

# GUIDE TO THE MODIFIED SPECTATING SCRIPT

v1.33

Welcome

If you often find spectating script to be abused and would like to do something about it, you came to the right place. I've made some modifications to Kegetys' script to limit player spying.

## 1. Features

- disabled control over seagull,
- only 1<sup>st</sup>, 3<sup>rd</sup> and gunner view available,
- decreased slider range,
- bigger 'namelist' box and unit name field,
- spectating starts from the active unit,
- quick switch between 1<sup>st</sup> person view and gunner view,
- no hint messages,
- when leaving 1<sup>st</sup> person or gunner view, script restores last slider and boxes state; keeps spectating the same unit as in previous view,
- localization (Polish by Faguss, German by Spiderman, Finnish by Osku, Czech by Kalasnikov471, French by Nikiller),
- easy customization (#conditions).

## 2. Usage

Copy scripts to your mission folder. If you already have *Init.sqs* / *Description.ext* / *Stringtable.csv*, just copy & paste lines from files in this archive to their counterparts in mission directory.

These are conditions of the three sides. Player can spectate units from his side only:

```
DeathCamArray = [ ]  
? strona == east : DeathCamArray = [ ]  
? strona == resistance : DeathCamArray = [ ]  
? strona == civilian : DeathCamArray = [ ]
```

`DeathCamArray` is a list of spectated units. Fill bracket gaps with unit names. If this division doesn't suit you, delete conditions leaving only 1<sup>st</sup> line.

If player wants to leave the game, he has 5 seconds to press Escape key after quitting the script,. Otherwise, program will return to the script. Of course player can quit spectating again.

If all units are dead, program will leave spectating mode.

### Using script with different respawn types:

#### a) Instant (2) and Base (3):

Place a trigger and set it to "repeatedly". In the fields write:

```
Condition: !alive player  
onActivation: [ ] exec "onPlayerKilled.sqs"
```

Script is terminated each time player spawns.

#### b) Group (4):

Rename script to *OnPlayerRespawnAsSeagull.sqs*.

# 3. Enhancing

Feel free to modify this script. You'll find exemplary enhancements below:

## a) Spectating player's squad only

Move `DeathCamArray` definition to *init.sqs*:

```
DeathCamArray = units group player
```

## b) Specifying `DeathCamArray` for given player

In *OnPlayerKilled.sqs*, after conditions of the three sides, add line:

```
? player == unitname : DeathCamArray = [ENTER, UNITS, NAMES, HERE]
```

Where `unitname` is name of player's unit.

## c) Using `#conditions`

Code at this label is executed non-stop during spectating. You may change `DeathCamArray` in game. If there's more than one side of the conflict you can make, for example, west players spectate east after they have been eliminated (and vice versa).

Change done to the array must be performed only once so you need to define local variable `_b` that will control it.

```
DeathCamArray = array1
? strona==east : DeathCamArray = array2

_a="0"
_b=false
#conditions
? "alive _x" count DeathCamArray==0 && !_b : DeathCamArray=array1+array2; _b=true;
⇒ goto "begin"
? "alive _x" count DeathCamArray==0 || alive player : titleCut ["" ,"BLACK IN", 1];
⇒ _exit=true; goto "begin"
goto _a
```

Arrow indicates that this is NOT a new line.

To display units in current `DeathCamArray`, program has to jump to `#begin`.

## d) Revealing player's killer

A cutscene that shows player's killer:

Copy below line to *init.sqs*:

```
player addEventHandler ["killed", {morderca = _this select 1}]
```

It will provide information on the killer. Now, *OnPlayerKilled.sqs* should start executing these lines:

```
~1
_cam = "CAMERA" CamCreate getpos player
_cam CameraEffect ["EXTERNAL", "FRONT"]
_cam CamSetTarget player
_cam CamSetPos [(getpos player select 0)-3, (getpos player select 1)-3, 3]
_cam CamCommit 1
@camCommitted _cam
_cam CamSetPos [(getpos morderca select 0)-3, (getpos morderca select 1)-3, 3]
_cam CamSetTarget morderca
_cam CamCommit 2
```

## e) Alive player spectate

This can prove very useful while testing your mission.

**1)** *OnPlayerkilled.sqs* needs to be executed before player's death. For example, add to *init.sqs*:

```
player addaction ["Spectate", "OnPlayerKilled.sqs"]
```

**2)** Delete condition to leave the script when player is alive (#conditions):

```
|| alive player - delete this expression
```

**3)** To quit observing, add this to the beginning of the #seagull label:

```
#seagull
_exit=true; goto "begin"
```

## f) More examples?

See the script in my missions.