Subject: Re: Scripting Help
Posted by reborn on Thu, 28 Jan 2010 00:52:45 GMT
View Forum Message <> Reply to Message

Tell me if this works, I have not tested it... If it does, then I'll release it properly...

TeamReBalancer.ccp

```
#include "scripts.h"
#include <stdarg.h>
#ifdef WIN32
#define WIN32_LEAN_AND_MEAN
#include <windows.h>
#endif
#include "engine.h"
#include "gmmain.h"
#include "TeamReBalancer.h"

trbSettingsStruct *trbSettings = 0;
bool maploading, cannodchange, cangdichange = false;
int gdiflag, nodflag = 0;

void trbSettingsStruct::Load() {
 SettingsLoader::Load();
LoadInt(min,"MinimumPlayerDifferential");
LoadInt(time,"TimeAllowedForVolunteer");
}




void Plugin_Load() {
 trbSettings = new trbSettingsStruct("teamrebalancer.ini");
}
// Our own get team player count function, as the existing one is actually broken...
int reb_Get_Team_Player_Count(int Team)
{
 int Total = 0;
 GenericSLNode *x = BaseGameObjList->HeadNode;
 while (x)
 {
  GameObject *o = (GameObject *)x->NodeData;
  if (o && Commands->Is_A_Star(o))
  {
   if (Get_Team(Get_Player_ID(o)) == Team)
   {
   Total++;
  }
```

```
  }
  x = x->NodeNext;
 }
 return Total;
}



void Plugin_Unload() {
 delete trbSettings;
}

void reteam(){
 if(maploading == false){ // Make sure the map isn't loading
  // Get the team sizes
  int gdisize = reb_Get_Team_Player_Count(1);
  int nodsize = reb_Get_Team_Player_Count(0);
   if(nodsize - gdisize >= trbSettings->min || gdisize - nodsize >= trbSettings->min){
    // One team has more than the allowed differential
    if(nodsize > gdisize){ // Need to move a Nod player to GDI
     if(cannodchange == true){
      // means there is already a need rebalance, so sets a flag to make sure the old timer doesn't
cut off the new one
      nodflag++;
     }
     cannodchange = true;
     GameObject *thingy = Commands->Create_Object("Invisible_Object",Vector3(0.0f,0.0f,0.0f));
     Commands->Attach_Script(thingy,"g_volunteer_timer","");
     Console_Input(StrFormat("msg Nod have more players on their team than the server owner
deems to be fair, if you're on team Nod and want to change teams, please type \"!balance\". Nod
players have %i seconds to volunteer before the server chooses a player for them.",
trbSettings->time).c_str());
    }
    else{ // Need to move a GDI player to Nod
     if(cangdichange == true){
      // means there is already a need rebalance, so sets a flag to make sure the old timer doesn't
cut off the new one
      gdiflag++;
     }
     cangdichange = true;
     GameObject *thingy = Commands->Create_Object("Invisible_Object",Vector3(0.0f,0.0f,0.0f));
     Commands->Attach_Script(thingy,"n_volunteer_timer","");
     Console_Input(StrFormat("msg GDI have more players on their team than the server owner
deems to be fair, if you're on team GDI and want to change teams, please type \"!balance\". GDI
players have %i seconds to volunteer before the server chooses a player for them.",
trbSettings->time).c_str());
    }
```

```
   }
  }
  else{
   // Nothing, the map is loading so don't fuck up the player stats board at the end of the map.
   // I only included this else to show you why I made the conditional in the first place really.
  }
}

void g_volunteer_timer::Created(GameObject *obj){
 Commands->Start_Timer(obj,this,(float)trbSettings->time,1);
}

void g_volunteer_timer::Timer_Expired(GameObject *obj,int number){
 if(number == 1){
  if(gdiflag == 0){ //make sure this is the only active timer, so as not to stop a later one
prematurely.
   if(cangdichange == true){ // make sure no one has volunteered already using the chat command
    int iterations = 0;
    int score = 0;
    GameObject *volunteer;
   GenericSLNode *x = BaseGameObjList->HeadNode;
   while (x){
    GameObject *o = (GameObject *)x->NodeData;
    if (o && Commands->Is_A_Star(o) && Get_Team(Get_Player_ID(o)) == 1){
     if(iterations == 0){
      score = (int)Get_Score(Get_Player_ID(o));
      volunteer = o;
     }
     else{
      int newscore = (int)Get_Score(Get_Player_ID(o));
       if(newscore < score){
        score = newscore;
        volunteer = o;
       }
      }
     }
     iterations++;
     x = x->NodeNext;
    }
   Change_Player_Team(volunteer,false,false,true);
   }
   cangdichange = false;
   reteam();
  }
  gdiflag--;
 }
}
```

```
ScriptRegistrant<g_volunteer_timer> g_volunteer_timer_Registrant("g_volunteer_timer","");

void n_volunteer_timer::Created(GameObject *obj){
 Commands->Start_Timer(obj,this,(float)trbSettings->time,1);
}

void n_volunteer_timer::Timer_Expired(GameObject *obj,int number){
 if(number == 1){
  if(nodflag == 0){ //make sure this is the only active timer, so as not to stop a later one
prematurely.
   if(cannodchange == true){ // make sure no one has volunteered already using the chat
command
    int iterations = 0;
    int score = 0;
    GameObject *volunteer;
   GenericSLNode *x = BaseGameObjList->HeadNode;
   while (x){
    GameObject *o = (GameObject *)x->NodeData;
    if (o && Commands->Is_A_Star(o) && Get_Team(Get_Player_ID(o)) == 0){
     if(iterations == 0){
      score = (int)Get_Score(Get_Player_ID(o));
      volunteer = o;
     }
     else{
      int newscore = (int)Get_Score(Get_Player_ID(o));
       if(newscore < score){
        score = newscore;
        volunteer = o;
       }
      }
     }
     iterations++;
     x = x->NodeNext;
   }
   Change_Player_Team(volunteer,false,false,true);
   }
   cannodchange = false;
   reteam();
  }
  nodflag--;
 }
}

ScriptRegistrant<n_volunteer_timer> n_volunteer_timer_Registrant("n_volunteer_timer","");


class balanceChatCommand : public ChatCommandClass {
 void Triggered(int ID,const TokenClass &Text,int ChatType){
```

```
  int team = Get_Team(ID);
  if(team == 1 && cangdichange == true){
   Change_Player_Team(Get_GameObj(ID),false,false,true);
   Console_Input(StrFormat("msg Player %s has changed teams to rebalance the
game.",Get_Player_Name(Get_GameObj(ID))).c_str());
   Console_Input(StrFormat("ppage %d You have volunteered to balance the game and changed
teams.",ID).c_str());
   cangdichange = false;
   reteam();
  }
  else if(team == 0 && cannodchange == true){
   Change_Player_Team(Get_GameObj(ID),false,false,true);
   Console_Input(StrFormat("msg Player %s has changed teams to rebalance the
game.",Get_Player_Name(Get_GameObj(ID))).c_str());
   Console_Input(StrFormat("ppage %d You have volunteered to balance the game and changed
teams.",ID).c_str());
   cannodchange = false;
   reteam();
  }
  else{
   Console_Input(StrFormat("ppage %d You do not need to balance the teams right
now.",ID).c_str());
  }
 }
};
ChatCommandRegistrant<balanceChatCommand>
balanceChatCommandReg("!Balance;!balance;!BALANCE",CHATTYPE_ALL,0,GAMEMODE_AL
L);

extern "C" {



DLLEXPORT void SSGM_Player_Leave_Hook(int ID) {
reteam();
}


DLLEXPORT void SSGM_Player_Join_Hook(int ID, const char *Nick) {
 int gdisize = reb_Get_Team_Player_Count(1);
 int nodsize = reb_Get_Team_Player_Count(0);
 if(nodsize - gdisize < trbSettings->min && cannodchange == true){
  cannodchange = false;
  Console_Input(StrFormat("msg Nod's team no longer needs balancing now that %s has joined.",
Nick).c_str());
 }
 else if(gdisize - nodsize < trbSettings->min && cangdichange == true){
  cangdichange = false;
```

```
  Console_Input(StrFormat("msg GDI's team no longer needs balancing now that %s has joined.",
Nick).c_str());
 }
}

DLLEXPORT void SSGM_Level_Loaded_Hook() {
 trbSettings->Load();
maploading, cangdichange, cannodchange = false;
gdiflag, nodflag = 0;
}

DLLEXPORT void SSGM_GameOver_Hook() {
maploading = true;
}


}
```

TeamReBalancer.h

```
#include "gmsettingsclass.h"

#define PluginName "reborn's team rebalancer plugin"
#define PluginVersion "1.0"

struct trbSettingsStruct : public virtual SettingsLoader {
 trbSettingsStruct(const char *ININame) : SettingsLoader(ININame) {
  min = 0;
  time = 30;
 }

 void Load();
 int min, time;
};

void Plugin_Load();

int reb_Get_Team_Player_Count(int Team);

void reteam();

class g_volunteer_timer : public ScriptImpClass {
 void Created(GameObject *obj);
 void Timer_Expired(GameObject *obj,int number);
};

class n_volunteer_timer : public ScriptImpClass {
```

```
 void Created(GameObject *obj);
 void Timer_Expired(GameObject *obj,int number);
};

void Plugin_Unload();
```

teamrebalancer.ini
Quote:
[General]

; This plugin is designed to rebalance teams when players leave and cause an imbalance.

; This setting defines how many more players on One team must a side have before the code takes effect.
; DO NOT set this to 0! The default is 2, and that's a good number, I wouldn't set it any lower.
MinimumPlayerDifferential = 2

; This setting defines how long a team has to volunteer to !balance before the server does it for them. The time is in seconds.
TimeAllowedForVolunteer = 30