money comes in the faster we can develop this Project

Credits:
Of Course I will add your Name to the Credit List together with a short description on how you supported this Project.

Note:
Please keep in mind that the WSAG Project is open to everybody what means that everybody can use and change everything in this Project for his own use…
Of course it would be great if they would give Credit…

Download and Install Blender and Python (LU 23.10.07)

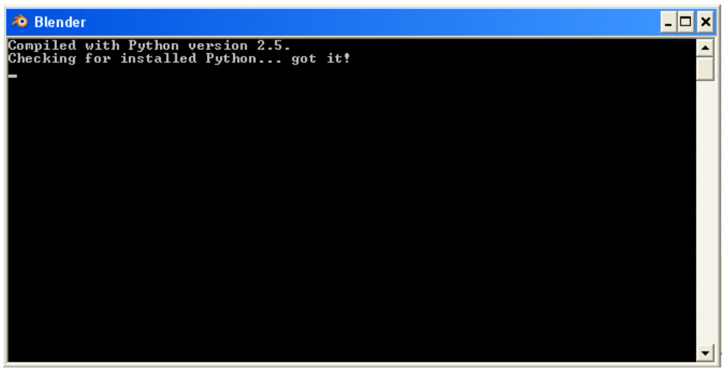
First of all you will need some Software.

Blender (GPL licence free Download at: http://www.blender.org/download/get-blender/)
Double Click the downloaded file and Blender will be installed. In the Folder of the installed Blender Program (Standard C:\Programme\Blender Foundation\Blender) you will find 2 Files I recommend that you read. (‘’Blender.html’’ and ‘’BlenderQuickStart.pdf’’) Also read the Manual at: http://wiki.blender.org/

Python (Open Source licence free Download at: http://www.python.org/download/)
Blender has some basic Python functions build into it – But for the more complex functions that we will need for the Network setup you will need the full version of Python.
Double Click the downloaded file and Python will be installed. In the Folder of the installed Python Program (Standard C:\Python25\Doc\) you will the file ‘’Python25.chm’’ that I recommend you read.
Also read the Manual at: http://www.python.org/doc/

(1)WSAG – First Start (BV 2.44 PV 2.5.1) (LU 23.10.07)

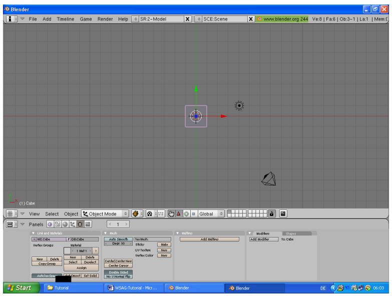
When you start up Blender 2 Windows will open. The first one is your Consol Window and it should look like this:



```
Compiled with Python version 2.5.
Checking for installed Python... got it!
_
```

Make Sure that the Text in this window is:
Compiled with Python version **2.5**
Checking for installed Python… got it!
This shows that it has found your installed Python, and that is imported because if it dos not find you installed full version of Python some of the Network function won’t work…

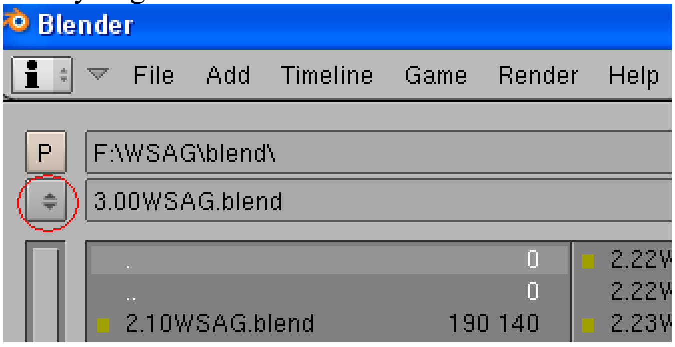
The Second Window is the actual Bender Working window and at its first Stat up it looks like this:



The 3 Objects that you see are a cube, a light and a camera.
Click on one of them and press Delete. A Question will come up: Erase Selected Object(s)?
Click Ok. Do the same Thing with the other 2 Objects.
Now we have an Empty 3 D Window.

<u>(2)WSAG – Save</u> (BV 2.44 PV 2.5.1) (LU 23.10.07)

Now its time to save or Project the first Time:
There are tow ways to save the first is the classic way go to Files and in the Dropdown menu select Save.
The Second way is much quicker just press the Ctrl-Key and W-Key at the same time.
Now you get to a menu that looks like this:



F:\WSAG\blend\
3.00WSAG.blend

.	0
..	0
2.10WSAG.blend	190 140

2.22W
2.22W
2.23W

You see where I made the red mark? Click on that Symbol to select the Path were you want to save your file.
Now in the second line you write Your File Name. Name it 3.00WSAG. Click on the Save as Button in the right upper Corner.
Its 3.00 because: This is the 3 Version of WSAG and the .00 is because this will allow us to easy save subversion. If you save the next time you just do this:
Press F press the +-Key and your same name will change to 3.01WSAG.
Click on the Save as Button in the right upper Corner.
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

<u>(3)WSAG – Program Setup Tree</u> (BV 2.44 PV 2.5.1) (LU 11.11.07)

Here is a Diagram of how the WSAGbasicSetup is set up.

```
                        Intro
                          ¦
                          ¦
----------------------    Start Up Menu----------------
¦                         ¦                                ¦
¦                         ¦                                ¦
Server                    Explore                          Client
¦                         ¦                                ¦
¦                         ¦                                ¦
```

```
Location          Player Selection          IPInput
|                 |                         |
|                 Loctation                 Player Selection
|                 |                         |
|                 |                         PlayerNameImput
|                 |                         |
--------------------------------------------
                  |
                  Gameplay
                  |
                  Location
```

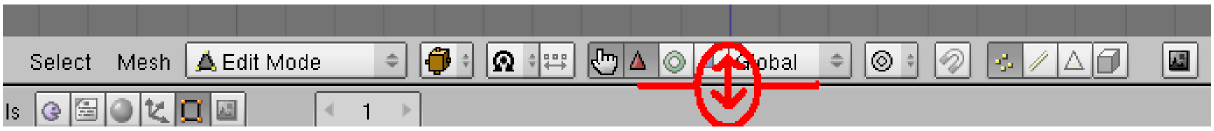
Keep this Diagram in Mind it will help you understand how different things will work together.

<u>(4)WSAG – Intro / Add Object</u> (BV 2.44 PV 2.5.1) (LU 23.10.07)

Press the 1-Key in your Numpad wile your mouse is in the 3D window. This will change the Perspective from top View to Front View. If you don’t have a Numpad change the View by clicking on the View Menu it the bottom left corner and select Front.
Add a Plane by:
Wile Mouse is over the 3D Window press Space now select Add => Mesh => Plane.

<u>(5)WSAG – Intro / Split Screen</u> (BV 2.44 PV 2.5.1) (LU 23.10.07)

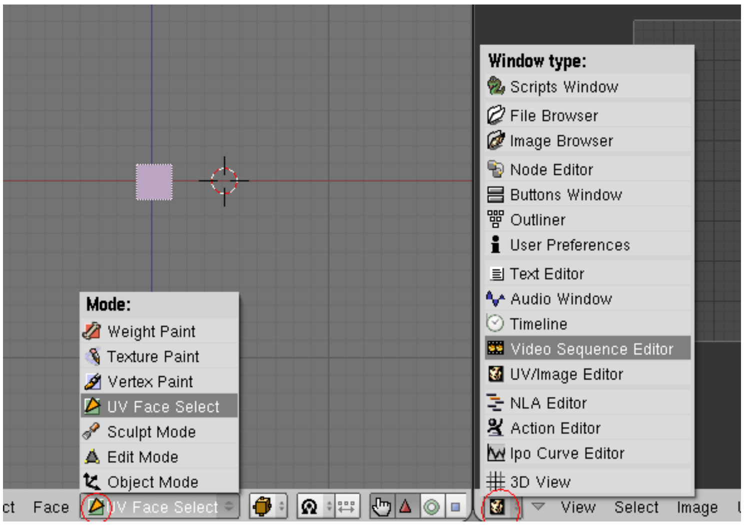
Now you Split the screen by moving the mouse right in to the marked line in the Screenshot below:



Your mouse courser will change to the same Symbol as you can see inside of the Red circle. Now press The Right Mouse Button and select split Area in the Popup Menu. Now you will see a line that you can move with your mouse center the line in the middle of the Screen and press the Right Mouse Button. You now have 2 3D Screens showing the same thing.

<u>(6)WSAG – Intro / Text UV Mapping</u> (BV 2.44 PV 2.5.1) (LU 24.10.07)

Now change left and right window as shown on the picture:

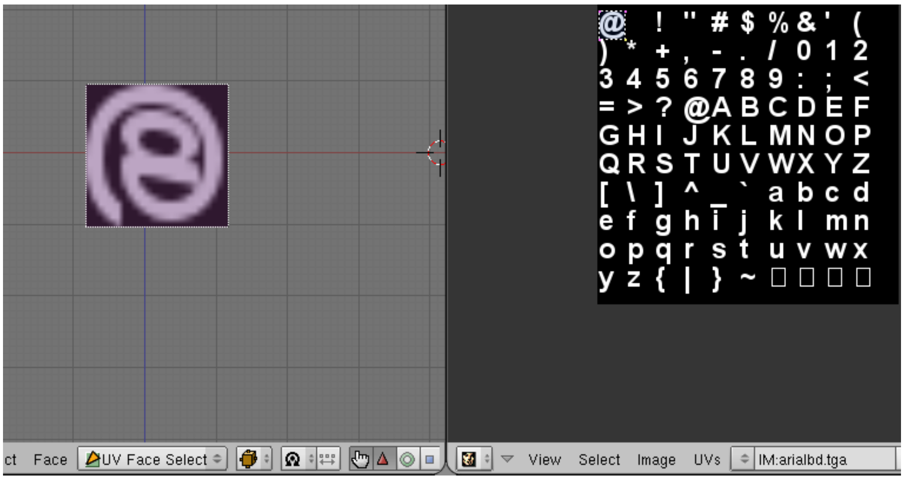


With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.01WSAG.
Click on the Save as Button in the right upper Corner.

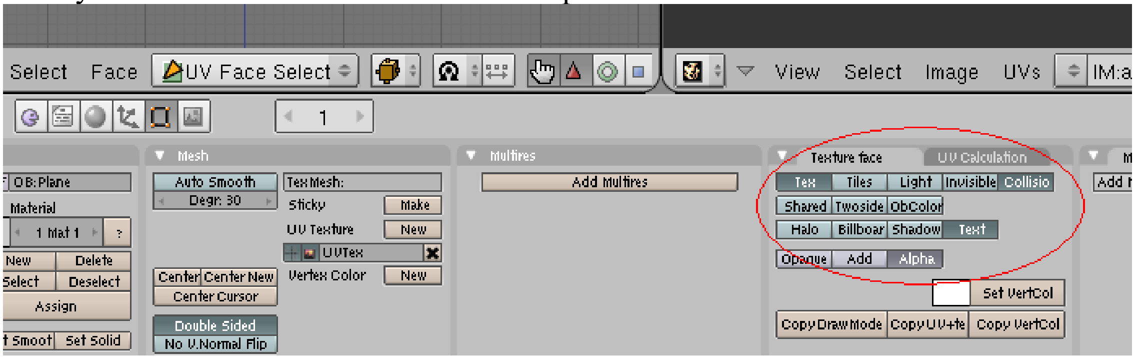
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

Now we need an Font Image you can download it form this Webpage.:
http://www.blending-online.co.uk/8501/17001.html
The direct download link is:
http://www.blending-online.co.uk/arialbd.tga
Save this File in the Folder of your Blender File.

In the Right window click on Image => open and Navigate to your Image (arialbd.tga) and open it.
Now you can see the Alphabet layout in the Right Window and in the Left window printed on the Plane.
In the Right Window the Alphabet layout is surrounded by a dotted line and in every corner you see a purple dot.
Click on one of the Purple dots with your right-mouse-button and the dot will turn yellow.
Now press the G-Key and Move the Yellow Dot Press the left-mouse-button to select the dot to wished place.
Move all the Dots the same way until the Surround the @ Symbole.
Your Screen should look like this now:

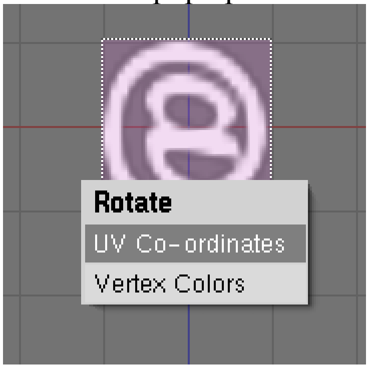


With the Mouse over the Bottom Window press
With your mouse over the bottom Window press F9.



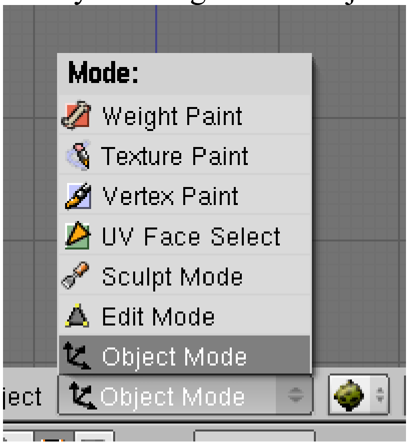
Make sure that your selections look the same as inside the Red circle in the Image above.

With the Mouse over the Left 3D Window press R-Key.
A Window pop up Menu will come up.

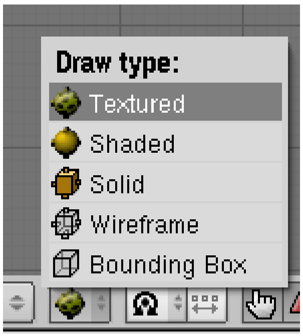


Click on the UV Co-ordinates, this will rotate The Image. Do this again until the @ is rotated the right way.

Now you change to the Object Mode.

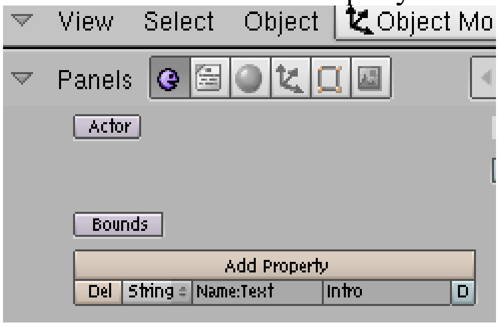


Change the View port to Textured



Press G-Key and Center the Plane in to middle of the Screen. Press Right Mouse Button to let go of the Plane.

With your mouse over the bottom Window press F4.
Now Click on the Add Property and Change the settings so that they look like this:

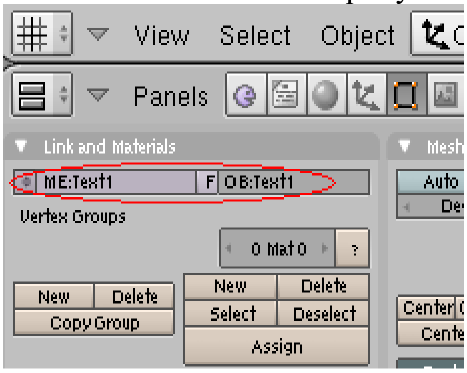


String Name:Text Intro D

Change to String, Name = Text (not text), there where I wrote Intro you can write what ever Text you want for your Intro.

(7)WSAG – Intro / Change Object name (BV 2.44 PV 2.5.1) (LU 28.10.07)

With your mouse over the bottom Window press F9.
Now Click on the Add Property and Change the settings so that they look like this:



ME:Text1 F OB:Text1

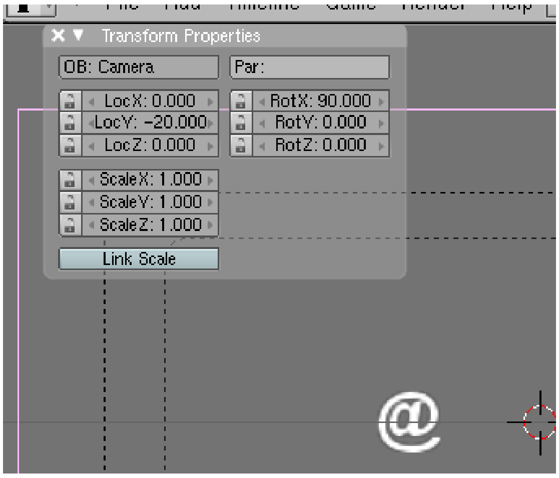
ME: Text and OB: Text
ME = Material OB = Object Name

(8)WSAG – Intro / Transform Objects (BV 2.44 PV 2.5.1) (LU 24.10.07)

With your Mouse over the Left 3D Window press Space and in the Pop up Menu select Add => Camera
Now:

Press the 0-Key in your Numpad wile your mouse is in the 3D window. This will change the Perspective from top View to Front View. If you don't have a Numpad change the View by clicking on the View Menu it the bottom left corner and select Camera.

Click on the Camera with a Right mouse click, press N-Key a in the Pop up menu Change the Values like this:



Transform Properties

OB: Camera

Par:

LocX: 0.000

LocY: -20.000

LocZ: 0.000

RotX: 90.000

RotY: 0.000

RotZ: 0.000

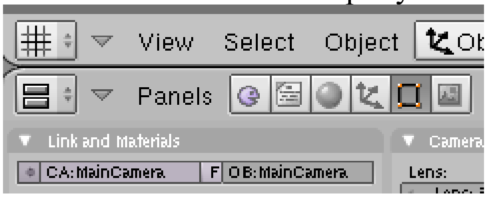
ScaleX: 1.000

ScaleY: 1.000

ScaleZ: 1.000

Loc X = 0.0 // LocY= -20.0 // LocZ = 0.0
This will put your camera into the right position.
Now close the Transform Properties Pop up menu by clicking on the x in the left upper corner.
With your mouse over the bottom Window press F9

Now Click on the Add Property and Change the settings so that they look like this:



CA:MainCamera

F

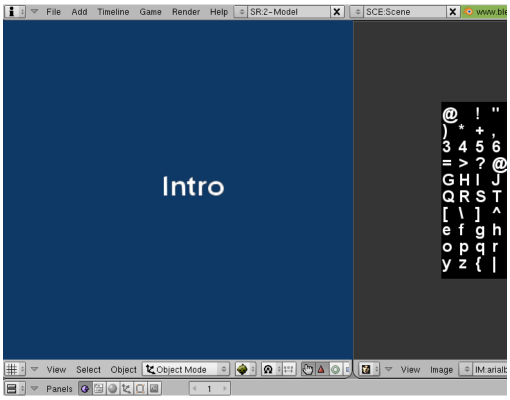
OB:MainCamera

ME: MainCamera and OB: MainCamera
ME = Material OB = Object Name
Now its time, too save again.

With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.02WSAG.
Click on the Save as Button in the right upper Corner.
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(9)WSAG – Intro / Start Game Engines (BV 2.44 PV 2.5.1) (LU 24.10.07)

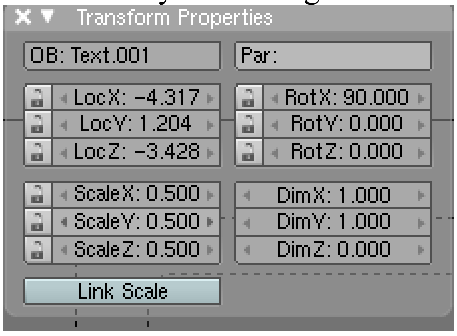
Now we can look at the resultant:
With the Mouse over the 3D Window press the P-Key this will start the Game engine.
You now should get a picture that looks like this:



Press Esc-Key to stop the Engine.

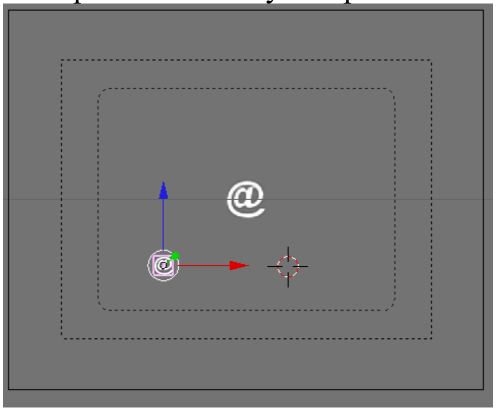
(10)WSAG – Intro / Copy/Scale Object (BV 2.44 PV2.5.1) (LU 16.11.07)

Left Mouse Click on the Text1 Object so that it’s surrounded by a purple line (That is the sign that it’s selected.)
Now press the Shift-Key and D-Key at the same time, this will duplicate the Object.
Press N- Key and Change the Scale x y z Value to 0.5 like the picture below shows



OB: Text.001
Par:
LocX: -4.317
LocY: 1.204
LocZ: -3.428
RotX: 90.000
RotY: 0.000
RotZ: 0.000
ScaleX: 0.500
ScaleY: 0.500
ScaleZ: 0.500
DimX: 1.000
DimY: 1.000
DimZ: 0.000

Now press the G-Key and place the new Object like you can se on the picture below.

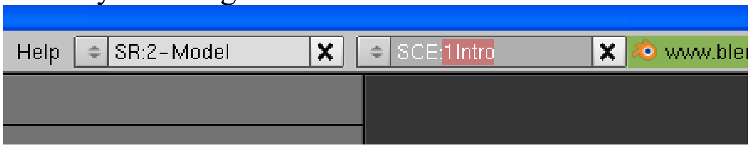


The new created Object will have the Name Text.001.
Now with your mouse over the bottom window, press F4.
In The Properties were at the moment it says Intro, change it to: “To Skip Intro press Space”.
Like always without the “”.

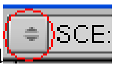
(11)WSAG – Add a Scene (BV 2.44 PV 2.5.1) (LU 4.11.07)

To Skip the Intro we need a Scene to Skip to.

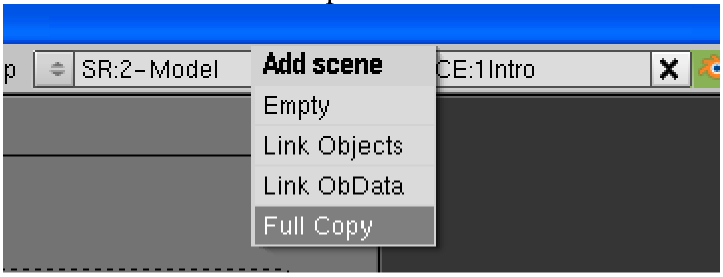
In the Top header you find the Popup Menu SCE: And the first Scene is Standard called Scene. Change it to 1Intro. We call it 1Intro and not just Intro because this way the Scenes will stay in the right order. It should look like this now:



Now we want to add a new scene by clicking on the left side of the SCE Popup menu and clicking on ADD NEW.



Now a menu will come up that looks like this:



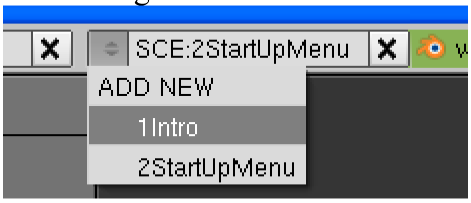
We will use the Full Copy. It will give us a new scene with all the objects of your current scene. That way we won't have to rebuild the Text plane….
So click in Full Copy and name the new Scene 2StartUpMenu

With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.03WSAG.
Click on the Save as Button in the right upper Corner.
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

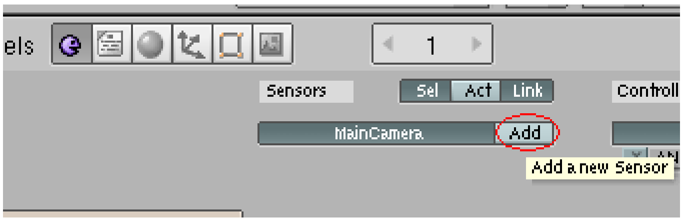
(12)WSAG – Interact with the Engine / Change Scenes – Logic Bricks (BV 2.44 PV 2.5.1)

(LU 16.11.07)
What makes an Engine interesting is that you can interact with it – You can influence what happens when.
We now will interact with the Engine so that it skips the Intro when you press Space.
For this time I will show the pure Logic Bricks option. There is also a option where You use Python that I will show later.
The Interaction is Build up with Logic Bricks that you can combine with each other. The Main Logic Bricks are Sensor Controller and Actuator. Each of this Logic Bricks has sub functions.
Here is how to set the Logic Bricks up so that the Scene Changes(skip Intro) when you press Space.

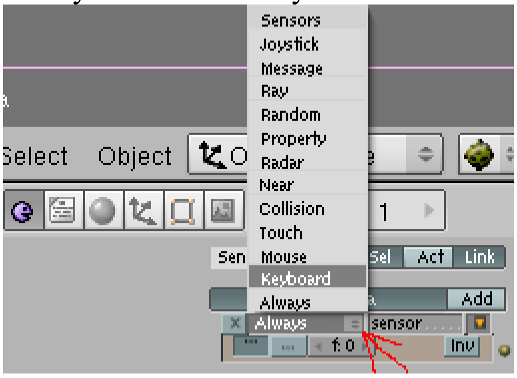
Now change back to the 1Intro scene.



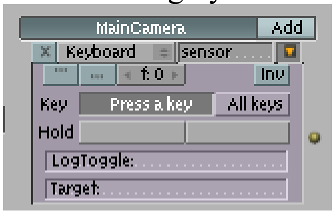
Click on the Camera so that it changes Purple.
With the Mouse over the 3D Screen press F4.
In the bottom window you click on Add button in the Sensor section.



Now you click on there were the red arrow points at in the Picture below and change the Always sensor to a Keyboard Sensor.

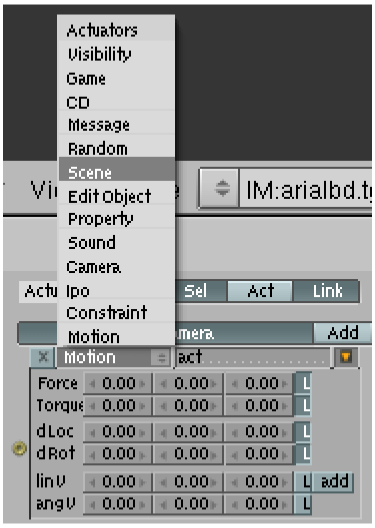


You remember we wrote that you can press the Space Key to skip the Intro?
That's so we will set the Keyboard sensor so that it reacts if you press the Space Key.
Click on the grey button next to were it says Key. Now it should look this:

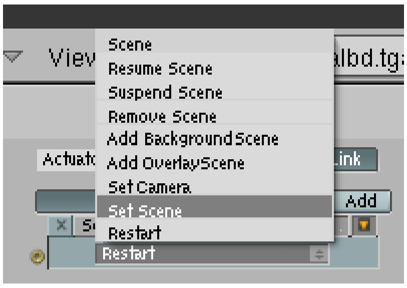


Now press the Space Key and Voila it's done. The Sensor is set to react if you press the Space Key.
So the next step is to tell the Engine what to do if you press the Space Key.

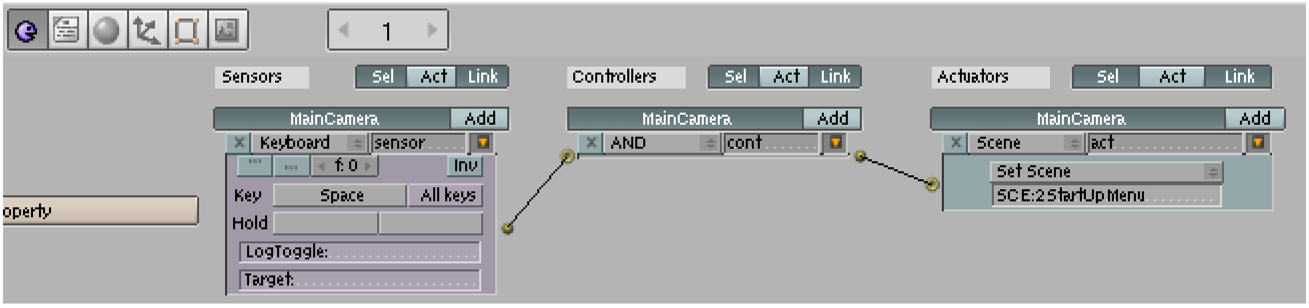
Next you add a Controller and an Actuator. The Same way we added the Keyboard Sensor.
The Controller we leave as it is. The Actuator we change to Scene.



The Button restart we change to Set Scene

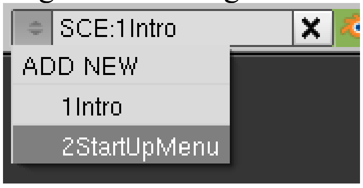


And in the Box where it says SCE we write the Scene we what to change to in this chase 2StartUpMenu
Now click on the yellow dot next to the Sensor menu, keep the Left-Mouse-button pressed and pull a line to the yellow dot on the left side of the Controllers Menu let the Mouse Button go. Do the same with the dot right to the Controllers Menu and the one left to the Actuators Menu. It then should look like this.



With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.04WSAG.
Click on the Save as Button in the right upper Corner.
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

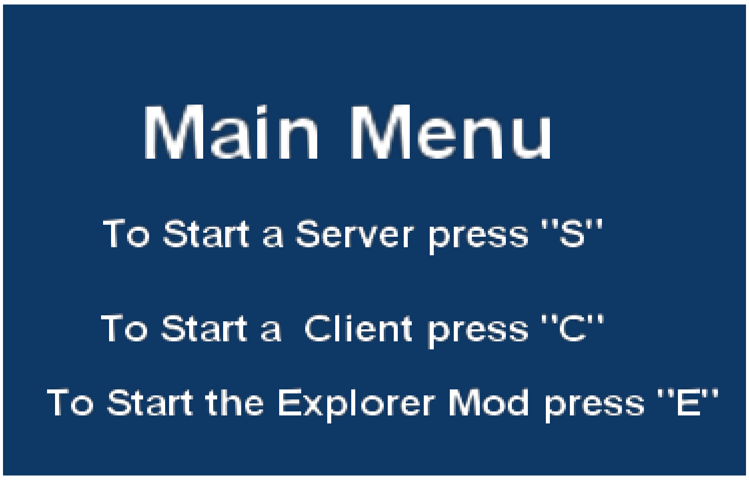
Now if you Start the Engine by Pressing the P-Key with the Mouse over the 3D window and then press the Space-Key it will Change the Scene.
But because both scenes look the same you wont notice it. So Press the ESC-Key to stop the Engine and Change to the 2StarUpMenu Scene.



(13)WSAG – Startup Menu / Logic Brick Setup to use Python(BV 2.44 PV 2.5.1) (LU 24.11.07)

First Thing we will do here is to Display the menu Information's. For I have already have shown how to Display Text in the Section **WSAG – Intro / Text UV Mapping** I will only Give you a Screenshot of how the End product should look like. To duplicate the Text Panes read the **WSAG – Intro / Copy/Scale Object** section.

So here is the Screen Shoot:
Wile Engine is running



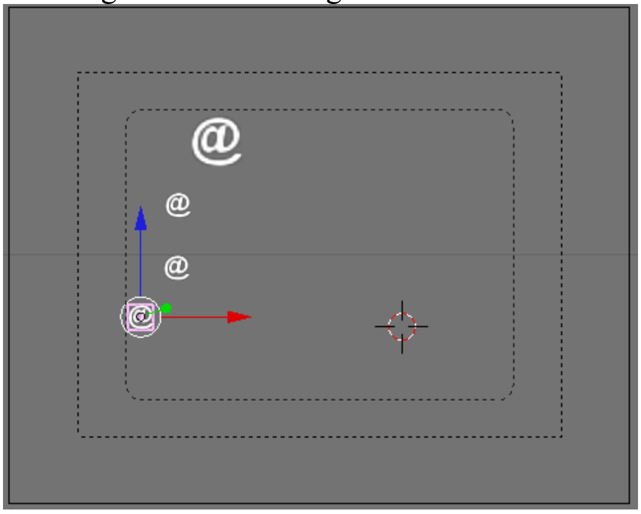
Main Menu

To Start a Server press "S"

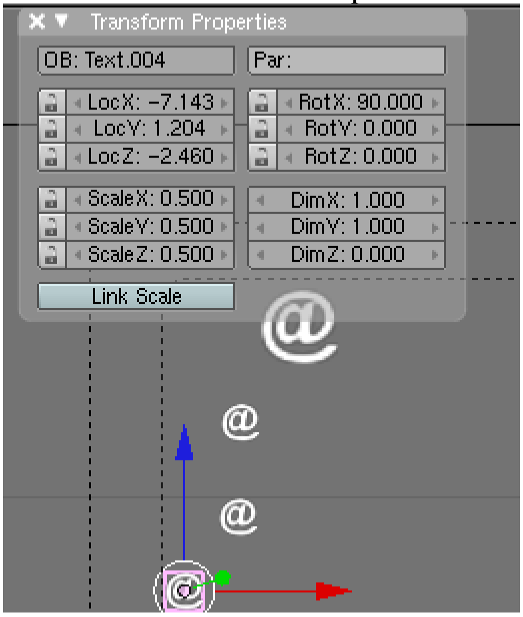
To Start a Client press "C"

To Start the Explorer Mod press "E"

Wile Engine is not running



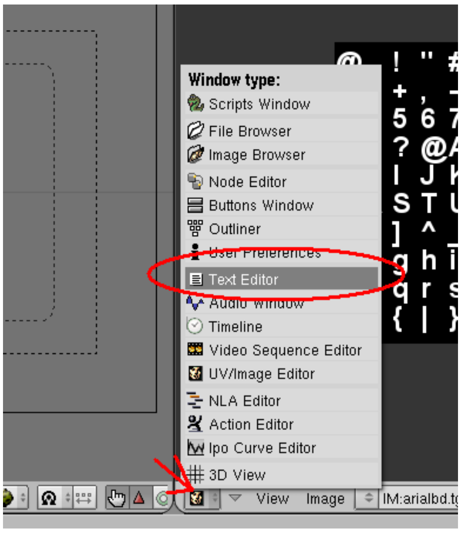
Note that the Lower 3 Textpanes were Scaled down to 0.05



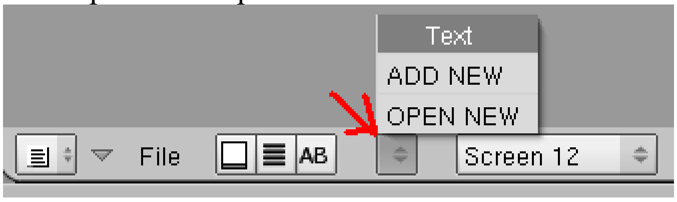
OB: Text.004
Par:
LocX: -7.143
LocY: 1.204
LocZ: -2.460
RotX: 90.000
RotY: 0.000
RotZ: 0.000
ScaleX: 0.500
ScaleY: 0.500
ScaleZ: 0.500
DimX: 1.000
DimY: 1.000
DimZ: 0.000

For more detail an how to scale Objects read the section **WSAG – Intro / Copy/Scale Object**

No change the Right window to Text Editor:

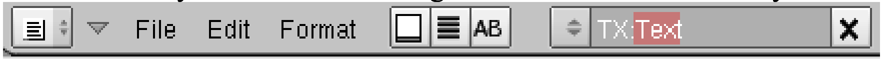


Now open the script file TEXT.



And rename it to MainMenu.py

You do this by left mouse clicking in to the field where it says TEXT now.



And then write MainMenu.py into that field and press Enter.

No select the camera in the left side of your window with a right mouse click and press F4.

In the bottom Part of your Window you now have Logic setup for your selected camera.

Add a Sensor, a Controller and an Actuator and connect them . How too do this you can read in the section **WSAG – Interact with the Engine / Change Scenes – Logic Bricks**.

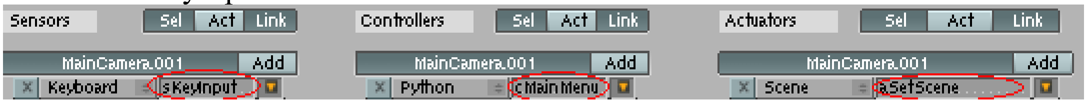
Now change the Sensor to Keyboard and select all Keys

The Controller we change to Python and in the field Scripts we write the name of the Text file we just created that would be MainMenu.py

The Actuator we set to scene and change the Button restart to Set Scene, but leave the SCE: Field empty.

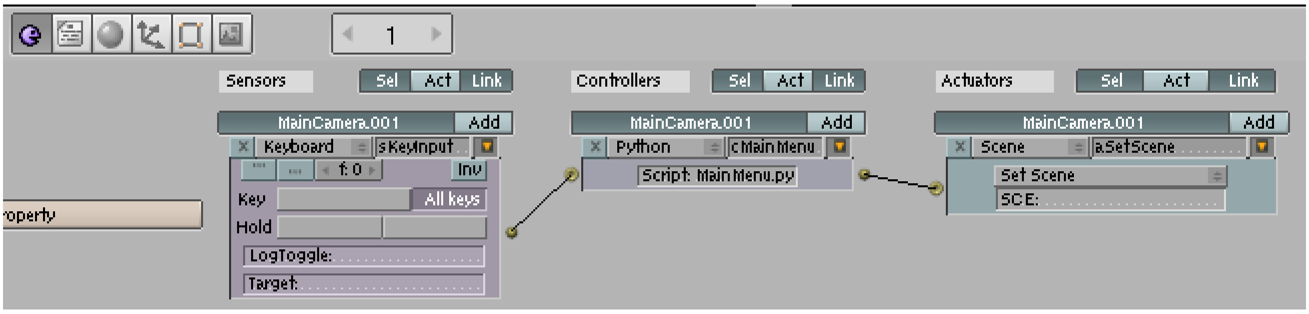
Now rename The Sensor, Controller and Actor by clicking in to the fields that I have circled Red.

Sensor = sKeyInput // Controller = cMainMenu // Actuator = aSetScene



The Small letter in front for the Names stand for s=Sensor c=Controller a=Actuator, this will help us to easy define what kind of Logic Brick we are using when we Program in the Python Text Window.

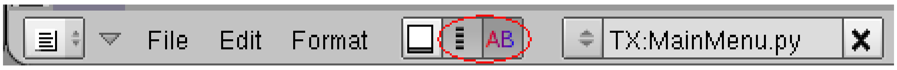
Now your Logic Brick setup should look like this:



Now the Logic Brick setup will start the MainMenu.py script every time you press a Key and again when you let the Key go.
So we can start to write in to the MainMenu.py script what it should do when you press/let go a Key.

<u>**(14)WSAG – Startup Menu / Read out Key codes for Python**</u>(BV 2.44 PV 2.5.1) (LU 28.3.08)

Before we start make sure that the to Buttons that I have Marked in the Red circle below are pressed… This Will add the Line Number on the side and highlight the Python commands with color.



So now click in to the Text Window on your right so that you can start writing into it.
Now the first thing you always have to do is to import the Libraries that you will need for your Script.
The Main Library that you almost always if not always use is GameLogic.
To import this Library you simple type:

```
from GameLogic import*
```

The * tells the Engine to import everything from the GameLogic Library, this way we don't have to bother anymore what exactly we want to use from this Library… If you don't want to import all of the Library you can just type `import GameLogic`. But then you will always have to reference to the Library by typing `GameLogic.???` Instead of the `???` you would Type the Actual Command… If you use the `from GameLogic import*` Command you will only have to the Actual command for example: `getCurrentController()` instead of `GameLogic.getCurrentController()`

The first thing we will need to know is what Code the different Key inputs send to the Python Script.

We dot this with this simple little script:

```
Cont = getCurrentController()
KeySens = Cont.getSensor("sKeyInput")
Keyinput = KeySens.getPressedKeys()
print Keyinput
```

So now what will this Code line do?

Cont = getCurrentController()
This will tell the script that Cont will call up the command getCurrentController()
This helps us that we only have to type Cont if we want to call up the command getCurrentController() instead of typing out the howl command every time… This will not only speed things up a bit but also help us to keep a nice and easy to overlook structure of the Script.

The command = getCurrentController() this will get (import) the CurrentController in to the Script and allows us to take action on the Logic Bricks (Sensor, Controller and Actuator) that we have connected to the Python Logic Brick (the yellow dots mentioned in **WSAG – Interact with the Engine / Change Scenes – Logic Bricks**)

KeySens = Cont.getSensor("sKeyInput")
This line of code will tell the Engine that the Variable KeySens will call up the command Cont.getSensor("sKeyInput")
The Command Cont.getSensor("sKeyInput") starts with the Variable Cont that we have created in the Line above.
The second Part of the Command is getSensor this tells the Script that from he Logic Brick that the Cont Command imported in to the Script it should get the Sensor.
The last Part of the Script is ("sKeyInput") this is simply the name of the Sensor that the Script (Engine) should get (import).

After understanding the first tow lines in the Script its not hard to figure out this next line:
Keyinput = KeySens.getPressedKeys()
This line of code will tell the Engine that the Variable Keyinput will call up the command = KeySens.getPressedKeys()
The Command = KeySens.getPressedKeys() starts with the Variable KeySens that we have created in the Line above.
The second Part of the Command is getPressedKeys() this tells the Script that to get the Code that is send when a Key is Pressed.

Last Line in this Small script is:
print Keyinput
This Simply Prints out the Code the command getPressedKeys() gets when a Key is Pressed.
By the way always make sure that the you write the script just as I do because for example **T**ext ist not the same as **t**ext and **P**rint will do something else then **p**rint.

So now its Time To save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.05WSAG.
Click on the Save as Button in the right upper Corner.
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

So we will start the Engine and see what this Script will do for us.
With the Mouse over the 3D Window (Window your Left), press the P-Key.
Now press the E-Key and let it go again (This is the Key we want to start the Explorer Mode with….)

When you look in to the Consol window now (That's by the way the second Window that Starts when you start Blender the Window with the Black background and Withe Text)
You can se that the numbers [[101; 1]] and [[101,3]] where Printed the 101 Stands For the E-Key the ;1 Stands for Press Key and the ;3 Stand for let Key go…

Here is a code List of the Main Keys
0 = 48 a = 97 k = 107 u = 117 Enter = 13
Up-Arrow = 146
1 = 49 b = 98 l = 108 v = 118 Left Shift = 129
Down-Arrow = 144
2 = 50 c = 99 m = 109 w = 119 Right Shift = 128
Left-Arrow = 143
3 = 51 d = 100 n = 110 x = 120 Left Ctrl = 124
Right-Arrow = 145
4 = 52 e = 101 o = 111 y = 121 Right Ctrl = 127
Num Minus = 159
5 = 53 f = 102 p = 112 z = 122 Left Alt = 125
Num Plus = 161
6 = 54 g = 103 q = 113 Right Alt = [[124;1],[126;1]]
Page Up = 177
7 = 55 h = 104 r = 114 Space = 32
PageDown = 178
8 = 56 I = 105s = 115 Backspace = 133
9 = 57 j = 106 t = 116

So now that we know what the different Key-Inputs will send for a code we can delete the last file of the Script (print Keyinput), we can define what will happen when the scripts receives different Codes.

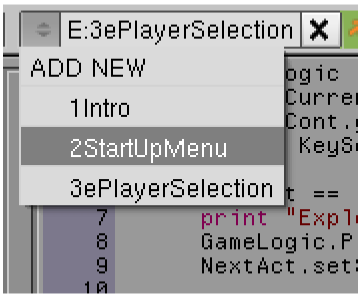
(15)WSAG – Add PlayerSelection Scene –Explorer Mod (BV 2.44 PV 2.5.1) (LU 24.11.07)

Before we can use a script to change scenes we need to create the Scene that it should switch to.
So add a scene with the name 3ePlayerSelection… How to do that is described in the section **(11)WSAG – Add a Scene**

By the Way the number 3 in the name helps to keep the scenes in a Logical Order… The are Ordered by Name… And if you get a lot of scenes having them in a Number order will help us find the right scene faster…
The e in the Name tells us that this Scene belongs to the explorer Mod.

Now in the 3D Window we will change the Text so that we see a difference between the two scenes
The Text **Main Menu** we change to **Select Player**
The Text **To Start a Server Press "S"** we change to **For Blue Player press "B"**
The Text **To Start a Client Press "C"** we change to **For Red Player press "R"**
The Text **To Start the Explorer Mod Press "E"** we change to **For Green Player press "G"**

Now switch back to the Scene 2StartUpMenu



(16)WSAG – Startup Menu / Change Scenes - Python(BV 2.44 PV 2.5.1) (LU 6.4.08)

Now we have everything setup and can add this little bit of code.
Please Note that you cant just copy this Code!!!
In this Code if there are 2 “=” I made a space between them so you can better see that there are 2 “=” and not your one “=” for 2”=” with out Spaces looks like this “==” and whit spaces it looks like “==” <= Much easyer to see that there 2 right?
But in the Code you will have to write it whitout spaces between the “=”!

```
SetSceneAct = Cont.getActuator("aSetScene")

if Keyinput = = [[101, 1]]:
        print "Explore"
        GameLogic.PlayerMod = "Explore"
        NextAct.setScene("3ePlayerSelection")
addActiveActuator(SetSceneAct,1)
```

The whole Code looks should look like this now

```
from GameLogic import*
Cont = getCurrentController()
KeySens = Cont.getSensor("sKeyInput")
Keyinput = KeySens.getPressedKeys()
SetSceneAct = Cont.getActuator("aSetScene")

if Keyinput = = [[101, 1]]:
        print "Explore"
        GameLogic.PlayerMod = "Explore"
        SetSceneAct.setScene("3ePlayerSelection")

addActiveActuator(SetSceneAct,1)
```

Now if Start the Engine (Press P-Key with the mouse over the 3D Window) and press the E-Key it will change the Scene to the Player Selections Menu of the Explorer Mod.

So what do this Lines of Code do?

```
SetSceneAct = Cont.getActuator("aSetScene")
```

Gets the Actuator that we named aSetScene and saves it in to the variable SetSceneAct

```
if Keyinput = = [[101, 1]]:
```

Tells the Script that if the Code that is send with the Keyinput, is equal with [[101, 1]]: it should do what is written in the lines below that are indented.
Note that different from other Program Languages Python not uses Brackets to tell what the scrip should do after an if command.
Instead it takes the code lines below that are indented.

print "Explore"
This prints the text Explore in to the Consol window. The only thing this really is good for is that we get a nice clean Protocol in the consol window that tells us what was going on in the game.

GameLogic.PlayerMod = "Explore"
This saves the Text Explore in the Variable GameLogic.PlayerMod.
We will need this later to tell scripts if we are running the game in Server, Explorer or Client Mode…

SetSceneAct.setScene("3ePlayerSelection")
The Command SetSceneAct.setScene("3ePlayerSelection") starts with the Variable SetSceneAct that we have defined above.
The second Part of the Command is setScene, it tells that the game should be set to the scene defined in the last part of this command ("3ePlayerSelection").

addActiveActuator(SetSceneAct,1)
This line tells the script if the added Actuator should be executed or not. The 1 at the end of the command tells the script it school activate (execute) this added active Actuator. If it would be set to 0 it would mean that this actuator is shouted of (is passive).

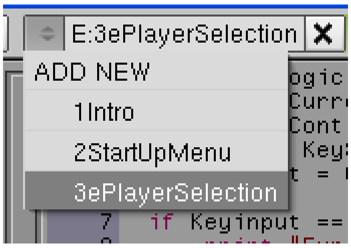
So now its Time To save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.06WSAG.
Click on the Save as Button in the right upper Corner.
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(17)WSAG – Explorer Mod - Player Selection / Bind to GameLogic (BV 2.44 PV 2.5.1) (LU 14.1.08)

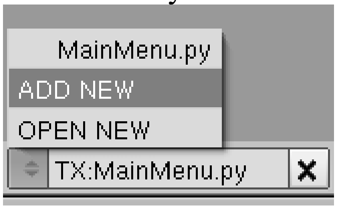
Add a scene with the name 4eLoctionSelection… How to do that is described in the section **(11)WSAG – Add a Scene**

Now in the 3D Window we will change the Text so that we see a difference between the two scenes
The Text **Select Player** we change to **Set Location**
The Text **For Blue Player press "B"** we change to **For Blue World press "B"**
The Text **For Red Player press "R"** we change to **For Red World press "R"**
The Text **For Green Player press "G"** we change to **For Green World press "G"**

Now switch to the Scene 3ePlayerSelecton



Add a new Python Text.



And name it: ePlayerMenu.py

Now select the Camera and with the mouse over the 3D Window press F4
In the Logic Brick Controller you select the field where it says script: MainMenu.py and change it to ePlayerMenu.py

In the Text Window of ePlayerMenu.py you enter this Code:

```
from GameLogic import*
Cont = getCurrentController()
KeySens = Cont.getSensor("sKeyInput")
Keyinput = KeySens.getPressedKeys()
SetSceneAct = Cont.getActuator("aSetScene")


if GameLogic.PlayerMod = = "Explorer":
        if Keyinput = = [[98, 1]]:
                print "Player = Blue"
                GameLogic.PlayerTyp = "ContBlue"
                SetSceneAct.setScene("4eLocationSelection")

        if Keyinput = = [[114, 1]]:
                print "Player = Red"
                GameLogic.PlayerTyp = "ContRed"
                SetSceneAct.setScene("4eLocationSelection")

        if Keyinput = = [[103, 1]]:
                print "Player = Green"
                GameLogic.PlayerTyp = "ContGreen"
                SetSceneAct.setScene("4eLocationSelection")
addActiveActuator(SetSceneAct,1)
```

The first Code Block:

```
from GameLogic import*
Cont = getCurrentController()
KeySens = Cont.getSensor("sKeyInput")
Keyinput = KeySens.getPressedKeys()
```

```
SetSceneAct = Cont.getActuator("aSetScene")
```

Has nothing new to it you can read what this lines do in
(14)WSAG – Startup Menu / Read out Key codes for Python
and
(16)WSAG – Startup Menu / Change Scenes – Python

The Second Block:

```
if Keyinput == [[98, 1]]:
	print "Type = Blue"
	GameLogic.PlayerTyp = "Blue"
	SetSceneAct.setScene("4ePLocationSelection")
```

Is mostly explained in the **(16)WSAG – Startup Menu / Change Scenes – Python**
The Line I want to take a bit closer Look here is

```
if GameLogic.PlayerMod == "Explorer":
```

Tells the Engine to only run this Script if the Variable saved to GameLogic.PlayerMod is Explorer

```
GameLogic.PlayerTyp = "Blue"
```

This saves the Text ContBlue in the Variable GameLogic.PlayerTyp
But why do we use the Gamelogic. In front of PlayerTyp? Would it not be easier to just type PlayerTyp= “ContBlue” ?
Well because at the Beginning of every Script we use this Line

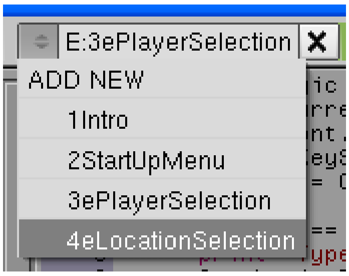
```
from GameLogic import*
```

We have the possibility to Save a Variable with GameLogic. In front of it, that we can use in every Scene or Script… Otherwise the Variable only will be valid inside of the Script that it’s written in…
And because we will need the information that we store in the Variable GameLogic.PlayerTyp not only in a different Script but also in a different Sceene we have to store it with the variable GameLogic. In front of it…

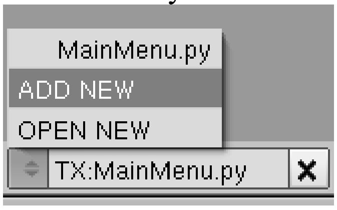
So the rest of the Script is should now be clear again otherwise read this capters again:
(14)WSAG – Startup Menu / Read out Key codes for Python
and
(16)WSAG – Startup Menu / Change Scenes – Python
So now its Time To save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.07WSAG.
Click on the Save as Button in the right upper Corner.
PLEASE DON’T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(18)WSAG – Explorer Mod - Location Selection (BV 2.44 PV 2.5.1) (LU 23.1.08)

So now cange to the scene 4eLocationSelection



Add a new Python Text.



And name it: eLoctionMenu.py

Now select the Camera and with the mouse over the 3D Window press F4
In the Logic Brick Controller you select the field where it says script: ePlayerMenu.py and change it to eLocationMenu.py

In the Text Window of ePlayerMenu.py you enter this Code:

```
from GameLogic import*
Cont = getCurrentController()
KeySens = Cont.getSensor("sKeyInput")
Keyinput = KeySens.getPressedKeys()
SetSceneAct = Cont.getActuator("aSetScene")

if Keyinput == [[98, 1]]:
	print "Location = Blue"
	GameLogic.Location = "Blue"
	GameLogic.LocationC = "6LocationBlue"
	SetSceneAct.setScene("6LocationBlue")

if Keyinput == [[114, 1]]:
	print "Location = Red"
	GameLogic.Location = "Red"
	GameLogic.LocationC = "6LocationRed"
	SetSceneAct.setScene("6LocationRed")

if Keyinput == [[103, 1]]:
	print "Location = Green"
	GameLogic.Location = "Green"
	GameLogic.LocationC = "6LocationGreen"
	SetSceneAct.setScene("6LocationGreen")

addActiveActuator(SetSceneAct,1)
```

If you have read
(14)WSAG – Startup Menu / Read out Key codes for Python
and
(16)WSAG – Startup Menu / Change Scenes – Python

and

(17)WSAG – Explorer Mod - Player Selection / Bind to GameLogic

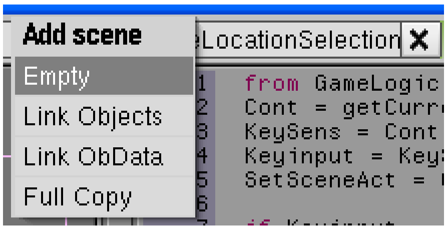
This script should be self explaining. Depending on what for a Key input you give, it will change to the Location ("6LocationBlue"), ("6LocationRed") or ("6LocationGreen") and save the Location Display Name to Variable GameLogic.Location and the Real Location Name to the Variable GameLogic.LocationC so before we can run the Script we will have to create these scenes.
The Display Name is The Name that will be Displayed In the Real-time Window
The Real Location Name is the Name of the Scene and is uses to change to the Scene it's the same as in the Brackets of the Variable SetSceneAct.setScene

So now its Time To save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.08WSAG.
Click on the Save as Button in the right upper Corner.
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(19)WSAG – Gameplay – Setting up the Basics (BV 2.44 & BV 2.46 PV 2.5.1) (LU 3.6.08)

Before we start to create the different locations we will setup the basics that have to be Included in every location.
That would be the spawning Point Were the Players Start in the Game and of course the Players themselves.

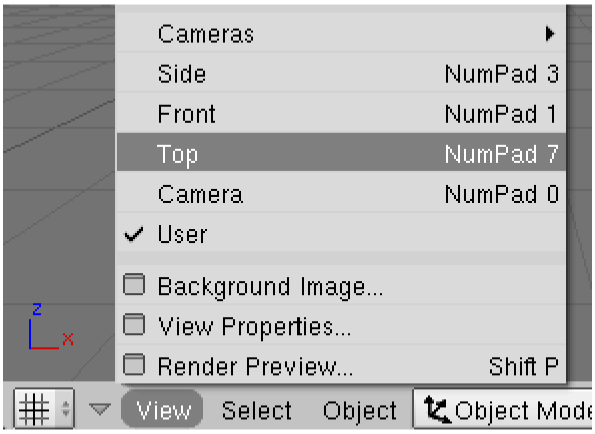
So we will ad a new scene with the name 5BasicGamePlay



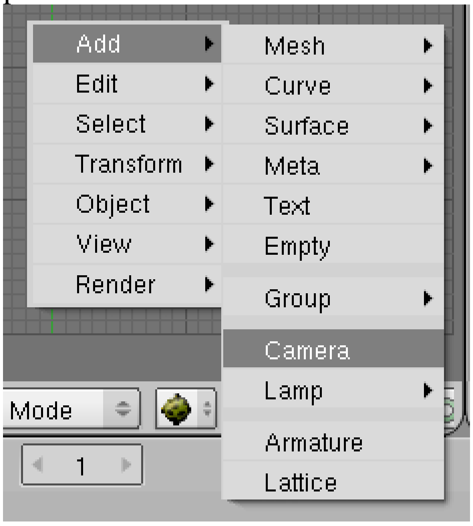
```
from GameLogic
Cont = getCurr
KeySens = Cont
Keyinput = Key
SetSceneAct =
```

These time an Empty one.
Don't forget to name it 5BasicGamePlay.

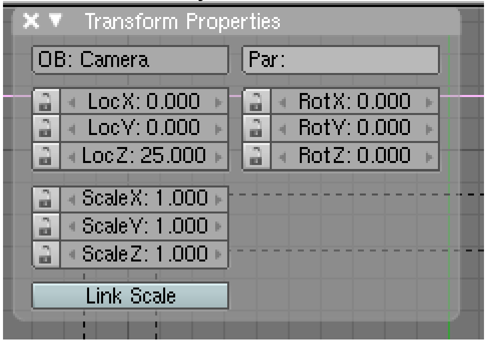
There you change to the top View: As the picture below shows you do this by clicking on View and then on Top or with the mouse over the 3D window press the 7-Key of your NumPad.



First ting we do is add a camera you do this by pressing space and select as shown in the picture below.



Change to the Camera View by pressing the 0-Key of the NumPad.
Press the N-Key and set the Location as the Picture shows below.

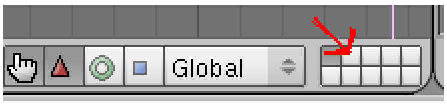


OB: Camera
Par:
LocX: 0.000
LocY: 0.000
LocZ: 25.000
RotX: 0.000
RotY: 0.000
RotZ: 0.000
ScaleX: 1.000
ScaleY: 1.000
ScaleZ: 1.000

LocX: 0.000
LocY: 0.000
LocZ: 25.000

Close the Transform Window by pressing on the X in the left upper corner.

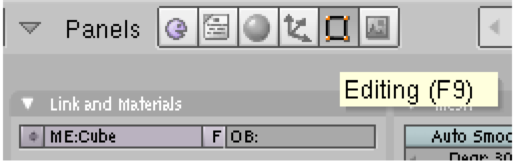
Change to the second layer by Clicking on the little square that I have marked with a red Arrow in the Picture below:



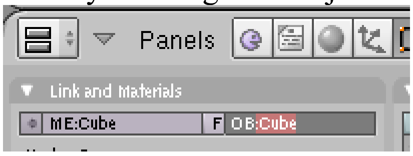
If the scene selection squares are not visible then move your mouse right over the line that divides the 3D Window for the Text Window.
When the Mouse courser changes to a symbol of 2 Arrows hold down your left mouse button hand move the line until you can see the scene Selection squares in the bottom right corner.

Now in this Layer we will create the Players… For right now we will keep it very simple. We will create a Cube.
Space-Key => Add => Mesh => Cube (mouse cursor has to be over the 3D window)

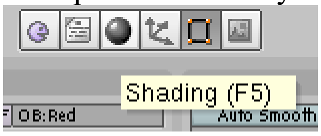
Now move your mouse down over the Button Window and press the F9-Key or click on the highlighted symbol in the picture below.



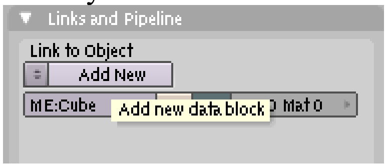
There you change the Object name form cube to Red.



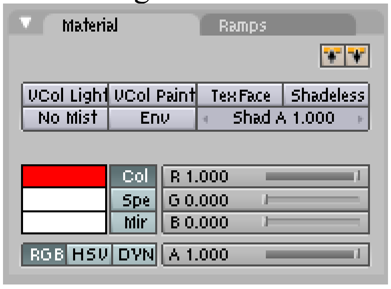
Now press the F5-Key or click on the highlighted symbol of the Picture below.



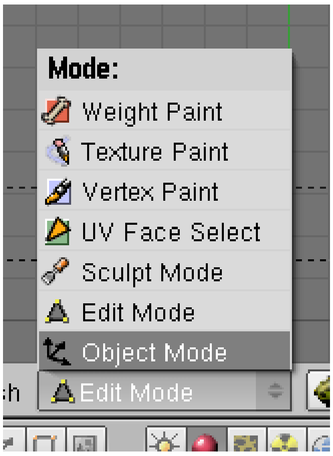
There you click on the Add New button.



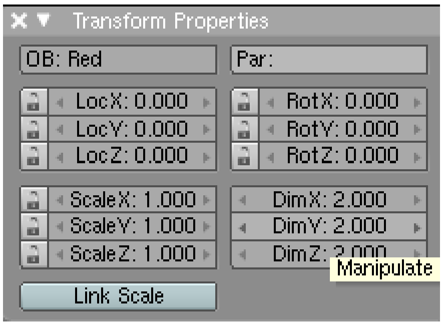
Now change the Slidebares so that the Cube has a Red Color



Click on Object Mode



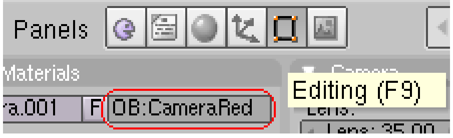
Press the N-Key and Change the Location value:
LocX 0.000
LocY 0.000
LocZ 0.000



OB: Red
Par:
LocX: 0.000
LocY: 0.000
LocZ: 0.000
RotX: 0.000
RotY: 0.000
RotZ: 0.000
ScaleX: 1.000
ScaleY: 1.000
ScaleZ: 1.000
DimX: 2.000
DimY: 2.000
DimZ: 2.000

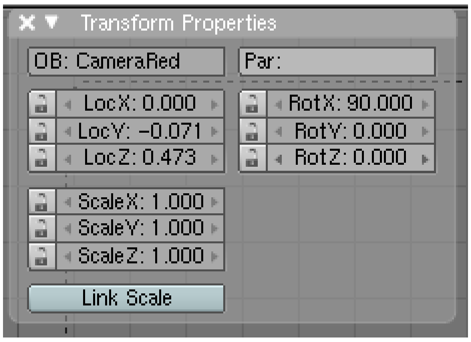
Close the Transform Window by clicking on the X in the right upper corner.

Add a Camera (Space-Key => Add => Camera)
Press F9-Key and Change the Camera Name to CameraRed



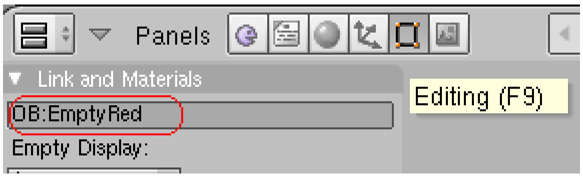
OB:CameraRed
Lens: 35.00

Press the N-Key and set:
LocX 0.000
LocY -0.071
LocZ 0.473
RotX 90.00



OB: CameraRed
Par:
LocX: 0.000
LocY: -0.071
LocZ: 0.473
RotX: 90.000
RotY: 0.000
RotZ: 0.000
ScaleX: 1.000
ScaleY: 1.000
ScaleZ: 1.000

Add a Empty (Space-Key => Add => Empty)
Press F9-Key and Change the Empty Name to EmtyRed

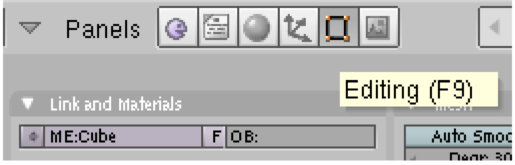


OB:EmptyRed

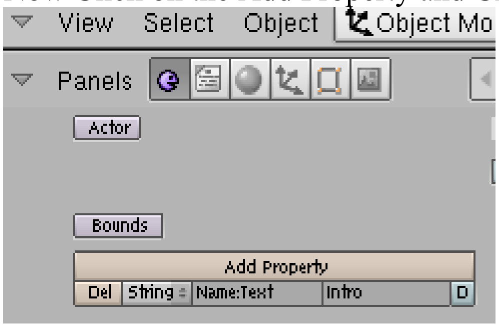
Pres N-Key and set:
LocX 0.000
LocY 1.500
LocZ 0.000
RotX 0.000
RotY 0.000
RotZ 0.000

Add a other Empty (Space-Key => Add => Empty)
Press F9-Key and Change the Empty Name to ContRed
Pres N-Key and set:
LocX 0.000
LocY 0.000
LocZ 0.000
RotX 0.000
RotY 0.000
RotZ 0.000

So now we will add some Text Planes how to do this you can read at (**(6)WSAG – Intro / Text UV Mapping**)
I will now only give you the name of each Texture plane and its location.
To change the Texture Planes Name press the F9-Key and enter the name in the field were it says "OB:"



To Change the Location just press the N-Key and Enter the Locations Numbers in to the place I will write you.
To make the Plane Display Text goes like this:
With your mouse over the bottom Window press F4.
Now Click on the Add Property and Change the settings so that they look like this:



String Name:Text Intro

Change to String, Name = Text (not text), there where I wrote Intro you write the Text that I will give you marked whit Text in the first Example below that would be :"GP".
So here we go:

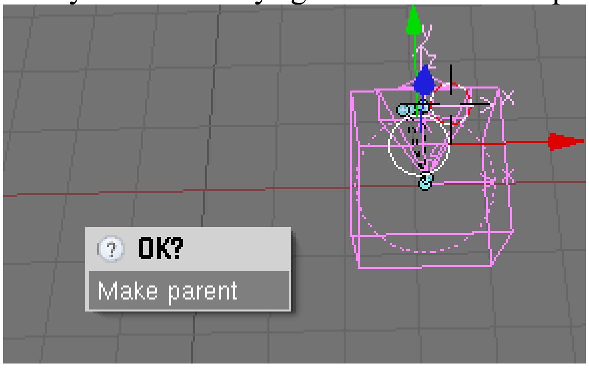
OB: GPRed
Text: GP

LocX: -0.300
LocY: 0.850
LocZ: 0.700
ScaleX: 0.025
ScaleY: 0.025
ScaleZ: 0.025
RotX: 90.000

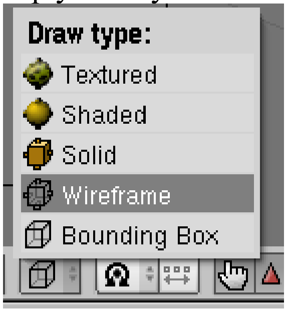
OB: GPCRed
Text: 100
LocX: -0.200
LocY: 0.850
LocZ: 0.700
ScaleX: 0.025
ScaleY: 0.025
ScaleZ: 0.025
RotX: 90.000

Select the The RedEmpty with a right mouse click, now press the Shift-Key and keep it pressed will you select one of the Text Planes und then the other one and at last the Camera “CameraRed”.
Let the Shift-Key go and press the Ctrl-Key keep it pressed and press the P-Key. Let both Keys go and Select Make parent.

Next you Select the Camera “CameraRed” and again press the Shift-Key and keep it pressed wile you select the Red Cube “Red” and at last the ContRed (Empty)… (The order in which you select the Objects is imported for it will decide the Hierarchy of the Parenting Model we will create now..) Let the Shift-Key go and press the Ctrl-Key keep it pressed and press the P-Key. Let both Keys go and Select Make parent.

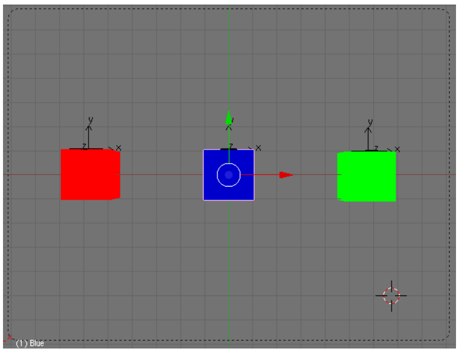


Tip you may want to change to the Wirefrime View so that you can see all the Objects better



Press the G-Key and move the ContRed (Empty) a bit to the left (Camera Empty Text planes and the Cube will fallow.)

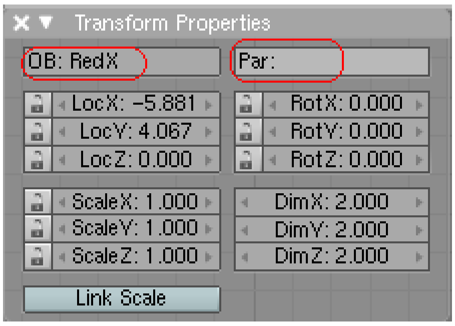
Now in the same way as described above create a Blue and a Green Cube.
Change back to Textured View.
The screen should look like this now:



As long as you just would like to make a single Player Game this setup would just do the trick. But If you want to make a Multiplayer game we run in to a small little problem:
For example if your Player is the Red Cube and the Other Player is the Blue Cube and both Players have the same move setup so If you give a move command (like pressing the UpArrow-Key) both players will move up.
O so why not have just have the Cube whit out the control Empty be created for the Second Player:

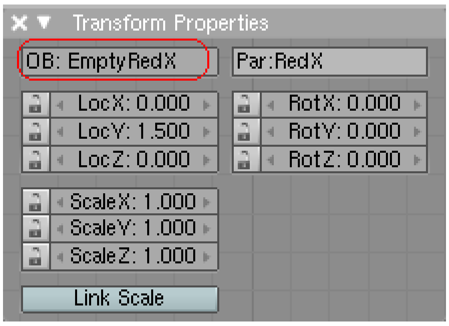
So Player One has the Red Cube whit the Control Empty and Camera and all that stuff created and as a second Player just the Blue Cube.(The Blue cube is moved by the Information send by the Server)
And Player Two has the Blue Cube whit the Control Empty and Camera and all that stuff created and as a second Player just the Red Cube.(The Red cube is moved by the Information send by the Server)

This actually would work but since the Cube (example Red) itself is a Child of the Control Empty (example: ContRed) and Childs when the get crated in Game Play act like Ghosts you would walk right through the other clients. So we will have to make a dubble of the Players (Cubes):
Select the Red Cube (Red)and whit the Shift-Key Pressed select the Red Empty (EmptyRed)
And Press The Shift-Key and the D-Key => This will Duplicate it.
Select the duplicated Red Cube (Red.001) and Press the N-Key Rename it to RedX and delete the Field where its says (Par:ContRed) because we don't want this Cube to be controlled be the ContRed.



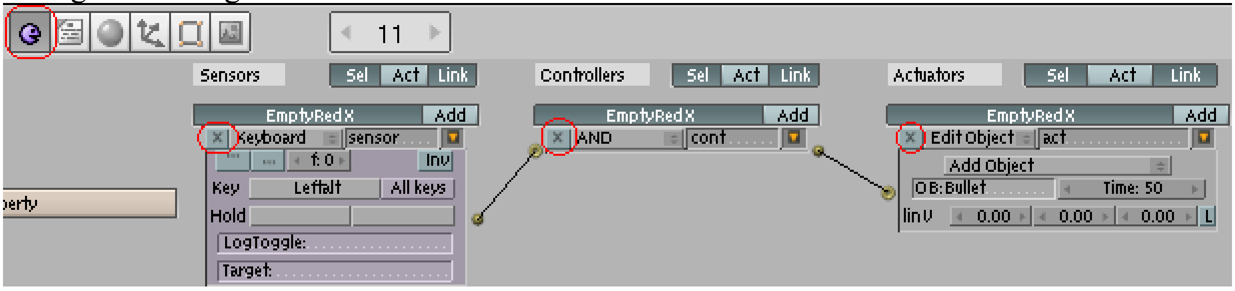
OB: RedX
Par:
LocX: -5.881
LocY: 4.067
LocZ: 0.000
RotX: 0.000
RotY: 0.000
RotZ: 0.000
ScaleX: 1.000
ScaleY: 1.000
ScaleZ: 1.000
DimX: 2.000
DimY: 2.000
DimZ: 2.000

Now Select the duplicated Red Empty (EmptyRed.001) press the N-Key and Rename it to EmptyRedX



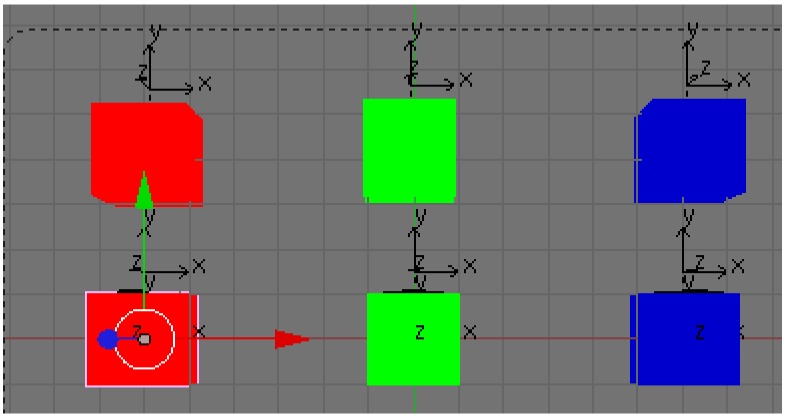
OB: EmptyRedX
Par:RedX
LocX: 0.000
LocY: 1.500
LocZ: 0.000
RotX: 0.000
RotY: 0.000
RotZ: 0.000
ScaleX: 1.000
ScaleY: 1.000
ScaleZ: 1.000

Change to the Logic Bricks and delete them:

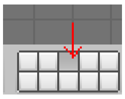


Keyboard sensor
f: 0
Inv
Key LeftAlt All keys
Hold
LogToggle:
Target:
AND cont
Edit Object act
Add Object
OB:Bullet
Time: 50
linV 0.00 0.00 0.00 L

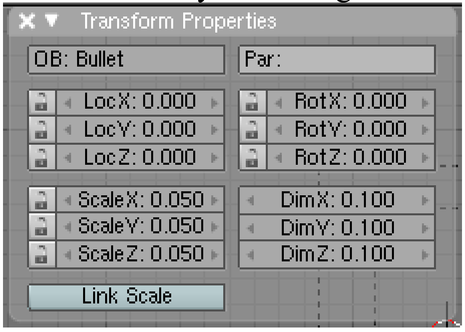
Now do the same thing whit the Blue and The Green Setups…
It now will look like this:



Go to the 3 Layer.

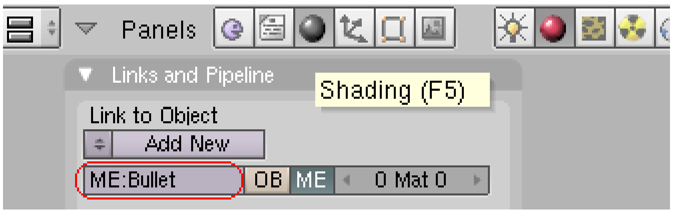


There you create a Cube and change to object mode.
Press the N-Key and change these values:

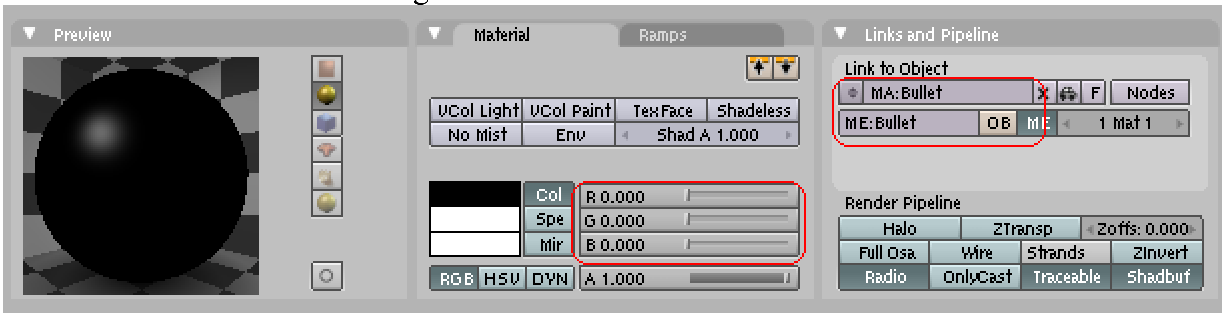


OB: Bullet
Par:
LocX: 0.000
LocY: 0.000
LocZ: 0.000
RotX: 0.000
RotY: 0.000
RotZ: 0.000
ScaleX: 0.050
ScaleY: 0.050
ScaleZ: 0.050
DimX: 0.100
DimY: 0.100
DimZ: 0.100

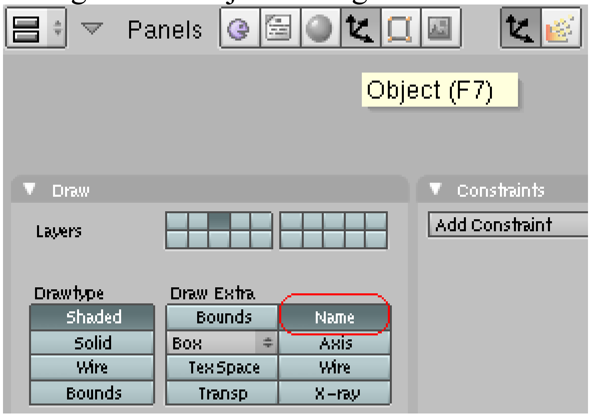
Now go to the Material settings and Change the Material to Bullet



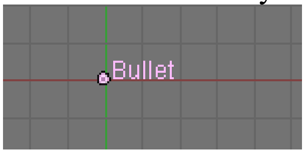
Click on the "Add New Button" and change the color to black.
Make sure the two circled settings are Correct:



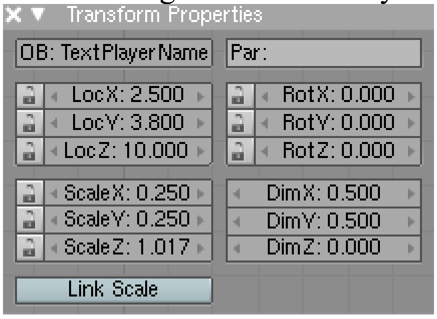
Change to the Object settings and there were I made a red Circle you push the button Name…



This will help us find this Object even if it's so small because it now displays with the name written next to it as you can see in the picture below…



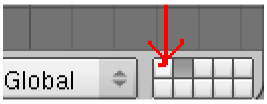
Now you create a Text Plane as Described in **(6)WSAG – Intro / Text UV Mapping**
Set its settings with the N-Key like shown in the Picture below:



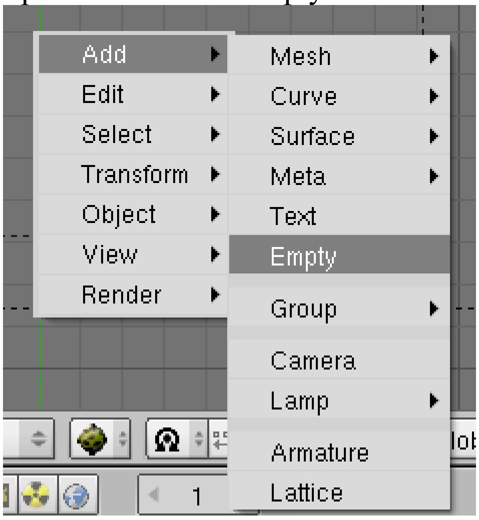
OB: TextPlayerName
Par:
LocX: 2.500
LocY: 3.800
LocZ: 10.000
RotX: 0.000
RotY: 0.000
RotZ: 0.000
ScaleX: 0.250
ScaleY: 0.250
ScaleZ: 1.017
DimX: 0.500
DimY: 0.500
DimZ: 0.000

Copy this TextPlane (Shift-Key and D-Key) and change this Transform Properties:
OB: TextGPs
LocX: 3.800
LocY: 3.800

Now go back to the first layer

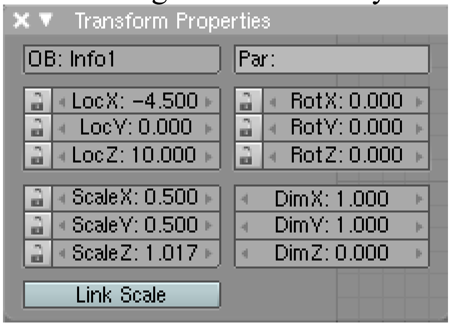


And create an Empty (This will be the spawning Point for the Players)
Space => Add => Empty

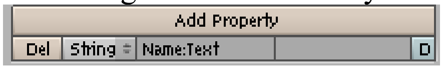


Pres N-Key and set:
LocX 0.000
LocY 0.000
LocZ 1.000

So now We need so Add some Info Text Planes That Mainly The Server Mode Will use later but we already have to set them up now so that the will be linked to all Scenes that we will add later:
Now you create a Text Plane as Described in **(6)WSAG – Intro / Text UV Mapping**
Set its settings with the N-Key like shown in the Picture below:


Transform Properties
OB: Info1
Par:
LocX: -4.500
LocY: 0.000
LocZ: 10.000
RotX: 0.000
RotY: 0.000
RotZ: 0.000
ScaleX: 0.500
ScaleY: 0.500
ScaleZ: 1.017
DimX: 1.000
DimY: 1.000
DimZ: 0.000

In the Logic Bricks Section you add this the Property as show in the Picture below



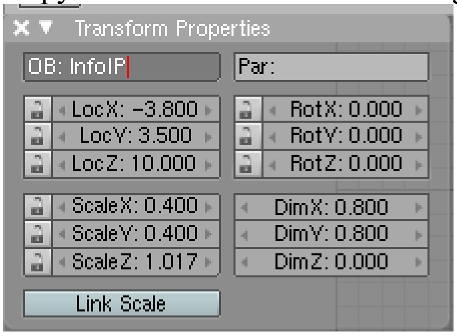
String
Name:Text

So now we will have to add a some more Text Planes. So Copy this Text Planes (Shift-Key and D-Key) and change this Values in the Menu that comes up when you press the N-Key (Transform Properties Menu):
OB: Info2
LocX: -4.500
LocY: -1.400
LocZ: 10.000

Copy another Text Plane and Change it to these Values:
OB: Info3
LocX: -4.500
LocY: -2.800

LocZ: 10.000

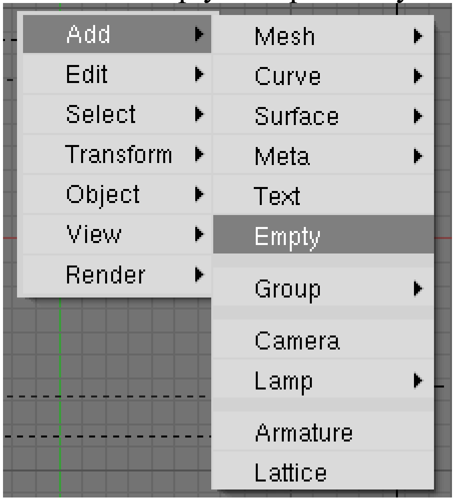
Copy another Text Plane and Change it to the Values in the Picture Below:



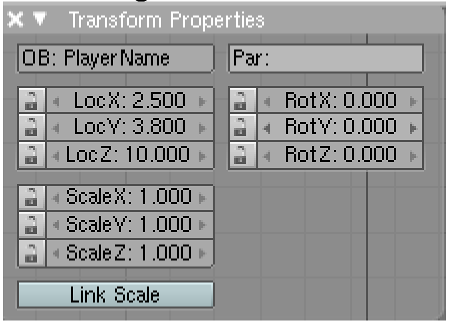
OB: InfoP
Par:
LocX: -3.800
LocY: 3.500
LocZ: 10.000
RotX: 0.000
RotY: 0.000
RotZ: 0.000
ScaleX: 0.400
ScaleY: 0.400
ScaleZ: 1.017
DimX: 0.800
DimY: 0.800
DimZ: 0.000

Copy this New Scaled Plane and change it to these Values:
OB: InfoLocation
LocX: -3.800
LocY: -2.500
LocZ: 10.000

Now ad a Empty => Space-Key => Add => Empty



And Change the Values shown in the Picture below:



OB: PlayerName
Par:
LocX: 2.500
LocY: 3.800
LocZ: 10.000
RotX: 0.000
RotY: 0.000
RotZ: 0.000
ScaleX: 1.000
ScaleY: 1.000
ScaleZ: 1.000

Copy this Empty and change the Copy to this Values
OB: GPs
LocX: 4.000
LocY: 3.800
LocZ: 10.000

Now Select the Empty (the one in the Middle of the Scene that we have created first and that is named Empty) and add these Logic Bricks to it
Sensor = Keybord = End-Key
Controller = AND
Actuator = Game = Restart this Game

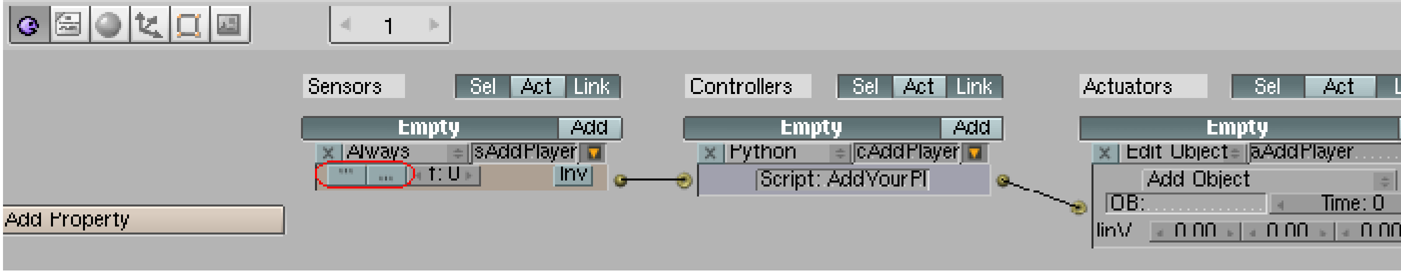
Connect the Yellow dots whit each other.
Now if we press ESC in the Game it will restart the Program.
Note: In Blender 2.46 it won't restart the game but Quit it… (A bug?)

So now every Object that we need for the Basic Setup is in the right Place.
Before we start to Program the single Objects, it's time to save again.

So now its Time To save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.09WSAG.
Click on the Save as Button in the right upper Corner.
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(20)WSAG – Gameplay – Setting up the spawning Point – Explorer Mod (BV 2.44 PV 2.5.1) (LU 23.12.07)

Create a new Text file with the Name "AddYourPlayer.py"
To the Empty you add the LogicBricks
Sensor => Allways (Name it "sAddPlayer")
Controller => Python => AddPlayers.Py (Name it "cAddPlayer")
Actuator => EditObject => AddObject (Name it "aAddPlayer")
The Setup should look like this now:



Make sure the Tow buttons in the red circle are not pushed… because we only want the script to run one time

For More Details on how to do this Read
(12)WSAG – Interact with the Engine / Change Scenes – Logic Bricks

I the AddYourPlyers.py Text Window we add this Script:

```
from GameLogic import *

if GameLogic.PlayerMod = = "Explore" or GameLogic.PlayerMod = = "Client":
        cont = GameLogic.getCurrentController()
        AddPlayerAct = cont.getActuator("aAddPlayer")
        AddPlayerAct.setObject(GameLogic.PlayerTyp)
        addActiveActuator(AddPlayerAct,1)
```

The first tow lines of the Script should be clear by now. If not read **(14)WSAG – Startup Menu / Read out Key codes for Python**

The Next Line:

```
if GameLogic.PlayerMod = = "Explore" or GameLogic.PlayerMod = = "Client":
```

It tells the Script to only execute the lines below if the Variable Saved to GameLogic.PlayerMod is either Explore or Client

The Next Line:
AddPlayerAct = cont.getActuator("aAddPlayer")
This line of code will tell the Engine that the Variable AddPlayerAct will call up the command cont.getActuator("aAddPlayer")
The Command cont.getActuator("aAddPlayer")starts with the Variable Cont that we have created in the Line above.
The second Part of the Command is getActuator this tells the Script that from he Logic Brick that the Cont Command imported in to the Script it should get the Actuator.
The last Part of the Script is ("aAddPlayer") this is simply the name of the Actuator that the Script (Engine) should get (import).

AddPlayerAct.setObject(GameLogic.PlayerTyp)
The Command AddPlayerAct.setObject(GameLogic.PlayerTyp)starts with the Variable AddPlayerAct that we have defined above.
The second Part of the Command is setObject, it tells that the game should create the Object defined in the last part of this command (GameLogic.PlayerTyp)
In the Scene 3ePlayerSelection we defined what the Variable GameLogic.PlayerTyp stands for.

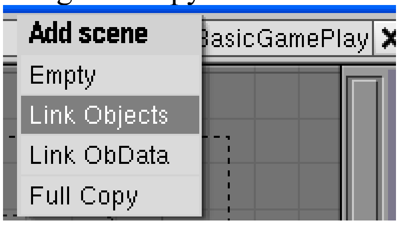
addActiveActuator(AddPlayerAct,1)
This line tells the script if the added Actuator should be executed or not. The 1 at he end of the command tells the script it school activate (execute) this added active Actuator. If it would be set to 0 it would mean that this actuator is shouted of (is passive).

So now the spawning Point is set up that it will create the Player defined in the Scene 3ePlayerSelection. Of course the Player has no interaction intergraded in to it jet so you wont be able to move it around … but we will take care of that later

So now it's Time to save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.10WSAG.
Click on the Save as Button in the right upper Corner.
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(21)WSAG – Gameplay – Creating the Locations (BV 2.44 PV 2.5.1) (LU 7.12.07)

Now we will add the 3 Locations ("6LocationBlue"), ("6LocationRed") and ("6LocationGreen").
You do this just as shown in **(11)WSAG – Add a Scene with** one little difference. Instead of using full Copy we use Link Objects.

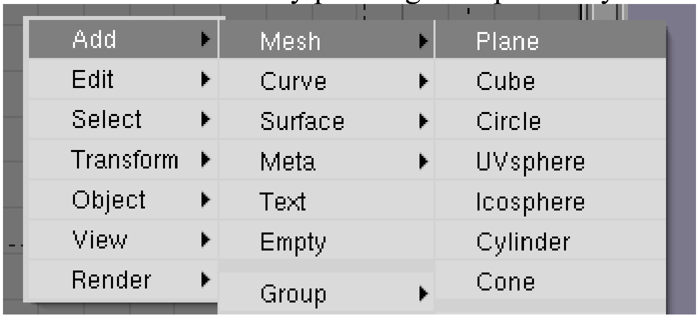


By doing this every change that we make on one Of the Objects in 5BasicGamePlay it will change them in all the Scenes….

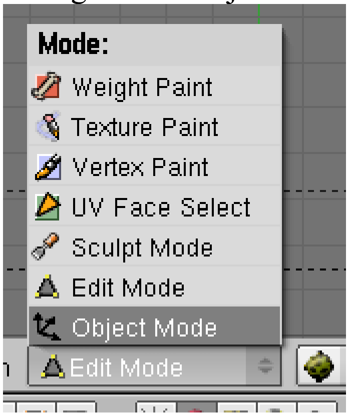
Now in the 3D Window of each scene (layer one, that's the one with the Empty) we will create a Plane with the colors of the of the Location.

I will show step by step for the 6LocationBlue and you then can adapt it for the other tow Locations

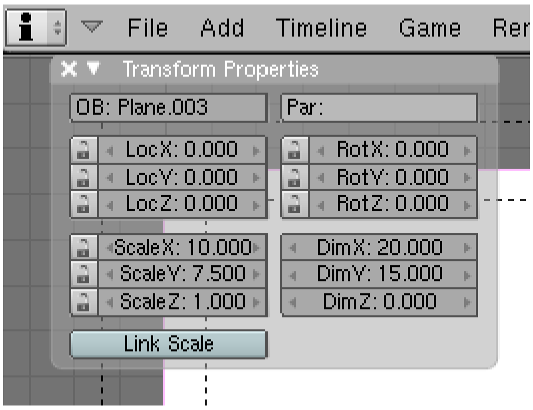
After adding a new scene with the Link Objects Method we will change the scene Name to 6LocationBlue.
Now we add a Plane by pressing the Space-Key =>Add=>Mesh=>Plane



Change to the Object mode:

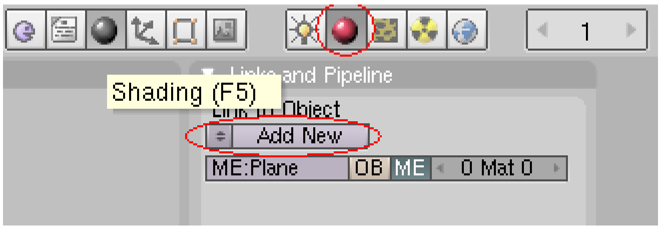


With the mouse over the 3D window press the N-Key and change the Settings as shown in the Picture below:

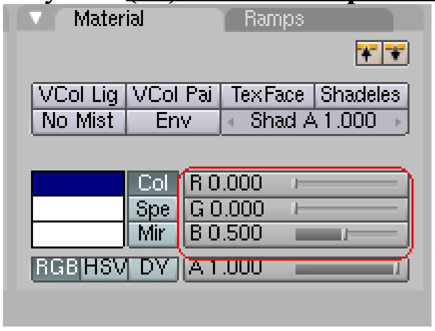


OB: Plane.003
Par:
LocX: 0.000
LocY: 0.000
LocZ: 0.000
RotX: 0.000
RotY: 0.000
RotZ: 0.000
ScaleX: 10.000
ScaleY: 7.500
ScaleZ: 1.000
DimX: 20.000
DimY: 15.000
DimZ: 0.000

Now Press F5-Key and make sure that the Material Button is pressed (That's the Red ball that I have circled)
And the Click on Add Material (that's the Second button that I have circled)



Change the Colors to the Plane Color You wish. In this case that would be Blue (have R at 0.0 // G at 0.0 and B at 0.5 so that you get a different Blue then the blue We used for the Blue Player in **<u>(18)WSAG – Explorer Mod - Location Selection</u>**)



Shad A 1.000
R 0.000
G 0.000
B 0.500
A 1.000

No do this same thing for the Other 2 Locations (make sure you always make an Add Scene from the 5BasicGamePlay scene with the Link Object Option.

So now its Time To save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.11WSAG.
Click on the Save as Button in the right upper Corner.
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

<u>(22)WSAG – Gameplay – Setting up the Players – Moving</u> (BV 2.44 PV 2.5.1) (LU 5.4.08)

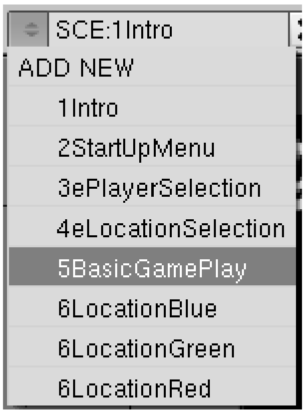
Create a new Text file with the Name "Move.py"
In that Text File you add this Code:

```
from GameLogic import*
Cont = getCurrentController()
Obj = Cont.getOwner()
KeySens = Cont.getSensor("sKeyimput")
TouchSens = Cont.getSensor("sTouch")
MoveAct = Cont.getActuator("aMove")
Keyimput = KeySens.getPressedKeys()


if TouchSens.isPositive():
        if Keyimput = = [[146, 1]]:
                yMove = 0.1
        if Keyimput = = [[144, 1]]:
                yMove = -0.1
        if Keyimput = = [[145, 1]]:
                zRot = - 0.1
        if Keyimput = = [[143, 1]]:
```

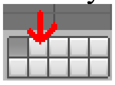
```
        zRot = 0.1
    if Keyimput = = [[32, 1]]:
        zForce = 300
        Obj.Jump = 3
    if Keyimput = = [[114, 1]]:
        Ori = [[1,0,0],[0,1,0],[0,0,1]]
        Obj.setOrientation(Ori)


if Keyimput = = [[146, 3]]:
        yMove = 0.0
if Keyimput = = [[144, 3]]:
        yMove = 0.0
if Keyimput = = [[143, 3]]:
        zRot = 0.0
if Keyimput = = [[145, 3]]:
        zRot = 0.0
if Keyimput = = [[32, 3]]:
        zForce = 0.0
if Obj.Jump > 0:
    Obj.Jump -= 1
if Obj.Jump = = 0:
    zForce = 0.0


try:
    MoveAct.setDRot(0.0,0.0,zRot,1)
except:
    pass
try:
    MoveAct.setDLoc(0.0,yMove,0.0,1)
except:
    pass
try:
    MoveAct.setForce(0.0,0.0,zForce,1)
except:
    pass
addActiveActuator(MoveAct,1)
```

Before I will explain what this Code will do we will set up the Logic Brick that this Code Works with.

Change to the scene 5BasicGamePlay



There you change to the second layer.



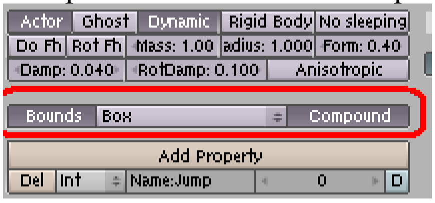
Select the ContGreen (Empty) with a right mouse Click and press F4
Press the “Actor Button” and the in the new menu “Dynamic Button”
We also press the “Button Bounds” and the in the new menu the “Button Compound”

This will make the Cube relay on Physics. For Example it will fall down on the Plane….

Now we Add tow Sensors, one Controllers and one Actuators.
Now set:
The 1st. Sensor = Kebord = All Keys
The 2nd.Sensor = Touch = Pulse = MA: (leave empty for all Materials)
The Controler = Python = Move.py
The Actuator = Motion

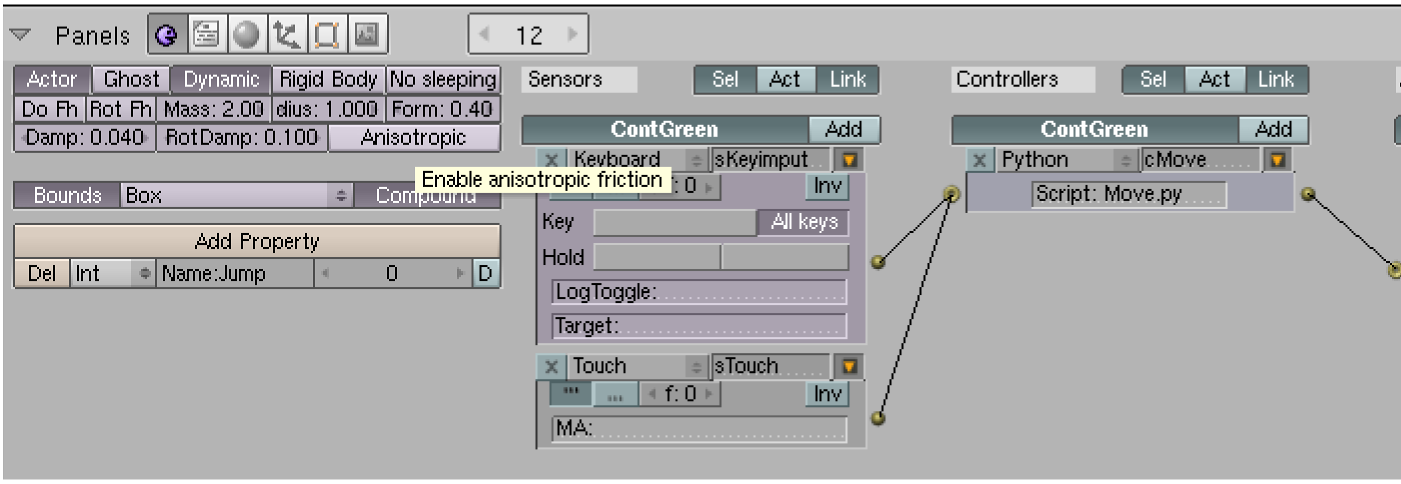
Now name:
The 1st. Sensor = sKeyimput
The 2nd.Sensor = sTouch
The Controler = cMove
The Actuator = a Move

Combined, the 2 Sensors with the Controller and the Controller with the Actuator, by connecting the Yellow dots with each other.
Now click on the “Add Property Button” and set the new Property to Int. with the name Jump.
For the last you click on the Bounds Button and then on the new appeared button called Compound. Make sure the dropdown menu is set to Box.



Mass: 1.00
adius: 1.000
Form: 0.40
Damp: 0.040
RotDamp: 0.100
Box
Int
Name:Jump
0

This will draw a box around the Object and this Box is what will react to the Physics.
Your Button Window should look like this now…



Mass: 2.00
dius: 1.000
Form: 0.40
Damp: 0.040
RotDamp: 0.100
Bounds: Box
Int Name:Jump 0
Keyboard sKeyimput
Key
Hold
LogToggle:
Target:
Touch sTouch
f: 0
MA:
Python cMove
Script: Move.py

Do the Same to the Other to Players.

So now that everything is setup I will try to explain what the Code Move.py dose.
Get yourself a strong cup of Coffee and now here we go:

```
from GameLogic import*
Cont = getCurrentController()
Obj = Cont.getOwner()
KeySens = Cont.getSensor("sKeyimput")
TouchSens = Cont.getSensor("sTouch")
MoveAct = Cont.getActuator("aMove")
Keyimput = KeySens.getPressedKeys()
```

This first block should be pretty familiar by now:
A quick Repetition:

```
from GameLogic import*
```

Imports the Lybary GameLogic into the Script

```
Cont = getCurrentController()
```

Gets The Logic Brick that are attached to the Script and saves them in to the Variable Cont

```
Obj = Cont.getOwner()
```

Gets the Object that the Logic Bricks belong to… I our Case the ContGreen (Empty) object.
By getting this Object into or Script we can take action on all the Properties of the Object.

```
KeySens = Cont.getSensor("sKeyimput")
```

Gets the Sensor sKeimput from the Logic Bricks we got saved in the Variable Cont. And saves them to the Variable KeySens

```
TouchSens = Cont.getSensor("sTouch")
```

Gets the Sensor sTouch from the Logic Bricks we got saved in the Variable Cont. And saves them to the Variable TouchSens

```
MoveAct = Cont.getActuator("aMove")
```

Gets the Actuator aMove from the Logic Bricks we got saved in the Variable Cont. And saves them to the Variable MoveAct

```
Keyimput = KeySens.getPressedKeys()
```

Command `getPressedKeys()` tells the Script to get the Code that is send when a Key is Pressed and save it to the Variable `Keyimput`.

```
if TouchSens.isPositive():
        if Keyimput = = [[146, 1]]:
                yMove = 0.1
        if Keyimput = = [[144, 1]]:
                yMove = -0.1
        if Keyimput = = [[145, 1]]:
                zRot = - 0.1
        if Keyimput = = [[143, 1]]:
                zRot = 0.1
        if Keyimput = = [[32, 1]]:
                zForce = 300
                Obj.Jump = 3
        if Keyimput = = [[114, 1]]:
                Ori = [[1,0,0],[0,1,0],[0,0,1]]
                Obj.setOrientation(Ori)
```

This is the Second Block and it contains a lot of familiar stuff, but also some new stuff:

```
if TouchSens.isPositive():
```

If the Touch sensor is Positive (meaning if the Player (ContGeen (Empty)) is touching something) it should do what is written in the lines below that are indented.

```
if Keyimput = = [[146, 1]]:
```

If the Code that was saved in the Variable `Keyimput` is equal to the Code `146.1` it should do what is written in the lines below that are indented.

```
yMove = 0.1
```

Saves the Value `0.1` in to the Variable `yMove`

```
if Keyimput = = [[144, 1]]:
                yMove = -0.1
if Keyimput = = [[145, 1]]:
                zRot = - 0.1
if Keyimput = = [[143, 1]]:
                zRot = 0.1
if Keyimput = = [[32, 1]]:
                zForce = 300
```

All this lines to the same as the last tow lines I described just with different Variables and Values.

```
Obj.Jump = 3
```

The command `Obj.Jump` first calles up the Object that is saved in the Variable `Obj.` in are case the Empty ContGreen.

Then it will call up the property `Jump` and set it to the Value `3`

This is the Property that we have created in the Logic Bricks of the Object Empty ContGreen.

```
if Keyimput = = [[114, 1]]:
                Ori = [[1,0,0],[0,1,0],[0,0,1]]
```

```
        Obj.setOrientation(Ori)
```

The first tow line of this Code Block is the same as all the ones above but then an totally new lines come up.

The command Obj.setOrientation(Ori) first calles up the Object that is saved in the Variable Obj. in are case the Empty ContGreen.

Then it tells the Object to set its Orientation with the Values saved in the Variable Ori

This line is use so that if your Player rolls over you can press the R-Key and it will sand up on his feet again.

If you want to understand what the Value that was saved to the Variable Ori mean, you will have to learn about the 3 x 3 Matrix. This is quite complex so I will explain it in a separate Chapter:

```
if Keyimput = = [[146, 3]]:
        yMove = 0.0
if Keyimput = = [[144, 3]]:
        yMove = 0.0
if Keyimput = = [[143, 3]]:
        zRot = 0.0
if Keyimput = = [[145, 3]]:
        zRot = 0.0
if Keyimput = = [[32, 3]]:
        zForce = 0.0
```

This Code Block dose almost the same thing as the Code block above that was indented under the command if TouchSens.isPositive():

The divergence is that the Values are set to 0.0 if you let the Key go and in the script above it is set to 0.1 when you press the Key.

```
if Obj.Jump > 0:
    Obj.Jump -= 1
```

If the Value of Obj.Jump is bigger then 0 then subtract 1 from the Value of Obj.Jump.

```
if Obj.Jump = = 0:
    zForce = 0.0
```

If the Value of Obj.Jump is 0 then set the Value of zForce to 0.0

```
try:
    MoveAct.setDRot(0.0,0.0,zRot,1)
except:
    pass
try:
    MoveAct.setDLoc(0.0,yMove,0.0,1)
except:
    pass
try:
    MoveAct.setForce(0.0,0.0,zForce,1)
except:
    pass
addActiveActuator(MoveAct,1)
```

So the last Block of Code from this Script

You will notice here that every command line is between the lines

```
try:
```

and

```
except:
	pass
```

This tells the script I should try to execute the Command and if it fails it should go on… I it trys to set a Value of one of the Variables and that Varable has not Value saved to it the Script would get a error if it would not have the

```
try:
```

and

```
except:
	pass
```

Commands in it.

So what do the Commands do that sand between the

```
try:
```

and

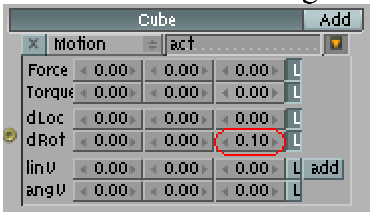
```
except:
	pass
```

lines?

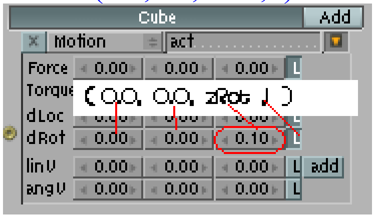
```
MoveAct.setDRot(0.0,0.0,zRot,1)
```

In the Variable MoveAct the Actuator aMove is saved. In that Actuator it will set the Variable zRot to DRot at the Position given in (0.0,0.0,zRot,1).

Do Imagine this Graphical the Picture below shows where this Value would be set if we would do the same thing with Logic Bricks



So the (0.0,0.0,zRot,1) stand for:



The First number stands for the Rotation on the X axis
The Second number stands for the Rotation on the Y axis.
The third number stands for the Rotation on the Z axis (this is the axis we want are Player to Rotate on if you Press/let go the Left Arrow Key (Python Code [[143, 3]] or [[143, 1]]) or the Right Arrow Key (Python Code [[145, 3]] or [[145, 1]])
The last number is set to 1 meaning the same as if the L in the Logic Bricks where pushed what will let the Engine know that the Values relay to the Local settings. If it were set to 0 it would mean that the Values are related to the Objects own settings.

```
MoveAct.setDLoc(0.0,yMove,0.0,1)
```

This line dose almost the same as the one above it just sets the Movement on the different axis, I are case the Y axis. When you Press/let go the Up Arrow Key (Python Code [[146, 3]] or [[146, 1]]) or the Down Arrow Key (Python Code [[144, 3]] or [[144, 1]])

```
addActiveActuator(MoveAct,1)
```

This line tells the script if the added Actuator should be executed or not. The 1 at the end of the command tells the script it school activate (execute) this added active Actuator. If it would be set to 0 it would mean that this actuator is shouted of (is passive).

So now it's Time to save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.12WSAG.
Click on the Save as Button in the right upper Corner.
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(23)WSAG – Gameplay – Setting up the Players – 3 x 3 Matrix(BV 2.44 PV 2.5.1) (LU 22.12.07)

I hope the cup of coffee I recommended earlier has already kicked in because mow we will come to a complex Part, were I will try to explain the basics of the 3 x3 Matrix.

So here we go:

[[1,0,0],[0,1,0],[0,0,1]]
The 3 brackets stand for the World Axis
[[World X axis], [World Y axis], [World Z axis]]

The 3 number inside everyone of this brackets, stand for the Object Axis.
[[Object X axis, Object Y axis, Objext Z axis], [Object X axis, Object Y axis, Objext Z axis], [Object X axis, Object Y axis, Objext Z axis]]

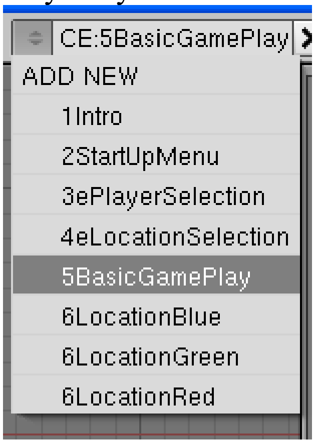
[[1,0,0],[0,1,0],[0,0,1]] So these values mean:
In the 1st bracket the Object X axis points in World X axis direction.
In the 2nd bracket the Object Y axis points in World Y axis direction.
In the 3rd bracket the Object Z axis points in World Z axis direction.

The Values can be anything between -1 and 1
So if we would rotate the Object 90° on its Y axis the Value would Change to
[[0.0, 0.0, 1.0], [0.0, 1.0, 0.0], [-1.0, 0.0, 0.0]]
Meaning that:
In the 1st bracket the Object Z axis new points in World X axis direction.
In the 2nd bracket the Object Y axis still points in World Y axis direction.
In the 3rd bracket the Object X axis new points in World -Z axis direction.

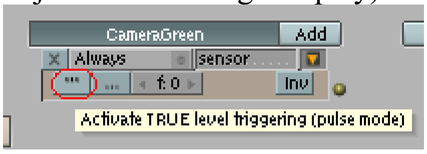
So now it's Time to save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.12WSAG.
Click on the Save as Button in the right upper Corner.
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(24)WSAG – Gameplay – Setting up the Players – Camera (BV 2.44 & BV 2.46 PV 2.5.1) (LU 30.5.08)

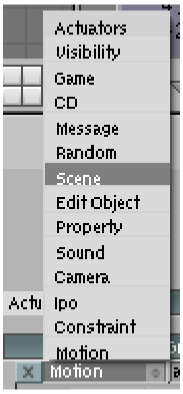
So now we want to make our game in First Person Perspective. This Means that you will be looking tough the Camera of the Player you selected.
So we change to the SCE: 5BasicGamePlay and there change to the 2nd layer where our Player lay.



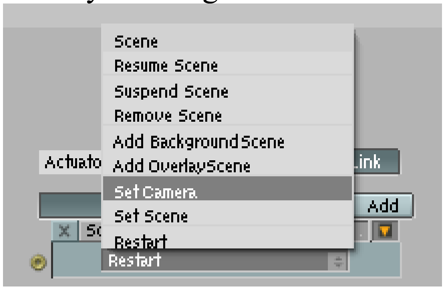
There we select the Object CameraGreen.
Pres the F5-Key to change to the Logic Bricks setup and add
A Sensor => Always => Pulse Mode of (this way it will only call up the action once when this object comes into game play)



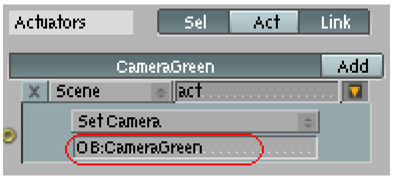
Then a Controller => And
Then a Actuator = Scene



There you change to the Set Camera option:

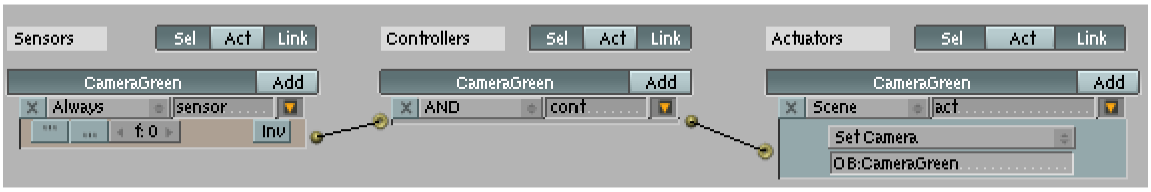


And in the Field that is circled red you write in the Name of the Camera you want to change to. I our chase: CameraGreen.



NOTE: For Blender 2.45 and higher leave the Field to what Camera it should change to **Empty**.
In these newer Versions it will automatically set the Camera that this actuator is linked to. And if you type in a Name it will point to the Default camera and not the new created one…

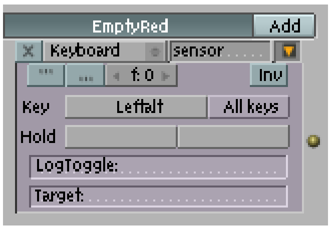
Combine the Logic Bricks with the yellow dots.
It now should look like this:



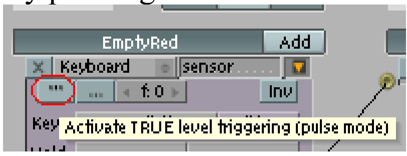
Do the same thing with the other two Player Cameras.
So now it’s Time to save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.13WSAG.
Click on the Save as Button in the right upper Corner.
PLEASE DON’T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(25)WSAG – Gameplay – Setting up the Players – Shoot (BV 2.44 PV 2.5.1) (LU 27.3.08)

You may have asked your self why added a Empty in front of each Players Camera well now we will use it to fire of a bullet.
Select the EmptyRed.
Press F5-Key to setup its logic Bricks.
Add a Senor => Keyboard
And have the Alt-Key be the Key that activates this Sensor:
It now should look like this:

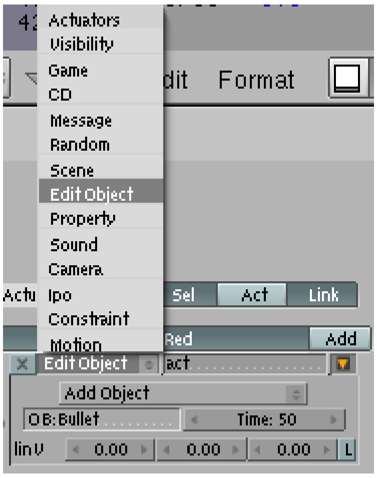


By pressing the “Pulse button” you can make Rapid Fire…



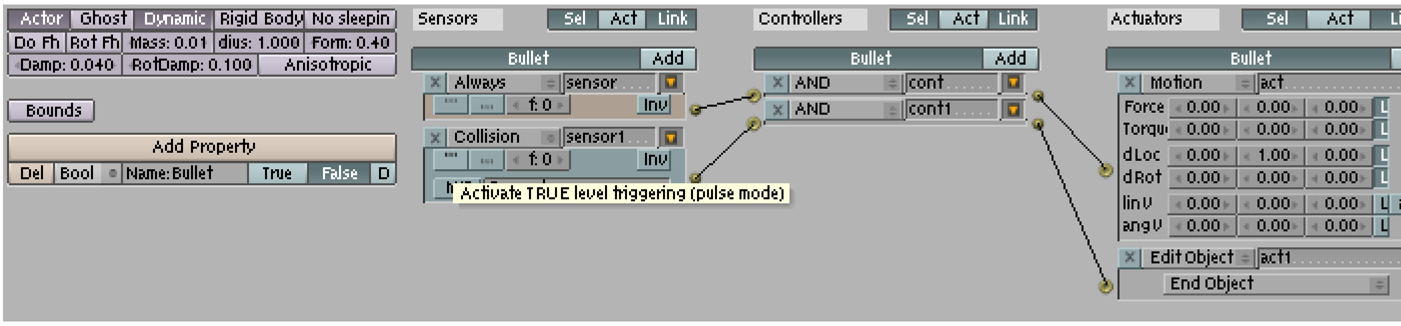
Activate TRUE level triggering (pulse mode)

Add a Controller = AND
And a Actuator = Edit Object = Add Object and in the OB filed you the write Bullet.
An the Time you set to 50 (That means the Bullet will exist for 50 Frames and then will be removed again.



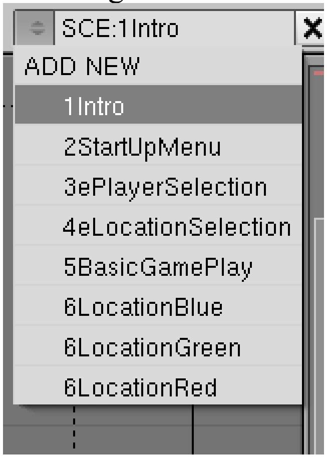
Now with the Yelllow dots you combined all the Logic Bricks with each other.
Do the Same with the Other Players.

No you change to the 3rd Layer and select the Bullet and add The Logic Bricks as shown in the Picture below:



First Line will give the Bullet a Forward Motion and the Second line will end The Object if it colliding with anything.

So now go back to the Intro Scene



So now it’s Time to save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.13WSAG.
Click on the Save as Button in the right upper Corner.

PLEASE DON’T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

If you now press the P-Key with the mouse over the 3D Scene you can select the Explorer Mode the Location and the Player and you will be able to walk and Shoot Around in the selected Location with you're from the First Person View of your Selected Player. With this, the Explorer Mode is up And Running. Of course it's nothing pretty to look at yet – That will have to be improved by some advanced Modeling but that is not the Subject we want to look at yet. We first of all want a basic setup for Explore Mod
Client Mod
Server Mod
The Explorer Mode is now up and running so next will be to get the other two Modes up.
But before we move on you can get yourself a good Glass of Champagne and celebrate your first step of success.

(26)WSAG – Server Mod - Location Selection (BV 2.44 PV 2.5.1) (LU 23.12.07)

Will start with the basic setup for the Server…
So you change to the 2StartUpMenu scene there you see that if you press the S-Key it should start the Server Mode. In the Text Window we call up the Script
MainMenu.py

There we add these lines:

```
if Keyinput == [[115, 1]]:
        print "Server"
        GameLogic.PlayerMod = "Server"
        SetSceneAct.setScene("4eLocationSelection")
```

After reading the **(18)WSAG – Explorer Mod - Location Selection** it should be clear what this lines do…
The Howl MainMenu.py will now look like this:

```
from GameLogic import*
Cont = getCurrentController()
KeySens = Cont.getSensor("sKeyInput")
Keyinput = KeySens.getPressedKeys()
SetSceneAct = Cont.getActuator("aSetScene")

if Keyinput = = [[101, 1]]:
        print "Explore"
        GameLogic.PlayerMod = "Explore"
        SetSceneAct.setScene("3ePlayerSelection")

if Keyinput = = [[115, 1]]:
        print "Server"
        GameLogic.PlayerMod = "Server"
        SetSceneAct.setScene("4eLocationSelection")


addActiveActuator(SetSceneAct,1)
```

Now it will go to the Scene Selection Window if you press The S-Key in the Main Menu und after the scene selection it will load the requested scene.
So the Next Changes will be in the scene 5BasicGamePlay.

So now it's Time to save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.14WSAG.
Click on the Save as Button in the right upper Corner
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial
.

(27)WSAG – Server Mod - Display Your IP (BV 2.46 & 2.44 PV 2.5.1) (LU 21.6.08)
Change to the scene 5BasicGamePlay.
Add a new Text file that we will call YourIP.py
And insert this script to it:

```
from urllib import*
from GameLogic import*
Cont = getCurrentController()
Obj = Cont.getOwner()

if GameLogic.PlayerMod == "Server":
	try:
		url = URLopener().open("http://myip.dk")
		html = url.read(300)

		start = html.find("<title>Your IP address is:") + 26
		end = html.find("</title>")

		YourIP = html[start:end].strip()
		print "Server IP =",YourIP
		Obj.Text = YourIP
	except:
		print "No Internet Connection"
		Obj.Text = "No Internet"
```

So and now what dose this Code do?

`from  urllib import*`
Imports the urllib Library

`from GameLogic import*`
Imports the GameLogic Library

`Cont = getCurrentController()`
Gets The Logic Brick that are attached to the Script and saves them to the Variable Cont

`Obj = Cont.getOwner()`
Gets the Object that the Logic Bricks belong to and saves it to the Variable Obj

`if GameLogic.PlayerMod = = "Server":`
Only reads the script if your in Server Mod

`try:`

This command tells the Engine to try to run the next script lines (if you don’t but in a try here and you run this programm an a local network with out Internet connection you will get a error because the falowing script can’t connect to the internet.

```
url = URLopener().open(“http://myip.dk”)
```

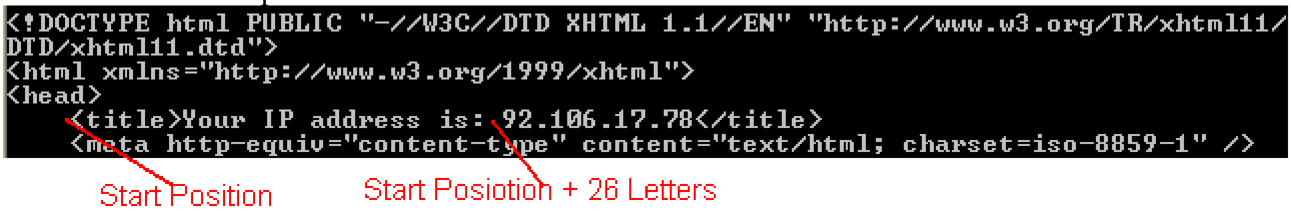
The Command URLopener().open(“http://myip.dk”) opens the url between the (“ and the “) in are case http://myip.dk and saves it to the Variable url.

```
html = url.read(300)
```

The Command url.read(300) reads the first 300 letters of the Information saved to the Variable url and saves this 300 letters to the variable html.

```
start = html.find("<title>Your IP address is:") + 26
```

The Command find("<title>Your IP address is:") searches the information saved to the variable html for the letters between the (“ and the “) in are case <title>Your IP address is
If it finds this letters; it defines the start position were they are written and then the command + 26 will move this position 26 letters forward…



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/
DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>Your IP address is: 92.106.17.78</title>
    <meta http-equiv="content-type" content="text/html; charset=iso-8859-1" />
```

This position is now saved to the variable start

```
end = html.find("</title>")
```

More or less dose the same thing as the line above it searches the information saved in html for the letters </title> if it finds this letters; it defines the start position were they are written and saves this Position to the variable end

```
YourIP = html[start:end].strip()
```

The Command html[start:end] tells the script to read the information saved to html between the Positions defined in the variable start and end. And then it saves that information in to the variable YourIP

```
print "Server IP =", YourIP
```

Prints the information written between the tow " " and the information saved in the variable YourIP in to the consol window.

```
Obj.Text = YourIP
```

Displays the information saved to YourIP on to the Text plane that we will attach to this script in the next step:

```
except:
```

If the script after the try: command failed the except: command will run the script that fallows it.

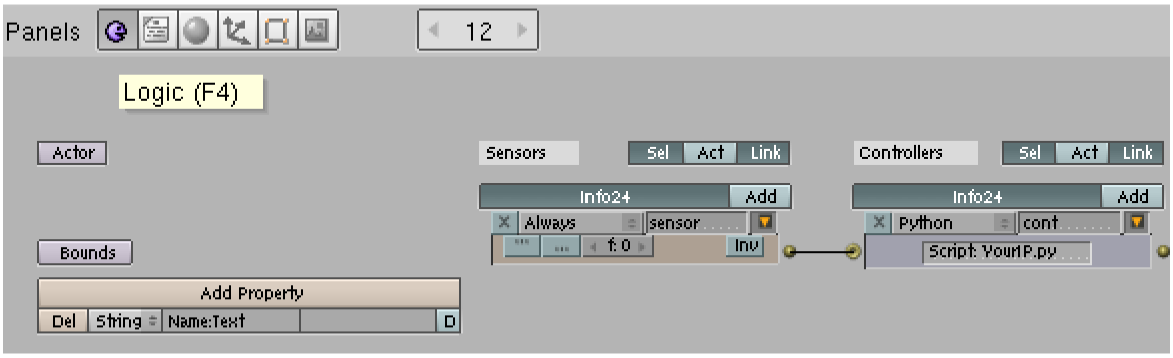
```
print "No Internet Connection"
```

Prints "No Internet Connection" in to the Consol window.

```
Obj.Text = "No Internet"
```

Dispalys the Text "No Internet" on the Text plane that we will attach to this script in the next step:

Now select the Text Plane “Info24” and add the Logic Bricks to it as shown on the Picture Below:



And Voila Now when you play the Server Mode It will display your IP.

So now it’s Time to save again.

With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.15WSAG.

Click on the Save as Button in the right upper Corner

PLEASE DON’T FORGET TO SUPPORT THIS PROJECT

How you can support this project is written at the beginning of this Tutorial

(28)WSAG – Server Mod - Display Your Location (BV 2.44 PV 2.5.1) (LU 23.1.08)

Change to the scene 5BasicGamePlay.

Add a new Text file that we will call YourLocation.py

And insert this script to it:

```
from GameLogic import*
Cont = getCurrentController()
Obj = Cont.getOwner()
if GameLogic.PlayerMod = = "Server":
        Obj.Text = GameLogic.Location
```

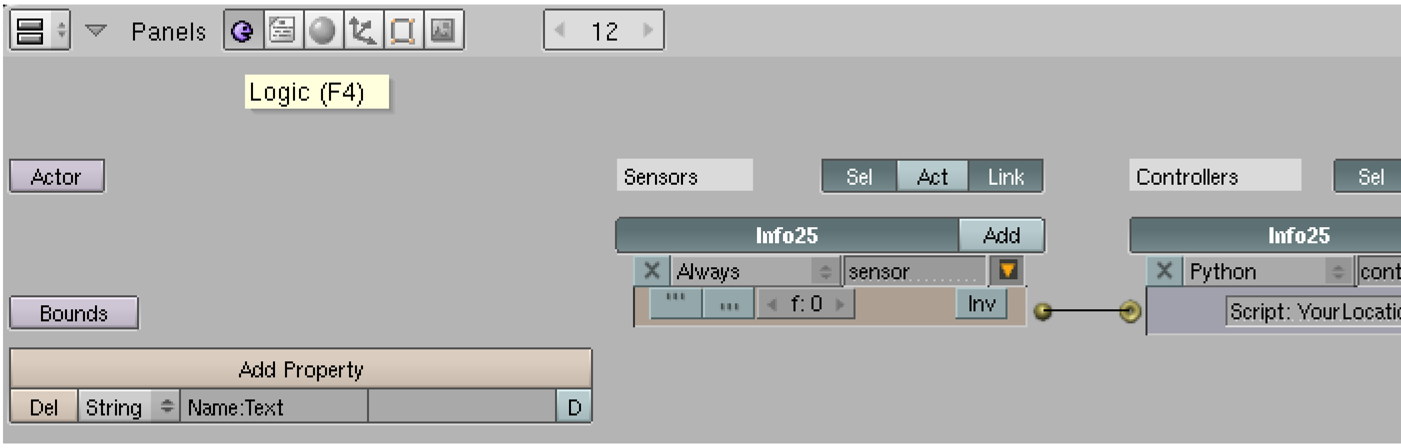
So and now what dose this Code do?

The first 3 Lines should be standard by now

The Line `if GameLogic.PlayerMod = = "Server":` tell the Engine to only run the Script If you are in Server Mod.

The last line calls up the Objects Properties with the name `Text` and Sets them to the Value saved in the Variable `GameLogic.Location`

Now select the Text Plane “Info25” and add the Logic Bricks to it as shown on the Picture Below:



And Voila now when you play the Server Mode It will display your Location.

So now it's Time to save again.

With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.16WSAG.

Click on the Save as Button in the right upper Corner

PLEASE DON'T FORGET TO SUPPORT THIS PROJECT

How you can support this project is written at the beginning of this Tutorial

(29)WSAG – Server Mod – Socket Setup /Server Main Port (BV 2.44 PV 2.5.1) (LU 26.1.08)

Make sure you are in the scene 5BasicGamePlay.
Add a new Text file that we will call ServerMainPort.py
And insert this script to it:

```
from GameLogic import *
from socket import*
if GameLogic.PlayerMod = = Server":
        host =  " "
        port = 1000
        GameLogic.sMain = socket(AF_INET,SOCK_DGRAM)
        GameLogic.sMain.bind((host,port))
        GameLogic.sMain.setblocking(0)
        print "Main Port"  ,port,  "is up"
```

```
from GameLogic import *
from socket import*
```

Imports the Libraries that we will need for this script

```
if GameLogic.PlayerMod = = "Server":
```

Tells the Engine to only start the script below if you are in the Server Mod
This Information has been saved to the Variable GameLogic.PlayerMod in the MainMenu.py script.

```
host =  " "
```

Saves a Empty Information to the Variable host
We leave this information Empty because this way it will automatic use the Computer that it runs on as its Host.

You could enter your Computers Name between the " ",but then It would only run on your Computer, so we leave it empty to keep it Dynamic.
Also you could write localhost between the " ", that would also tell the Script to use the Computer that it runs on as its Host, but why write more then necessary?

```
port = 1000
```

Saves the information 1000 to the Variable port
This will be the Port that this Socket will communicate over. Make sure that you're Firewall and Router won't Block this Prot for your Blender game.
You will have to read the Manual of your Router and Firewall to make sure that all the Ports that we will add in this game are forwarded.

```
GameLogic.sMain = socket(AF_INET,SOCK_DGRAM)
```

The command socket will create a socket with the Protocol type defined between the brackets () and save it to the variable GameLogic.sMain . AF_INET,SOCK_DGRAM stands for the UPD protocol.

```
GameLogic.sMain.bind((host,port))
```

This Command will bind the information of the Variables Saved to host and port to the socket we just have created and saved to the variable GameLogic.sMain.
Inside the double bracket ((,)) the first Information will tell the Socket what Computer it should use to host his socket service from and the second Information tells the Socket over what Port it should communicate…

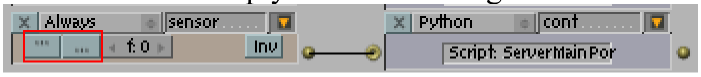
```
 GameLogic.sMain.setblocking(0)
```

The command setblocking tells the Script if it should stop until the socket has correctly operated. If the Number in the brackets if 1 it means that it should stop and if it is set to 0 if means it should not stop the Script. We don't want too stop the script, because if we do this will mean that if the socket has a problem all of the Game will stop working… But if the Script/Engine will just keep on working even if the socket has a problem then the rest of the game will keep on working and there is a good chance that the socket will also start to work again… This way you will not even notice if the socket once fails…

```
print "Main Port"  ,port,  "is up"
```

This command should be clear by now…
In the consol window it will print the information in the fist " " then the Information saved to the variable port and the Information in the second " ".
Note that if you have more then one Information that should be printed on the same line you can list them up with a comma between them.

Now select the Empty and add the Logic Bricks to it as shown on the Picture Below:



Script: ServerMainPor

Make sure the Pulls Mode is turned **OF** so that this script is only stared one Time…(see the buttons in the red marked field)
So now the server has a Port installed that can be addressed by the clients.

So now it's Time to save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.17WSAG.
Click on the Save as Button in the right upper Corner

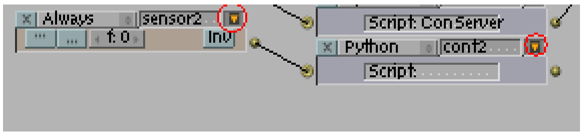
PLEASE DON’T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(30)WSAG – ClientMod/ServerMod –Lists and Dictionaries(BV 2.44 PV 2.5.1) (LU 26.1.08)

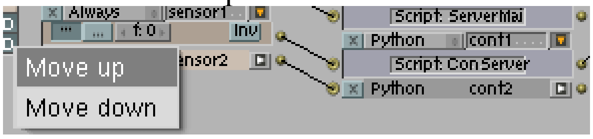
Go to the Scene 4GamePlay
Open a new Text file and name it Lists.py and insert this script to it:

```
from GameLogic import *
GameLogic.PlayerList = []
GameLogic.PlayerNameList = []
GameLogic.PlayerDict = {}
GameLogic.PortList = []
GameLogic.PlayerNameList = []
```

Now select the Empty and add a:
Always Sensor with the Pulse Mode of
Python Controller that executes the script Lists.py
Connect the two with each other.
Now minimize the Sensor and the Controller by clicking on the little triangle marked in the Picture below.



Now if you click (left Mouse button) one of the Minimized Logic Bricks you get a Menu to move the Brick up or down.



Move it totally to the Top this way it will be executed first…

So and here we get two totally new things the [] and the {}.
The [] tells the script that this is a list of information.
And the {} tells the Script that this is a Dictionary of Information.
These Lists and Dictionaries will be used In the Next few scripts to come…
So you may ask yourself why we but this Lists and Dictionaries here and not in the script were they are used.
The answer is simply this Script here runs only one time at start up. The other script will run all the time. Since there is noting written between the [] and the {} it would tell the script each time it reads this code that the list / Dictionary is empty…
What the different Lists and Dictionaries are used for will be explained in the Script were they are used.

XXXX Description of what is a List / Dictionary and what’s the difference between the two.

So now it’s Time to save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.18WSAG.
Click on the Save as Button in the right upper Corner
PLEASE DON’T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

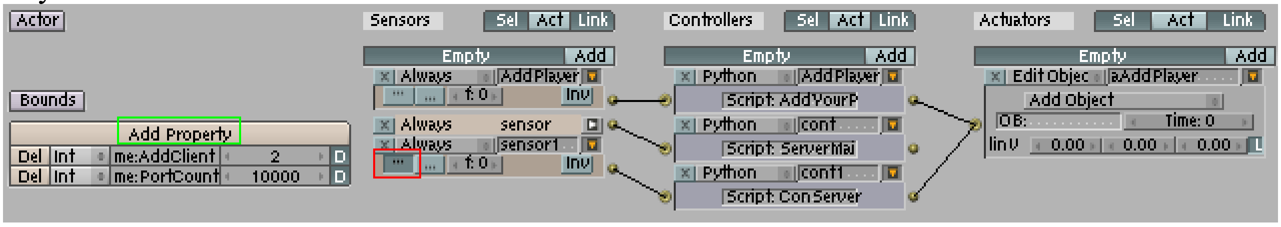
(31)WSAG – Server Mod – Connect and add Clients (BV 2.44 PV 2.5.1) (LU 23.1.08)

Make sure you are in the scene 5BasicGamePlay.
Add a new Text file that we will call ConServer.py
And insert this script to it:

```
from GameLogic import *
from socket import *
from cPickle import *
cont = GameLogic.getCurrentController()
Obj = cont.getOwner()

if GameLogic.PlayerMod == "Server":

#-----Reseive New Client-----
	try:
		RData, CLIP = sMain.recvfrom(1024)
		Data = loads(RData)
		PlayerName = Data[0]
		PlayerTyp = Data[1]
		print "---Conecting New Client---"
		if PlayerName not in GameLogic.PlayerNameList:
			print "Player Name =", PlayerName
			print "Player Typ =", PlayerTyp
			print CLIP
			GameLogic.PlayerNameList.append(PlayerName)
			Obj.PortCount = Obj.PortCount + 1
			Data = dumps((Obj.PortCount,GameLogic.LocationC))
			GameLogic.sMain.sendto(Data, CLIP)
			Obj.AddClient = 1
		else:
			print "Playername", Data[0], "already in List"
			print "-------------------------"
			Info = "Playername already in use"
			Data = dumps((Info))
			GameLogic.sMain.sendto(Data, CLIP)
	except:
		pass

#-----Add New Client-----

	if Obj.AddClient == 1:
		host = ''
		port = Obj.PortCount
		GameLogic.sPort = socket(AF_INET,SOCK_DGRAM)
		GameLogic.sPort.bind((host,port))
		GameLogic.sPort.setblocking(0)
		AddPlayerAct = cont.getActuator("aAddPlayer")
		AddPlayerAct.setObject(PlayerTyp)
		addActiveActuator(AddPlayerAct,1)
```

```
        GameLogic.PortList.append(GameLogic.sPort)
        print "Port"  ,port,  "is up"
        print "-------------------------"
        Obj.AddClient = 2
```

Now select the Empty and add the Logic Bricks to it as shown on the Picture Below:
Make sure the Pulls Mode is turned **ON** so that this script is only stared one Time…(see the buttons in the red marked field)
Now click on the Add Property Button 2x (see Green Mark in the Picture below) And set it up as you can see in the Picture below.



Del Int me:AddClient 2 D
Del Int me:PortCount 10000 D

Sensors: Always – AddPlayer, f: 0; Always – sensor; Always – sensor1, f: 0
Controllers: Python – AddPlayer, Script: AddYourP; Python – cont, Script: ServerMai; Python – cont1, Script: ConServer
Actuators: Edit Objec – aAddPlayer, Add Object

So now the Server not only has a Port that the Clients connect to but can also Communicate with the Client and Add the Client to the Play scenario…

So now it's Time to save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.19WSAG.
Click on the Save as Button in the right upper Corner
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(32)WSAG – ClientMod - Player Selection (BV 2.44 PV 2.5.1) (LU 21.3.08)
Will start with the basic setup for the Client…
So you change to the 2StartUpMenu scene there you see that if you press the C-Key it should start the Client Mode. In the Text Window we call up the Script
MainMenu.py
There we add these lines:

```
if Keyinput == [[99, 1]]:
        print "Client"
        GameLogic.PlayerMod = "Client"
        SetSceneAct.setScene("3ePlayerSelection")
```

In the Text Window we call up the Script
3ePlayerSelection.py

There we add these Lines too the script:

```
if GameLogic.PlayerMod == "Client":
        if Keyinput == [[98, 1]]:
                print "Player = Blue"
                GameLogic.PlayerTyp = "ContBlue"
                GameLogic.PlayerTypS = "BlueX"
                SetSceneAct.setScene("4cPlayerName")
```

```
	if Keyinput == [[114, 1]]:
		print "Player = Red"
		GameLogic.PlayerTyp = "ContRed"
		GameLogic.PlayerTypS = "RedX"
		SetSceneAct.setScene("4cPlayerName")

	if Keyinput == [[103, 1]]:
		print "Player = Green"
		GameLogic.PlayerTyp = "ContGreen"
		GameLogic.PlayerTypS = "GreenX"
		SetSceneAct.setScene("4cPlayerName")

addActiveActuator(SetSceneAct,1)
```

After reading the **(17)WSAG – Explorer Mod - Player Selection / Bind to** it should be clear what this lines do…
We need the `GameLogic.PlayerTypS` because this will be the info that we will send to the Server and the server should only swan the visual shell of the clients with out the controls…
That's why we can't send him the info from `GameLogic.PlayerTyp`

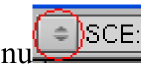
The Howl 3ePlayerSelection.py will now look like this:

```
from GameLogic import*
Cont = getCurrentController()
KeySens = Cont.getSensor("sKeyInput")
Keyinput = KeySens.getPressedKeys()
SetSceneAct = Cont.getActuator("aSetScene")

if GameLogic.PlayerMod == "Explorer":
	if Keyinput == [[98, 1]]:
		print "Player = Blue"
		GameLogic.PlayerTyp = "ContBlue"
		SetSceneAct.setScene("4eLocationSelection")

	if Keyinput == [[114, 1]]:
		print "Player = Red"
		GameLogic.PlayerTyp = "ContRed"
		SetSceneAct.setScene("4eLocationSelection")

	if Keyinput == [[103, 1]]:
		print "Player = Green"
		GameLogic.PlayerTyp = "ContGreen"
		SetSceneAct.setScene("4eLocationSelection")

if GameLogic.PlayerMod == "Client":
	if Keyinput == [[98, 1]]:
		print "Player = Blue"
		GameLogic.PlayerTyp = "ContBlue"
		GameLogic.PlayerTypS = "Blue"
		SetSceneAct.setScene("4cPlayerName")

	if Keyinput == [[114, 1]]:
        print "Player = Red"
        GameLogic.PlayerTyp = "ContRed"
        GameLogic.PlayerTypS = "Red"
        SetSceneAct.setScene("4cPlayerName")

    if Keyinput == [[103, 1]]:
        print "Player = Green"
        GameLogic.PlayerTyp = "ContGreen"
        GameLogic.PlayerTypS = "Green"
        SetSceneAct.setScene("4cPlayerName")

addActiveActuator(SetSceneAct,1)
```

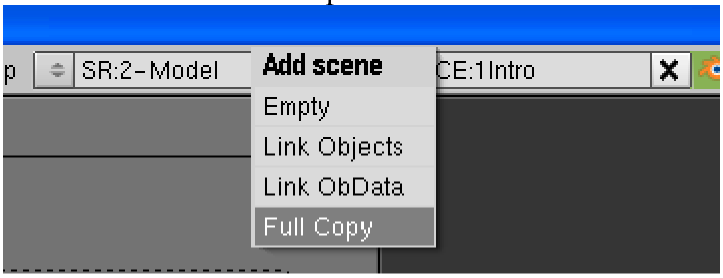
XXXX = Explaining will be Inserted here later…

So now it’s Time to save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.20WSAG.
Click on the Save as Button in the right upper Corner
PLEASE DON’T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(33)WSAG – ClientMod –Player Name/Dynamic text input Display (BV 2.44 PV 2.5.1) (LU 21.1.08)

In The last script we wrote we tell the script to change to the scene 4cPlayerName
This Scene we will now create:
Change to the scene 1Intro and now we want to add a new scene by clicking on the left side of the SCE Popup menu  and clicking on ADD NEW.

Now a menu will come up that looks like this:



We will use the Full Copy. It will give us a new scene with all the objects of your current scene. That way we won’t have to rebuild the Text plane….
So click in Full Copy and name the new Scene 4cPlayerName

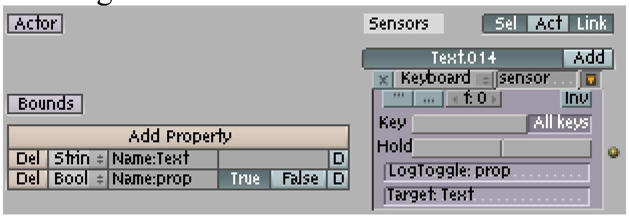
Now click on the Bigger Text Plane and in the Logic Bricks you change the Text “Intro” to “Enter Your Player Name”

Now with the Plane still selected (its lines are Purple) and the Mouse in the 3D Window you press the N-Key and change these Settings:
LocX = - 6.500
LocY = 1.000
LocZ = 3.000

DimX = 1.500
DimY = 1.500

Now press the Shift-Key and D-Key at the same time, this will duplicate the Object.
Make a Left Mouse Click so that the Plane is selected, then press the N-Key and change these Settings:
LocX = - 6.500
LocY = 1.000
LocZ = -3.000
DimX = 1.500
DimY = 1.500
In the Logic Bricks you change the Text “Enter Your Player Name” to “Press Enter to Confirm”

Now select the small Text Plane and press the N-Key and change these Settings:
LocX = - 3.500
LocY = 1.000
LocZ = 0.000
DimX = 1.000
DimY = 1.000
So now we want to make this Plane Display the Letters that you Type:
In the Logic Bricks you delete the Text “Press Space to skip Intro” so that the field is empty
Now add an other Property (Bool) with the name prop and select it to true.
Now add a Sensor (keyboard) and click on the All Keys button.
In the LogToggel field you write: prop
In the Target feld you write: text
The Logic Bricks should look like this now:



Name:Text
Name:prop
LogToggle: prop
Target: Text

So nw the small Text Plane will display what your Typing.
Next Step is to record the enter text to a Python script so that we can make us of it.

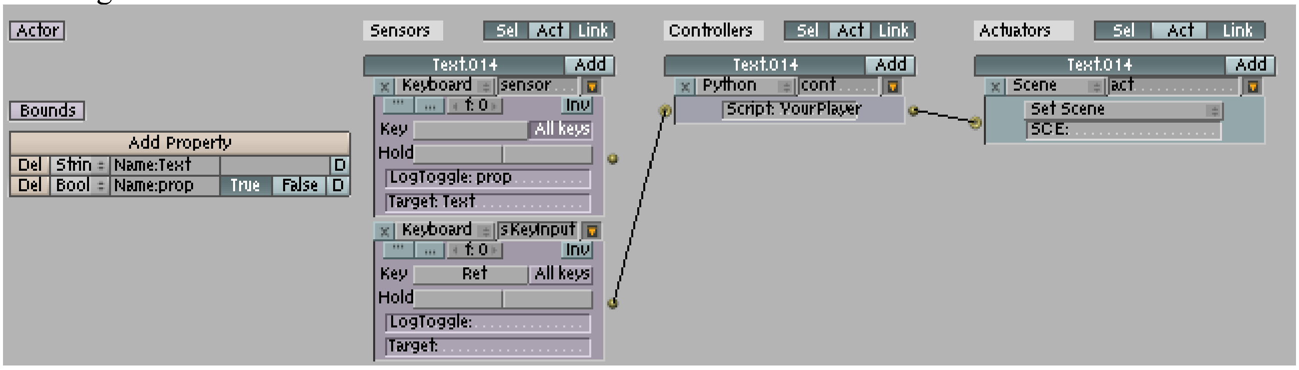
Open a new Text field and call it YourPlayerName.py
And ad this Script to it

```
from GameLogic import*
Cont = getCurrentController()
KeySens = Cont.getSensor("sKeyInput")
Keyinput = KeySens.getPressedKeys()
SetSceneAct = Cont.getActuator("act")
Obj = Cont.getOwner()

if Keyinput == [[13, 3]]:
	GameLogic.YourPlayerName = Obj.Text

	print "Your Name =", Obj.Text

	SetSceneAct.setScene("4cIPinput")
```

addActiveActuator(SetSceneAct,1)

By now it should be clear what this Script dose… The text you typed was saved to the Variable Obj.Text
And now we save that Information to the Variable GameLogic.YourPlayerName so that we can always have access to it.
And then goes to the Scene that we set with the in the brackets behind the Variable SetSceneAct.setScene
Then Last line then activates the Actuator with the Name act
The if Keyinput = = [[13, 1]]: is here so that this script is only rune one Time when you let go of the Enter Key and not already when you press the Enter Key. Other wise It would run the Script the first time when you press the Enter-Key and change the Scene now it would run a second time and give you an Error because it cant find the Sensors that the script calls on because they are not in the new scene anymore.

The next Step is to define when the typed Information should be recorded to the Variable GameLogic.YourPlayerName
We can do this simply by:
- Adding a Sensor (Keybord) that will react on the Enter-Key. Make sure you name this Sensor sKeyInput otherwise the script won’t find it.
- And adding a Controller (Python) with the Script: YourPlayerName.py
- Now add an Actuator (Scene) set it to “Set Scene”
- Combine the Yellow dots of The Sensor, Controller and Actuator
The Logic Bricks now look like this:



Del Strin Name:Text
Del Bool Name:prop True False

Keyboard sensor
Key All keys
Hold
LogToggle: prop
Target: Text

Keyboard sKeyInput
Key Ret All keys
Hold
LogToggle:
Target:

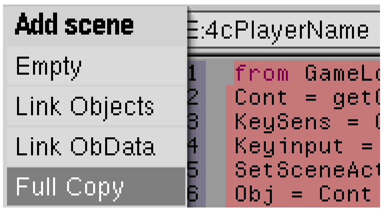
Python cont
Script: YourPlayer

Scene act
Set Scene
SCE:

So now it’s Time to save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.21WSAG.
Click on the Save as Button in the right upper Corner
PLEASE DON’T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(34)WSAG – ClientMod –IP Input /Connect to Server / Dynamic Info Text(BV 2.44 PV 2.5.1) (LU 20.2.08)

In the last Chapter we wrote a script that tells the Engine to change to the Scene 4cIPinput
So we will have to create that scene now.
Make a full Copy of the 4cPlayerName scene



```
from GameLo
Cont = get(
KeySens = (
Keyinput =
SetSceneAct
Obj = Cont
```

Open a New Text file and Name it ConClient.py

And add this Script to it:

```
from GameLogic import *
from socket import *
from cPickle import *
Cont = getCurrentController()
KeySens = Cont.getSensor("sKeyInput")
Keyinput = KeySens.getPressedKeys()
SetSceneAct = Cont.getActuator("act")
Obj = Cont.getOwner()

if Keyinput == [[13, 1]]:
        if Obj.Next == 0:
                Obj.Next = 1
                try:
                        Serverhost = Obj.Text
                        Serverport = 10000
                        Clienthost = ''
                        Clientport = 50000
                        GameLogic.sClient = socket(AF_INET,SOCK_DGRAM)
                        GameLogic.sClient.bind((Clienthost,Clientport))
                        GameLogic.host = (Serverhost,Serverport)
                        GameLogic.sClient.setblocking(0)
                        Data = dumps((GameLogic.YourPlayerName,GameLogic.PlayerTypS))
                        GameLogic.sClient.sendto(Data,GameLogic.host)
                        Obj.Connecting = 120
                        print "Connecting to Server:", Obj.Text
                        GameLogic.Info1 = "Connecting to Server"
                except:
                        print "Cant access requested IP"
                        GameLogic.Info1 = "Cant access requested IP"
                        GameLogic.Info2 = "Press Enter to Retry"
                        GameLogic.Info3 = "Press Backsace to enter new IP"
                        Obj.Next = 0

if Obj.Next == 1:
        if Obj.Connecting > 0:
                Obj.Connecting = Obj.Connecting - 1
                try:
                        Data, SRIP = sClient.recvfrom(1024)
                        Info = loads(Data)
                        if Info == "Playername already in use":
                                print Info
                                GameLogic.Info1 = "Playername already in use"
                                GameLogic.Info2 = ""
                                GameLogic.Info3 = ""
                                Obj.Text = ""
                                SetSceneAct.setScene("4cPlayerNameR")
                        else:
                                print "Received PortNR:",Info[0], "From Server"
                                GameLogic.Info1 = "Received PortNR from Server"
                                Obj.Connecting = 0
                                GameLogic.sClient.close()

                                Serverhost = Obj.Text
                                Serverport = Info[0]
                                Clienthost = ''
                                Clientport = Info[0] + 40000
                                GameLogic.sClient = socket(AF_INET,SOCK_DGRAM)
                                GameLogic.sClient.bind((Clienthost,Clientport))
                                GameLogic.host = (Serverhost,Serverport)
                                GameLogic.sClient.setblocking(0)
                                print "Port" ,Clientport, "is Up"
                                GameLogic.AddYou = 0
                                SetSceneAct.setScene(Info[1])
                except:
                        pass
        if Obj.Connecting == 1:
                GameLogic.sClient.close()
                Obj.Connecting = 0
                print "Connection Failed"
                GameLogic.Info1 = "Connection Failed"
                GameLogic.Info2 = "Press Enter to Retry"
                GameLogic.Info3 = "Press Backsace to enter new IP"
                Obj.Next = 0

if Keyinput == [[133, 1]]:
        GameLogic.Info1 = ""
        GameLogic.Info2 = ""
        GameLogic.Info3 = ""
        Obj.Text = ""


addActiveActuator(SetSceneAct,1)
```

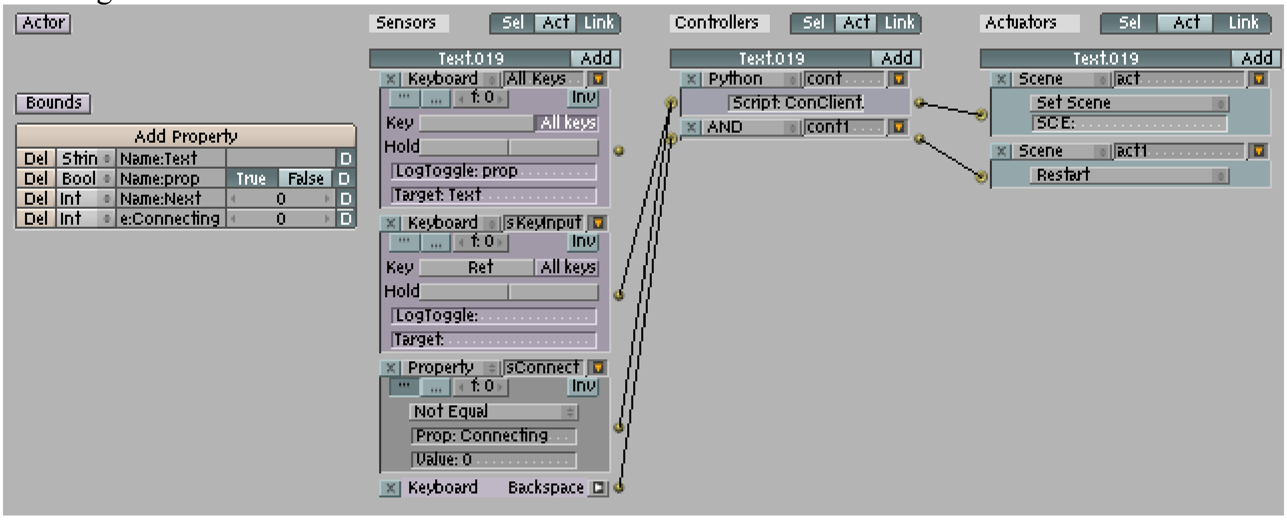
XXXX = Explaining will be Inserted here later…

Now select the Top Plane (its lines are Purple) and the Mouse in the 3D Window you press the N-Key and change these Settings:
LocX = - 4.500
LocY = 1.000
LocZ = 4.500
DimX = 1.500
DimY = 1.500
In the Logic Bricks you change the Text “Enter Your Player Name” to “Enter Server IP”

Now select the Middle Plane (its lines are Purple) and the Mouse in the 3D Window you press the N-Key and change these Settings:
LocX = - 2.000
LocY = 1.000
LocZ = 3.000
DimX = 1.000
DimY = 1.000
In the Logic Bricks you change the Controller (Python) from YourPlayerName.py to IPinput.py
And add a Property(int) that we call Next.
An also a Property(int) that we call Connecting.
And add a Property Sensor => Not Equal => Prop: Connecting => Value: 0 with the PulseMode on and connect it with the Python Controller that will start the IPinput.py script If the Value of the Connecting Property is not = 0

And add a Keyboard Sensor => Enter-Key (Return-Key) with the PulseMode off and connect it with the Python Controller that will start the IPinput.py script every time you press or let go if the Return Key.

Add a Keybord Sensor => Backspace-Key with the PulseMode of
And a And Controller
And a Scene Actuator => Restart
Connect these 3 with each other…

The Logic Bricks should look like this now:


Now select the lowest Plane (its lines are Purple) and the Mouse in the 3D Window you press the N-Key and change these Settings:
LocX = - 6.500
LocY = 1.000
LocZ = 1.000
DimX = 1.500
DimY = 1.500
Leave the Logic Bricks as they are.

With the lowest Plane still selected you press the Shift-Key and D-Key at the same time, this will duplicate the Object.
Make a Left Mouse Click so that the Plane is selected, then press the N-Key and change these Settings:
OB: Info1
LocX = - 6.500
LocY = 1.000
LocZ = -1.000
DimX = 1.500
DimY = 1.500
In the Logic Bricks you delete the Text “Press Enter to Confirm” so that the field is empty
Open a new Text file and name it Info1.py and insert this script to it:

```
from GameLogic import *
cont = GameLogic.getCurrentController()
Obj = cont.getOwner()
try:
	Obj.Text = GameLogic.Info1
except:
	pass
```

This Script will make the Text Plane any Information saved to the Variable GameLogic.Info1
The rest of the script should be clear by now.
And add a always Sensor with the PulseMode on
And a Python Controller that will Open the Script: Info1.py

Copy the Plane again and make a Left Mouse Click so that the Plane is selected, then press the N-Key and change these Settings:
OB: Info2
LocX = - 6.500
LocY = 1.000
LocZ = -2.800
DimX = 1.500
DimY = 1.500
In the Logic Bricks you delete the Text “Press Enter to Confirm” so that the field is empty
Open a new Text file and name it Info2.py and insert this script to it:

```
from GameLogic import *
cont = GameLogic.getCurrentController()
Obj = cont.getOwner()
try:
	Obj.Text = GameLogic.Info2
except:
	pass
```

This Script will make the Text Plane any Information saved to the Variable GameLogic.Info1
The rest of the script should be clear by now.
And add a always Sensor with the PulseMode on
And a Python Controller that will Open the Script: Info2.py

Copy the Plane again and make a Left Mouse Click so that the Plane is selected, then press the N-Key and change these Settings:
OB: Info3
LocX = - 6.500

LocY = 1.000
LocZ = -4.600
DimX = 1.500
DimY = 1.500

In the Logic Bricks you delete the Text “Press Enter to Confirm” so that the field is empty

Open a new Text file and name it Info3.py and insert this script to it:

```
from GameLogic import *
cont = GameLogic.getCurrentController()
Obj = cont.getOwner()
try:
	Obj.Text = GameLogic.Info3
except:
	pass
```

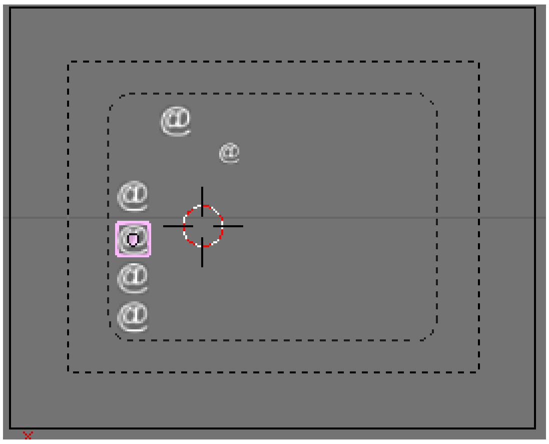
This Script will make the Text Plane any Information saved to the Variable GameLogic.Info1

The rest of the script should be clear by now.

And add a always Sensor with the PulseMode on

And a Python Controller that will Open the Script: Info3.py

Your screen should look like this now:



So now it’s Time to save again.

With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.22WSAG.

Click on the Save as Button in the right upper Corner

PLEASE DON’T FORGET TO SUPPORT THIS PROJECT

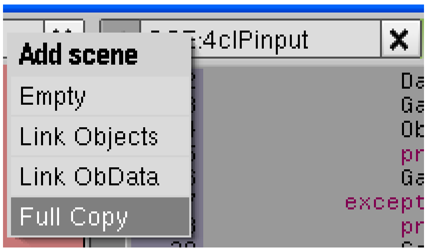
How you can support this project is written at the beginning of this Tutorial

(35)WSAG – ClientMod –Player Name already in use on Server(BV 2.44 PV 2.5.1) (LU 20.2.08)

In the Last Chapter We tell the Script to change to the Scene 4cPlayerNameR if already a Client with the same Player name you gave in, is logged on to the Server.

So now e will have to create this Scene.

Make a full Copy of the Scene 4cIPinput and name it 4cPlayerNameR



Select the Top Text Plane and Change the Text in the Logic Bricks from “Enter Server IP” to “Enter New Player Name”
With the Plane still selected press the N-Key and change to this Values:
LocX = - 6.500
LocY = 1.000
LocZ = 4.500
DimX = 1.500
DimY = 1.500
Close the Transform Property Window by Clicking on the X in the left upper corner.

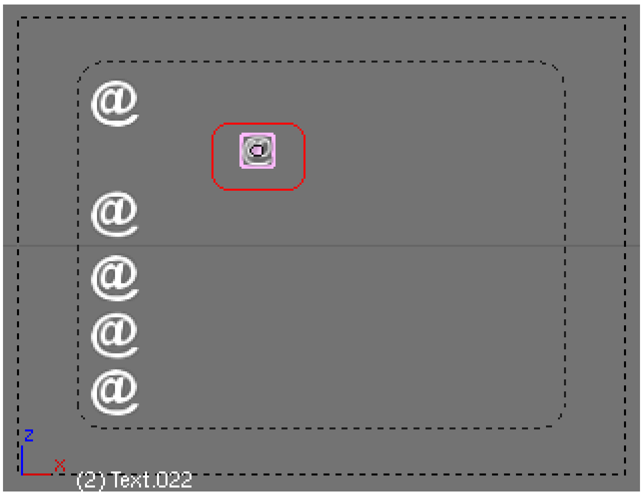
Open a new Text file and name it PlayerNameR.py and insert this script to it:

```
from GameLogic import*
from socket import *
from cPickle import *
Cont = getCurrentController()
KeySens = Cont.getSensor("sKeyInput")
Keyinput = KeySens.getPressedKeys()
SetSceneAct = Cont.getActuator("act")
Obj = Cont.getOwner()

if Keyinput == [[13, 3]] and Obj.Text != "":
    if Obj.Next == 0:
        Obj.Next = 1
        GameLogic.YourPlayerName = Obj.Text
        print "Your New Name =", GameLogic.YourPlayerName
        try:
            Data = dumps((GameLogic.YourPlayerName,GameLogic.PlayerTypS))
            GameLogic.sClient.sendto(Data,GameLogic.host)
            Obj.Connecting = 120
        except:
            print "Cant access Server"
            GameLogic.Info1 = "Cant access Server"
            GameLogic.Info2 = "Press Enter to Retry"
            GameLogic.Info3 = "Press F1 to enter new IP"
            Obj.Next = 0


if Obj.Next == 1:
    if Obj.Connecting > 0:
        Obj.Connecting = Obj.Connecting - 1
        try:
            Data, SRIP = sClient.recvfrom(1024)
            Info = loads(Data)
            if Info == "Playername already in use":
                print Info
```

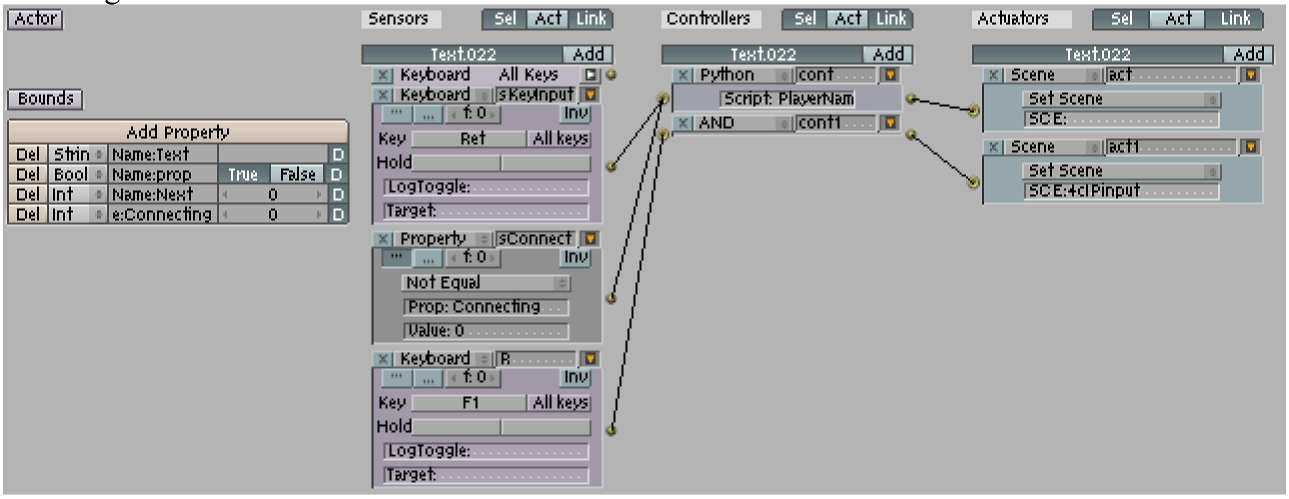
```
                        GameLogic.Info1 = "Playername already in use"
                        GameLogic.Info2 = "Please Enter an other Name"
                        GameLogic.Info3 = ""
                        Obj.Text = ""
                        Obj.Connecting = 0
                        Obj.Next = 0
                else:
                        print "Received PortNR:",Info[0], "From Server"
                        GameLogic.Info1 = "Received PortNR from Server"
                        Obj.Connecting = 0
                        GameLogic.sClient.close()

                        Serverhost = Obj.Text
                        Serverport = Info[0]
                        Clienthost = ''
                        Clientport = Info[0] + 40000
                        GameLogic.sClient = socket(AF_INET,SOCK_DGRAM)
                        GameLogic.sClient.bind((Clienthost,Clientport))
                        GameLogic.host = (Serverhost,Serverport)
                        GameLogic.sClient.setblocking(0)
                        print "Port" ,Clientport, "is Up"
                        GameLogic.AddYou = 0
                        SetSceneAct.setScene(Info[1])
        except:
                pass

if Obj.Connecting == 1:
        Obj.Connecting = 0
        print "Transmission Failed"
        GameLogic.Info1 = "Transmission Failed"
        GameLogic.Info2 = "Press Enter to Retry"
        GameLogic.Info3 = "Press F1 to enter new IP"
        Obj.Next = 0

if Keyinput == [[114, 1]]:
        GameLogic.Info1 = ""
        GameLogic.Info2 = ""
        GameLogic.Info3 = ""
        Obj.Text = ""


addActiveActuator(SetSceneAct,1)
```

XXXX = Explaining will be Inserted here later…

Select the Second Text Plane from the Top.



And in its Logic Brick section:
Change the Python Controller from ConClient.py to PlayerNameR.py
Change the Keyboard Controller that reacts to the Backspace-Key so that it reacts to the F1-Key.
Change the Actuator that restarts the Scene to Set Scene => 4cIPinput.
The Logic Bricks should look like this now:



So now it’s Time to save again.
With the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.23WSAG.
Click on the Save as Button in the right upper Corner

PLEASE DON’T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(37)WSAG – ServerMod/ClientMod–Register to Player list (BV 2.44 PV 2.5.1) (LU 20.2.08)

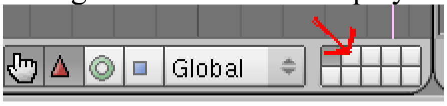
Open a new Text file and name it RegisterToList.py and insert this script to it:

```
from GameLogic import*

Cont = getCurrentController()
Obj = Cont.getOwner()

if GameLogic.PlayerMod == "Client":
	if GameLogic.AddYou == 1:
```

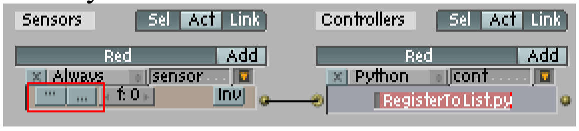
```
            GameLogic.PlayerDict[GameLogic.NewPlayer] = Obj
            print "------------------------"
            print "New Player Connected:", GameLogic.NewPlayer
            print "------------------------"
            GameLogic.AddNew = 1
    else:
            GameLogic.AddYou = 1
            GameLogic.PlayerDict[GameLogic.YourPlayerName] = Obj
            GameLogic.AddNew = 1
if GameLogic.PlayerMod == "Server":
    GameLogic.PlayerList.append(Obj)
```

XXXX = Explaining will be Insert here later…

Change to the 5BasicGameplay Scene. Switch to the 2. Layer



There you select the Red Cube and add this Logic Bricks to it:



Make sure the Pulls mode is of (see red circle in the Picture above) for this Script should only run one Time.

Do the same thing to the Green and Blue Cube.

So now it's Time to save again.

Whit the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.24WSAG.

Click on the Save as Button in the right upper Corner

PLEASE DON'T FORGET TO SUPPORT THIS PROJECT

How you can support this project is written at the beginning of this Tutorial

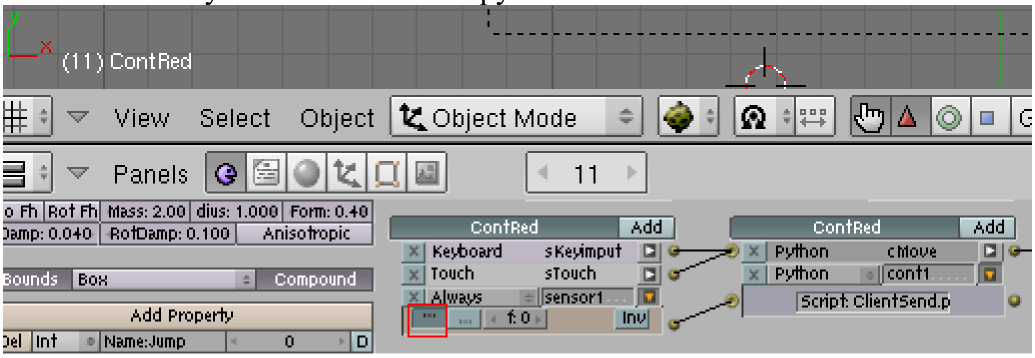
(38)WSAG –ClientMod– Send Orientation/Position (BV 2.44 PV 2.5.1) (LU 26.2.08)

Open a new Text file and name it ClientSend.py and insert this script to it:

```
from GameLogic import *
#from struct import *
from socket import *
from cPickle import *
if GameLogic.PlayerMod == "Client":
    cont = GameLogic.getCurrentController()
    obj = cont.getOwner()
    PosClient = obj.getPosition()
    OriClient = obj.getOrientation()
    Data =
dumps((GameLogic.YourPlayerName,GameLogic.PlayerTypS,PosClient,OriClient))
    GameLogic.sClient.sendto(Data,GameLogic.host)
```

XXXX = Explaining will be Insert here later…

There you select the Red Empty (ContRed) and add this Logic Bricks to it:
Sensor =>Always
Controller => Python => ClientSend.py



Note that the Pulls Mode Always Sensor is Turned on. Because we have to continuously send the Orientation and Position of the Clients.

So now it’s Time to save again.
Whit the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.25WSAG.
Click on the Save as Button in the right upper Corner
PLEASE DON’T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(39)WSAG –ServerMod– Receive/Send Orientation/Position of Client’s (BV 2.44 PV 2.5.1) (LU 4.2.08)

Open a new Text file and name it ServerReceive.py and insert this script to it:

```
#---------------IMPORT-----------------#
from GameLogic import *
from socket import *
from cPickle import *
cont = GameLogic.getCurrentController()
obj = cont.getOwner()
#----------------Start Setup------------------#
if GameLogic.PlayerMod == "Server":


#----------------DATA------------------#
        PosClient = [0,0,0]
        OriClient = [[0,0,0],[0,0,0],[0,0,0]]
        GameLogic.IPList =[]
        DataList =[]
        DataPack =[]
        DataSize = 0
#------------------RECEIVE-------------------#

        for Port,Player in zip(GameLogic.PortList,GameLogic.PlayerList):
                try:
                        Data, CLIP = Port.recvfrom(1024)
                        GameLogic.IPList.append(CLIP)
#---------------Set Game Play----------------#
```

```
                        UPD = loads(Data)
                        DataList.append(UPD)
                        DataSize = DataSize + 1
                        if DataSize == 3:
                                DataPack.append(DataList)
                                DataList = []
                                DataSize = 0
                        PosClient = [UPD[2][0],UPD[2][1],UPD[2][2]]
                        OriClient = [UPD[3][0],UPD[3][1],UPD[3][2]]
                        Player.setPosition(PosClient)
                        Player.setOrientation(OriClient)

                except:
                        pass
#-----------------SEND------------------#
        if DataSize  != 0:
                DataSend1 = dumps((DataList))
                for CLIP in GameLogic.IPList:
                        try:
                                GameLogic.sMain.sendto(DataSend1, CLIP)
                        except:
                                pass
        for Data in DataPack:
                DataSend2 = dumps((Data))
                for CLIP in GameLogic.IPList:
                        try:
                                GameLogic.sMain.sendto(DataSend2, CLIP)
                        except:
                                pass
```

XXXX = Explaining will be Inserted here later…

Change to the 5BasicGameplay Scene. Make sure your In Layer 1.
Select the Empty
And Add an Always Sensor (pulls Mode On) to the Logic Bricks.
And Add a Python Controller whit the Script ServerReceive.py.
Connect them with each other.
So now it's Time to save again.
Whit the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.26WSAG.
Click on the Save as Button in the right upper Corner
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(40)WSAG –ClientMod– Receive Client's Orientation/Position from Server (BV 2.44 PV 2.5.1) (LU 26.2.08)
Open a new Text file and name it ClientReceive.py and insert this script to it:

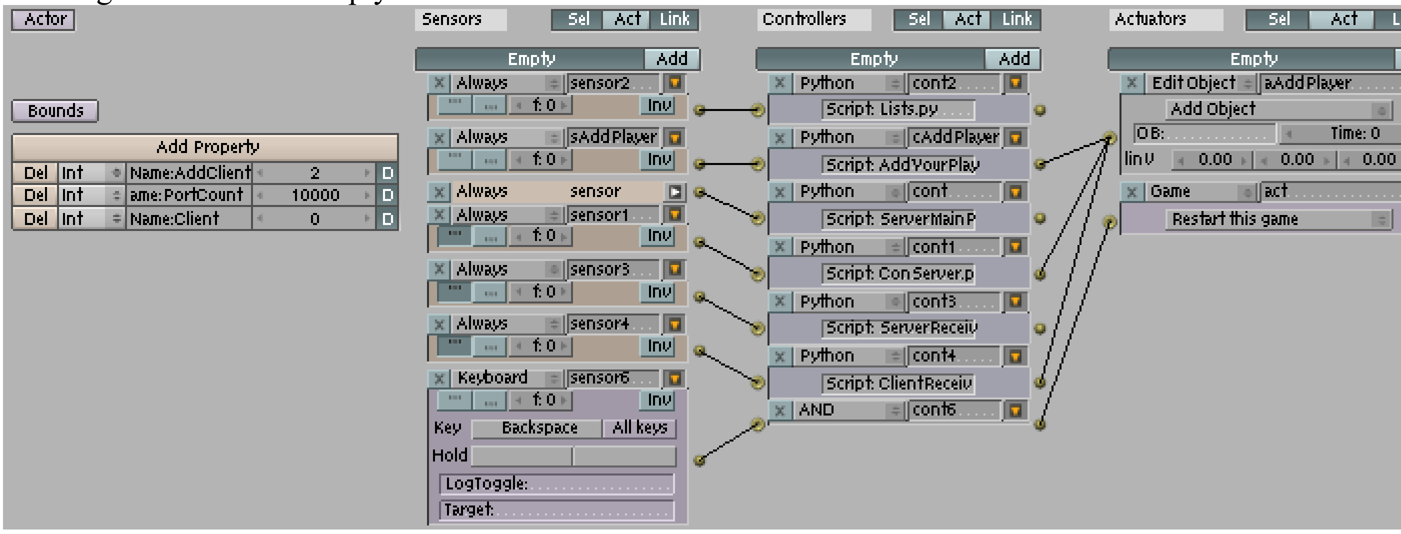
```
#--------------IMPORT----------------#
from GameLogic import *
from socket import *
from cPickle import *
cont = GameLogic.getCurrentController()
obj = cont.getOwner()
#----------------DATA------------------#
if GameLogic.PlayerMod == "Client":
    PosClientX = [0,0,0]
    OriClientX = [[0,0,0],[0,0,0],[0,0,0]]

#------------RECEIVE-------------#
    Client = 1
    try:
        Data, SRIP = sClient.recvfrom(1024)
    except:
        Client = 0
    if Client ==1:
        if Data:
            UData = loads(Data)
            for PData in UData:
                if PData[0] in GameLogic.PlayerDict:
                    #if PData[0] == GameLogic.YourPlayerName:
                        #print "------------------"
                        #print obj.Time
                        #obj.Time = 0.0
                        #print "------------------"
                    if PData[0] != GameLogic.YourPlayerName:
                        PosClientX = [PData[2][0],PData[2][1],PData[2]
[2]]
                        OriClientX = [PData[3][0],PData[3][1],PData[3]
[2]]

    GameLogic.PlayerDict[PData[0]].setPosition(PosClientX)

    GameLogic.PlayerDict[PData[0]].setOrientation(OriClientX)
                if PData[0] not in GameLogic.PlayerDict:
                    if GameLogic.AddNew == 1:
                        GameLogic.NewPlayer = PData[0]
                        AddNewPlayer =
cont.getActuator("aAddPlayer")
                        AddNewPlayer.setObject(PData[1])
                        addActiveActuator(AddNewPlayer,1)
                        GameLogic.AddNew = 0


        else:
            GameLogic.s.close()
            obj.Client = 1
#---------------THE-END---------------#
```

XXXX = Explaining will be Inserted here later…

Change to the 5BasicGameplay Scene. Make sure your In Layer 1.
Select the Empty
And Add an Always Sensor (pulls Mode On) to the Logic Bricks.
And Add a Python Controller whit the Script ClientReceive.py.
Connect them with each other.
So now it's Time to save again.
Whit the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.27WSAG.
Click on the Save as Button in the right upper Corner
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(41)WSAG –Restart Gameplay (BV 2.44 PV 2.5.1) (LU 26.2.08)

Select The Empry in the 5BasicGameplay and Add this Logic Bricks to it:
Sensor => Keybord => Backspace-Key
Controller => AND
Actuator => Game => Restart this Game
The Logic Brick of the Empty should now look like this:



Now if you press the Backspace-Key in game Play the Game will restart.

(42)WSAG –Transmitting Shoot Signal between Clients (BV 2.44 PV 2.5.1) (LU 27.3.08)

Open the Script Lists.py and add this 2 lines

```
GameLogic.ShootDict = {}
GameLogic.ShootList = []
```

This will open a Dictory and a List where we will be able to Save the Object that will Shoot to (EmptyRedX, EmptyGreenX, EmptyBlueX)

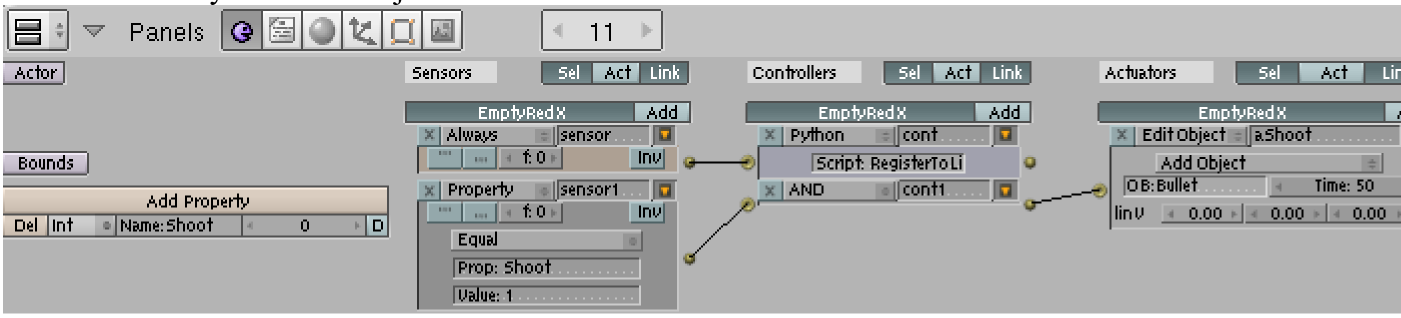
Create a New Script called RegisterToList(S).py and add this Text to it:

```
from GameLogic import*

Cont = getCurrentController()
Obj = Cont.getOwner()

if GameLogic.PlayerMod == "Client":
	if GameLogic.AddYou == 1:
		GameLogic.ShootDict[GameLogic.NewPlayer] = Obj
		print GameLogic.NewPlayer
```

```
                GameLogic.AddNew = 1
        else:
                GameLogic.AddYou = 1
                GameLogic.ShootDict[GameLogic.YourPlayerName] = Obj
                GameLogic.AddNew = 1

if GameLogic.PlayerMod == "Server":
        GameLogic.ShootList.append(Obj)
```

XXXX = Explaining will be Inserted here later…

Now go to SCE:5BasicGamePlay on the second Layer there you Select the EmptyRedX and Add this Logic Bricks to it:
Property => Int => Shoot => 0
Always Sensor (NonPulse)=> Python Controller (RegisterToList(S).py)
Proberty Sensor(NonPulse) (Equal / Prop:Shoot / Value:1) => AND Controller => EditObject Actuator (Add Object / OB: Bullet / Time: 50)
The Time:50 Says that this Object will exist for 50Frames…



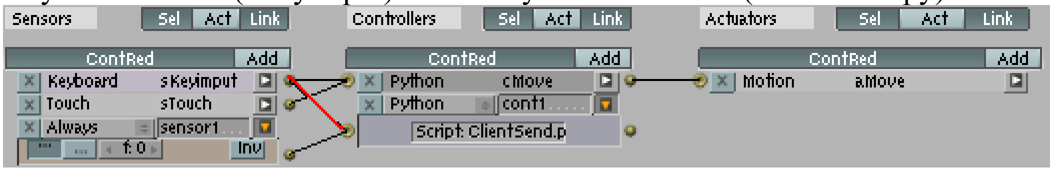
Now Do the Same whit the , EmptyGreenX and EmptyBlueX

Open the ClientSend.py: and Change the Script so that it look like this The Changes are Highlighted:

```
from GameLogic import *
from socket import *
from cPickle import *

if GameLogic.PlayerMod == "Client":
        cont = GameLogic.getCurrentController()
        obj = cont.getOwner()
        KeySens = cont.getSensor("sKeyimput")
        Keyinput = KeySens.getPressedKeys()

        PosClient = obj.getPosition()
        OriClient = obj.getOrientation()
        if Keyinput == [[125, 1]]:
                Shoot = 1
        else:
                Shoot = 0
        Data = dumps((GameLogic.YourPlayerName,GameLogic.PlayerTypS,PosClient,OriClient,Shoot))
        GameLogic.sClient.sendto(Data,GameLogic.host)
```

XXXX = Explaining will be Inserted here later…

Now select the Empty (ContRed) in the SCE:5BasicGamePlay on the second Layer
And make a logic Brick Conection between:
Keyboard Sensor (sKeyimput) and the Python Conrtoller (ClientSend.py)



Now Do the Same whit the , ContGreen and ContBlue

Open the ServerReceive.py: Change the Script so that it look like this The Changes are Highlighted:

```
#---------------IMPORT-----------------#
from GameLogic import *
from socket import *
from cPickle import *
cont = GameLogic.getCurrentController()
obj = cont.getOwner()

#----------------Start Setup-----------------#
if GameLogic.PlayerMod == "Server":

#----------------DATA-----------------#
    PosClient = [0,0,0]
    OriClient = [[0,0,0],[0,0,0],[0,0,0]]
    GameLogic.IPList =[]
    DataList =[]
    DataPack =[]
    DataSize = 0

#-----------------RECEIVE------------------#
    for Port,Player,Shoot in zip(GameLogic.PortList,GameLogic.PlayerList,GameLogic.ShootList):
        try:
            Data, CLIP = Port.recvfrom(1024)
            GameLogic.IPList.append(CLIP)

#---------------Set Game Play---------------#
            UPD = loads(Data)
            #print UPD [0], UPD[2]
            #print "=  =  =  =  =  =  =  =  =  =  =  =  =  =  =  =  =  =  =  =  =  =  ="
            DataList.append(UPD)
            DataSize = DataSize + 1
            if DataSize == 3:
                DataPack.append(DataList)
                DataList = []
                DataSize = 0
            PosClient = [UPD[2][0],UPD[2][1],UPD[2][2]]
            OriClient = [UPD[3][0],UPD[3][1],UPD[3][2]]
            Player.setPosition(PosClient)
            Player.setOrientation(OriClient)
            Shoot.Shoot = UPD[4]
        except:
            pass

#------------------SEND------------------#
    if DataSize  != 0:
        DataSend1 = dumps((DataList))
        for CLIP in GameLogic.IPList:
            try:
                GameLogic.sMain.sendto(DataSend1, CLIP)
            except:
                pass
    for Data in DataPack:
        DataSend2 = dumps((Data))
        for CLIP in GameLogic.IPList:
            try:
                GameLogic.sMain.sendto(DataSend2, CLIP)
            except:
                pass
```

XXXX = Explaining will be Inserted here later…

Open the ClientReceive.py: Change the Script so that it look like this The Changes are Highlighted:

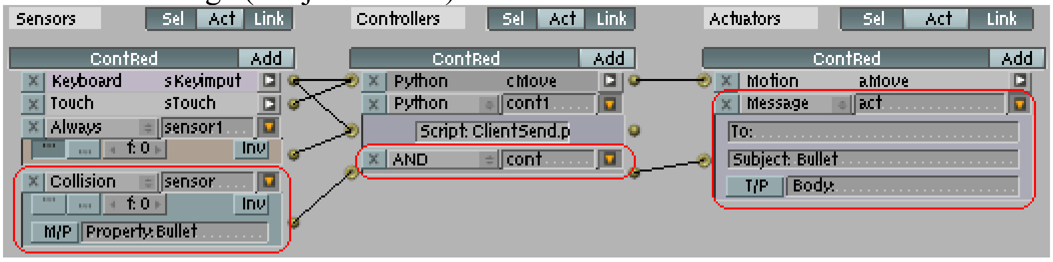
```
#---------------IMPORT----------------#
from GameLogic import *
from socket import *
from cPickle import *
cont = GameLogic.getCurrentController()
obj = cont.getOwner()
#----------------DATA-----------------#
if GameLogic.PlayerMod == "Client":
    PosClientX = [0,0,0]
    OriClientX = [[0,0,0],[0,0,0],[0,0,0]]

#------------RECEIVE-------------#
    Client = 1
    try:
        Data, SRIP = sClient.recvfrom(1024)
    except:
        Client = 0
    if Client ==1:
        if Data:
            UData = loads(Data)
            for PData in UData:
                if PData[0] in GameLogic.PlayerDict:
                    #if PData[0] == GameLogic.YourPlayerName:
                        #print "------------------"
                        #print obj.Time
                        #obj.Time = 0.0
                        #print "------------------"
```

```
                                        if PData[0] != GameLogic.YourPlayerName:
                                              PosClientX = [PData[2][0],PData[2][1],PData[2]
[2]]
                                              OriClientX = [PData[3][0],PData[3][1],PData[3]
[2]]

      GameLogic.PlayerDict[PData[0]].setPosition(PosClientX)

      GameLogic.PlayerDict[PData[0]].setOrientation(OriClientX)
                                              GameLogic.ShootDict[PData[0]].Shoot =
PData[4]

                                  if PData[0] not in GameLogic.PlayerDict:
                                        if GameLogic.AddNew == 1:
                                              GameLogic.NewPlayer = PData[0]
                                              AddNewPlayer =
cont.getActuator("aAddPlayer")
                                              AddNewPlayer.setObject(PData[1])
                                              addActiveActuator(AddNewPlayer,1)
                                              GameLogic.AddNew = 0


                  else:
                        GameLogic.s.close()
                        obj.Client = 1

#---------------THE-END----------------#
```

XXXX = Explaining will be Inserted here later…

So now it's Time to save again.
Whit the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.28WSAG.
Click on the Save as Button in the right upper Corner
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(43)WSAG –Counting GamePoint(GP) (BV 2.44 PV 2.5.1) (LU 27.3.08)
Now go to SCE:5BasicGamePlay on the second Layer there you Select the EmptyRedX and Add this Logic Bricks to it:
Sensor Collision (Property: Bullet)
Controller AND
Actuator Message (Subject: Bullet)



Keyboard sKeyimput
Touch sTouch
Always sensor1
f: 0
Collision sensor
f: 0
Property: Bullet

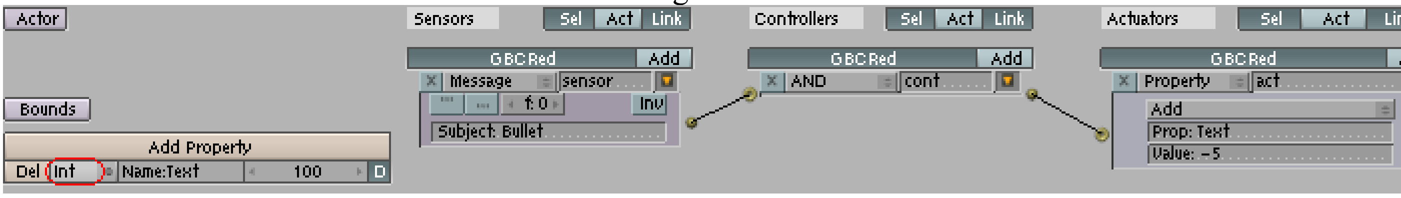
Python cMove
Python cont1
Script: ClientSend.p
AND cont

Motion aMove
Message act
To:
Subject: Bullet
Body:

This will send a Message whit the Subject Bullet and any Object that has a Sensor Message will read it.

You also could send it only to one Object by Tying the Object Name into the To: field in the Message Actuator,

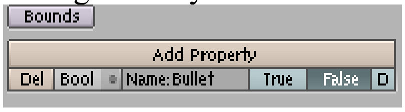
Now Select the Text Plane GPCRed and add this Logic Bricks to it:



When the Sensor Message gets the Message Bullet it will change the Property by -5. So now I a Player gets hit by a Bullet
it will loose 5 Gamepoints.

Now do the Same whit the Green and Blue Players…

Now got to Layer 3 and select the Bullet and add this Logic Bricks to it:



So now it's Time to save again.
Whit the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.29WSAG.
Click on the Save as Button in the right upper Corner

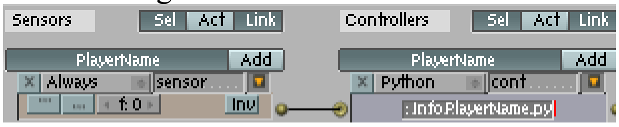
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(44)WSAG –Transmitting GamePoints(GP)between Clients (BV 2.44 PV 2.5.1) (LU 30.3.08)

Add a New Script Whit the Name InfoPlayerName.py and add this Script to it

```
from GameLogic import *
cont = GameLogic.getCurrentController()
Obj = cont.getOwner()
GameLogic.InfoPlayerName = Obj
```

This registers the Object to the Variable `GameLogic.InfoPlayerName`
Go to SCE:5BasicGamePlay on the first Layer there you Select the Empty PlayerName and Add this Logic Bricks to it:



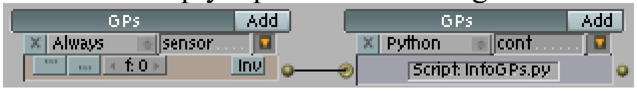
Now we can take action on this Object from every script over the Variable `GameLogic.InfoPlayerName`

Add a other Script Whit the Name InfoGps.py and add this Script to it

```
from GameLogic import *
cont = GameLogic.getCurrentController()
Obj = cont.getOwner()
GameLogic.InfoGps = Obj
```

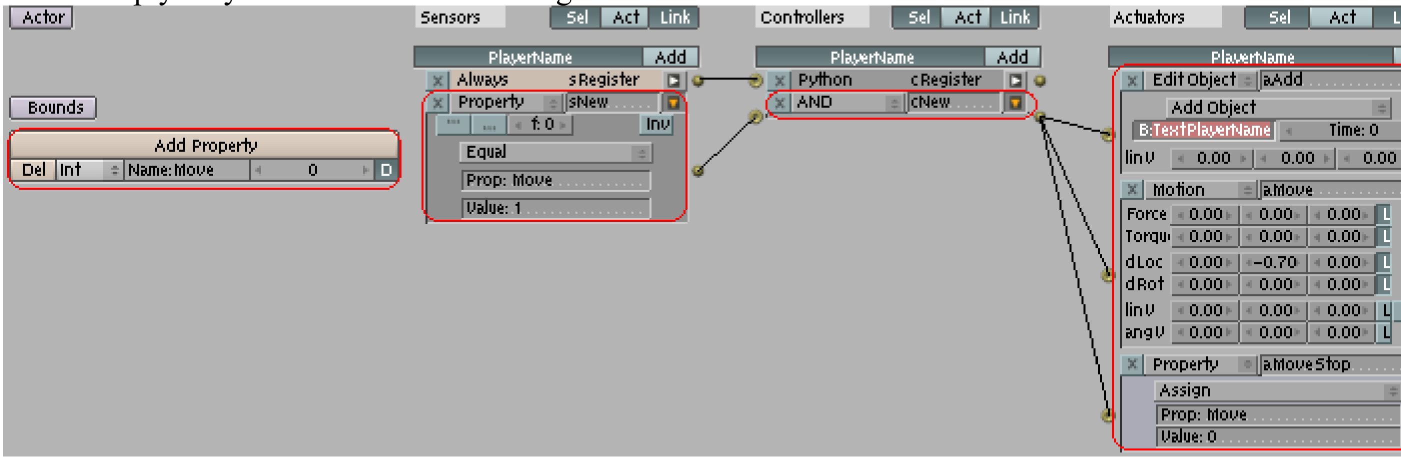
This registers the Object to the Variable `GameLogic.Gps`

Select the Empty Gps and ad this Logic Bricks to it:

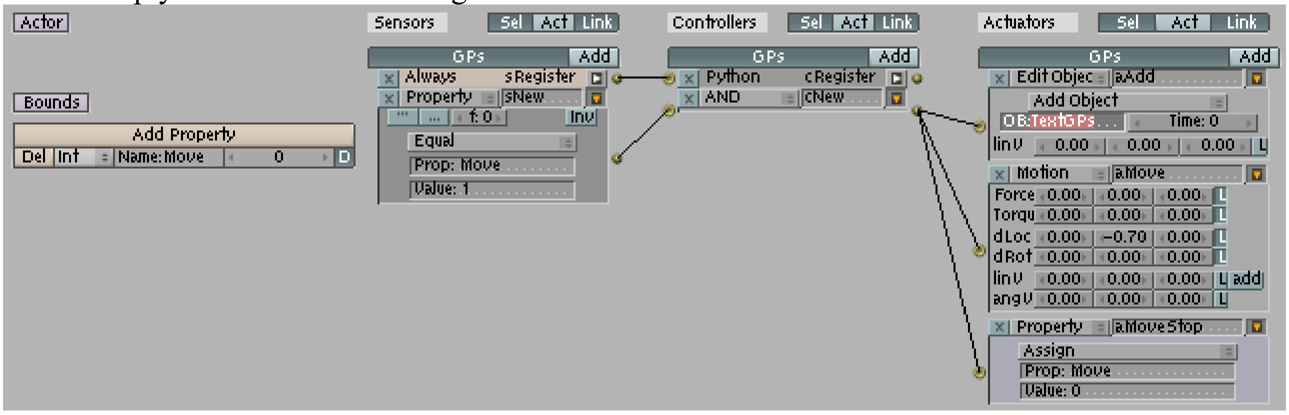


Now we can take action on this Object from every script over the Variable GameLogic.InfoGps

Select Empty PlayerName and Add this Logic Bricks to it:



Select Empty GPs and Add this Logic Bricks to it:



Now open the Script ConServer.py and make the red marked additions to it:

```
from GameLogic import *
from socket import *
from cPickle import *
cont = GameLogic.getCurrentController()
Obj = cont.getOwner()

if GameLogic.PlayerMod == "Server":

#-----Reseive New Client-----
	try:
		RData, CLIP = sMain.recvfrom(1024)
		Data = loads(RData)
		PlayerName = Data[0]
		PlayerTyp = Data[1]
		print "---Conecting New Client---"
		if PlayerName not in GameLogic.PlayerNameList:
			print "Player Name =", PlayerName
			print "Player Typ =", PlayerTyp
```

```
            print CLIP
            GameLogic.PlayerNameList.append(PlayerName)
            Obj.PortCount = Obj.PortCount + 1
            Data = dumps((Obj.PortCount,GameLogic.LocationC))
            GameLogic.sMain.sendto(Data, CLIP)
            Obj.AddClient = 1
        else:
            print "Playername", Data[0], "alredy in List"
            print "-------------------------"
            Info = "Playername alredy in use"
            Data = dumps((Info))
            GameLogic.sMain.sendto(Data, CLIP)
    except:
        pass

#-----Add New Client-----

    if Obj.AddClient == 1:
        host = ''
        port = Obj.PortCount
        GameLogic.sPort = socket(AF_INET,SOCK_DGRAM)
        GameLogic.sPort.bind((host,port))
        GameLogic.sPort.setblocking(0)
        AddPlayerAct = cont.getActuator("aAddPlayer")
        AddPlayerAct.setObject(PlayerTyp)
        addActiveActuator(AddPlayerAct,1)
        GameLogic.InfoPlayerName.Move = 1
        GameLogic.InfoGPs.Move = 1
        GameLogic.PortList.append(GameLogic.sPort)
        print "Port"  ,port,  "is up"
        print "-------------------------"
        Obj.AddClient = 2
```

XXXX = Explaining will be Inserted here later…

Open the Script List.py and add these lines at the bottom:

```
GameLogic.PNList = []
GameLogic.GPList = []
```

XXXX = Explaining will be Inserted here later…

Add New Script with the Name RegisterToList(PN).py and add this Text to it:

```
from GameLogic import*
Cont = getCurrentController()
Obj = Cont.getOwner()
GameLogic.PNList.append(Obj)
#Note PN = PlayerName
```

XXXX = Explaining will be Inserted here later…

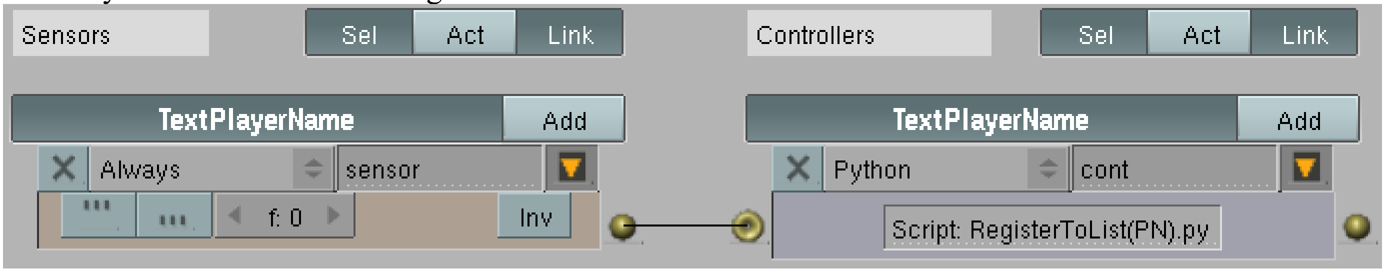
Add New Script with the Name RegisterToList(GP).py and add this Text to it:

```
from GameLogic import*
Cont = getCurrentController()
Obj = Cont.getOwner()
```

```
GameLogic.GPList.append(Obj)
#Note GP = GamePoints
```

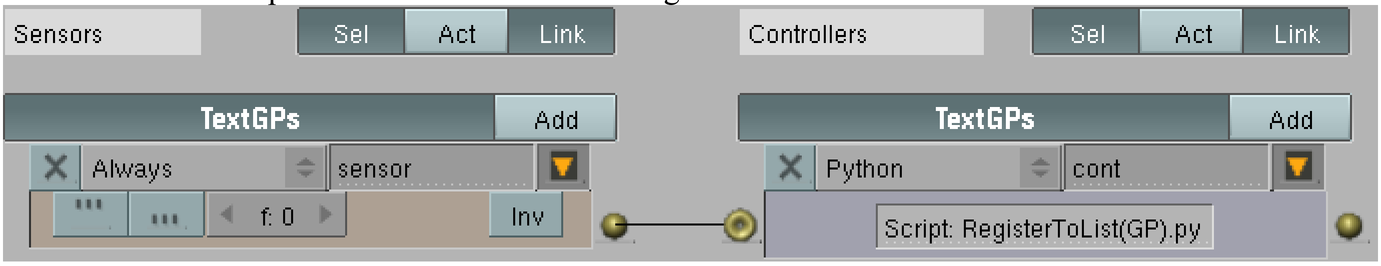
XXXX = Explaining will be Inserted here later…

Now go to the Layer 3 in SCE:5BasicGamePlay and there You select the Text plane TextPlayerName and add this Logic Bricks to it:



TextPlayerName
Always
sensor
TextPlayerName
Python
cont
Script: RegisterToList(PN).py

Now select the Text plane TextGPs and add this Logic Brick to it:



TextGPs
Always
sensor
TextGPs
Python
cont
Script: RegisterToList(GP).py

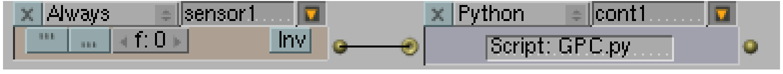
Now when this Text Planes will be created in Game Play the will register to there List and Can be called from there,

Add a New Scrip whit the Name GPC.py and add this Script to it:

```
from GameLogic import *
cont = GameLogic.getCurrentController()
Obj = cont.getOwner()
GameLogic.GPC = Obj
#Note GPC = GamePointCount
```

XXXX = Explaining will be Inserted here later…

Go to the second Layer in SCE:5BasicGamePlay there you add this Logic Bricks to the Text Planes GPCRed, GPCBlue, GPCGreen:



Always
sensor1
Python
cont1
Script: GPC.py

Now open the Script Client send and make the Red highlighted Changes to it:

```
from GameLogic import *
from socket import *
from cPickle import *

if GameLogic.PlayerMod == "Client":
    cont = GameLogic.getCurrentController()
    obj = cont.getOwner()
    KeySens = cont.getSensor("sKeyimput")
    Keyinput = KeySens.getPressedKeys()

    PosClient = obj.getPosition()
    OriClient = obj.getOrientation()
    if Keyinput == [[125, 1]]:
        Shoot = 1
    else:
        Shoot = 0

    try:
        Data = dumps((GameLogic.YourPlayerName,GameLogic.PlayerTypS,PosClient,OriClient,Shoot,GameLogic.GPC.Text))
        GameLogic.sClient.sendto(Data,GameLogic.host)
    except:
        pass
```

XXXX = Explaining will be Inserted here later…

This change will now send the Game Points to to Server together with all the other Information.

Now in the ServerReceive.py Script we make Red highlighted Changes below:

```
#---------------IMPORT-----------------#
from GameLogic import *
from socket import *
from cPickle import *
cont = GameLogic.getCurrentController()
obj = cont.getOwner()
#----------------Start Setup------------------#
if GameLogic.PlayerMod == "Server":
#----------------DATA------------------#
    PosClient = [0,0,0]
    OriClient = [[0,0,0],[0,0,0],[0,0,0]]
    GameLogic.IPList =[]
    DataList =[]
    DataPack =[]
    DataSize = 0
#------------------RECEIVE-------------------#
    for Port,Player,Shoot,PN,GPC in zip(GameLogic.PortList,GameLogic.PlayerList,GameLogic.ShootList,GameLogic.PNList,GameLogic.GPList):
        try:
            Data, CLIP = Port.recvfrom(1024)
            GameLogic.IPList.append(CLIP)
#---------------Set Game Play----------------#
            UPD = loads(Data)
            DataList.append(UPD)
            DataSize = DataSize + 1
            if DataSize == 3:
                DataPack.append(DataList)
                DataList = []
                DataSize = 0
            PosClient = [UPD[2][0],UPD[2][1],UPD[2][2]]
            OriClient = [UPD[3][0],UPD[3][1],UPD[3][2]]
            Player.setPosition(PosClient)
            Player.setOrientation(OriClient)
            Shoot.Shoot = UPD[4]
            PN.Text = UPD[0]
            GPC.Text = UPD[5]
        except:
            pass
#------------------SEND------------------#
    if DataSize  != 0:
        DataSend1 = dumps((DataList))
        for CLIP in GameLogic.IPList:
            try:
                GameLogic.sMain.sendto(DataSend1, CLIP)
            except:
                pass
    for Data in DataPack:
        DataSend2 = dumps((Data))
        for CLIP in GameLogic.IPList:
            try:
                GameLogic.sMain.sendto(DataSend2, CLIP)
            except:
                pass
```

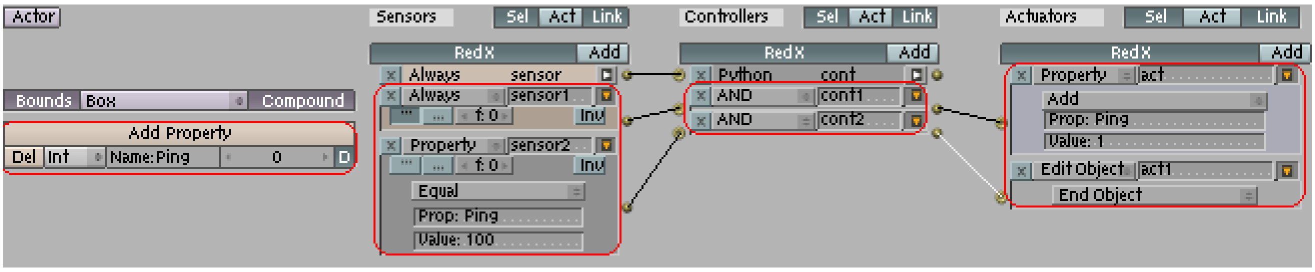
XXXX = Explaining will be Inserted here later…

So now it's Time to save again.
Whit the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.30WSAG.
Click on the Save as Button in the right upper Corner

PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

(45)WSAG – Disconnect Clients(BV 2.44 PV 2.5.1) (LU 18.4.08)

Got to the second layer of SCE:5BasicGamePlay. Select the Cube RedX and add this Logic Bricks to it.



This will make that it will count the Property "Ping" up to a 100 and then Ends the Object.
Do the Same thing to the BlueX and GreenX Cube.
Now we will make it so tat as long as this Object gets Signals for the Client that is controlling it, we will reset the Ping to 0.
For this open the Script ServerReceive.py and adding the Red highlighted line to it:

```
#---------------IMPORT----------------#
from GameLogic import *
from socket import *
from cPickle import *
```

```
cont = GameLogic.getCurrentController()
obj = cont.getOwner()
#----------------Start Setup-----------------#
if GameLogic.PlayerMod = = "Server":
#----------------DATA-----------------#
    PosClient = [0,0,0]
    OriClient = [[0,0,0],[0,0,0],[0,0,0]]
    GameLogic.IPList =[]
    DataList =[]
    DataPack =[]
    DataSize = 0
#------------------RECEIVE------------------#
for Port,Player,Shoot,PN,GPC in zip(GameLogic.PortList,GameLogic.PlayerList,GameLogic.ShootList,GameLogic.PNList,GameLogic.GPList):
        try:
            Data, CLIP = Port.recvfrom(1024)
            GameLogic.IPList.append(CLIP)
#---------------Set Game Play---------------#
            UPD = loads(Data)
            DataList.append(UPD)
            DataSize = DataSize + 1
            if DataSize == 3:
                DataPack.append(DataList)
                DataList = []
                DataSize = 0
            PosClient = [UPD[2][0],UPD[2][1],UPD[2][2]]
            OriClient = [UPD[3][0],UPD[3][1],UPD[3][2]]
            Player.setPosition(PosClient)
            Player.setOrientation(OriClient)
            Shoot.Shoot = UPD[4]
            PN.Text = UPD[0]
            GPC.Text = UPD[5]
            Player.Ping = 0
        except:
            pass
#------------------SEND------------------#
    if DataSize  != 0:
        DataSend1 = dumps((DataList))
        for CLIP in GameLogic.IPList:
            try:
                GameLogic.sMain.sendto(DataSend1, CLIP)
            except:
                pass
    for Data in DataPack:
        DataSend2 = dumps((Data))
        for CLIP in GameLogic.IPList:
            try:
                GameLogic.sMain.sendto(DataSend2, CLIP)
            except:
                pass
```

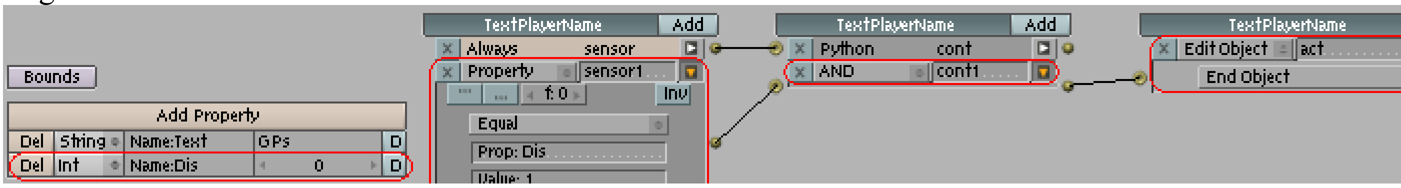
This will reset the Property Ping to 0 ever time the Server Receives a Signal of that Client. Now we will have to do the Same ting on the Client Side.
Open the Script ClientReceive.py and adding the Red highlighted line to it:

```
#---------------IMPORT-----------------#
from GameLogic import *
from socket import *
from cPickle import *
cont = GameLogic.getCurrentController()
obj = cont.getOwner()
#----------------DATA------------------#
if GameLogic.PlayerMod == "Client":
    PosClientX = [0,0,0]
    OriClientX = [[0,0,0],[0,0,0],[0,0,0]]
#------------RECEIVE--------------#
    Client = 1
    try:
        Data, SRIP = sClient.recvfrom(1024)
    except:
        Client = 0
    if Client ==1:
        if Data:
            UData = loads(Data)
            for PData in UData:
                if PData[0] in GameLogic.PlayerDict:
                    #if PData[0] == GameLogic.YourPlayerName:
                        #print "-------------------"
                        #print obj.Time
                        #obj.Time = 0.0
                        #print "-------------------"
                    if PData[0] != GameLogic.YourPlayerName:
                        PosClientX  =  [PData[2][0],PData[2][1],PData[2]
[2]]
                        OriClientX  =  [PData[3][0],PData[3][1],PData[3]
[2]]

    GameLogic.PlayerDict[PData[0]].setPosition(PosClientX)

    GameLogic.PlayerDict[PData[0]].setOrientation(OriClientX)
                        GameLogic.ShootDict[PData[0]].Shoot            =
PData[4]
                        GameLogic.PlayerDict[PData[0]].Ping = 0

                if PData[0] not in GameLogic.PlayerDict:
                    if GameLogic.AddNew == 1:
                        GameLogic.NewPlayer = PData[0]
                        AddNewPlayer                                   =
cont.getActuator("aAddPlayer")
                        AddNewPlayer.setObject(PData[1])
                        addActiveActuator(AddNewPlayer,1)
                        GameLogic.AddNew = 0


        else:
            GameLogic.s.close()
            obj.Client = 1
#---------------THE-END---------------#
```

This will do the same thing on the Client side as we looked at before for the Server side. So now I actually will delete the Visual Part of the Client on both sides if the Client dose not sends a Signal for 100frames.

Next step is to delete all the other things that where Created together with the Client such as (List and Dictionary entry's or the Text Planes on the Server side that display the Player name and its Live Points).

Now Go the SCE:5BasicGamePlay on layer 3 and select the Text Plane GP and add this Logic Bricks to it:



This will give us the Possibility to end this Object by changing the Proberty value Dis = 1.

Do the Same thing to the Text Plane PN.

Open the Script ServerReceive.py and add the Red Highlighted Text to it:

```
#---------------IMPORT-----------------#
from GameLogic import *
from socket import *
from cPickle import *
cont = GameLogic.getCurrentController()
obj = cont.getOwner()
#----------------Start Setup------------------#
if GameLogic.PlayerMod = = "Server":
#----------------DATA------------------#
    PosClient = [0,0,0]
    OriClient = [[0,0,0],[0,0,0],[0,0,0]]
    GameLogic.IPList =[]
    DataList =[]
    DataPack =[]
    DataSize = 0
#------------------RECEIVE-------------------#
    GL = GameLogic
    for Port,Player,PlayerName,Shoot,PN,GPC in zip(GL.PortList,GL.PlayerList,GL.PlayerNameList,GL.ShootList,GL.PNList,GL.GPList):
        #print PN.Text, "=", Player.Ping
        if Player.Ping > 90:
            print "Disconnect", PN.Text
            Port.close()
            print "Port closed"
            GameLogic.PortList.remove(Port)
            GameLogic.PlayerList.remove(Player)
            GameLogic.PlayerNameList.remove(PlayerName)
            GameLogic.ShootList.remove(Shoot)
            GameLogic.PNList.remove(PN)
            GameLogic.GPList.remove(GPC)
            PN.Dis = 1
            GPC.Dis = 1
            print "List Entry Deleted"
            print "-------------------------"
        try:
            Data, CLIP = Port.recvfrom(1024)
            GameLogic.IPList.append(CLIP)
#---------------Set Game Play----------------#
            UPD = loads(Data)
            DataList.append(UPD)
            DataSize = DataSize + 1
            if DataSize == 3:
                DataPack.append(DataList)
                DataList = []
                DataSize = 0
            PosClient = [UPD[2][0],UPD[2][1],UPD[2][2]]
            OriClient = [UPD[3][0],UPD[3][1],UPD[3][2]]
            Player.setPosition(PosClient)
            Player.setOrientation(OriClient)
            Shoot.Shoot = UPD[4]
            PN.Text = UPD[0]
            GPC.Text = UPD[5]
            Player.Ping = 0
        except:
            pass
#-----------------SEND------------------#
    if DataSize  != 0:
        DataSend1 = dumps((DataList))
        for CLIP in GameLogic.IPList:
            try:
                GameLogic.sMain.sendto(DataSend1, CLIP)
            except:
                pass
    for Data in DataPack:
        DataSend2 = dumps((Data))
        for CLIP in GameLogic.IPList:
            try:
                GameLogic.sMain.sendto(DataSend2, CLIP)
            except:
                pass
```

So what do this lines we just added do?

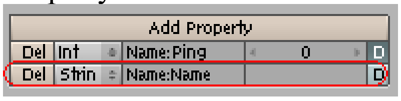
GL = GameLogic = This Way we only will have to type GL instead of GameLogic. I normaly don’t like to use this Shortcutm, because if I see everything written out its easyer for me to finde Mistakes. But here the “for…in zip(….)” Line is geting way to long if we don’t use this Shortcut.
for …,PlayerName,…in zip(…,GL.PlayerNameList,…): = We added the calling of the PlayerNameList so that we can Delete the Playername Entrys if a Client Disconects.
#print PN.Text, "=", Player.Ping = if you delete the # it will print out the Pings of all clients in the Consol window. This helps to find Ping Problems
if Player.Ping > 90: = Will look if the Ping Value of The Player is Bigger the 90 if so it will run the Lines below:
print "Disconnect", PN.Text = Pints first Disconnect and then the Value that is Saved in the Variable PN.Text (The PlayerName)
Port.close() = Closes the Port that this Client was communicating offer
GameLogic.PortList.remove(Port) = Removes the Post from the PortList
GameLogic.PlayerList.remove(Player) = Removes The Player from the PlayerList
GameLogic.PlayerNameList.remove(PlayerName) = Removes The PlayerName From the PlayerNameList this way someone else can use the name gain
GameLogic.ShootList.remove(Shoot) = Removes the Shoot empty from the ShootList
GameLogic.PNList.remove(PN) = Removes the Entry of the Text plane PN from the PN List
GameLogic.GPList.remove(GPC) = Removes the Entry of the Text plane GPC from the GPList
PN.Dis = 1 = Sets the Value of the Property Dis from the Text Plane PN to 1 (As we set up the Logic Bricks above the Object will be ended if The Property Value of Dis is equal to 1).
GPC.Dis = 1 = Sets the Value of the Property Dis from the Text Plane GP to 1 (As we set up the Logic Bricks above the Object will be ended if The Property Value of Dis is equal to 1).
print "List Entry Deleted" = Prints List Entry Deleted in the Consol window
print "-------------------------" = Prints ------------------------- in the Consol window

Note that in my WSAGdev.blend file I use Higher Ping values because I test Server and 3 Clients all on one PC and whit a lower Ping value I get disconnections all the Time so try out for your self what values work best for you.

Actually everything is working now. But I like to clean up on the Client side to...This will help tat if you stay connected to the server for a long time and a lot of clients come and go not only the Player Object will be deleted but also all its registries in the Dictionaries.

Okay so now we will have to do the same Trick on the Client side. We will have to do this a bit different then on the Server side because we use Dictionaries on the Client side and not lists like on the Server side. Off course we could always have all the Lists and Dictionaries created on both the Client and Server Side. But this way we get to learn how to do the same things in 2 different ways…

So go to the second layer of SCE:5BasicGamePlay. Select the Cube RedX and add a String Property called Name:



Del Int Name:Ping 0 D
Del Strin Name:Name D

Now do the same thing to the Objects BlueX and GreenX.

Open the Script RegisterToList.py and Add the Highlighted Text to it:

```
from GameLogic import*
Cont = getCurrentController()
```

```
Obj = Cont.getOwner()
if GameLogic.PlayerMod == "Client":
    if GameLogic.AddYou == 1:
        GameLogic.PlayerDict[GameLogic.NewPlayer] = Obj
        print "-----------------------"
        print "New Player Connected:", GameLogic.NewPlayer
        print "-----------------------"
        GameLogic.AddNew = 1
        Obj.Name = GameLogic.NewPlayer
    else:
        GameLogic.AddYou = 1
        GameLogic.PlayerDict[GameLogic.YourPlayerName] = Obj
        GameLogic.AddNew = 1
if GameLogic.PlayerMod == "Server":
    GameLogic.PlayerList.append(Obj)
```

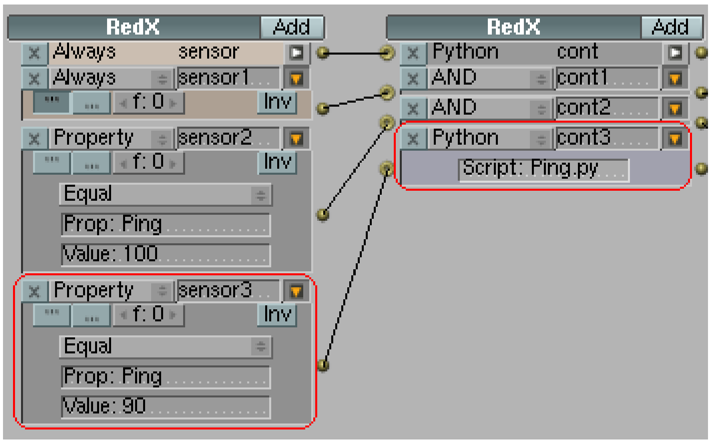
This Will save the Name of this Player in to the Property Name.

Create a new Script whit the Name Ping.py

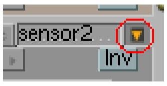
```
from GameLogic import *
Cont = getCurrentController()
Obj = Cont.getOwner()
if GameLogic.PlayerMod == "Client":
    if Obj.Ping == 200:
        print "Disconnect", Obj.Name
        del GameLogic.PlayerDict[Obj.Name]
        del GameLogic.ShootDict[Obj.Name]
        print "List Entry Deleted"
        print "-------------------------"
```

XXXX = Explaining will be Inserted here later…

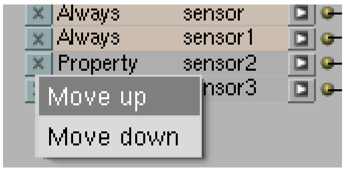
So go to the second layer of SCE:5BasicGamePlay. Select the Cube RedX and add this Logic Bricks to it:



Sensor =>Poperty => (Prop: Ping) (Value: 90)
Controller => Python => (Ping.py)
Now minimize the Logic Bricks by clicking on the little yellow triangle of each logic Brick.



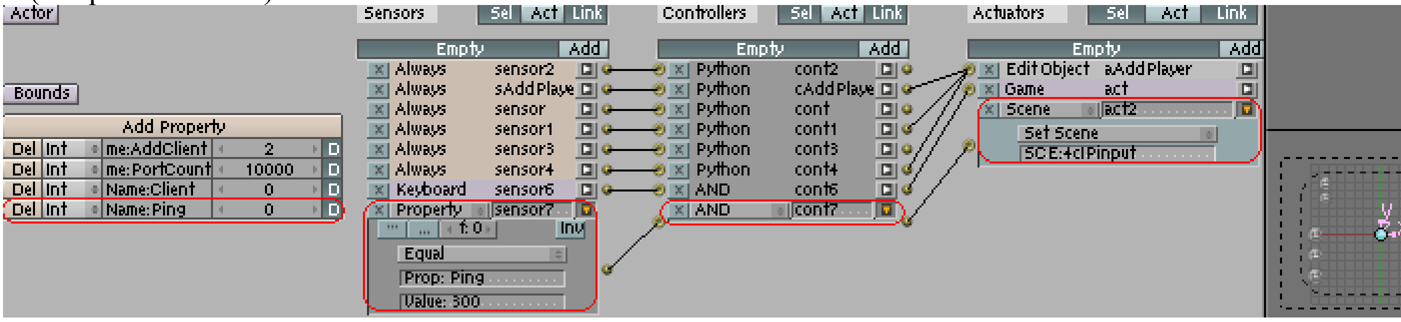
Now click on the Logic Brick whit the (Prop Ping Value 90) and in the Popup Menu select Move Up.



Do the Same Thing to the Logic Brick that contains the Python script Ping.py
This way this Logic brick will be read in each cycle before the Logic bricks that will have the Object deleted. (Proberty Ping Value 100/ AND / Edit Object =>End)
It will always read the Logic Bricks from the top to the bothem and then the cycle will start over again.

Now do the same thing to the Objects BlueX and GreenX.

So one last thing to do; if you as a client get disconnected from the server you should change to a Login scene where you can reconnect whit the server:
Go to the first Layer of SCE:5BasicGamePlay. Select the Empty and add this Logic Bricks to it (see picture below)



Del Int me:AddClient 2
Del Int me:PortCount 10000
Del Int Name:Client 0
Del Int Name:Ping 0

f: 0
Equal
Prop: Ping
Value: 300

Set Scene
SCE:4clPinput

well this logic bricks should be self explaining by now…

No open the Script ClientReceive.py and add the Highlighted Text to it:

```
#---------------IMPORT-----------------#
from GameLogic import *
from socket import *
from cPickle import *
cont = GameLogic.getCurrentController()
obj = cont.getOwner()
#-----------------DATA------------------#
if GameLogic.PlayerMod == "Client":
        PosClientX = [0,0,0]
        OriClientX = [[0,0,0],[0,0,0],[0,0,0]]

#------------RECEIVE--------------#
        Client = 1
        try:
                Data, SRIP = sClient.recvfrom(1024)
        except:
                obj.Ping = obj.Ping +1
                if obj.Ping = = 100:
                        print "No Server Connection"
                if obj.Ping = = 290:
                        print "Disconecting from Server"
                        print "-------------------------"
                Client = 0
```

```
    if Client = =1:
        if obj.Ping > 100:
            print "Connection of Server is Back"
            print "--------------------------"
        obj.Ping = 0
        if Data:
            UData = loads(Data)
            for PData in UData:
                if PData[0] in GameLogic.PlayerDict:
                    #if PData[0] = = GameLogic.YourPlayerName:
                        #print "-------------------"
                        #print obj.Time
                        #obj.Time = 0.0
                        #print "-------------------"
                    if PData[0] != GameLogic.YourPlayerName:
                        PosClientX = [PData[2][0],PData[2][1],PData[2]
[2]]
                        OriClientX = [PData[3][0],PData[3][1],PData[3]
[2]]

    GameLogic.PlayerDict[PData[0]].setPosition(PosClientX)

    GameLogic.PlayerDict[PData[0]].setOrientation(OriClientX)
                        GameLogic.ShootDict[PData[0]].Shoot             =
PData[4]
                        GameLogic.PlayerDict[PData[0]].Ping = 0

                if PData[0] not in GameLogic.PlayerDict:
                    if GameLogic.AddNew = = 1:
                        GameLogic.NewPlayer = PData[0]
                        AddNewPlayer                                    =
cont.getActuator("aAddPlayer")
                        AddNewPlayer.setObject(PData[1])
                        addActiveActuator(AddNewPlayer,1)
                        GameLogic.AddNew = 0

        else:
            GameLogic.s.close()
            obj.Client = 1
#---------------THE-END----------------#
```

So what do the lines just added do:

```
obj.Ping = obj.Ping +1
```

If it dose not receive data from the server it will add the Ping count by one (current Ping count + 1)

```
if obj.Ping = = 100:
    print "No Server Connection"
```

If the Ping count reaches a 100 it will print “No Server Connection” in to the consol window

```
if obj.Ping = = 290:
    print "Disconecting from Server"
    print "--------------------------"
```

If the Ping count reaches a 290 it will print "Disconecting from Server" / "-------------------------"
in to the consol window

```
if obj.Ping > 100:
        print "Connection of Server is Back"
        print "-------------------------"
```

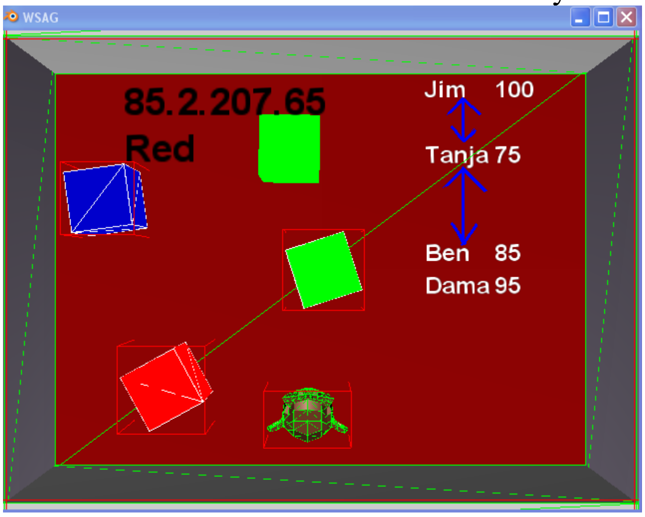
If the Ping count went over a 100 and then it gets a server signal again it will print "Connection of Server is Back" in to the consol window

And Viola Disconnection is now not only working but it also cleans up all its Bits and Pieces.

So now it's Time to save again.
Whit the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.31WSAG.
Click on the Save as Button in the right upper Corner
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial

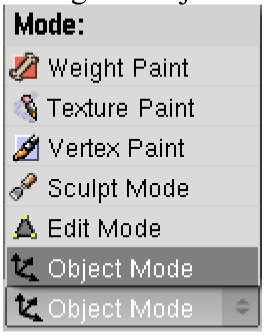
(46)WSAG – Disconnect (Cosmetic lift up) (BV 2.46 PV 2.5.1) (LU 3.6.08)

Now the only thing left is some Cosmetic lift up that has nothing to do whit the Functionality. You may have noticed that the List on the Server that displays all the Clients will have holes there where Clients where listed before they were disconnected.

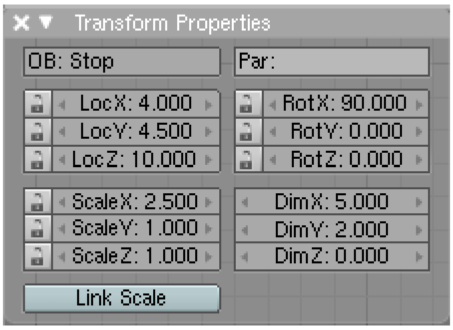


So we will have to move the Listings together so that these holes will disappear.

Go to the first Layer of SCE:5BasicGamePlay and add a Plane:
Change to Object Mode

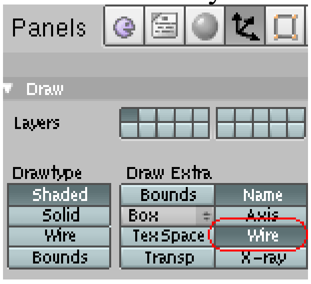


Press the N-Key and set the position to the Values as the Picture shows below:

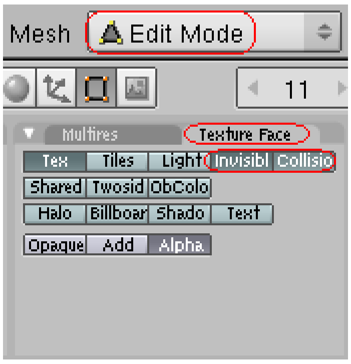

Transform Properties

OB: Stop
Par:
LocX: 4.000
LocY: 4.500
LocZ: 10.000
RotX: 90.000
RotY: 0.000
RotZ: 0.000
ScaleX: 2.500
ScaleY: 1.000
ScaleZ: 1.000
DimX: 5.000
DimY: 2.000
DimZ: 0.000

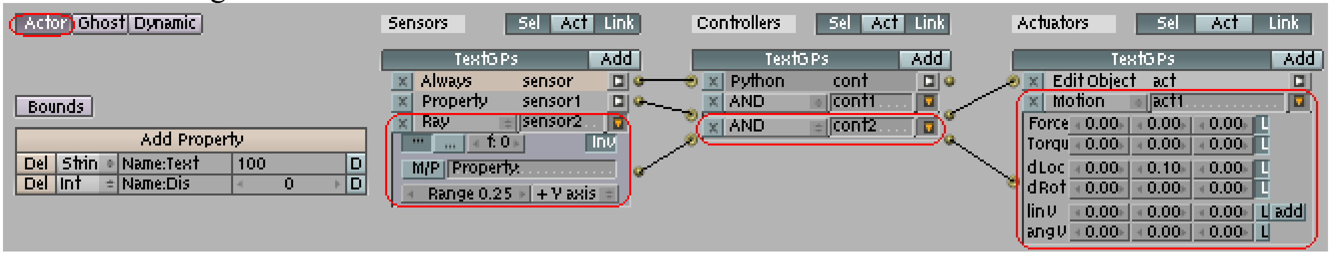
Press the F7-Key and make these Changes:



This way you will still see the Object in Edit Mode after we now will make it Transparent:
Change to Edit mode Press the F9-Key and select the Texture Face Panel:
No Select the Invisible and Collisions Button. See Picture below.



Now go to the 3 Layer and select the Text Plane GPs (make sure you changed back to Object mode)
And add this Logic Bricks to it:



Del Strin Name:Text 100 D
Del Int Name:Dis 0 D

Sensors (TextGPs):
Always sensor
Property sensor1
Ray sensor2
f: 0
Property:
Range 0.25
+Y axis

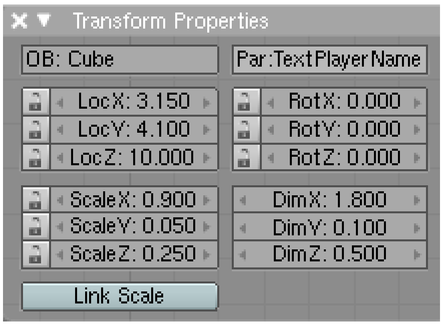
Controllers (TextGPs):
Python cont
AND cont1
AND cont2

Actuators (TextGPs):
Edit Object act
Motion act1

Force	0.00	0.00	0.00
Torqu	0.00	0.00	0.00
dLoc	0.00	0.10	0.00
dRot	0.00	0.00	0.00
linV	0.00	0.00	0.00
angV	0.00	0.00	0.00

Do the Same to the Text Plane “TextPlayerName”

Add Cube, press the N-Key and set this Properties.



OB: Cube
Par:TextPlayerName
LocX: 3.150
LocY: 4.100
LocZ: 10.000
RotX: 0.000
RotY: 0.000
RotZ: 0.000
ScaleX: 0.900
ScaleY: 0.050
ScaleZ: 0.250
DimX: 1.800
DimY: 0.100
DimZ: 0.500

Note that you have made the Cube a Parent of the Text Plane “TextPlayerName”

On the first Layer we select the Empty “PlayerName” and “GPs” and move the down a bit this will give a cool effect of the New Player Name floating from the bottom up… I Changed the Location to LocY – 2…

Because we added a new Object (the and New Objects will not be created in the Linked Scenes we will have to delete the Scenes ("6LocationBlue"), ("6LocationRed") and ("6LocationGreen"). And recreate them as shown in **(21)WSAG – Gameplay – Creating the Locations.**

This is way at the Moment we keep the Location Scenes very simple so it won’t take long to delete and recreate them.

So now it’s Time to save again.

Whit the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.32WSAG.

Click on the Save as Button in the right upper Corner

PLEASE DON’T FORGET TO SUPPORT THIS PROJECT

How you can support this project is written at the beginning of this Tutorial

(47)WSAG – And here comes the Mouse(BV 2.46 PV 2.5.1) (LU 1.7.08)

So now we will add some mouse Controlls So that we can look around better...

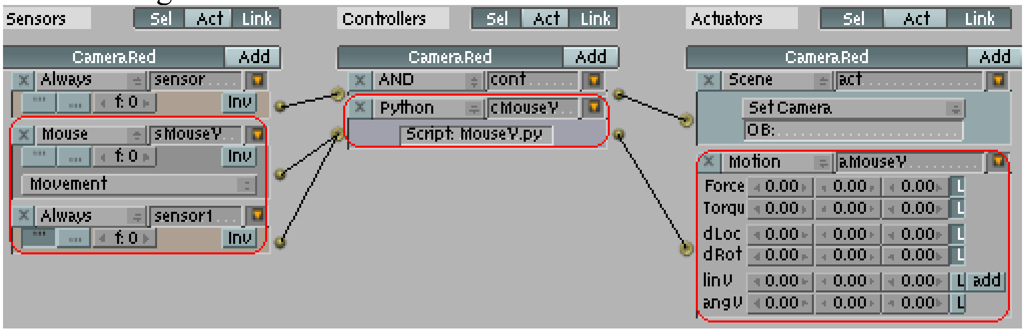
First we add the Vertikal Movement. For this The Players dose not move, only the Camera will.

Go to the second Layer of SCE:5BasicGamePlay

And select the CameraRed.

Add a new Script window and name it MouseY.py

Add this Logic Bricks to the CameraRed:



Always sensor f: 0
Mouse sMouseY f: 0
Movement
Always sensor1 f: 0
AND cont
Python cMouseY
Script: MouseY.py
Scene act
Set Camera
OB:
Motion aMouseY
Force 0.00 0.00 0.00
Torqu 0.00 0.00 0.00
dLoc 0.00 0.00 0.00
dRot 0.00 0.00 0.00
linV 0.00 0.00 0.00
angV 0.00 0.00 0.00

This Logic Bricks Make that the Script will always run and have access to the Sensor Mouse movement and the Actuator Motion.

Now in the Script MouseZ.py we wil write this Code:

```
from GameLogic import *
from Rasterizer import *

cont = GameLogic.getCurrentController()
obj = cont.getOwner()
Sensor = cont.getSensor("sMouseY")
Actuator = cont.getActuator("aMouseY")

YPosition = Sensor.getYPosition()

Sensitivity = 0.003
Max = -0.900
Min = 0.900
CenterY = getWindowHeight()/2
CenterX = getWindowWidth()/2
Rotation = obj.getOrientation()[2][2]
MoveY = (CenterY - YPosition)*Sensitivity

Actuator.setDRot(0,0,0,1)
if MoveY > 0 and Rotation > Max:
	Actuator.setDRot(MoveY,0,0,1)
if MoveY < 0 and Rotation < Min:
	Actuator.setDRot(MoveY,0,0,1)

addActiveActuator(Actuator,1)
setMousePosition(CenterX,CenterY)
```

So and What is this Script doing?

```
from GameLogic import *
from Rasterizer import *
```

I think we now what this scripts do by now; The Improt the Libarys that we need.

```
cont = GameLogic.getCurrentController()
obj = cont.getOwner()
Sensor = cont.getSensor("sMouseY")
Actuator = cont.getActuator("aMouseY")
```

This part of the Script should be clear by now aswell; It gives us access to the Sensors and Acutators that we will need and saves them a Variable.

```
YPosition = Sensor.getYPosition()
```

Gets the Position of the Mouse on the Y Axis and saves it to the Variable Yposition.

```
Sensitivity = 0.003
```

This Variable will be used to set the Sensitiviy of the Mouse (Change its Value until the Mousemove feels comfortable)

```
Max = -0.900
Min = 0.900
```

This Varible will be used to set the Max and Minimum that you can look up or down. If you do not set a Max/Min then you could look around in a loop. But a Player is limitet on how far it can look up or down, depending on how long and Flexible its neck is.

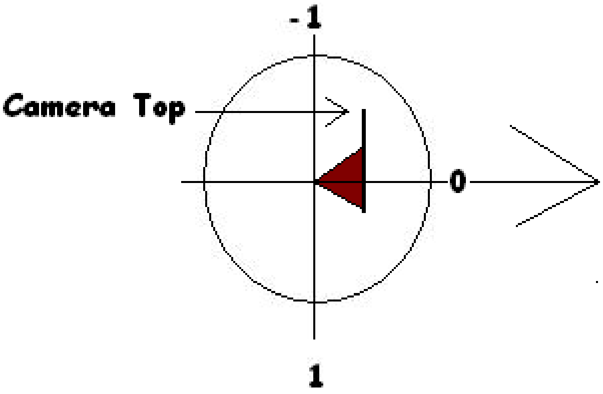
```
CenterY = getWindowHeight()/2
CenterX = getWindowWidth()/2
```
Both Variable together will give you the Center of your screen.

```
Rotation = obj.getOrientation()[2][2]
```
This Variable holds the Value on how muche up or dow the Camara has ben Rotated or in other Words it gives you the Orienations-Position of the Camera.

```
MoveY = (CenterY - YPosition)*Sensitivity
```
CenterY - YPosition gives you the distance that the Mouse was moved by multiplying it whit the Value that is saved in Sensitivity
You geht the Value on how muche you whant the Camera to move and this Value is saved in the Variable MoveY

```
Actuator.setDRot(0,0,0,1)
```
Sets the Rotations Value of the Actuator to 0 . This Way if the mouse is not being moved the Rotation of the Cammera will stop.

```
if MoveY > 0 and Rotation > Max:
        Actuator.setDRot(MoveY,0,0,1)
```
If the Variable saved in MoveY is a positive nummer (bigger then 0) and the Orient Position of the Cammera is bigger the the Value Saved in Max: then this script will move the Camera the distance of the Value saved in the Variabele MoveY.
In the picture below you can see how the Y Axis that the Camara is rotationg on is build up. Looking Strait up in to the Sky will give you a Orientations Value of **-1** if you look strait aheat you get a value of **0** and if you look down at the Floor you get a **1**



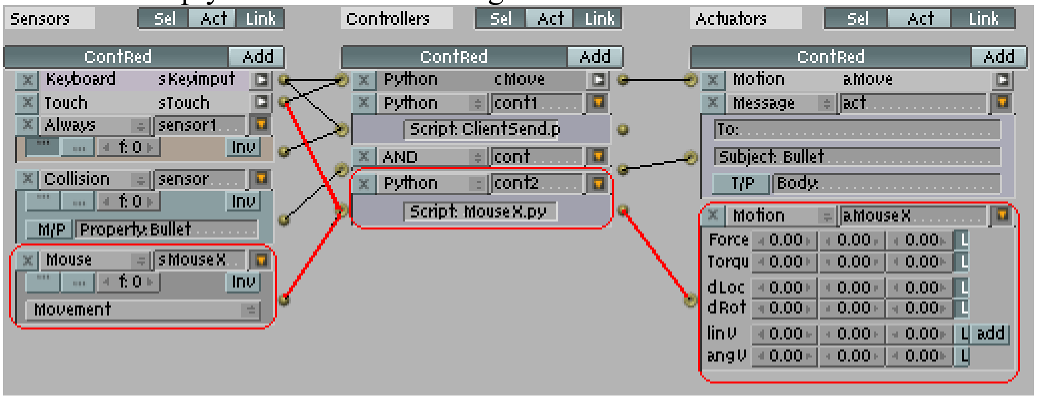
```
if MoveY < 0 and Rotation < Min:
        Actuator.setDRot(MoveY,0,0,1)
```
This handels the same action as the script above just for the part of looking down.

```
addActiveActuator(Actuator,1)
```
This is a well know part of script by now... It aktivatest the Actuator by seting it to the Value 1 = Positive...
```
setMousePosition(CenterX,CenterY)
```
Sets the mouse position in to the center of the Display. This way in the next time this script runs it can calculate the distance the the mouse was moved from the center again.

So now the Player can look up and down whit the Mouse. By carful if you now look total down and shoot you will hit yourself !

So next steep is to be able to turn the Player horizontal by moving the mouse on the X Axis.

Add a new Script window and name it MouseX.py
Select the Empty Red and add this Logic Bricks to it:



In the MouseX.py Script Window you add this Script:

```
from GameLogic import *
from Rasterizer import *

cont = GameLogic.getCurrentController()
obj = cont.getOwner()
Sensor = cont.getSensor("sMouseX")
TouchSens = cont.getSensor("sTouch")
Actuator = cont.getActuator("aMouseX")

if TouchSens.isPositive():
        XPosition = Sensor.getXPosition()
        Sensitivity = 0.003
        CenterY = getWindowHeight()/2
        CenterX = getWindowWidth()/2
        Rotation = obj.getOrientation()[2][2]
        MoveX = (CenterX - XPosition)*Sensitivity

        Actuator.setDRot(0,0,MoveX,1)

        addActiveActuator(Actuator,1)
        setMousePosition(CenterX,CenterY)
```

Its allmost the same script as MouseY.py so i will only explane the difference:

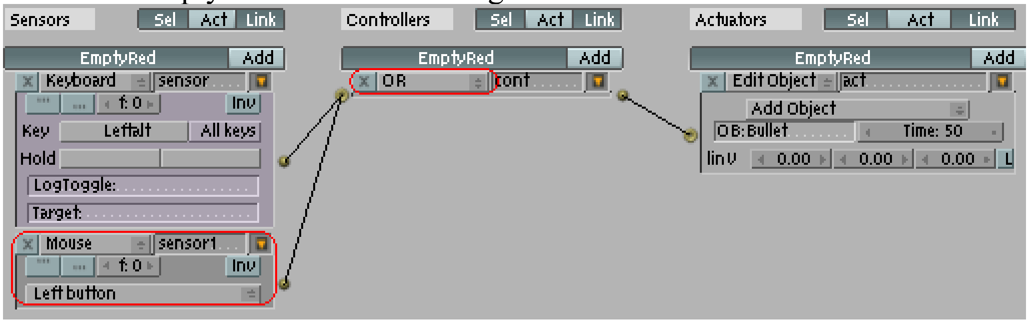
The name of the Variables changed from ...Y to ...X

```
TouchSens = cont.getSensor("sTouch")
if TouchSens.isPositive():
```

Gets the Touch Sensor and makes that the script will only run if the Touchsensor is positive. This way you cand turn around wile you are in the air...

As you have noticed there now Max Min Values, because you can turn around 360°

So you now can fully look around whit the mouse. But if you are using the mouse to look around it would help If you could use the mouse to shoot to.
Select the Empty Red and add this Logic Brick to it:



Keyboard sensor
f: 0
Key LeftAlt
LogToggle:
Target:
Mouse sensor1
f: 0
Left button
OR cont
Edit Object act
Add Object
OB:Bullet Time: 50
linV 0.00 0.00 0.00

Note the the Controller is changed to OR.

Now do all the Logic Brick addings we did whit the Red Player whit all the other Player.

So now it's Time to save again.
Whit the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.33WSAG.
Click on the Save as Button in the right upper Corner
PLEASE DON'T FORGET TO SUPPORT THIS PROJECT
How you can support this project is written at the beginning of this Tutorial
Status 3.86

So now there is still one little Problem to solve. At the moment the camera orientation is not beeing transmittet over the Network.
So we will have to change the network script a bit.

Add a New Text window and name it ShootOriPos.py
In that Script Window Write this Text:

```
from GameLogic import *
cont = GameLogic.getCurrentController()
obj = cont.getOwner()
GameLogic.ShootOri = obj.getOrientation()
GameLogic.ShootPos = obj.getPosition()
```

This will save the Orientation of the EmptyRed in to the Variable GameLogic.ShootOri and its Position in to the Variable GameLogic.ShootPos

Select the EmptyRed and add an Allways Sensor that is connectet whit the Python Controller that points to the script ShootOriPos.py
Do the same to all the other Players Shoot Empty.

So now we have to transmit this Info to the Server.
Open the Script ClientSend.py and add this Red Highlighted Text to it:

```
from GameLogic import *
from socket import *
from cPickle import *

if GameLogic.PlayerMod = = "Client":
        cont = GameLogic.getCurrentController()
```

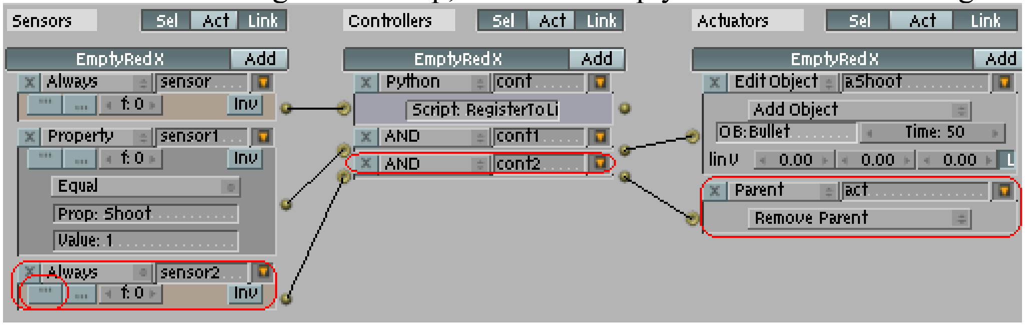
```
    obj = cont.getOwner()
    KeySens = cont.getSensor("sKeyimput")
    Keyinput = KeySens.getPressedKeys()

    PosClient = obj.getPosition()
    OriClient = obj.getOrientation()
    if Keyinput == [[125, 1]]:
        Shoot = 1
    else:
        Shoot = 0
    try:
        Data = dumps((GameLogic.YourPlayerName,GameLogic.PlayerTypS,PosClient,OriClient,Shoot,GameLogic.GPC.Text, GameLogic.ShootOri, GameLogic.ShootPos))
        GameLogic.sClient.sendto(Data,GameLogic.host)
    except:
        pass
```

So now the Client Sends the Info to the Server and the Server sends it to all the other Clients. Next step is that the Server and the Client can use this information to set the Camera Position (the Shoot Empty).
Open the Script ServerReceive.py and add the Red Highlighted text.

```
#---------------IMPORT-----------------#
from GameLogic import *
from socket import *
from cPickle import *
cont = GameLogic.getCurrentController()
obj = cont.getOwner()
#----------------Start Setup------------------#
if GameLogic.PlayerMod == "Server":
#----------------DATA------------------#
    PosClient = [0,0,0]
    OriClient = [[0,0,0],[0,0,0],[0,0,0]]
    GameLogic.IPList =[]
    DataList =[]
    DataPack =[]
    DataSize = 0
#------------------RECEIVE-------------------#
    GL = GameLogic
    for Port,Player,PlayerName,Shoot,PN,GPC in zip(GL.PortList,GL.PlayerList,GL.PlayerNameList,GL.ShootList,GL.PNList,GL.GPList):
        #print PN.Text, "=", Player.Ping
        if Player.Ping > 200:
            print "Disconnect", PN.Text
            Port.close()
            print "Port closed"
            GameLogic.PortList.remove(Port)
            GameLogic.PlayerList.remove(Player)
            GameLogic.PlayerNameList.remove(PlayerName)
            GameLogic.ShootList.remove(Shoot)
            GameLogic.PNList.remove(PN)
            GameLogic.GPList.remove(GPC)
            PN.Dis = 1
            GPC.Dis = 1
            print "List Entry Deleted"
            print "--------------------------"
        try:
            Data, CLIP = Port.recvfrom(1024)
            GameLogic.IPList.append(CLIP)
#---------------Set Game Play----------------#
            UPD = loads(Data)
            DataList.append(UPD)
            DataSize = DataSize + 1
            if DataSize == 3:
                DataPack.append(DataList)
                DataList = []
                DataSize = 0
            PosClient = [UPD[2][0],UPD[2][1],UPD[2][2]]
            OriClient = [UPD[3][0],UPD[3][1],UPD[3][2]]
            Player.setPosition(PosClient)
            Player.setOrientation(OriClient)
            Shoot.Shoot = UPD[4]
            Shoot.setOrientation(UPD[6])
            Shoot.setPosition(UPD[7])
            PN.Text = UPD[0]
            GPC.Text = UPD[5]
            Player.Ping = 0
        except:
            pass
#------------------SEND-------------------#
    if DataSize  != 0:
        DataSend1 = dumps((DataList))
        for CLIP in GameLogic.IPList:
            try:
                GameLogic.sMain.sendto(DataSend1, CLIP)
            except:
                pass
    for Data in DataPack:
        DataSend2 = dumps((Data))
        for CLIP in GameLogic.IPList:
            try:
                GameLogic.sMain.sendto(DataSend2, CLIP)
            except:
                pass
```

The added lines will set the Positon and Orientation of the Shoot Empty (EmptyRedX), whit the Information that it received over the network…

Now if you run the Setup by pressing the P-Key over the 3D Window in the Intro scene and start a server and a Client you will note that the bullet cube that will fly in strange directions on the Server side if you shoot. This is as far as I can tell a bug in the GameEngine. So we will have to undo the Parenting of the RedX (Player Cube) and the EmptyRedX (The shoot Empty). That's Why we ar not only transmitting the Orienation but also its Position…
To undo the Parenting relationship, select the EmptyRedX and add this Logic Bricks to it.



So you may ask yourself why is this Empty Parentet in the first Place?
Well this way we only have to create one object in realtime and all its Childs will be automaticly createt too.. Otherwise we would need to Emptys that would create the to obects at the same time… Now we only have 2 but what if you have lik 30 objects that together give you the setup of a Player…

So next and last step of the the integration of the Mouse in to this script is that the Clients also can use the Information the are receiving to set the Positon and Orientation of the other Clients Shoot Empty…
Open the script ClientReceive.py and add the red highlighted lines…

```
#---------------IMPORT-----------------#
from GameLogic import *
from socket import *
from cPickle import *
cont = GameLogic.getCurrentController()
obj = cont.getOwner()
#----------------DATA------------------#
if GameLogic.PlayerMod == "Client":
        PosClientX = [0,0,0]
        OriClientX = [[0,0,0],[0,0,0],[0,0,0]]
#------------RECEIVE--------------#
        Client = 1
        try:
                Data, SRIP = sClient.recvfrom(1024)
        except:
                obj.Ping = obj.Ping +1
                if obj.Ping == 100:
                        print "No Server Connection"
                if obj.Ping == 290:
                        print "Disconecting from Server"
                        print "-------------------------"
                Client = 0
        if Client ==1:
                if obj.Ping > 100:
                        print "Connection of Server is Back"
                        print "-------------------------"
        obj.Ping = 0
        if Data:
            UData = loads(Data)
            for PData in UData:
                if PData[0] in GameLogic.PlayerDict:
                    #if PData[0] == GameLogic.YourPlayerName:
                        #print "-------------------"
                        #print obj.Time
                        #obj.Time = 0.0
                        #print "-------------------"
                    if PData[0] != GameLogic.YourPlayerName:
                        PosClientX = [PData[2][0],PData[2][1],PData[2]
[2]]
                        OriClientX = [PData[3][0],PData[3][1],PData[3]
[2]]

    GameLogic.PlayerDict[PData[0]].setPosition(PosClientX)

    GameLogic.PlayerDict[PData[0]].setOrientation(OriClientX)
                        GameLogic.ShootDict[PData[0]].Shoot =
PData[4]

    GameLogic.ShootDict[PData[0]].setOrientation(PData[6])
                        GameLogic.ShootDict[PData[0]].setPosition(PDa
                    ta[7])
                        GameLogic.PlayerDict[PData[0]].Ping = 0

                if PData[0] not in GameLogic.PlayerDict:
                    if GameLogic.AddNew == 1:
                        GameLogic.NewPlayer = PData[0]
                        AddNewPlayer =
cont.getActuator("aAddPlayer")
                        AddNewPlayer.setObject(PData[1])
                        addActiveActuator(AddNewPlayer,1)
                        GameLogic.AddNew = 0
        else:
            GameLogic.s.close()
            obj.Client = 1
#---------------THE-END----------------#
```

This 2 added Lines do the same thing on the Clint side as the 2 lines we addet on the Server side.
So the Mouse is up and Working. Lean back and enjoy. (Note that from now on you will need a separate Computer (or Virtual PC) for each Server and Client other wise the Programm will crasch…) Because the mouse script will tell the mouse coursor to center in the Middle of the Window… and if you have to windows that tell one Mouse to center in there window the mouse wont know where in now sould center and it will crasch…

So now it’s Time to save again.
Whit the mouse over the left 3DWindow press F2, press the +-Key and your same name will change to 3.34WSAG.

Click on the Save as Button in the right upper Corner

PLEASE DON’T FORGET TO SUPPORT THIS PROJECT

How you can support this project is written at the beginning of this Tutorial