Subject: Re: For those people who don't know C++
Posted by jnz on Fri, 22 Jun 2007 00:10:36 GMT
View Forum Message <> Reply to Message

Here's a list of functions that will be in the end plugin.

Toggle Spoiler

void Console_Output(const char *output) //writes to the console
void Console_Input(const char *input) //inputs to the console (as if you were typing. Same as renrem)
const char *Get_Definition_Name(unsigned long id); //Convert a definition/preset ID into a name
unsigned long Get_Definition_ID(const char *name); //Convert a definition/preset name into an ID
unsigned long Get_Definition_Class_ID(const char *name); //get the class ID of a definition (matches the #defines above)
bool Is_Valid_Preset_ID(unsigned long ID); //Is this a valid preset id
bool Is_Valid_Preset(const char *Preset); //Is this a valid preset name
void Kill_All_Buildings_By_Team(int Team); //kill all buildings of the specified team,ending the game with the other team winning,0 = Nod,1 = GDI
void Damage_All_Buildings_By_Team(int Team,float Damage,const char *Warhead,GameObject *Damager); //damages all buildings of the specified team,the Damager is the object that will be passed to the Damaged callback for the buildings and also the one that will get the points,0 = Nod,1 = GDI,2 = both
void Repair_All_Static_Vehicles_By_Team(int Team,int Message); //sends a custom to all vechicles of the team with the DecorationPhys physics type (i.e. all base defense vechicles etc),0 = Nod,1 = GDI,Message is the message to send. Use with JFW_Repair_On_Custom on the vechicles you want repaired to do the actual repairing
void Set_Max_Health(GameObject *obj,float health); //set the max health of obj
void Set_Max_Shield_Strength(GameObject *obj,float shieldstrength); //set the max shield strength of obj
const char *Get_Shield_Type(GameObject *obj); //gets the shield type of obj
const char *Get_Skin(GameObject *obj); //gets the skin type of obj
void Set_Skin(GameObject *obj,const char *Skintype); //sets the skin type of obj
void Repair_All_Turrets_By_Team(int team,float health); //repairs all vehicles that have mode = turret
float Get_Damage_Points(GameObject *obj); //get the damage points for an object
float Get_Death_Points(GameObject *obj); //get the death points for an object
void Kill_Occupants(GameObject *obj); //kill all the occupants of a vehicle
void Damage_Occupants(GameObject *obj,float Damage,const char *Warhead); //Damage all the occupants of a vehicle
void Power_Base(int team,bool powered); //Power a base up or down, correctly handles the doubled build time and costs
void Set_Can_Generate_Soldiers(int team,bool cangenerate); //Sets if soliders are purchasable
void Set_Can_Generate_Vehicles(int team,bool cangenerate); //Sets if vehicles are purchaseable
void Destroy_Base(int team); //Destroys a base completly
void Beacon_Destroyed_Base(int team, bool destroyed); //Sets if a base was destroyed by a beacon in the beacon zone

void Enable_Base_Radar(int team, bool enable); //Enables radar for a base, same thing as the communications center does
bool Is_Harvester(GameObject *obj); //Is this object one of the 2 team AI harvesters?
bool Is_Radar_Enabled(int team); //returns if the radar is enabled
int Building_Type(GameObject *obj); //returns the type of a BuildingGameObj
bool Is_Building_Dead(GameObject *obj); //Does the game consider the building dead
GameObject *Find_Building(int team,int type); //Find a building by team and type
GameObject *Find_Base_Defense(int team); //Find base defense for a team
bool Is_Map_Flying(); //Is the current map a flying map
GameObject *Create_Building(const char *preset,const Vector3 & Position); //Create a building controller
GameObject *Find_Harvester(int team); //Find this teams current harvester, if any
bool Is_Base_Powered(int team); //Is this teams base powered
bool Can_Generate_Vehicles(int team); //Can this team buy vehicles
bool Can_Generate_Soliders(int team); //Can this team buy soldiers
cGameData *The_Game(); //get the cGameData
cGameDataSkirmish *The_Skirmish_Game(); //get the cGameDataSkirmish
cGameDataCnc *The_Cnc_Game(); //get the cGameDataCnc
cGameDataSinglePlayer *The_Single_Player_Game(); //get the cGameDataSinglePlayer
bool Is_A_Building(GameObject *obj); //is this object any building type
int Get_Building_Count_Team(int Team); //number of buildings a team has that are alive
GameObject *Find_Building_By_Team(int Team); //finds the first building of this team,0 = Nod,1 = GDI
GameObject *Find_Building_By_Name(int Team,const char *Preset_Name); //finds the first building of team where preset name matches name,0 = Nod,1 = GDI
GameObject *Find_Power_Plant(int Team); //finds the first powerplant for Team,0 = Nod,1 = GDI,2 = either
GameObject *Find_Refinery(int Team); //finds the first refinery for Team,0 = Nod,1 = GDI,2 = either
GameObject *Find_Repair_Bay(int Team); //finds the first repair bay for Team,0 = Nod,1 = GDI,2 = either
GameObject *Find_Soldier_Factory(int Team); //finds the first soldier factory for Team,0 = Nod,1 = GDI,2 = either
GameObject *Find_Airstrip(int Team); //finds the first airstrip for Team,0 = Nod,1 = GDI,2 = either
GameObject *Find_War_Factory(int Team); //finds the first war factory for Team,0 = Nod,1 = GDI,2 = either
GameObject *Find_Vehicle_Factory(int Team); //finds the first airstrip for Team,if none is found,finds the first war factory for Team,0 = Nod,1 = GDI,2 = either
GameObject *Find_Com_Center(int Team); //finds the first communications center for Team,0 = Nod,1 = GDI,2 = either
bool Is_Gameplay_Permitted(); //is gameplay permitted
bool Is_Dedicated(); //returns false if game.exe, true if WFDS/LFDS
bool Is_Linux(); //returns true if linux, false if win32
unsigned int Get_Current_Game_Mode(); //Get the current game mode, 0 = function error, 1 = single player, 2 = skirmish, 3 = WOL, 4 = GameSpy, 5 = LAN. May not always differentiate correctly between WOL, Gamespy and LAN, especially when running as a non-dedicated server.
void Get_Private_Message_Color(unsigned int *red, unsigned int *green, unsigned int *blue); //get color for private messages

void Get_Public_Message_Color(unsigned int *red, unsigned int *green, unsigned int *blue); //get color for public messages
int Get_Harvester_Preset_ID(int Team); //Get the harvester preset ID for this team
bool Is_Harvester_Preset(GameObject *obj); //Checks if the object has the same preset used for the harvesters
void Destroy_Connection(int PlayerID); //Drop a player from the game by cutting off their network link
const char *Get_IP_Address(int PlayerID); //gets the IP address for a player as a string in 1.2.3.4 format
const char *Get_IP_Port(int PlayerID); //gets the IP address and port for a player as a string in 1.2.3.4;5 format
int Get_Bandwidth(int PlayerID); //Get players current bandwidth (same as set by sbbo)
unsigned long Get_Ping(int PlayerID); //get the ping of a player
unsigned long Get_Kbits(int PlayerID); //get the Kbits/s of a player
int Get_Object_Type(GameObject *obj); //get the team of an object,0 = Nod,1 = GDI,2 = neutral,works on buildings
void Set_Object_Type(GameObject *obj,int type); //set the team of an object,0 = Nod,1 = GDI,2 = neutral,works on buildings
bool Is_Building(GameObject *obj); //is a BuildingGameObj
bool Is_Soldier(GameObject *obj); //is a SoldierGameObj
bool Is_Vehicle(GameObject *obj); //is a VechicleGameObj
bool Is_Cinematic(GameObject *obj); //is a CinematicGameObj
bool Is_ScriptZone(GameObject *obj); //is a ScriptZoneGameObj
bool Is_Powerup(GameObject *obj); //is a PowerUpGameObj
bool Is_C4(GameObject *obj); //is a C4GameObj
bool Is_Beacon(GameObject *obj); //is a BeaconGameObj
bool Is_Armed(GameObject *obj); //is an ArmedGameObj
bool Is_Simple(GameObject *obj); //is a SimpleGameObj
bool Is_PowerPlant(GameObject *obj); //is a PowerPlantGameObj
bool Is_SoldierFactory(GameObject *obj); //is a SoldierFactoryGameObj
bool Is_VehicleFactory(GameObject *obj); //is a VehicleFactoryGameObj
bool Is_Airstrip(GameObject *obj); //is an AirstripGameObj
bool Is_WarFactory(GameObject *obj); //is a WarFactoryGameObj
bool Is_Refinery(GameObject *obj); //is a RefineryGameObj
bool Is_ComCenter(GameObject *obj); //is a ComCenterGameObj
bool Is_RepairBay(GameObject *obj); //is a RepairBayGameObj
bool Is_Scriptable(GameObject *obj); //is a ScriptableGameObj
const char *Get_Building_Type(GameObject *obj); //returns a string indicating the building type of an object
void Get_Object_Color(GameObject *obj, unsigned int *red, unsigned int *green, unsigned int *blue); //get color for a player
GameObject *Find_Smart_Object_By_Team(int Team); //will find the first SmartGameObj (i.e. soldier or vechicle) for the team,0 = Nod,1 = GDI
GameObject *Find_Object_By_Team(int Team); //will find the first object for the team,0 = Nod,1 = GDI
GameObject *Find_Non_Player_Object_By_Team(int Team); //will find the first non player object for the team,0 = Nod,1 = GDI
GameObject *Find_Object_By_Preset(int Team,const char *Preset_Name); //will find the first

object of team matching preset,0 = Nod,1 = GDI,2 = either
GameObject *Find_Closest_Non_Building_Object_By_Team(int Team,Vector3 position); //find the closest non building object to position
GameObject *Find_Closest_Preset_By_Team(int Team,const Vector3 &pos,const char *Preset); //Find the closest object of this preset to this position
GameObject *Find_Random_Preset_By_Team(int Team,const char *Preset); //Find a random object on this team with this preset
int Get_Object_Count(int Team,const char *Preset); //Get the number of objects that exist with this preset
void Send_Custom_To_Team_Buildings(int Team,GameObject *sender,int message,int param,float delay); //send custom to team buildings
void Send_Custom_To_Team_Preset(int Team,const char *PresetName,GameObject *sender,int message,int param,float delay); //send custom to team preset
void Send_Custom_All_Objects_Area(int message,const Vector3 &Position,float Distance,GameObject *sender,int team); //sends a custom to all objects in a given area
void Send_Custom_All_Objects(int message,GameObject *sender,int team); //sends a custom to all objects
void Send_Custom_Event_To_Object(GameObject *sender,const char *Preset,int message,int param,float delay); //this is like Send_Custom_Event except it sends to all objects of the passed in preset
bool Is_Unit_In_Range(const char *preset,float range,Vector3 location,int team); //Is the given unit type in range of a location
bool Get_Is_Powerup_Persistant(GameObject *obj); //Returns if this powerup is persistant
void Set_Is_Powerup_Persistant(GameObject *obj,bool Persist); //Sets if this powerup is persistant
bool Get_Powerup_Always_Allow_Grant(GameObject *obj); //Returns if this powerup is set to always allows grant
void Set_Powerup_Always_Allow_Grant(GameObject *obj,bool Grant); //Change if this powerup is set to always allows grant
int Get_Powerup_Grant_Sound(GameObject *obj); //Returns the sound that is played when this powerup is picked up
void Set_Powerup_Grant_Sound(GameObject *obj,int SoundID); //Set the sound that is played when this powerup is picked up
void Grant_Powerup(GameObject *obj,const char *Preset_Name); //grants a powerup
GameObject *Get_Vehicle(GameObject *obj); //gets the vechicle that obj is driving
void Grant_Refill(GameObject *obj); //triggers the same code as the refill button on the PT
bool Change_Character(GameObject *obj,const char *Preset_Name); //will change the character of the passed in object to the passed in preset
void Create_Vehicle(const char *Preset_Name,float Delay,GameObject *Owner,int Team); //creates a vechicle,dont know what Delay is for,Owner is for the owner of the vechicle,Team says which sides factories to create it at
void Toggle_Fly_Mode(GameObject *obj); //makes a soldier fly if they arent flying or not fly if they are
int Get_Vehicle_Occupant_Count(GameObject *obj); //gets the count of occupants in a vehicle
GameObject *Get_Vehicle_Occupant(GameObject *obj,int seat); //gets the occupant in the given seat of the vehicle
GameObject *Get_Vehicle_Driver(GameObject *obj); //gets the driver of a vehicle
GameObject *Get_Vehicle_Gunner(GameObject *obj); //gets the gunner of a vehicle,if there is

only one person this should be the driver as well as the gunner
void Force_Occupant_Exit(GameObject *obj,int seat); //kick the object in <seat> out of the vehicle,if anyone is in that seat
void Force_Occupants_Exit(GameObject *obj); //kick all occupants out of the vehicle
GameObject *Get_Vehicle_Return(GameObject *obj); //like Get_Vehicle but will return the soldier if its not inside a vehicle instead of NULL
bool Is_Stealth(GameObject *obj); //is this object stealth
bool Get_Fly_Mode(GameObject *obj); //is this infantry flying via Set_Fly_Mode
int Get_Vehicle_Seat_Count(GameObject *obj); //get the seat count for a vehicle
char *Get_Spawn_Char(int Team); //get the spawn character for a team
void Change_Spawn_Char(int Team,const char *Name); //change the spawn character for a team (does not reset after the map ends)
void Soldier_Transition_Vehicle(GameObject *obj); //makes the soldier exit the vehicle they are in (or if right next to a vehicle, get in)
unsigned int Get_Vehicle_Mode(GameObject *obj); //Gets the mode of a vehicle
int Get_Team_Vehicle_Count(int team); //Gets the current vehicle count for a team
GameObject *Get_Vehicle_Owner(GameObject *obj); //Gets the owner (if any) of a vehicle. Will not work if used within the first second after the ::Created event is called
void Force_Occupants_Exit_Team(GameObject *obj,int team); //kick all occupants not of a given team out of the vehicle
unsigned int Get_Vehicle_Definition_Mode(const char *preset); //get the mode of a vehicle given its preset name
GameObject *Find_Closest_Zone(Vector3 &Location,unsigned int Type); //Find the closest zone
bool IsInsideZone(GameObject *zone,GameObject *obj); //is <solder/vehicle> inside <zone>. Wont work if object is inside a zone when its created (e.g. spawns inside zone or zone is moved around them with Create_Zone or Set_Zone_Box)
unsigned int Get_Vehicle_Definition_Mode_By_ID(unsigned long ID); //Get the mode of a vehicle given its preset ID
unsigned int Get_Zone_Type(GameObject *obj); //Get the type of a script zone
OBBoxClass *Get_Zone_Box(GameObject *obj); //Get the box (size/position) of a ScriptZoneGameObj
void Set_Zone_Box(GameObject *obj,const OBBoxClass &box); //Set the box (size/position) of a ScriptZoneGameObj
GameObject *Create_Zone(const char *preset,const OBBoxClass &box); //Create a script zone and set its box
bool PointInZone(GameObject *obj,const Vector3 &v); //Is a point in a zone
unsigned int Overlap_Test(const OBBoxClass &box,const Vector3 &v); //Test if a vector is inside an OBBoxClass
bool IsAvailableForPurchase(GameObject *factory); //Is it possible to purchase a vehicle from this factory
bool Check_Transitions(GameObject *obj,bool unk); //Trigger vehicle transition on an object
GameObject *Get_Vehicle_Gunner_Pos(GameObject *obj); //Get the vehicle gunner, returns zero if there is no gunner
void Set_Vehicle_Is_Visible(GameObject *obj,bool visible); //works like Set_Is_Visible but for vehicles, makes them be ignored by Enemy_Seen
void Set_Vehicle_Gunner(GameObject *obj,int seat); //set gunner for this vehicle
const char *Get_Model(GameObject *obj); //get the name of the 3d model used by an object,opposite of Set_Model

float Get_Animation_Frame(GameObject *obj); //gets the current animation frame for obj,may not work for all objects
bool Is_TrackedVehicle(GameObject *obj); //has TrackedVechicle physics
bool Is_VTOLVehicle(GameObject *obj); //has VTOLVechicle physics
bool Is_WheeledVehicle(GameObject *obj); //has WheeledVechicle physics
bool Is_Motorcycle(GameObject *obj); //has Morotcycle physics
bool Is_Door(GameObject *obj); //has Door physics
bool Is_Elevator(GameObject *obj); //has Elevator physics
bool Is_DamageableStaticPhys(GameObject *obj); //has DamageableStaticPhys physics
bool Is_AccessablePhys(GameObject *obj); //has AccessablePhys physics
bool Is_DecorationPhys(GameObject *obj); //has DecorationPhys physics
bool Is_HumanPhys(GameObject *obj); //has HumanPhys physics
bool Is_MotorVehicle(GameObject *obj); //has MotorVehicle physics
bool Is_Phys3(GameObject *obj); //has Phys3 physics
bool Is_RigidBody(GameObject *obj); //has RigidBody physics
bool Is_ShakeableStatricPhys(GameObject *obj); //has ShakeableStaticPhys physics
bool Is_StaticAnimPhys(GameObject *obj); //has StaticAnimPhys physics
bool Is_StaticPhys(GameObject *obj); //has StaticPhys physics
bool Is_TimedDecorationPhys(GameObject *obj); //has TimedDecorationPhys physics
bool Is_VehiclePhys(GameObject *obj); //has VechiclePhys physics
bool Is_DynamicAnimPhys(GameObject *obj); //has DenamicAnimPhys physics
bool Is_BuildingAggregate(GameObject *obj); //has BuildingAggregate physics
bool Is_Projectile(GameObject *obj); //has Projectile physics
const char *Get_Physics(GameObject *obj); //returns a string indicating the physics type of an object
void Copy_Transform(GameObject *in,GameObject *out); //copies the complete transform (including rotation) from one object to another
float Get_Mass(GameObject *obj); //returns the mass of an object
const char *Get_Htree_Name(GameObject *obj); //returns the Hierarchy Tree name for an object
char Get_Sex(GameObject *obj); //gets the character (e.g. "A" for male or "B" for female) for use with animations
void Create_Effect_All_Stealthed_Objects_Area(const Vector3 &Position,float Distance,const char *object,const Vector3 &offset,int team); //creates an indicator object near every stealthed object in a given area
void Create_Effect_All_Of_Preset(const char *object,const char *preset,float ZAdjust,bool ZSet); //Create an object above all objects of a given preset, also set the facing to match the object its being created over.
GameObject *Get_GameObj(int PlayerID); //convert a player ID into a GameObject
long Get_Player_ID(GameObject *obj); //convert a GameObject into a player ID
const char *Get_Player_Name(GameObject *obj); //converts a GameObject into a player name
const char *Get_Player_Name_By_ID(int PlayerID); //gets the player name from a player ID. Will return NULL if that player doesnt exist.
void Change_Team(GameObject *obj,int Team); //changes the team of a player given their GameObject and also kills the player so they respwan,passing anything other than 0 = Nod,1 = GDI will crash
void Change_Team_By_ID(int PlayerID,int Team); //changes the team of a player given their ID and also kills the player so they respwan,passing anything other than 0 = Nod,1 = GDI will crash
int Get_Player_Count(); //gets the count of how many players there are

int Get_Team_Player_Count(int Team); //gets the count of players for a given team,0 = Nod,1 = GDI)
int Get_Team(int PlayerID); //get the team of a player
int Get_Rank(int PlayerID); //get the rank of a player
int Get_Kills(int PlayerID); //get the kills of a player
int Get_Deaths(int PlayerID); //get the deaths of a player
float Get_Score(int PlayerID); //get the score of a player
float Get_Money(int PlayerID); //get the money of a player
float Get_Kill_To_Death_Ratio(int PlayerID); //get the kill/death ratio of a player
GameObject *Get_Part_Name(const char *name1); //Will return the player with this string as part of their name if there is exactly one player with it
int Get_Part_Names(const char *name1); //Will return the count of players with this string as part of their name
void Get_Team_Color(unsigned int team, unsigned int *red, unsigned int *green, unsigned int *blue); //get color for a team
void Get_Player_Color(int PlayerID, unsigned int *red, unsigned int *green, unsigned int *blue); //get color for a player
GameObject *Get_GameObj_By_Player_Name(const char *name); //get the gameobject of a player given their name
bool Purchase_Item(GameObject *obj,int cost); //Pass a soldier object & a cost to deduct that much money from the soldier. Returns true if the transaction suceeded, false otherwise
void Set_Ladder_Points(int PlayerID,int amount); //Set the ladder points of a player
void Set_Rung(int PlayerID,int amount); //Set the rung of a player
void Set_Money(int PlayerID,float amount); //Set the money of a player
void Set_Score(int PlayerID,float amount); //Set the score of a player
GameObject *Find_First_Player(int Team); //finds the first player of Team,0 = Nod,1 = GDI,2 = either
bool Change_Player_Team(GameObject *obj,bool Retain_Score,bool Retain_Money,bool Show_Host_Message); //changes the players team
int Tally_Team_Size(int team); //gets the team size for a team
float Get_Team_Score(int team); //gets the score for a team (same as Game_Info uses)
char *stristr(const char *m_pStr1, const char *m_pStr2); //like strstr but case insenstive
void Send_Custom_All_Players(int message,GameObject *sender,int team); //send a custom to all players
float Steal_Team_Credits(float percentage, int team); //Steal credits from a team
float Get_Team_Credits(int team); //Count the total credits for a team
void Change_Team_2(GameObject *obj,int Team); //changes the team of a player given their GameObject without killing the player,passing anything other than 0 = Nod,1 = GDI will crash
int Get_Player_Type(GameObject *obj); //Get the player type of a player from the cPlayer
TeamPurchaseSettingsDefClass *Get_Team_Purchase_Definition(unsigned long team); //Gets the TeamPurchaseSettingsDefClass for a team
PurchaseSettingsDefClass *Get_Purchase_Definition(unsigned long type, unsigned long team); //Gets the PurchaseSettingsDefClass for a given team and type
unsigned int Get_Team_Cost(const char *preset,unsigned int team); //Get the cost of a preset for a given team. Returns zero if the preset is not found in any of the purchase terminal data or if it is one of the free units.
unsigned int Get_Cost(const char *preset); //Get the cost of a preset. Returns zero if the preset is not found in any of the purchase terminal data or if it is one of the free units.

void Set_Enlisted(unsigned int team, unsigned int position, unsigned int presetid, unsigned int stringid, const char *texture); //Change an enlisted soldier for a team
void Set_Beacon(unsigned int team, unsigned int cost, unsigned int presetid, unsigned int stringid, const char *texture); //Change the beacon for a team
void Set_Refill(unsigned int team, unsigned int stringid, const char *texture); //Change the refill for a team. Setting this to zero or blank does not actually disable refill.
void Set_Preset(unsigned int team, unsigned int type, unsigned int position, unsigned int cost, unsigned int presetid, unsigned int stringid, const char *texture); //Change a preset given a type and team
void Set_Alternate(unsigned int team, unsigned int type, unsigned int position, unsigned int altpos, unsigned int presetid, const char *texture); //Change an alternate given a type and a team
void Disable_Enlisted(unsigned int team, unsigned int position); //Disable an enlisted soldier
void Disable_Preset(unsigned int team, unsigned int type, unsigned int position); //Disable a preset
void Disable_Enlisted_By_Name(unsigned int Team,const char *Name); //Disable an enlisted unit by name
void Disable_Preset_By_Name(unsigned int Team,const char *Name); //Disable a preset by name
void Disable_All_Ground_Vehicles(unsigned int team); //Removes all ground vehicles from this teams PT menu
void Disable_All_Flying_Vehicles(unsigned int team); //Removes all flying vehicles from this teams PT menu
char *Get_Team_Icon(const char *preset,unsigned int team); //Get the icon texture name for a given preset of a given team. Returns NULL if it cant find that preset in the PT data.
char *Get_Icon(const char *preset); //Get the icon texture name for a given preset. Returns NULL if it cant find that preset in the PT data.
void Remove_Script(GameObject *obj,const char *script); //removes all copies of <script> from an object
void Remove_All_Scripts(GameObject *obj); //removes all scripts from an object
void Attach_Script_Preset(const char *script,const char *params,const char *preset,int team); //attached <script> to all objects of <preset> in team <team>
void Attach_Script_Type(const char *script,const char *params,unsigned long type,int team); //attaches <script> to all objects of <type> in team <team>
void Remove_Script_Preset(const char *script,const char *preset,int team); //removes all copies of <script> from all objects of <preset> in team <team>
void Remove_Script_Type(const char *script,unsigned long type,int team); //removes all copies of <script> from all objects of <type> in team <team>
bool Is_Script_Attached(GameObject *obj,const char *script); //is the script attached
void Attach_Script_Once(GameObject *obj,const char *script,const char *params); //attach a script if its not already attached
void Attach_Script_Preset_Once(const char *script,const char *params,const char *preset,int team); //attach a script to all objects of preset if its not already attached
void Attach_Script_Type_Once(const char *script,const char *params,unsigned long type,int team); //attach a script to all objects of type if its not already attached
void Attach_Script_Building(const char *script,const char *params,int team); //attach a script to all buildings
void Attach_Script_Is_Preset(GameObject *obj,const char *preset,const char *script,const char *params,int team); //attach the script if object is of preset
void Attach_Script_Is_Type(GameObject *obj,unsigned long type,const char *script,const char

*params,int team); //attach the script if object is of type
void Attach_Script_Player_Once(const char *script,const char *params,int team); //attach a script to all players if its not already attached
void Remove_Duplicate_Script(GameObject *obj, const char *script); //remove duplicate scripts from an object
void Attach_Script_All_Buildings_Team(int Team,const char *Script,const char *Params,bool Once); //attach a script to all buildings by team
void Attach_Script_All_Turrets_Team(int Team,const char *Script,const char *Params,bool Once); //attach a script to all static vehicles by team
GameObject *Find_Building_With_Script(int Team,int Type,const char *Script,GameObject *Caller); //Find a building of this type with this script attached to it
GameObject *Find_Object_With_Script(const char *script); //Find the first object with this script on it
int Get_Current_Bullets(GameObject *obj); //Get loaded bullets for an objects current gun
int Get_Current_Clip_Bullets(GameObject *obj); //Get clip/backpack bullets for an objects current gun
int Get_Current_Total_Bullets(GameObject *obj); //Get total bullets for an objects current gun
int Get_Total_Bullets(GameObject *obj,const char *weapon); //Get total bullets for a specific gun (if the object doesnt have the gun, return is zero)
int Get_Clip_Bullets(GameObject *obj,const char *weapon); //Get clip/backpack bullets for a specific gun (if the object doesnt have the gun, return is zero)
int Get_Bullets(GameObject *obj,const char *weapon); //Get loaded bullets for a specific gun (if the object doesnt have the gun, return is zero)
int Get_Current_Max_Bullets(GameObject *obj); //Get max loaded bullets for an objects current gun
int Get_Current_Clip_Max_Bullets(GameObject *obj); //Get max clip/backpack bullets for an objects current gun
int Get_Current_Total_Max_Bullets(GameObject *obj); //Get total bullets for an objects current gun
int Get_Max_Total_Bullets(GameObject *obj,const char *weapon); //Get max total bullets for a specific gun (if the object doesnt have the gun, return is zero)
int Get_Max_Clip_Bullets(GameObject *obj,const char *weapon); //Get max clip/backpack bullets for a specific gun (if the object doesnt have the gun, return is zero)
int Get_Max_Bullets(GameObject *obj,const char *weapon); //Get max loaded bullets for a specific gun (if the object doesnt have the gun, return is zero)
int Get_Position_Total_Bullets(GameObject *obj, int position); //Get total bullets for an objects gun at a specific position
int Get_Position_Bullets(GameObject *obj,int position); //Get loaded bullets for an objects gun at a specific position
int Get_Position_Clip_Bullets(GameObject *obj,int position); //Get clip/backpack bullets for an objects gun at a specific position
int Get_Position_Total_Max_Bullets(GameObject *obj, int position); //Get total bullets for an objects gun at a specific position
int Get_Position_Max_Bullets(GameObject *obj,int position); //Get loaded bullets for an objects gun at a specific position
int Get_Position_Clip_Max_Bullets(GameObject *obj,int position); //Get clip/backpack bullets for an objects gun at a specific position
void Set_Current_Bullets(GameObject *obj,int bullets); //Set current loaded bullets for an object

void Set_Current_Clip_Bullets(GameObject *obj,int bullets); //Set current clip/backpack bullets for an object
void Set_Position_Bullets(GameObject *obj,int position,int bullets); //Set position loaded bullets for an object
void Set_Position_Clip_Bullets(GameObject *obj,int position,int bullets); //Set position clip/backpack bullets for an object
void Set_Bullets(GameObject *obj,const char *weapon,int bullets); //Set loaded bullets for an object
void Set_Clip_Bullets(GameObject *obj,const char *weapon,int bullets); //Set clip/backpack bullets for an object
const char *Get_Powerup_Weapon(const char *Powerup); //Get the weapon name that a powerup will grant if collected
AmmoDefinitionClass *Get_Weapon_Ammo_Definition(const char *weapon,bool PrimaryFire); //Get the AmmoDefinitionClass of a weapon given its preset name
AmmoDefinitionClass *Get_Current_Weapon_Ammo_Definition(GameObject *obj,bool PrimaryFire); //Get the AmmoDefinitionClass of an objects current weapon
AmmoDefinitionClass *Get_Position_Weapon_Ammo_Definition(GameObject *obj,int position,bool PrimaryFire); //Get the AmmoDefinitionClass of an objects weapon at tbe specified position
WeaponDefinitionClass *Get_Weapon_Definition(const char *weapon); //Get the WeaponDefinitionClass of a weapon given its preset name
WeaponDefinitionClass *Get_Current_Weapon_Definition(GameObject *obj); //Get the WeaponDefinitionClass of an objects current weapon
WeaponDefinitionClass *Get_Position_Weapon_Definition(GameObject *obj,int position); //Get the WeaponDefinitionClass of an objects weapon at the specified position
ExplosionDefinitionClass *Get_Explosion(const char *explosion); //Get the ExplosionDefinitionClass of an explosion given its preset name
const char *Get_Powerup_Weapon_By_Obj(GameObject *Powerup); //Get the name of a powerup weapon given a PowerupGameObj
int Get_Current_Weapon_Style(GameObject *obj); //Get weapon style for an objects current gun
int Get_Position_Weapon_Style(GameObject *obj,int position); //Get weapon style for an objects gun at a specific position
int Get_Weapon_Style(GameObject *obj,const char *weapon); //Get weapon style for a specific gun (if the object doesnt have the gun, return is zero)
void Disarm_Beacon(GameObject *obj); //Disarm a beacon
void Disarm_Beacons(int PlayerID); //Disarm all beacons for a player
void Disarm_Nearest_Beacon(GameObject *Host,int Team,bool Nearest); //disarms the nearest beacon
void Disarm_C4(GameObject *obj); //Disarm a C4 object
void Disarm_All_Proxy_C4(int PlayerID); //Disarm all proximity C4 owned by a player
void Disarm_All_C4(int PlayerID); //Disarm all C4 owned by a player
const char *Get_Current_Weapon(GameObject *obj); //gets the current weapon of an object
int Get_Weapon_Count(GameObject *obj); //gets the weapon count for an object
const char *Get_Weapon(GameObject *obj,int position); //get the specified position in the weapon bag
bool Has_Weapon(GameObject *obj,const char *weapon); //does the object have the weapon
GameObject *Find_Beacon(int Number,int Team); //find a beacon
int Get_C4_Count(int Team); //get the C4 count

int Get_Beacon_Count(int Team); //get the beacon count
char Get_Mine_Limit(); //gets the current mine limit (use the MLIMIT console command to set it)
int Get_C4_Mode(GameObject *obj); //get the C4 mode for a C4GameObj, 1 = remote, 2 = timed, 3 = proximity, anything else is invalid
int Get_C4_Count_Proximity(int Team); //Get the proximity C4 count for a team
int Get_C4_Count_Remote(int Team); //Get the remote C4 count for a team
GameObject *Get_C4_Planter(GameObject *obj); //Gets the planter of a C4GameObj
GameObject *Get_C4_Attached(GameObject *obj); //Gets the object a C4GameObj is attached to (if any)
GameObject *Get_Beacon_Planter(GameObject *obj); //Gets the planter of a BeaconGameObj

I will probably release this without all the functions, because they are going to take some time to do.

---