HaZardModding Coop Mod for MOHAA Scripting

written by chrissstrahl on 2019.04.16
last update on 2019.04.16

PLEASE NOTE
This document is directed towards scripters that want to use the HZM Coop Mod and and know how to handle level scripts. This is not directed towards beginners.
The HZM Coop Mod for AA and BT was developed using MOH-Breaktrough so it run on BT.

FOREWORD
The HaZardModding Coop Mod might not be big in filesize, non the less the scripts of the mod are more complex than a regular map. This document was created to provide a overview and provide some help.

Imagine the HZM Coop Mod as a own gametype inside the actual gametype, handling player spawn iventory and so forth.

The HZM Coop Mod is designed to handle player actions/events correctly in single and multiplayer, this means if you are using a coop script function it will work in both sp/mp (with a few exceptions).

INDEX

VARIABLES

The HZM Coop Mod uses a few global variables that stay the same and are required for the mod to operate as expected. Other variables used are usually limited to the current level. Some Variables can be adjusted (green), but most are auto-managed by the coop mod and should not be changed.

NAME	TYPE	DESCRIPTION
level.coop_svmaxclients	int	Holds the max player/clients slots (speed benefit)
level.coop_mapname	string	Holds the map name (speed benefit)
level.coop_playerReady	bool	0 on default 1 if any player is ready and has joined
level.coop_musicCommandVol	int	Holds current volume for the music
level.coop_musicCommand	string	Holds current music command that is send to players
level.coop_inJeep	int	Used to check if player should be in jeep or not
level.coop_playerGlue	bool	Used to check if player should be glued
level.coop_playerHide	bool	Used to check if player should be hidden
level.coop_player	entity	Holds the player that the team ai is following
level.coop_playerTank	entity	Holds the player that is currently driving the tank
level.enemyhealthscalar	float	Health scaler - used in global/spawner.scr
level.coop_noWeapon	bool	Specifies if players should have no weapons
level.coop_threatbias	float	Specifies how much players are a threat to enemy AI
level.coop_aaMap	bool	Makes Coop & global scripts handle the map like in AA
level.coop_health	int	Sets players health (MP only)
level.coopMedicHealth	int	Sets health for Medic Friendly AI
level.coop_prespawn	bool	Used in replace.scr::waitTillPrespawn
level.script	string	Stores current script filename, used in global scripts

SPAWNLOCATIONS

The HZM Coop Mod holds all spawnlocations in a single file, this is convinient for us developers.
How ever, we recommend that you put the spawnlocations at the very top of your level script its main function, right before/above anything else. This is to make sure that your development does not get affected with changes occuring in the HZM Coop Mod and vice versa.

Each player has a own spawn point variable, which should contain the location the player can saftly spawn or respawn. To spawn all players at the same location, you need to specify only the spawnlocation with the number 1, the same applies for the facing angle variable. How ever, we do NOT recommend spawning all players at the same location, so use it only during testing.

We recommend that you update the spawnlocations whith the advancement of the mission progress, so that the players spawn close to where the other players are in the mission.

In the Example code you can see how you can implement the spawnlocations directly in your level script.
If you want to change/update the spawnlocations during the mission, you need to overwrite the value of each variable, by a script function, like shown in the example code.

```
1  main:{
2    level.flags["coop_spawn1origin"] = ( -2682 -3235 120 ) //player 1 spawn
3    level.flags["coop_spawn2origin"] = ( -2642 -3235 120 ) //player 2 spawn
4    level.flags["coop_spawn3origin"] = ( -2602 -3235 120 ) //player 3 spawn
5    level.flags["coop_spawn4origin"] = ( -2722 -3235 120 ) //player 4 spawn
5    level.flags["coop_spawn5origin"] = ( -2762 -3235 120 ) //player 5 spawn
7    level.flags["coop_spawn6origin"] = ( -2802 -3235 120 ) //player 6 spawn
8    level.flags["coop_spawn7origin"] = ( -2842 -3235 120 ) //player 7 spawn
9    level.flags["coop_spawn8origin"] = ( -2882 -3235 120 ) //player 8 spawn
10   level.flags["coop_spawn1angles"] = ( 0 1 0 ) //player 1 (applies to every other that is not set)
11    ...//this represents other code, variables above need to be first inizialised before all else code
12 }end //end main
```

FUNCTIONS

The HZM Coop Mod has many new functions that handle things in the background. Some of these functions are meant to replace some commands who are used in singleplayer, and some are just specific for coop.
The Tables below hold a overview of the functions in the coop files.

Types and recommended execution of the used functions		
Short	Description	Code
s	for Singleplayer only	-
c	for Coop (works too in sp)	-
r	Replacment, used to replace other code. Keeping the usual parameters	-
et	Needs to be executed by a entity	**$ai exec** coop_mod/replace.scr::tmstop
ew	Entity and waitexec	**$ai waitexec** coop_mod/replace.scr::cansee 120 1200
we	Needs to be executed with waitexec	**waitexec** coop_mod/replace.scr::waittilldrive 0.25 0.1
ex	Should be executed normally	**exec** coop_mod/replace.scr::forceTeam $player[4]

coop_mod/main.scr	This file contains the coop mod core functions	
Function (with parameters)	**Types**	**Description**
forceTeam local.player	c,ex	Forces the given player in the allied team
resetSpawn local.player	c,ex	Deletes last dynamic respawn location for player
printInfo local.player local.message local.bold	c,ex	Prints a message to player hud, prevents next message to show before 3 sec after the last message
playerFace local.face	r,et	Faces executing player to the given direction Replaces: $player face (0 69 0)
playerTakeAll local.player	r,ex	Takes weapons and Items from player Replaces: $player takeall
playerPlaceAtSpawn local.i	c,ex	Places a player at the given spawnpoint number
playerMakeSolidAsap local.i	c,ex	Makes given player with the number solid as soon as it is saftely possible, so the player will not get stuck in other player, medics, teammates or tanks
itemGetAll	c,et	Gives all registered Items to executing player
friendlyPlayerOrigin	r,ex	Returns the origin of teh player, the Team AI is following. Replacs (in context): $player.origin
destination	r,et	Replacement command for destination. Replaces: $someAI destination $player
tether	r,et	Replacement command for tether Replaces: $someAI tether $player
isPlayerActive local.player	c,we	Checks if a player is active (alive,not a spectator)
getPlayerId local.player	c,we	Returns the ID of given player
getActiveWeapon local.player	c,we	Returns the current weapon of given player
isPrimaryWeaponActive local.player	c,we	Check if primary weapon of given player is active
isWeaponModel local.model	c,we	Checks if given model is in weapons folder
isPrimaryWeapon local.weapon	c,we	Checks if the given Model is a primary Weapon
playerForcePrimary local.player local.weapon	c,ex	Forces given Primary Weapon on given Player
playerForceSecondary local.player local.weapon	c,ex	Forces given Secondary Weapon on given Player

coop_mod/replace.scr	This file is primarly for replacing code

Function (with parameters)	Type	Description
waitTillPrespawn	r,we	Replaces: level waittill prespawn
waitForPlayer	r,we	Replaces: level waittill spawn, waits until players are on the server and ready. (in a desirable fashion)
isPlayerArray local.entity	c,we	Checks if given entity is $player and a array
player_anyCanBeSeen local.ent local.fov local.range	r,we	Replaces: $player cansee $someAi 120 1200
player_numActive	c,we	Returns number of active players
player_random	c,we	Returns a random valid player
player_anyValid	c,we	Returns any valid player (sorted by player number)
player_anyPreferValid	c,we	Returns any player (sorted by player number)
player_closestTo local.object local.origin	c,we	Returns closest player to a Entity or origin
player_origin	c,we	This function needs a update, don't use it
waittill_spawn	r,we	Waits until any player is present on the server. Replaces: level waittill spawn (in a simple fashion)
stopwatch local.time	r,ex	Replaces: $player stopwatch
viewangles local.vecAngles	r,ex	Replaces: $player.viewangles =
tmstop	r,ex	Replaces: $player stufftext "tmstop"
tmstartloop local.file	r,ex	Replaces: $player stufftext "tmstartloop ..."
tmstart local.file	r,ex	Replaces: $player stufftext "tmstart ..."
tmvolume local.vol	r,ex	Replaces: $player stufftext "tmvolume ..."
lookat	r,et	Replaces: $ai lookat $player (selects closest player)
eyeslookat	r,et	Replaces: $ai eyeslookat $player (closest player)
turnto	r,et	Replaces: $ai turnto $player (selects closest player)
istouching local.touchMeBaby	r,we	Replaces: $player istouching $entity (checks all players)
getToucher local.touchMeBaby	c,we	Returns: player touching given entity
aimat	r,et	Replaces: $ai aimat $player (selects closest once)
set_hasdisguise local.disguise	r,ex	Replaces: $player.has_disguise
ammo local.type local.amount local.sound	r,ex	Replaces: $player ammo
sighttrace local.offset local.vec local.pass local.min local.max	r,we	Replaces: $player sighttrace
item local.item local.use	r,ex	Replaces: $player item
take local.item	r,ex	Replaces: $player take
takeAll	r,ex	Replaces: $player takeall
cansee local.fov local.range	r,ew	Replaces: $ai cansee $player ...
canseenoents local.fov local.range	r,ew	Replaces: $ai canseenoents $player ...
playerCansee local.ent local.fov local.range	r,we	Replaces: $player cansee $ai
canseeGetClosest local.fov local.range	c,ew	Returns: Closest player for executing entity
threatbias local.val	r,ex	Replace: $player threadbias
killplayer	c,ex	Kills all players
glue local.entity local.angle	r,ex	Replaces: $player glue
unglue local.entity	r,ex	Replaces: $player unglue
physics_on	r,ex	Replaces: $player physics_on
physics_off	r,ex	Replaces: $player physics_off
show	r,ex	Replaces: $player show
hide	r,ex	Replaces: $player hide
forcelegsstate local.state:	r,ex	Replaces: $player forcelegstate
withinDistanceOf local.ent local.distance	c,we	Returns true if a player is within given distance of given entity
withinDistance local.origin local.distance	c,we	Returns true if a player is within given distance of given origin

playsound local.sound	r,ex	Replaces: $player playsound "..."
loopsound local.sound	r,ex	Replaces: $player loopsound "..."
stoploopsound local.sound	r,ex	Replaces: $player stoploopsound
runtoClosest	c,ex	Makes executing ai run to closest player
origin local.ent	c,we	Returns: Origin of closest player to given entity
holster	r,ex	Replaces: $ai holster (fixes issues)
unholster	r,ex	Replaces: $ai unholster (fixes issues)
spawnclip local.origin local.mins local.maxs local.targetname local.scale local.angle local.message local.requiredentity local.range	c,ex	Spawns a invisible wall/clip, so players/ai can not pass. The clip removes it self as soon as the given required-entity is within given range. A message is printed each time the clip is touched.
waittilldrive local.delay local.wait	c,we	Waits until vehicle stops, optional self specifified delay time. Needs to be called with waitexec.
waittillRange local.other local.range	c,we	Waits until self is in given range of given other.
waittillRangeVector local.vector local.range	c,we	Waits until self is in given range of given vector.
originOffset local.offset	c,ex	Used to move objects relative to their current origin
solid local.awaypos	c,et	Makes executing entity solid and moves all players to the given vector, which would get stuck in the object
skip local.useOnly	c,ex	Checks if players vote(click) to skip current cinematic/intro.
teleportToOnTouch local.ent local.vec local.offset	c,et	Teleport player touching executing entity to given entity or given vector with optional given offset
inpvs local.ent	r,we	Replaces: $player inpvs $actor
onTouchKill	c,ex	Kills player touching the executing Trigger entity

IMPLEMENTATION

The HZM Coop Mod needs to be implemented into each main level script. For each level there is only one main level script, it can be easily idendified, since it has the exact same name as the bsp-file of the level. For the implementation of the HZM Coop Mod only a few lines of code are nessesary, at the very start of the main function…

The main function of the HZM Coop Mod needs to be executed after the main variables are set and right before any kind of wait command or delaying function. If you need to specify spawnlocations in your script, make sure to put them before `thread coop_mod/main.scr::main`, because the HZM Coop Mod needs them to be set before the level starts.

```
1  main:{
2   level.script = "maps/e1l1.scr" //set current mapscript (for global script stuff)
3   level.coopPrevMap = "e1l1" //set previouse map (for mom coop voteing)
4   level.coopNextMap = "e1l3" //set next map (for mom coop voteing)
5   thread coop_mod/main.scr::main //start coop mod before any wait command, required
5
7   … //this represents non coop code
8
9   waitthread coop_mod/replace.scr::waitForPlayer //replacing:level waittill spawn
10
11  … //this represents non coop code
12 }end //end main
```

COMPAITBILITY

Many script commands in the level scripts are desinged to handle one player, because the game its Singleplayer nature. This makes many level scripts not work right during multiplayer, this gets the Missions stuck.
The HZM Coop Mod was designed to compensate and to make these levels compatible to multiplayer.
We tried to keep it as simple and efficent as possible, so that other scripters can use these features.

Making a level script Coop compatible is in most cases a simple search and replace procedure. How ever, some complex mission do still need advanced modifications to their level scripts.

In this Document we are only focusing on the simple search and replace procedure. How ever, if you need need our help with your project or need some more insights on the HZM Coop Mod scripts, then please join us on discord (see link at the very bottom).

REPLACEMENT

Replacing Singleplayer functions with Coop Mod replacement functions is done by just replacing the command call with a function call to the correct function in coop_mod/replace.scr, keeping all parameters as they are or removing them entierly.

This is how the original code looks:

```
1 self.turnto $player
```

This is how it would be done for coop, remember that almost all coop commands work also in singleplayer:

```
1 self exec coop_mod/replace.scr::turnto //chrissstrahl - coop compatible
```

The difference between these two lines of code is that the entity reference **$player** does not work in multiplayer as soon as there is more than one player on the server. The HZM Coop Mod replacement function handles this, by making **self** turn towards the closest player. The parameter **$player** is then no longer needed and therefor removed from the script.

Similar kinds of handling is done insdide almost all functions of the HZM Coop Mod replace.scr script! It is recommended that you configure your Codeeditor to highlight the keywords, so that you can find them quickly and replace/remove them.

Keyword	Description
$player	• As previously mentioned the entity $player will turn into a Array and any code using $player will stop working. This is because it expects a single entity, not a Array of entities. • You will need to lookup and use the corrospinding replacement function instead. • Sometimes you need to replace the $player entity. You can get the closest player or use $world if you just want to open a door.
lock	• You need to make sure that doors do not lock in/out other players, so you might need to disable later occurences of lock in the script during multiplayer.

GAMETYPES

There are several different gametypes, each gametype is defined by a number. Sometimes you need to have different code for the same event in the script. Use the Level-Varaiable `level.gametype` to retrive the current gametype. By using if-else you can make the code to be execute only in the gametype you desire, the Singleplayer is represented by 0, while Multiplayer starts with 1 for Deathmatch and continues from there.

```
someFunction:{
 if( level.gametype == 0 ){ //do this only in singleplayer
   $ai turnto $player
 }
 else{ //do this if NOT in singleplayer
   $ai turnto $otherAi
 }
}end
```

This code example above could also be used to lock a door after a event/sequence, but only in singleplayer, so it does not lock out/in players in Coop.

PLAYERS

The Mission scripts deal with a single player, that can eigther be dead (this usually also ends the mission) or alive and is stored in a static entity variable (`$player` or `$player[1]`).

This is much more complex in Multiplayer:
- If any player dies the mission should not fail
- From 0 to 8 players can be active and alive at the same time
- You might need to halt the mission if no player is on the server or active (this is especially important if dedicated server support is desired).
- The players can be at 8 different locations at the same time, triggering events
- The player that is closest or is using a specific object needs to be idendified and handled

In the HZM Coop Mod players can have these states:
- alive/dead (this is determined by the health of a player, 0 is dead)
- spectator/no-spectator (this is determined by the team of the player)
- active/inactive (this is managed by the HZM Coop Mod - alive, no-spectator and spawned with a weapon)
- As soon as there is more than one player on the server, the entity `$player` becomes a array, and all players are stored in it, starting with `$player[1]` and ending with `$player[8]`

Sometimes you need to handle all players at once, this seams complicated at first, but it is really simple, as you will gather from the example code, if you just take a moment to read it carefully.

In this example code the currently active weapon of every active players is printed into the console.
This code also works in singleplayer.

```
someFunction:{
 for(local.i = 1;local.i <= $player.size;local.i++){ //handle all players
  local.player =  $player[local.i] //get a single player from array
  if(local.player != NULL && local.player.health > 0){ //check if player exists and is alive
   if(level.gametype != 0){ //do this only in multi
    if(local.player.dmteam == "spectator" || local.player.flags["coop_isActive"] != 1){
     continue; //skip this player if spec or not active
    }
    //get name of active weapon and print to console
    println( exec coop_mod/main.scr::getActiveWeapon local.player )
   }
  }
 }
}end
```

In this example code all players are moved to their spawnlocation. This can be used to move all players to a new area. We want all players to be moved, regardless if they are dead, inactive or in spectator.
This code also works in singleplayer.

```
someFunction:{
 for(local.i = 1;local.i <= $player.size;local.i++){ //handle all players
  local.player =  $player[local.i] //get a single player from array
  if(local.player != NULL){ //check if player exists
   //moves player with given number to his spawnpoint
   thread playerPlaceAtSpawn local.i
  }
 }
}end
```

END
Thank you for reading, I hope I could give you some useful insight.
If you have any feedback, feel free to contact me.

Contact HaZardModding Group:
On Discord: https://discord.gg/vW7vskc (recommended)
On ModDB: https://www.moddb.com/messages/compose?to=Chrissstrahl