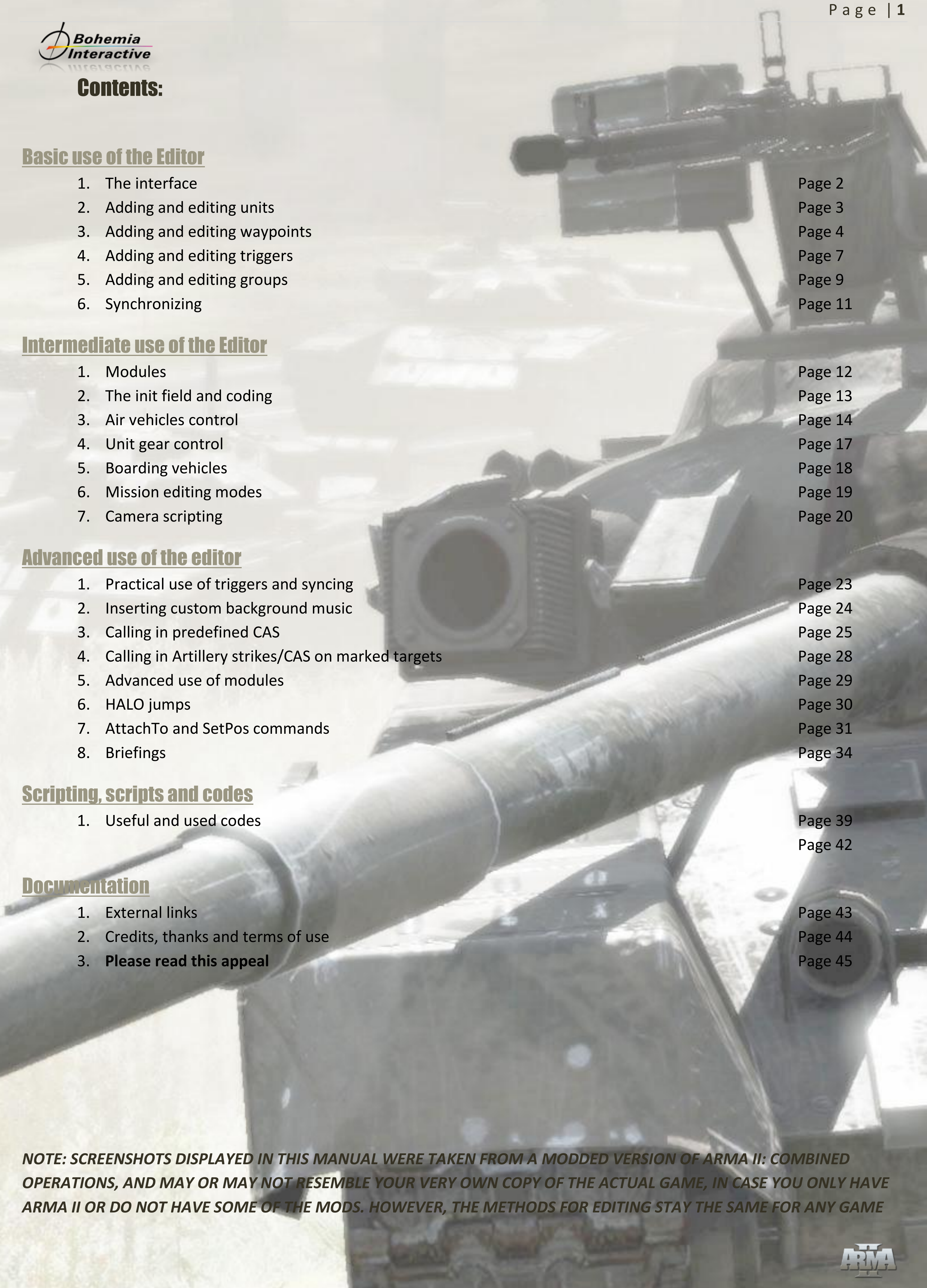
The complete guide from the simple to more advanced most commonly used features of the ArmA II 2D map editor

2D Editor

Must-Know User's Manual

Murdock 121

NOTE: THIS GUIDE IS BASED ONLY ON SINGLEPLAYER MISSION EDITING AND SCRIPTING. KNOWN AI AND SIMILAR BUGS ENCOUNTERED WITH DEDICATED SERVERS ARE NOTED HERE, HOWEVER NO GUIDELINES ON CREATING A MISSION FOR ONLINE GAMEPLAY, OR ADAPTING ONE ARE FOUND IN THIS MANUAL



Contents:

NOTE: SCREENSHOTS DISPLAYED IN THIS MANUAL WERE TAKEN FROM A MODDED VERSION OF ARMA II: COMBINED OPERATIONS, AND MAY OR MAY NOT RESEMBLE YOUR VERY OWN COPY OF THE ACTUAL GAME, IN CASE YOU ONLY HAVE ARMA II OR DO NOT HAVE SOME OF THE MODS. HOWEVER, THE METHODS FOR EDITING STAY THE SAME FOR ANY GAME

Basic use of the Editor

The interface

To enter the Editor, run ArmA II and select **Singleplayer > Editor**
From there you have the following options:

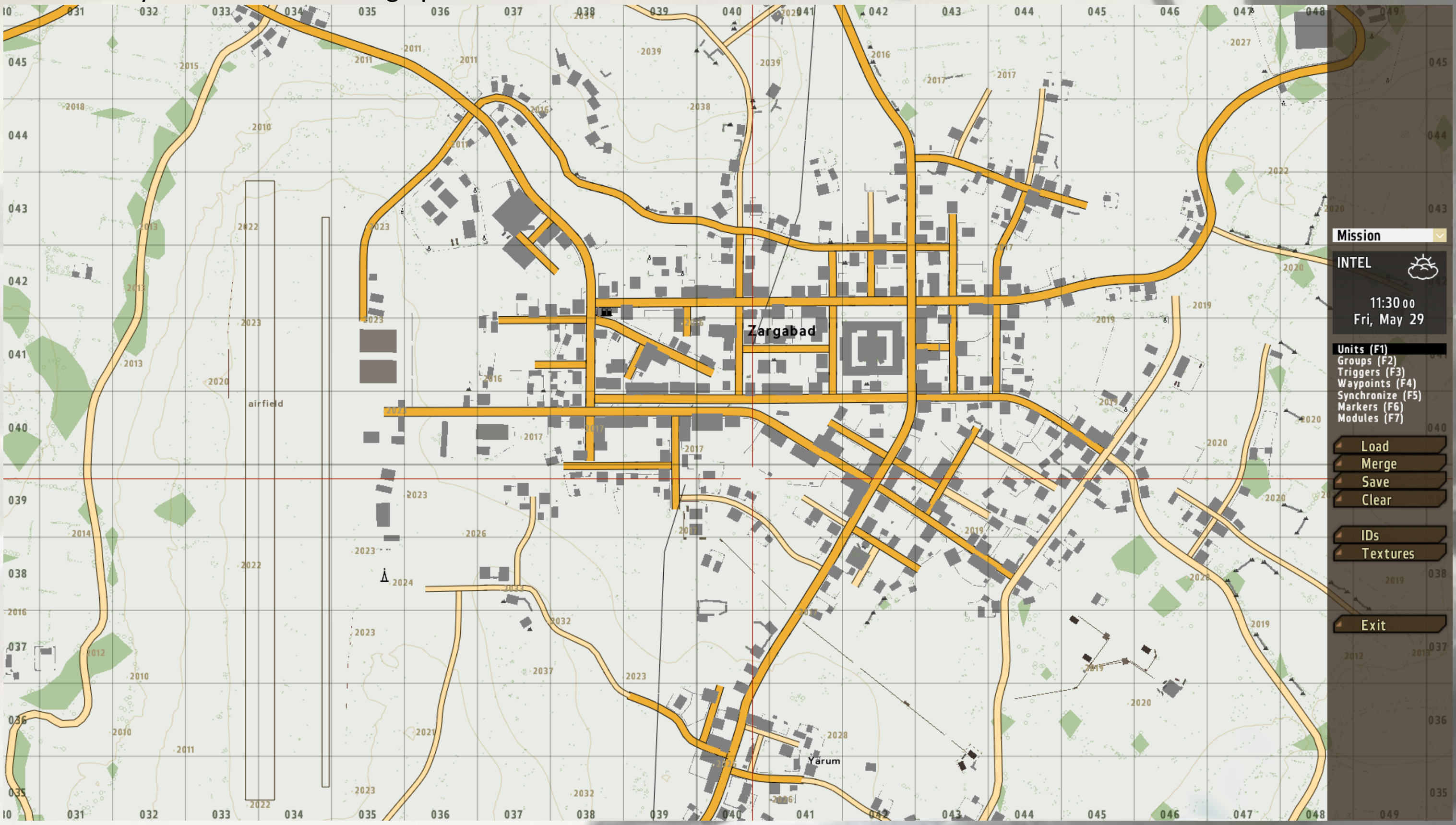


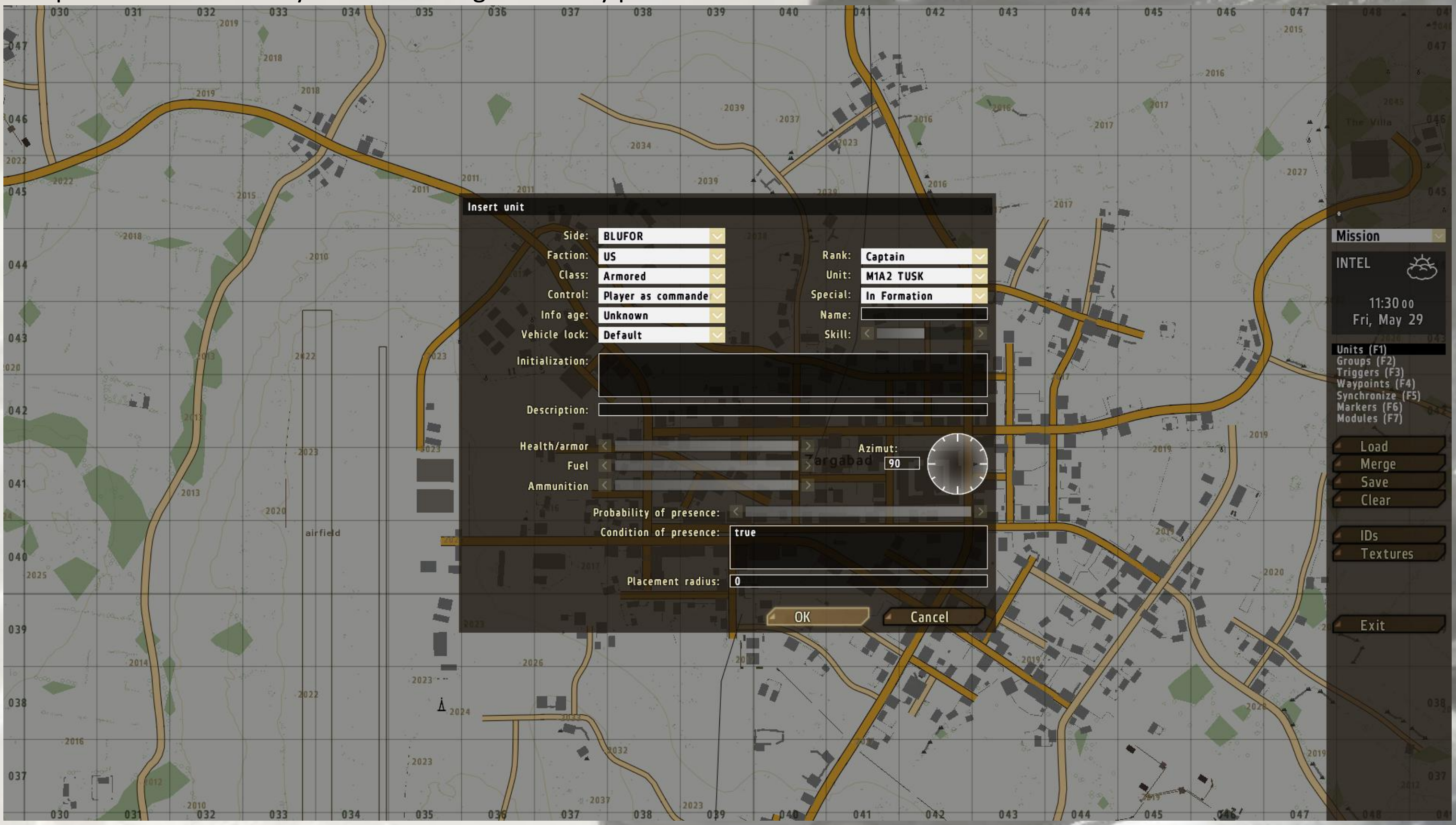
Image 1.0 – Editor view with Units (F1) selected

Mission field Here you select the mission editing modes – Mission, Intro, Outro – Win and Outro – Lose
Intel field Here you select the mission name, description, time of day, date, weather conditions etc.
Units (F1) Using the Units option you can place, move and edit units
Groups (F2) Using the Groups option you can place, move and edit groups and chains of command
Triggers (F3) With the Triggers option activated you place, move and edit triggers
Waypoints (F4) When selected, this option allows you to place, move and edit waypoints for units and groups already placed
Synchronize (F5) While the Synchronize option is activated, you can drag-and-drop to force simultaneous execution of orders
Markers (F6) Use this to place Markers on the map and enable yourself to create tasks and objectives (see Advanced: Briefing)
Modules (F7) Placing and editing Modules is achieved using the Modules option
Load, Merge, Save and Clear Using these buttons you can **Load** a previously saved map, **Merge** this mission with another one, **Save** it for later use or editing and Clear the entire map for a fresh start
NOTE: BY USING THE CLEAR OPTION THE EDITOR BEHAVES LIKE YOU BEGAN CREATING A COMPLETELY NEW MISSION
IDs and Textures These options enable you to show the IDs (numbers) and Textures of the ground.
Exit With the Exit option you exit the Editor (hopefully, we won't be needing it)
Map The **Map view** is what you'll use to edit the mission. There are four red lines indicating the current cursor position, wide roads are marked by yellowish-orange lines, and side roads are marked by bright yellow lines. Gray areas are houses, fences and other objects.
Controlling the Map view

  Zooming Use the **mousewheel** to **Zoom in** and **out** while in the **Map view**
  Moving To move around the map, **press-and-hold** the **right mouse button** while moving your mouse

Adding and editing units

Add units by selecting the **Units (F1)** option from the side menu (see image 1.0) and **double-clicking** on the map at the desired unit position. Edit them by double-clicking an already placed unit.


Insert unit

Side: BLUFOR
Faction: US
Class: Armored
Control: Player as commande
Info age: Unknown
Vehicle lock: Default
Rank: Captain
Unit: M1A2 TUSK
Special: In Formation
Name:
Skill:
Initialization:
Description:
Health/armor
Fuel
Ammunition
Azimut: 90
Probability of presence:
Condition of presence: true
Placement radius: 0

Image 1.1 – Placing a unit

In this example, we are placing a **BLUFOR United States Armored M1A2 Tusk** with the **Player as commander**, rank of **Captain** and **In Formation**. Explanation follows:

Side field Here you select a side for your unit – **BLUFOR**: The western forces – 'good guys' **OPFOR**: The eastern forces – 'bad guys' **INDEPENDENT**: Independent forces, can be neutral and can be allies to one side **EMPTY**: Empty vehicles and other useful objects

Faction field This is what you'll use to determine the army the unit belongs to

BLUFOR ARMIES USMC, CDF, US ***NOTE: DLC FACTIONS ARE NOT INCLUDED IN THE MANUAL***

OPFOR ARMIES Russia, Insurgents, Takistani Army and Militia

INDEPENTED FORCES Takistani locals, Guerillas, etc.

Class field Here you will select the class the unit belongs to – **Men**, **Air**, **Armored**, **Cars**, **Support**, **Static**, etc.

Control field Use this to determine whether the unit is controlled by the player or is playable at all

Rank field In order to establish a hierarchy in the team/squad, use the rank field

Example If a **Captain** dies, a **Lieutenant** will take his place and be **acting commander**

Unit field This is where you select the exact type of the unit. Here it is a **M1A2 Tusk** heavy armored vehicle (tank)

Special field Here you determine if the unit is **In Formation** (this automatically places it on the groud), **Flying** (air only) or **None** (no special functions available, the unit is placed exactly like you placed it)

Initialization field This field is used to enter custom parameters that are activated as soon as the unit is spawned

NOTE: THE INITIALIZATION FIELD WILL BE REFFERED TO AS 'INIT FIELD' IN THE MANUAL

Name field Very important field if the unit will be used in scripts or codes, it sets the name of the unit

Skill field Sets the skill of the unit. Values from 0.2 (stupid, blind and retarded) to 1.0 (war veterans)

Health/Armor, Fuel and Ammunition sliders These are used to determine the damage to the unit, amount of fuel and ammunition it has at the start of the mission

Azimut This is used to determine the facing direction of your unit. It is expressed in degrees (**0° = North**)

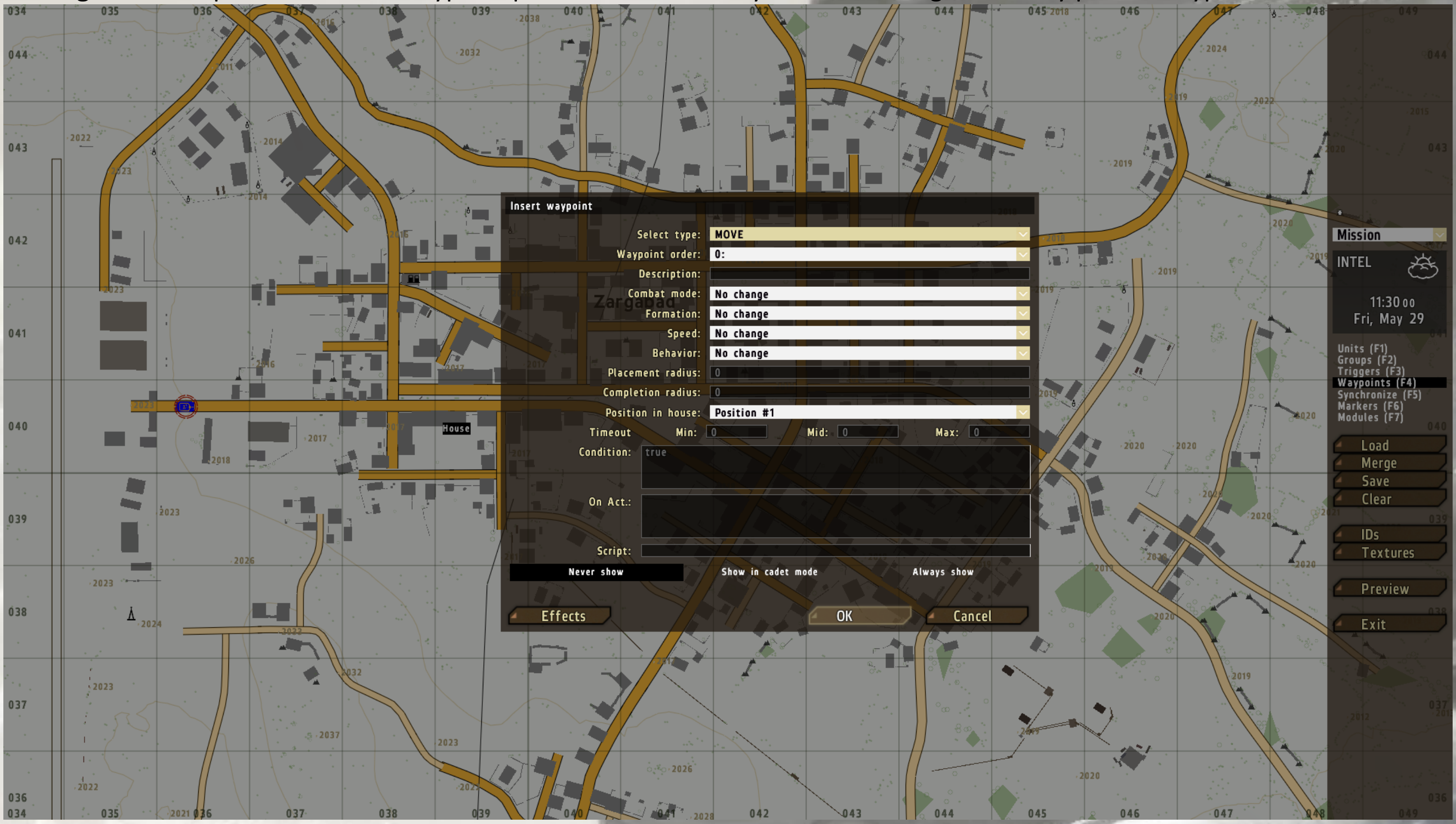
Units can also be **rotated** by left clicking on them, and holding shift while dragging with the left mouse button

Condition of presence Here you'll type in the code to determine the conditions at which the unit spawns

Placement radius This is the radius of the circle within the unit can spawn, and is randomly spawned inside it

Adding and editing waypoints

Add waypoints by selecting the **Waypoints (F4)** option from the side menu, **clicking** on an already palced unit and **double-clicking** on the map at the desired waypoint position. Edit them by double-clicking an already placed waypoint.


Insert waypoint

Select type:	MOVE
Waypoint order:	0:
Description:	
Combat mode:	No change
Formation:	No change
Speed:	No change
Behavior:	No change
Placement radius:	0
Completion radius:	0
Position in house:	Position #1
Timeout	Min: 0 Mid: 0 Max: 0
Condition:	true
On Act.:	
Script:	

[✓ Never show] Show in cadet mode Always show

Image 1.2 – Placing a Move waypoint. Note that in the background you can see I have selected the tank from the last picture

Select type field This is where you select the type of the waypoint. A list of waypoint types and functions is on the next page, and also in your game manual

Waypoint order Select the order in which the waypoint is executed.

Example A waypoint with a 0 order will be executed first, waypoint 1 will be executed second, waypoint 2 third, etc.

Description A description of the waypoint. It will be shown under the waypoint marker in-game (if the difficulty allows so)

Combat mode The ROE (rules of engagement) for the unit/s.

Example: **Never fire** will force the units not to open at all. **Hold fire** will force them to sustain from firing unless fired upon. **Open fire** will allow them to shoot. **Engage at will** allows units to chase their targets, without compromising their final objective.

Formation Determines the formation in which the group travels. A list of formation types is on the sixth page

Speed Selection of the speed the units will travel at.

Example: **Limited speed** units will move slowly (infantry will walk) to their objectives. **Normal speed** units will go normally (infantry will run). **Full speed** units will race to their objectives.

Behavior Choose the amount of awareness for your units. A list of behaviors is on the sixth page.

Placement radius field The waypoint is placed somewhere within the PLC radius field, randomly.

Completion radius field The radius of the waypoint's effect and completion.

Example For a **Move** type waypoint, as soon as the unit enters the desired radius, it is considered completed, although the center has not been reached.

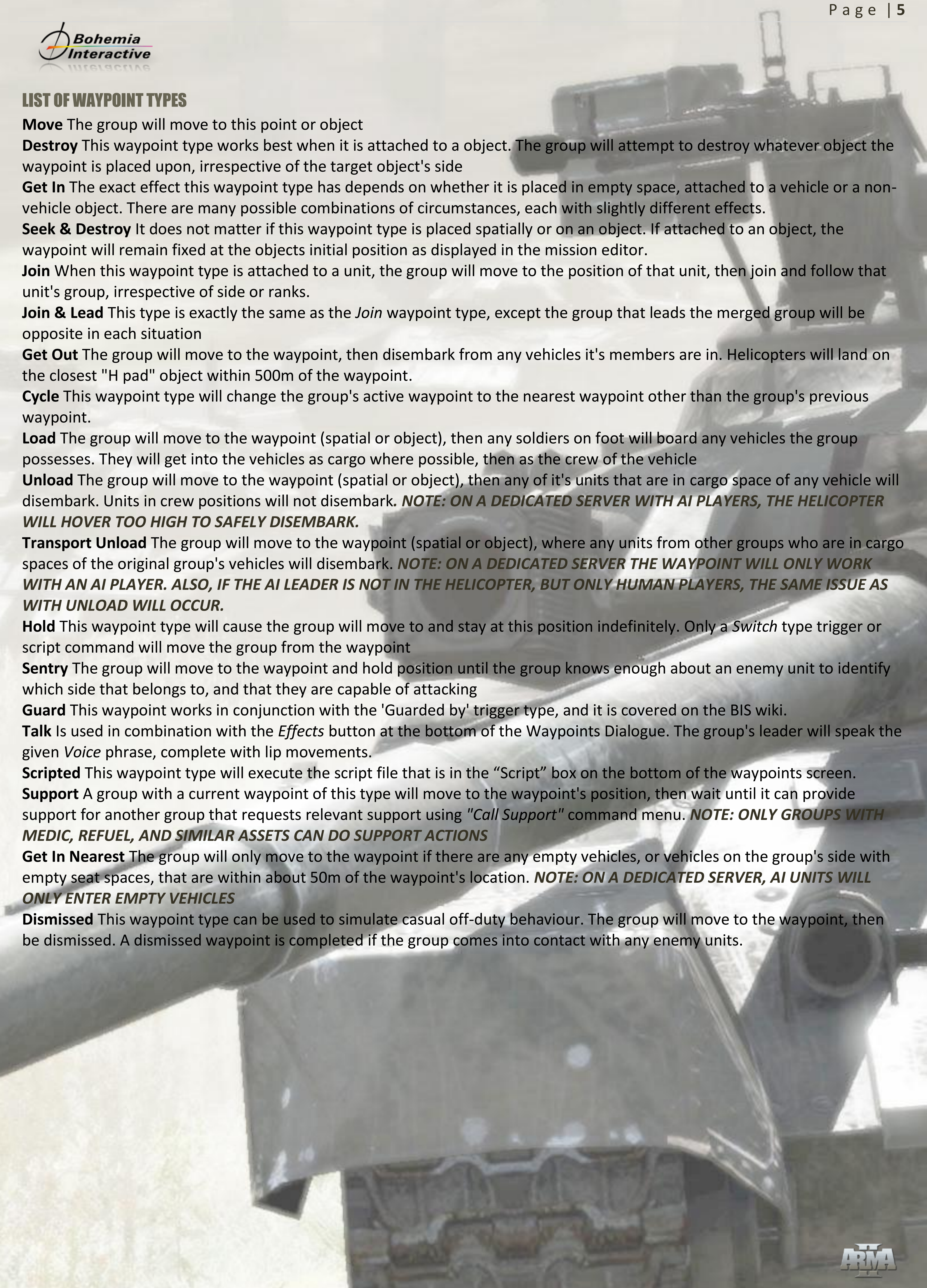
Position in house Lets you pick the position the units (only infantry!) will take inside a house, if the waypoint is placed inside one. ***NOTE: THE FEATURE IS MOSTLY BUGGY, AND IT TAKES SCRIPTING AND CODING TO AVOID NO-CLIP EFFECTS ON MULTIPLE UNITS WHEN USING THE POSITION IN HOUSE COMMAND***

Timeout The amount of time that has to pass after completion to send the units to their next waypoint.

On Act.(ivation) A piece of code that decides what happens on completion of the waypoint objective.

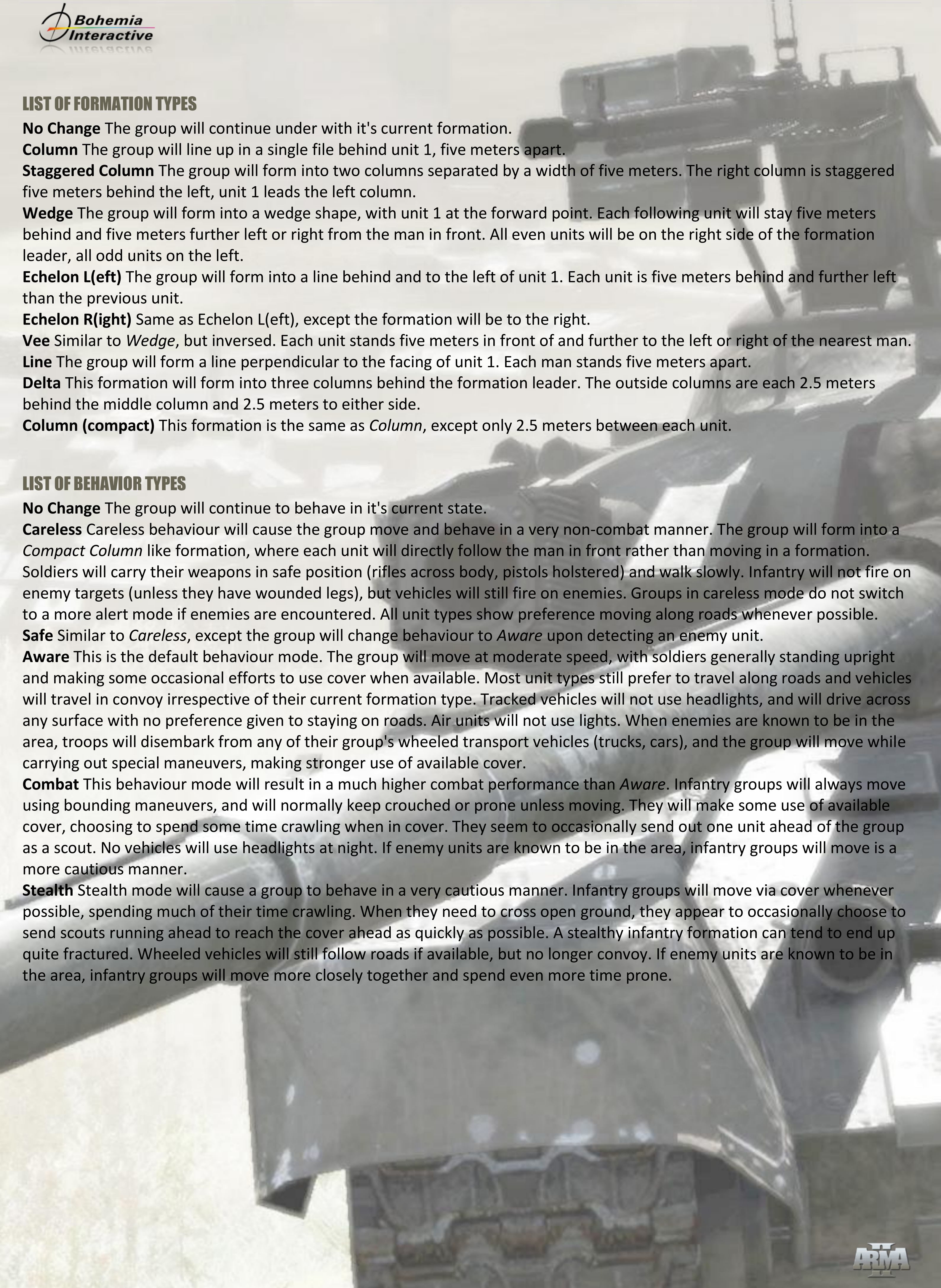
Script Only available for Scripted-type waypoints. Allows you to select a script to execute. More information on the BIS wiki.

Effects Allows you to select an effect that occurs upon completion. You can select a **Track** to play, **Voice** (for use with the **Talk**-type waypoints) and **Anonymus** sounds, etc.



LIST OF WAYPOINT TYPES

Move The group will move to this point or object
Destroy This waypoint type works best when it is attached to a object. The group will attempt to destroy whatever object the waypoint is placed upon, irrespective of the target object's side
Get In The exact effect this waypoint type has depends on whether it is placed in empty space, attached to a vehicle or a non-vehicle object. There are many possible combinations of circumstances, each with slightly different effects.
Seek & Destroy It does not matter if this waypoint type is placed spatially or on an object. If attached to an object, the waypoint will remain fixed at the objects initial position as displayed in the mission editor.
Join When this waypoint type is attached to a unit, the group will move to the position of that unit, then join and follow that unit's group, irrespective of side or ranks.
Join & Lead This type is exactly the same as the *Join* waypoint type, except the group that leads the merged group will be opposite in each situation
Get Out The group will move to the waypoint, then disembark from any vehicles it's members are in. Helicopters will land on the closest "H pad" object within 500m of the waypoint.
Cycle This waypoint type will change the group's active waypoint to the nearest waypoint other than the group's previous waypoint.
Load The group will move to the waypoint (spatial or object), then any soldiers on foot will board any vehicles the group possesses. They will get into the vehicles as cargo where possible, then as the crew of the vehicle
Unload The group will move to the waypoint (spatial or object), then any of it's units that are in cargo space of any vehicle will disembark. Units in crew positions will not disembark. ***NOTE: ON A DEDICATED SERVER WITH AI PLAYERS, THE HELICOPTER WILL HOVER TOO HIGH TO SAFELY DISEMBARK.***
Transport Unload The group will move to the waypoint (spatial or object), where any units from other groups who are in cargo spaces of the original group's vehicles will disembark. ***NOTE: ON A DEDICATED SERVER THE WAYPOINT WILL ONLY WORK WITH AN AI PLAYER. ALSO, IF THE AI LEADER IS NOT IN THE HELICOPTER, BUT ONLY HUMAN PLAYERS, THE SAME ISSUE AS WITH UNLOAD WILL OCCUR.***
Hold This waypoint type will cause the group will move to and stay at this position indefinitely. Only a *Switch* type trigger or script command will move the group from the waypoint
Sentry The group will move to the waypoint and hold position until the group knows enough about an enemy unit to identify which side that belongs to, and that they are capable of attacking
Guard This waypoint works in conjunction with the 'Guarded by' trigger type, and it is covered on the BIS wiki.
Talk Is used in combination with the *Effects* button at the bottom of the Waypoints Dialogue. The group's leader will speak the given *Voice* phrase, complete with lip movements.
Scripted This waypoint type will execute the script file that is in the "Script" box on the bottom of the waypoints screen.
Support A group with a current waypoint of this type will move to the waypoint's position, then wait until it can provide support for another group that requests relevant support using *"Call Support"* command menu. ***NOTE: ONLY GROUPS WITH MEDIC, REFUEL, AND SIMILAR ASSETS CAN DO SUPPORT ACTIONS***
Get In Nearest The group will only move to the waypoint if there are any empty vehicles, or vehicles on the group's side with empty seat spaces, that are within about 50m of the waypoint's location. ***NOTE: ON A DEDICATED SERVER, AI UNITS WILL ONLY ENTER EMPTY VEHICLES***
Dismissed This waypoint type can be used to simulate casual off-duty behaviour. The group will move to the waypoint, then be dismissed. A dismissed waypoint is completed if the group comes into contact with any enemy units.



LIST OF FORMATION TYPES

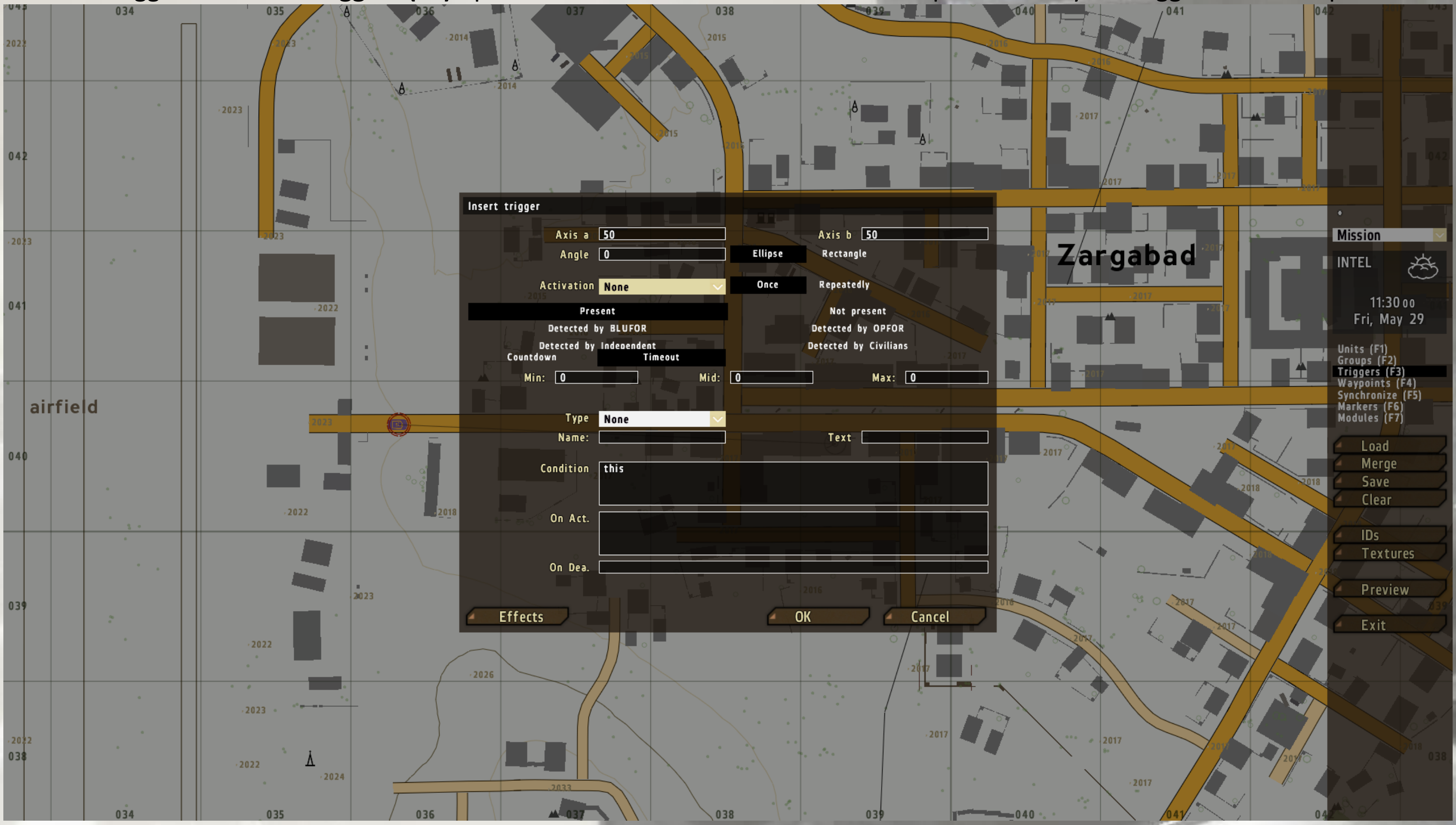
No Change The group will continue under with it's current formation.
Column The group will line up in a single file behind unit 1, five meters apart.
Staggered Column The group will form into two columns separated by a width of five meters. The right column is staggered five meters behind the left, unit 1 leads the left column.
Wedge The group will form into a wedge shape, with unit 1 at the forward point. Each following unit will stay five meters behind and five meters further left or right from the man in front. All even units will be on the right side of the formation leader, all odd units on the left.
Echelon L(eft) The group will form into a line behind and to the left of unit 1. Each unit is five meters behind and further left than the previous unit.
Echelon R(ight) Same as Echelon L(eft), except the formation will be to the right.
Vee Similar to *Wedge*, but inversed. Each unit stands five meters in front of and further to the left or right of the nearest man.
Line The group will form a line perpendicular to the facing of unit 1. Each man stands five meters apart.
Delta This formation will form into three columns behind the formation leader. The outside columns are each 2.5 meters behind the middle column and 2.5 meters to either side.
Column (compact) This formation is the same as *Column*, except only 2.5 meters between each unit.

LIST OF BEHAVIOR TYPES

No Change The group will continue to behave in it's current state.
Careless Careless behaviour will cause the group move and behave in a very non-combat manner. The group will form into a *Compact Column* like formation, where each unit will directly follow the man in front rather than moving in a formation. Soldiers will carry their weapons in safe position (rifles across body, pistols holstered) and walk slowly. Infantry will not fire on enemy targets (unless they have wounded legs), but vehicles will still fire on enemies. Groups in careless mode do not switch to a more alert mode if enemies are encountered. All unit types show preference moving along roads whenever possible.
Safe Similar to *Careless*, except the group will change behaviour to *Aware* upon detecting an enemy unit.
Aware This is the default behaviour mode. The group will move at moderate speed, with soldiers generally standing upright and making some occasional efforts to use cover when available. Most unit types still prefer to travel along roads and vehicles will travel in convoy irrespective of their current formation type. Tracked vehicles will not use headlights, and will drive across any surface with no preference given to staying on roads. Air units will not use lights. When enemies are known to be in the area, troops will disembark from any of their group's wheeled transport vehicles (trucks, cars), and the group will move while carrying out special maneuvers, making stronger use of available cover.
Combat This behaviour mode will result in a much higher combat performance than *Aware*. Infantry groups will always move using bounding maneuvers, and will normally keep crouched or prone unless moving. They will make some use of available cover, choosing to spend some time crawling when in cover. They seem to occasionally send out one unit ahead of the group as a scout. No vehicles will use headlights at night. If enemy units are known to be in the area, infantry groups will move is a more cautious manner.
Stealth Stealth mode will cause a group to behave in a very cautious manner. Infantry groups will move via cover whenever possible, spending much of their time crawling. When they need to cross open ground, they appear to occasionally choose to send scouts running ahead to reach the cover ahead as quickly as possible. A stealthy infantry formation can tend to end up quite fractured. Wheeled vehicles will still follow roads if available, but no longer convoy. If enemy units are known to be in the area, infantry groups will move more closely together and spend even more time prone.

Adding and editing triggers

To add a trigger select the **Triggers (F3)** option and then **double-click** the desired position for your trigger on the map view.



Insert trigger

Axis a 50 Axis b 50
Angle 0 Ellipse Rectangle
Activation None Once Repeatedly
Present Not present
Detected by BLUFOR Detected by OPFOR
Detected by Independent Detected by Civilians
Countdown Timeout
Min: 0 Mid: 0 Max: 0
Type None
Name: Text
Condition this
On Act.
On Dea.

Image 1.3 – Adding a trigger. Note the map is now quite zoomed if compared with the first interface image

Axis a and Axis b fields Settings for the width and length of the trigger. If you want a circle, make them equal and select **Ellipse**
Angle The angle of the trigger. Works similar as **Azimut** in unit selection.
Ellipse/Rectangle and Once/Repeatedly Select the shape of the trigger, and how many times can it be triggered
Activation This is the condition for the activation. A list of activation types is on the next page
Present/Not present The absolute setting for the condition.

> **Present** = Is and
> **Not present** = Is Not
> **Example**: A trigger with the condition **Activation BLUFOR** and condition **Present** will go off as soon as **BLUFOR** forces enter it's area of effect

Detected by BLUFOR/OPFOR/Independent/Civilian The relative setting for the condition.

> **Example**: A trigger with the condition **Activation BLUFOR** and condition **Detected by OPFOR** will go off when **OPFOR** forces think (or spot) **BLUFOR** entered it's area of effect

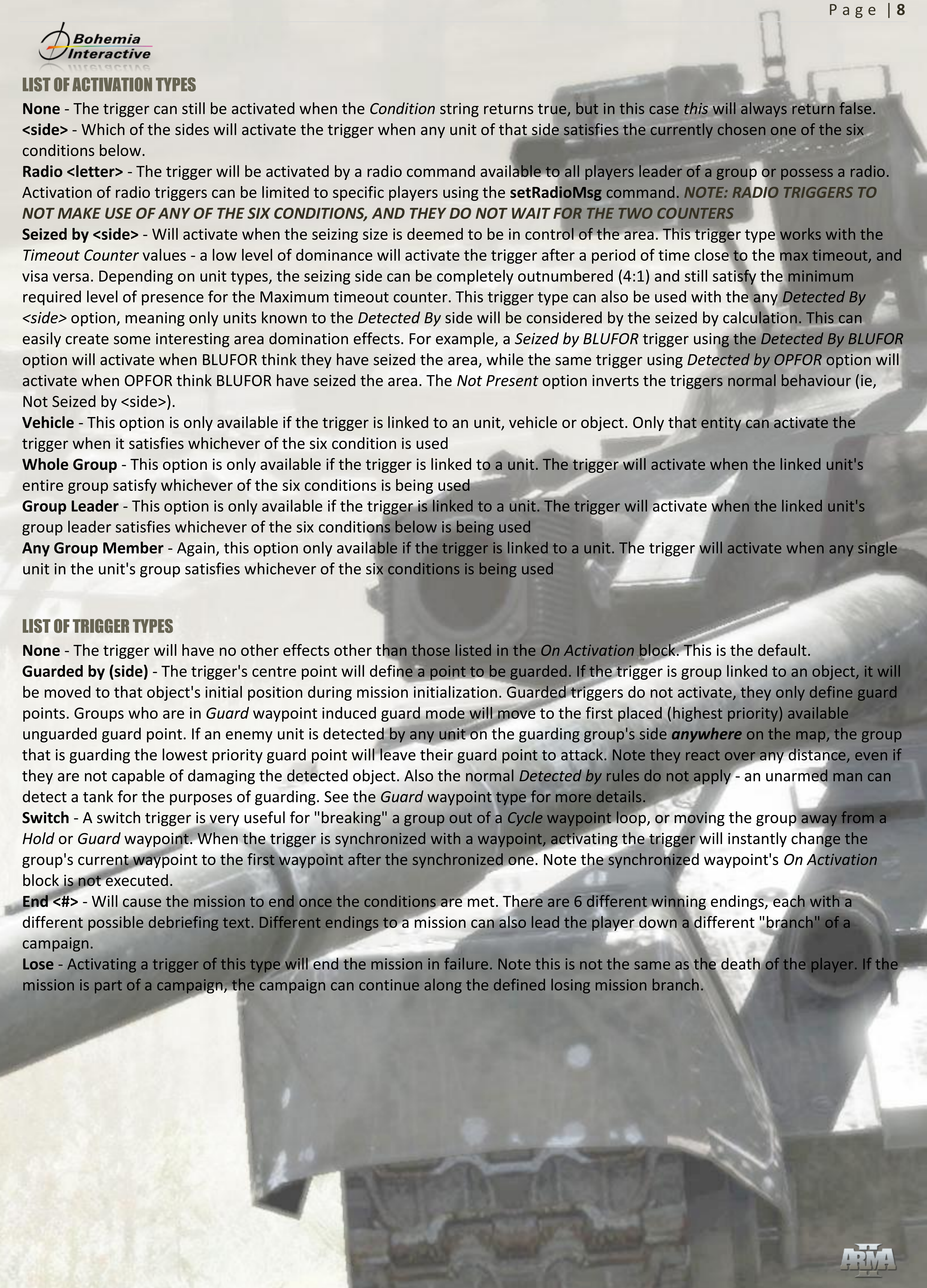
Countdown/Timeout Select the trigger's type of activation.

> **Countdown** Once the trigger's conditions are met the trigger will wait the specified amount of time before activating.
> **Timeout** The trigger's conditions must be met for the duration of the specified amount of time before the trigger activates

Type Select the type of the trigger here. A list of all types is on the following page
Name and Text fields The **Name field** is used in the same way as with units. The **Text field** is where you type the text displayed

> **Example**: In case of a **Radio Alpha** trigger called **CAS**, when in **Map view** in-game, you'll see a radio that says **"CAS"**

Condition field You can also set the condition for the trigger using codes, which you will type in here
On Act.(ivation) Same as with **Waypoints** (*see **Waypoints (F4)** on page 4*)
On Dea.(ctivation) Same as **On Act.**, but only when the trigger is **Deactivated**
Effects Same as with **Waypoints** (*see **Waypoints (F4)** on page 4*)



LIST OF ACTIVATION TYPES

None - The trigger can still be activated when the *Condition* string returns true, but in this case *this* will always return false.
<side> - Which of the sides will activate the trigger when any unit of that side satisfies the currently chosen one of the six conditions below.
Radio <letter> - The trigger will be activated by a radio command available to all players leader of a group or possess a radio. Activation of radio triggers can be limited to specific players using the **setRadioMsg** command. ***NOTE: RADIO TRIGGERS TO NOT MAKE USE OF ANY OF THE SIX CONDITIONS, AND THEY DO NOT WAIT FOR THE TWO COUNTERS***
Seized by <side> - Will activate when the seizing size is deemed to be in control of the area. This trigger type works with the *Timeout Counter* values - a low level of dominance will activate the trigger after a period of time close to the max timeout, and visa versa. Depending on unit types, the seizing side can be completely outnumbered (4:1) and still satisfy the minimum required level of presence for the Maximum timeout counter. This trigger type can also be used with the any *Detected By <side>* option, meaning only units known to the *Detected By* side will be considered by the seized by calculation. This can easily create some interesting area domination effects. For example, a *Seized by BLUFOR* trigger using the *Detected By BLUFOR* option will activate when BLUFOR think they have seized the area, while the same trigger using *Detected by OPFOR* option will activate when OPFOR think BLUFOR have seized the area. The *Not Present* option inverts the triggers normal behaviour (ie, Not Seized by <side>).
Vehicle - This option is only available if the trigger is linked to an unit, vehicle or object. Only that entity can activate the trigger when it satisfies whichever of the six condition is used
Whole Group - This option is only available if the trigger is linked to a unit. The trigger will activate when the linked unit's entire group satisfy whichever of the six conditions is being used
Group Leader - This option is only available if the trigger is linked to a unit. The trigger will activate when the linked unit's group leader satisfies whichever of the six conditions below is being used
Any Group Member - Again, this option only available if the trigger is linked to a unit. The trigger will activate when any single unit in the unit's group satisfies whichever of the six conditions is being used

LIST OF TRIGGER TYPES

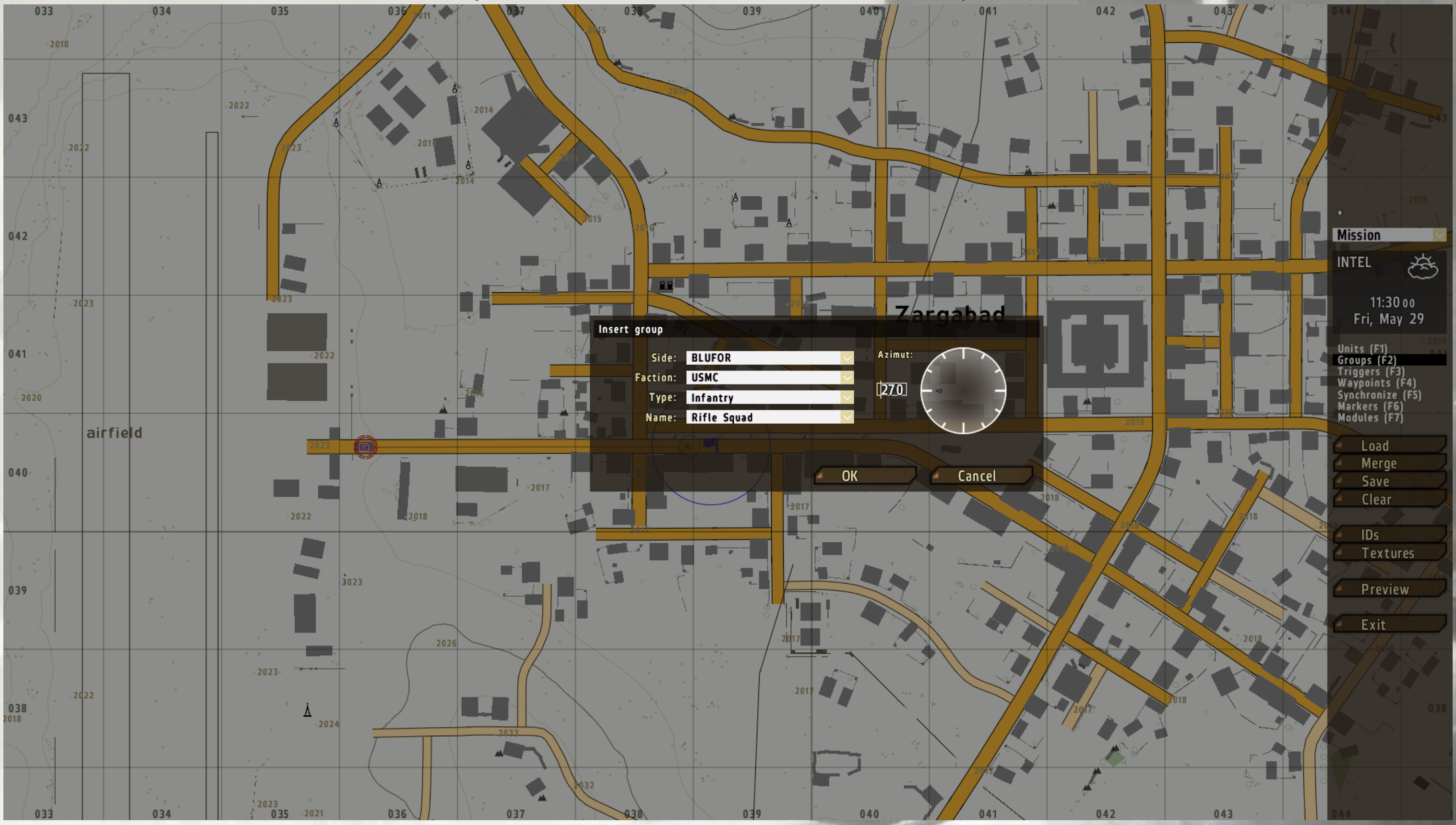
None - The trigger will have no other effects other than those listed in the *On Activation* block. This is the default.
Guarded by (side) - The trigger's centre point will define a point to be guarded. If the trigger is group linked to an object, it will be moved to that object's initial position during mission initialization. Guarded triggers do not activate, they only define guard points. Groups who are in *Guard* waypoint induced guard mode will move to the first placed (highest priority) available unguarded guard point. If an enemy unit is detected by any unit on the guarding group's side ***anywhere*** on the map, the group that is guarding the lowest priority guard point will leave their guard point to attack. Note they react over any distance, even if they are not capable of damaging the detected object. Also the normal *Detected by* rules do not apply - an unarmed man can detect a tank for the purposes of guarding. See the *Guard* waypoint type for more details.
Switch - A switch trigger is very useful for "breaking" a group out of a *Cycle* waypoint loop, or moving the group away from a *Hold* or *Guard* waypoint. When the trigger is synchronized with a waypoint, activating the trigger will instantly change the group's current waypoint to the first waypoint after the synchronized one. Note the synchronized waypoint's *On Activation* block is not executed.
End <#> - Will cause the mission to end once the conditions are met. There are 6 different winning endings, each with a different possible debriefing text. Different endings to a mission can also lead the player down a different "branch" of a campaign.
Lose - Activating a trigger of this type will end the mission in failure. Note this is not the same as the death of the player. If the mission is part of a campaign, the campaign can continue along the defined losing mission branch.

Adding and editing groups

Add groups by selecting the **Groups (F2)** option from the side menu and **double-clicking** on the desired group position.
This tool can also be used to set a hierarchy irrespective of ranks using **press-and-hold** the **left mouse button**

Setting hierarchies using the Groups (F2) tool: Follower → Follows → Leader

NOTE: IN CASE OF SETTING HIERARCHIES, WHEN YOU CHANGE THE UNIT'S LEADER, IT LOSES IT PREVIOUS WAYPOINTS



Insert group

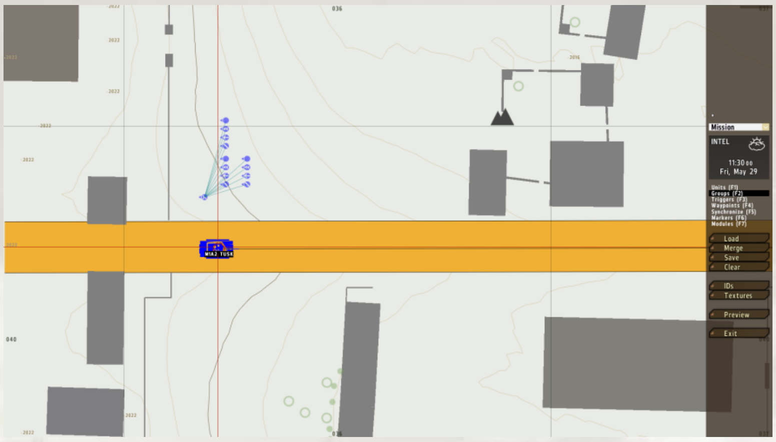
Side: BLUFOR
Faction: USMC
Type: Infantry
Name: Rifle Squad
Azimut: 270

Image 1.4 – Placing a group. I have not erased either my tank, waypoint or trigger in order to show how their symbols look like

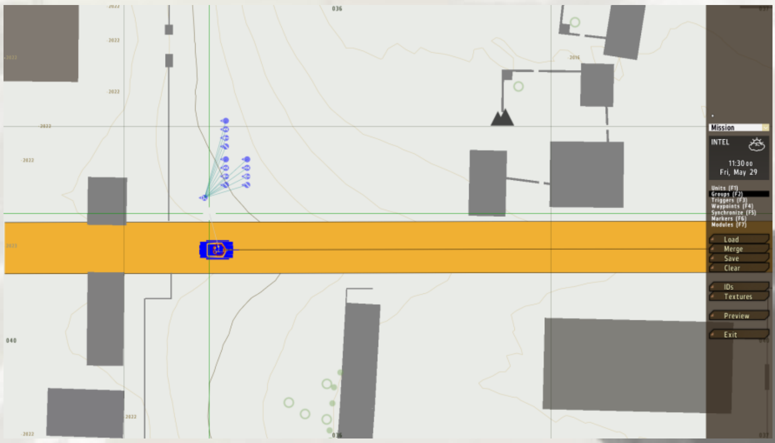
Side field Same as with placing **Units** (*see **Units**, page 3*)
Faction field Same as with placing **Units** (*see **Units**, page 3*)
Type Select the type of your group

- **Infantry** Men
- **Motorized Infantry** Men and two cars
- **Mechanized Infantry** Men and an APC
- **Armored** Tanks
- **Air** Helicopters, UAVs or aircraft

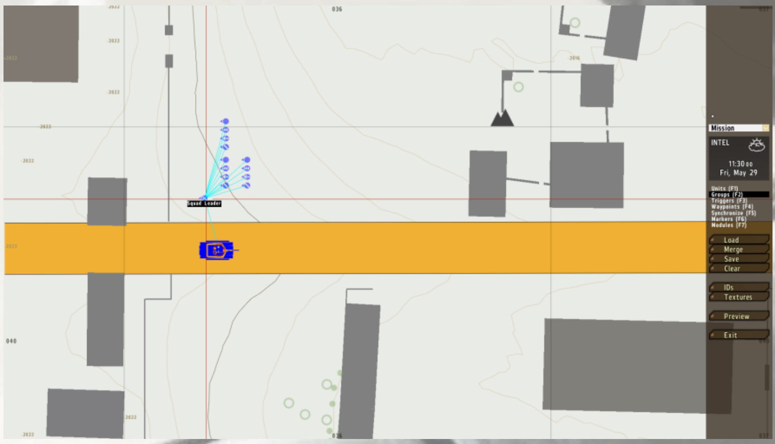
Name Defines the specific group and units to be placed in it
Azimut Same as with placing **Units** (*see **Units**, page 3*)



Selecting the M1A2 Tusk Tank that will follow the Leader



Dragging the marker to the Leader

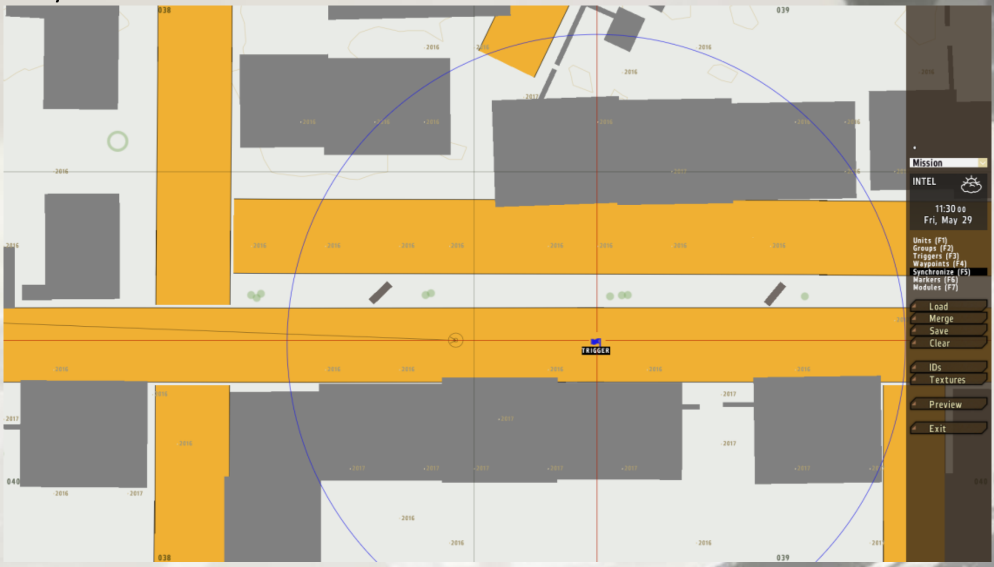


Dropping the marker at the Leader's position

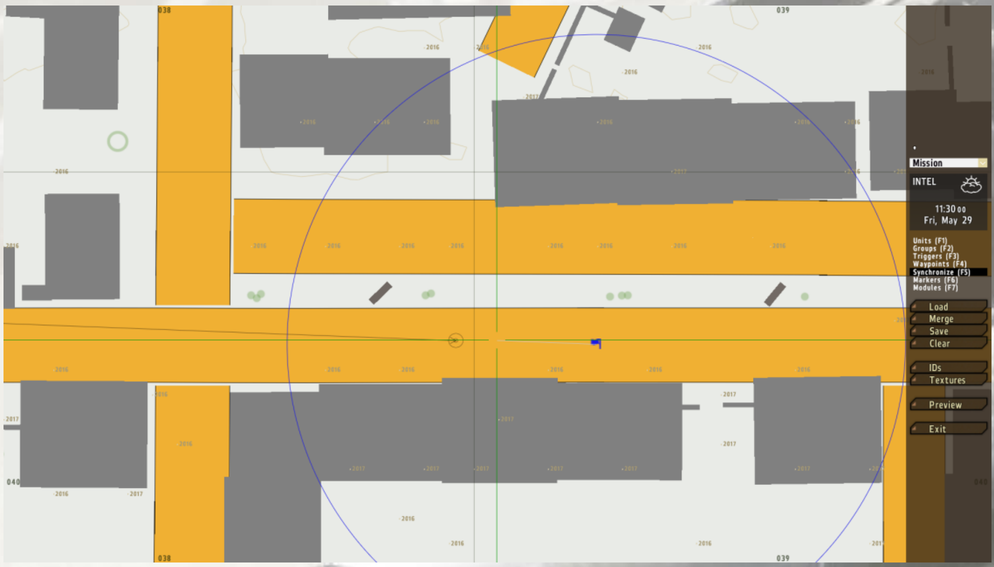
Image 1.41 – Setting a hierarchy using the Groups tool. Note in the last slide there is a blue line attached to the tank and group Note also that in the first and second slides, the tank has a black line to the right, marking it's current waypoint

Synchronizing

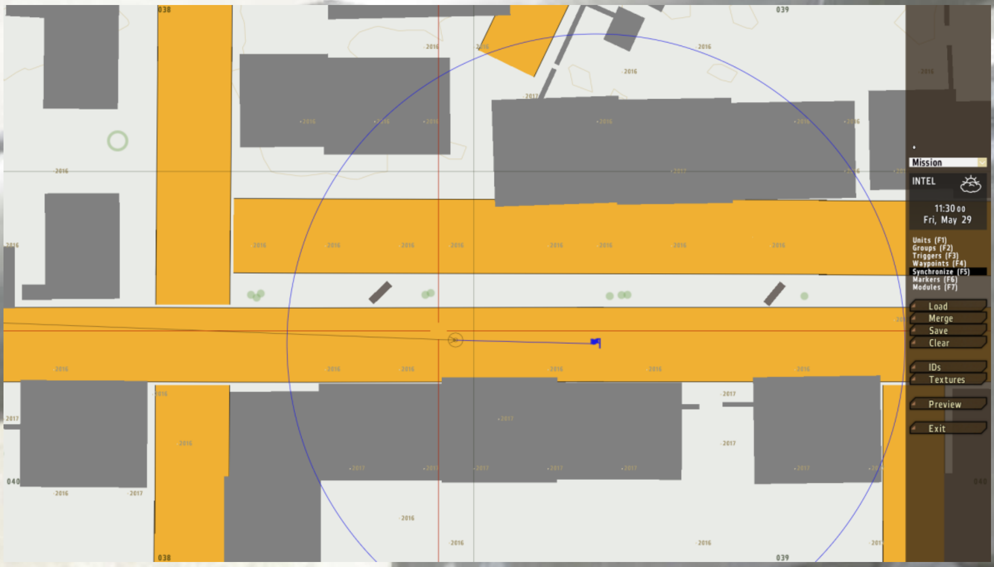
Synchronize waypoints with other waypoints and triggers with other triggers by selecting the **Synchronize (F5)** option from the side menu, selecting the waypoint/trigger to synchronize and **drag-and-dropping** from it onto the desired waypoint/trigger to be synchronized



Selecting the trigger to sync



Dragging the marker to the waypoint



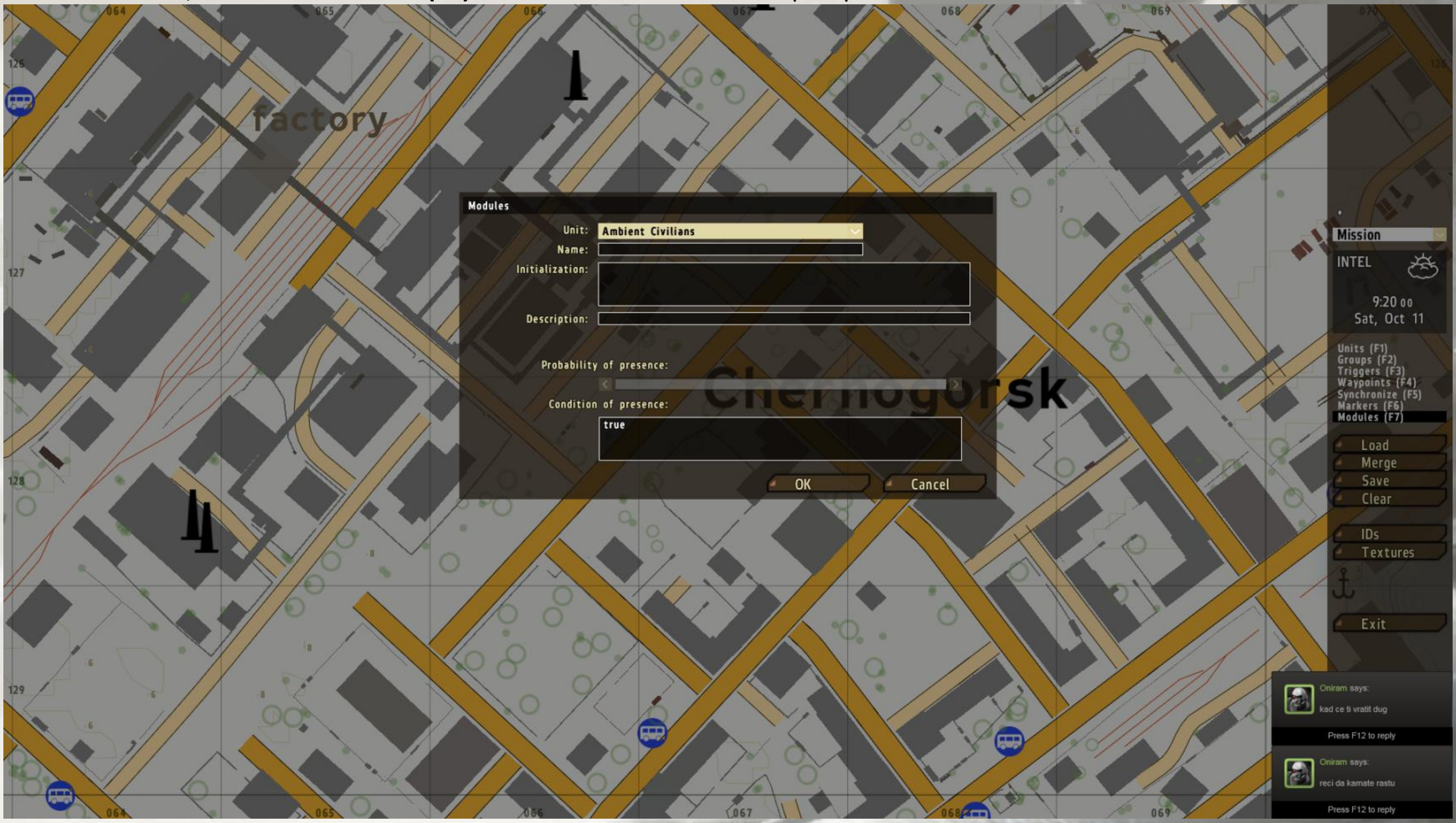
The completed synchronization

Image 1.5 – Synchronizing a trigger with a waypoint. Note in the last slide there is a blue line connecting the trigger and waypoint to mark the synchronization

Intermediate use of the Editor

Modules

To add modules, select the **Modules (F7)** and **double-click** on the map to place the module



Modules

Unit: Ambient Civilians

Name:

Initialization:

Description:

Probability of presence:

Condition of presence: true

Oniram says: kad ce ti vratit dug

Press F12 to reply

Oniram says: reci da kamate rastu

Press F12 to reply

Image 2.0 – Placing an Ambient Civilians Module

Unit Use this to select the type of the module. A list of modules is on the next page
Name field Here type in the name to be used in codes and scripts
Init This is where you'll insert codes to be executed on the mission beginning
Description A description for the module
Condition of presence Same as with placing **Units** (*see **Units**, page 3*)

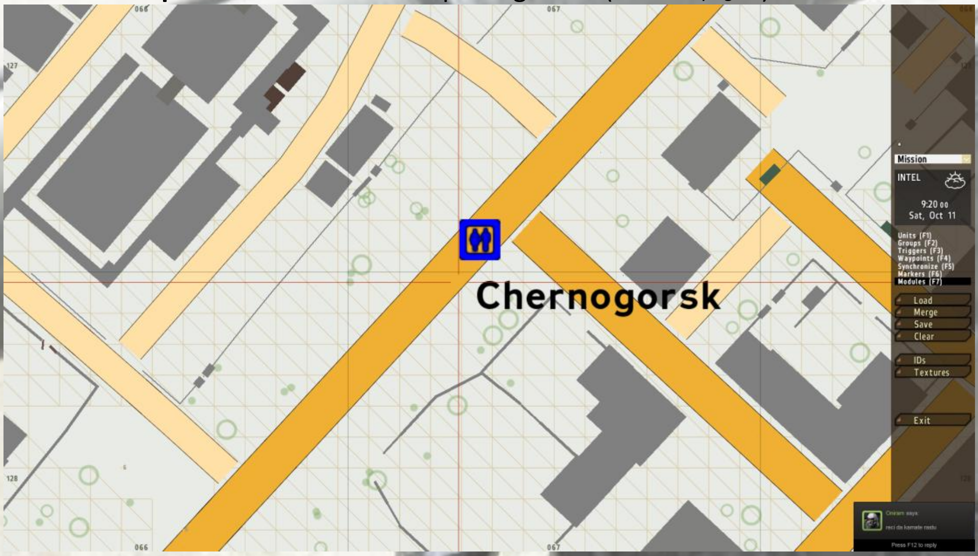
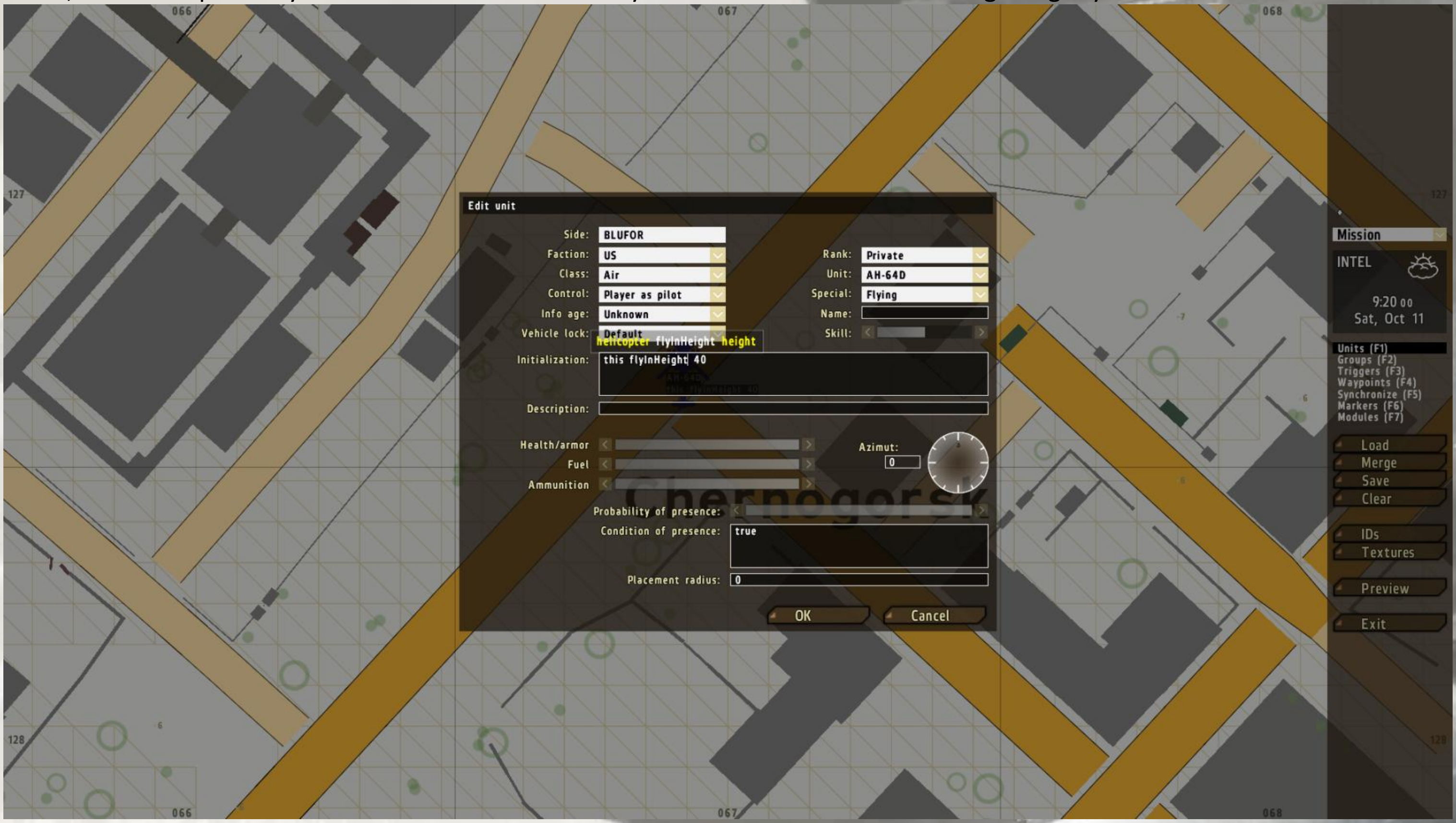


Image 2.01 – A placed Ambient Civilians Module

NOTE: SOME MODULES HAVE TO BE SYNCHRONIZED (F5) IN ORDER TO WORK PROPERLY

The init field and coding / Air vehicles control – height

The **Initialization** field and **coding** is probably something you'll be using **a lot** with creating missions. You will find it useful at all times, and there probably won't be a mission in which you won't use it. Here's an image to get you started:



Edit unit

Side: BLUFOR
Faction: US
Class: Air
Control: Player as pilot
Info age: Unknown
Vehicle lock: Default
Rank: Private
Unit: AH-64D
Special: Flying
Name:
Skill:
Initialization: this flyInHeight 40
helicopter flyInHeight height
Description:
Health/armor
Fuel
Ammunition
Azimut: 0
Probability of presence:
Condition of presence: true
Placement radius: 0

Image 2.2 – Placing an AH-64D Apache helicopter with the setting to fly at a height of 40 meters

An explanation of this situation follows

This helicopter has a **Flying Special** setting. That means it will fly on spawn
In it's initialization field is a piece of code:

> **this flyInHeight value** where **this** marks the **object** the command applies to, **flyInHeight** is the command, and **value** is the integer value (number) for the height

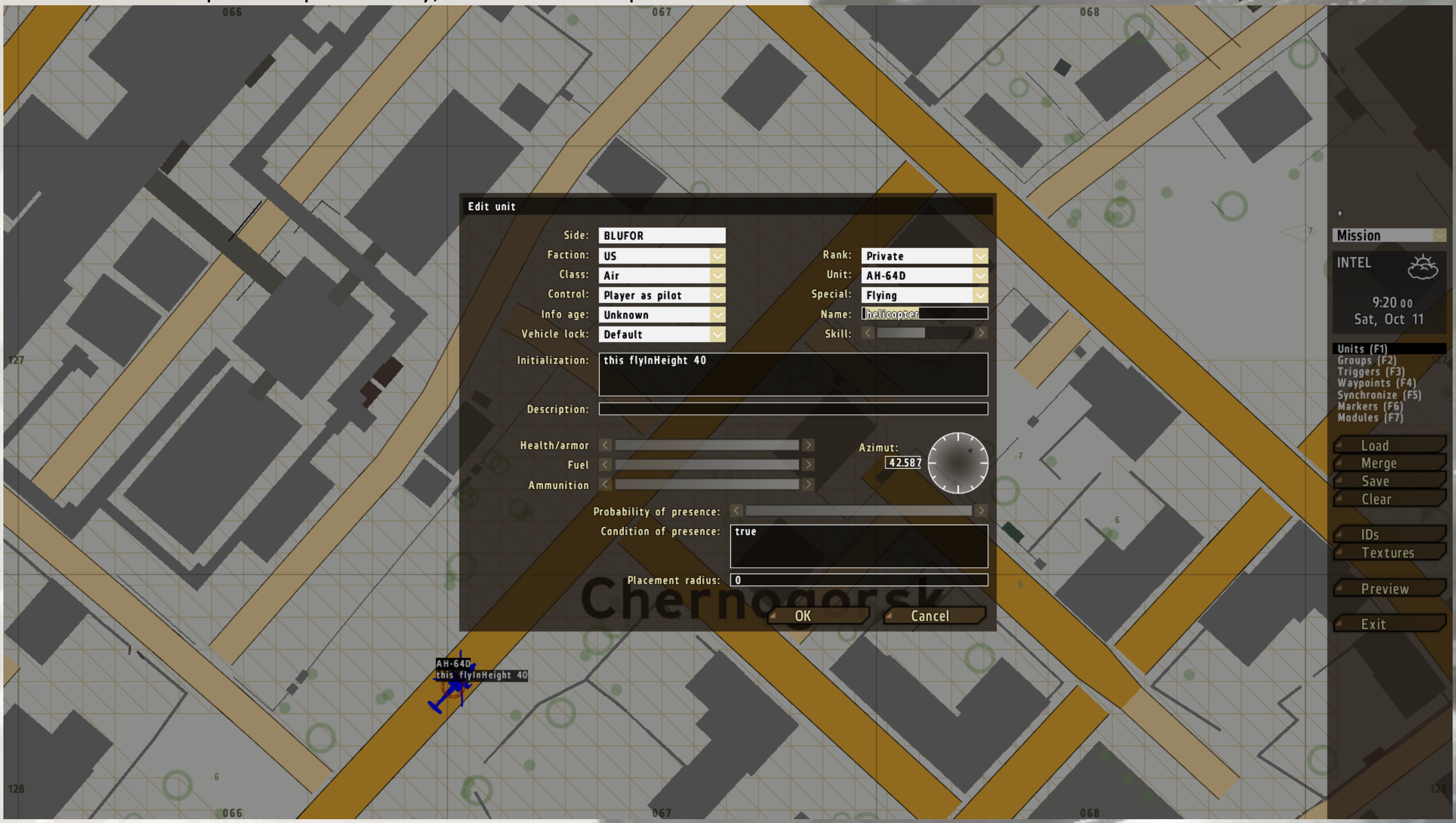
That means that the **this** variable marks theobject executing the command – the unit is applying the command to itself
The game provides assistance by showing this above your code:

> **helicopter flyInHeight height** and lets you know that the **object** the code applies to is a helicopter, and the value next to the command **flyInHeight** is supposed to be the **height** value which determines how high the chopper flies

Almost all editor objects have an init field, and keep an eye out for it, because it may be quite useful.

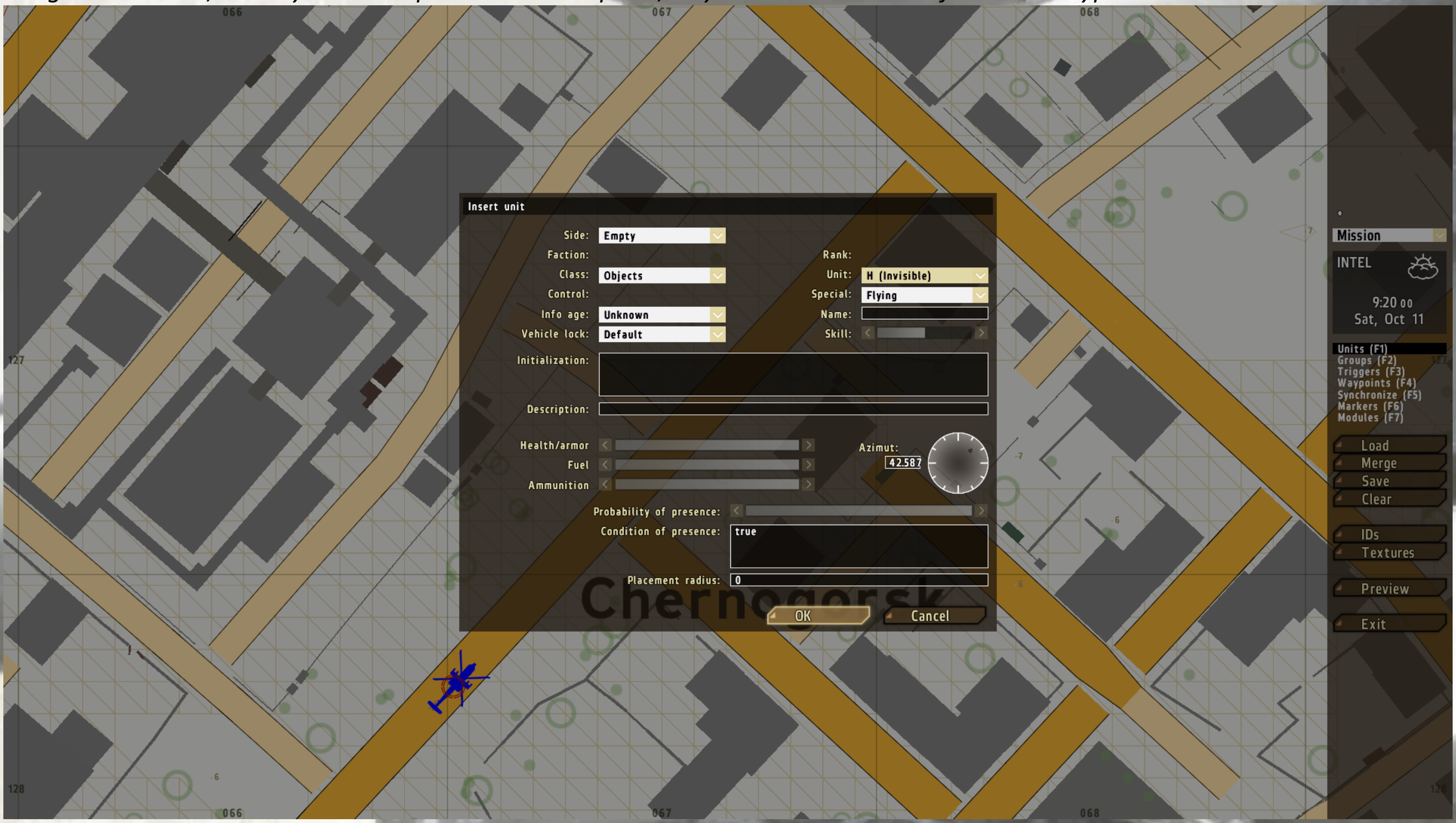
Air vehicles control - landing

Let's control some helicopters… let's say, for example, you need a helicopter to land somewhere. How on earth do you make him do that? It's quite simple actually, here are the steps to do it. ***NOTE: SET THE HELI'S FLYINHEIGHT TO 85, RECOMMENDED***


Edit unit

Side:	BLUFOR		
Faction:	US	Rank:	Private
Class:	Air	Unit:	AH-64D
Control:	Player as pilot	Special:	Flying
Info age:	Unknown	Name:	helicopter
Vehicle lock:	Default	Skill:	
Initialization:	this flyInHeight 40		
Description:			
Azimut:	42.587		
Condition of presence:	true		
Placement radius:	0		

Image 2.31 – First, name your helicopter unit "helicopter", so you can command it from the waypoint's On Act. code box


Insert unit

Side:	Empty		
Faction:		Rank:	
Class:	Objects	Unit:	H (Invisible)
Control:		Special:	Flying
Info age:	Unknown	Name:	
Vehicle lock:	Default	Skill:	
Initialization:			
Description:			
Azimut:	42.587		
Condition of presence:	true		
Placement radius:	0		

Image 2.32 – Then, create a helipad – Empty > Object > H (Invisible) – you will use it to make your heli land where you want to

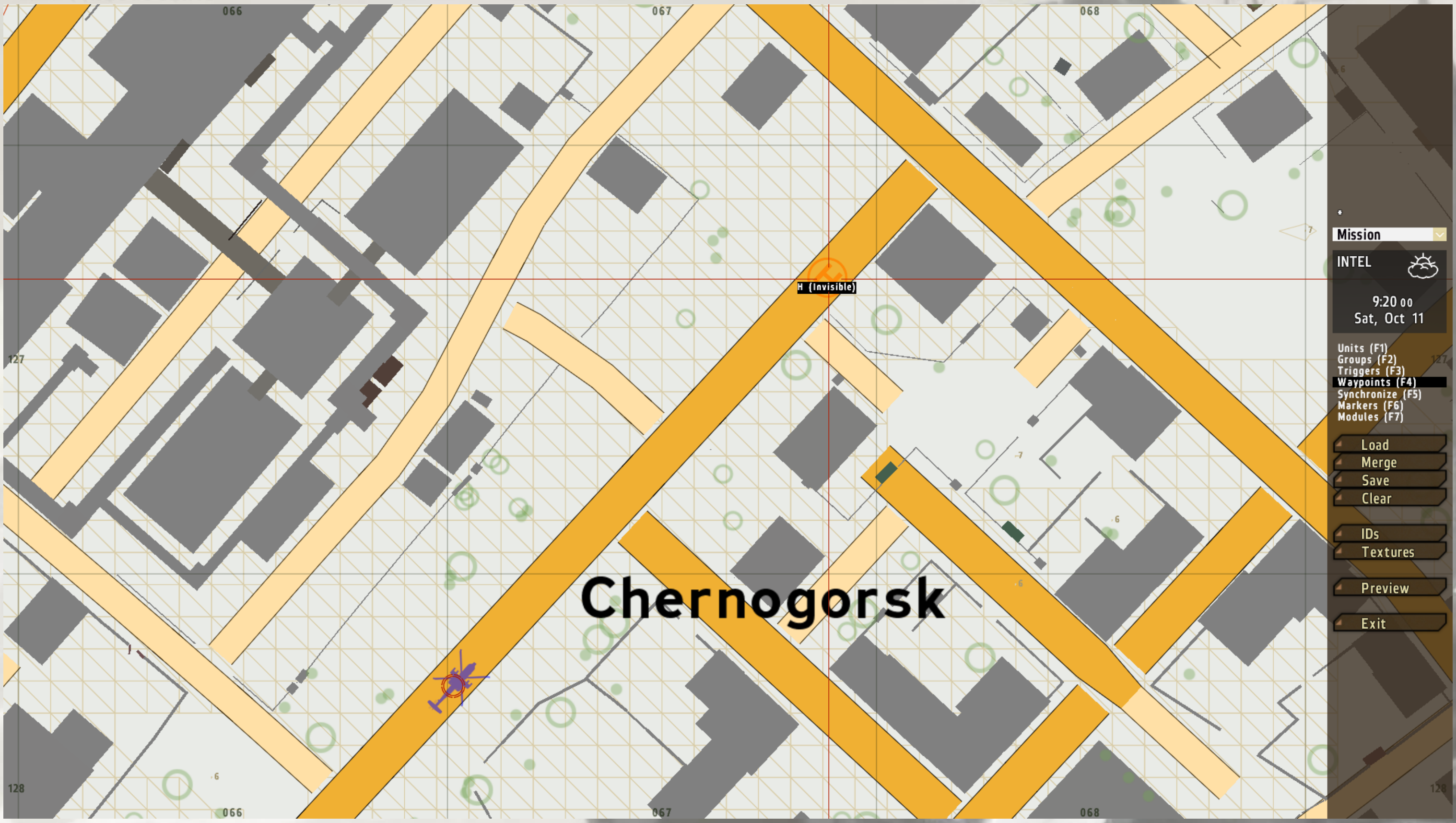
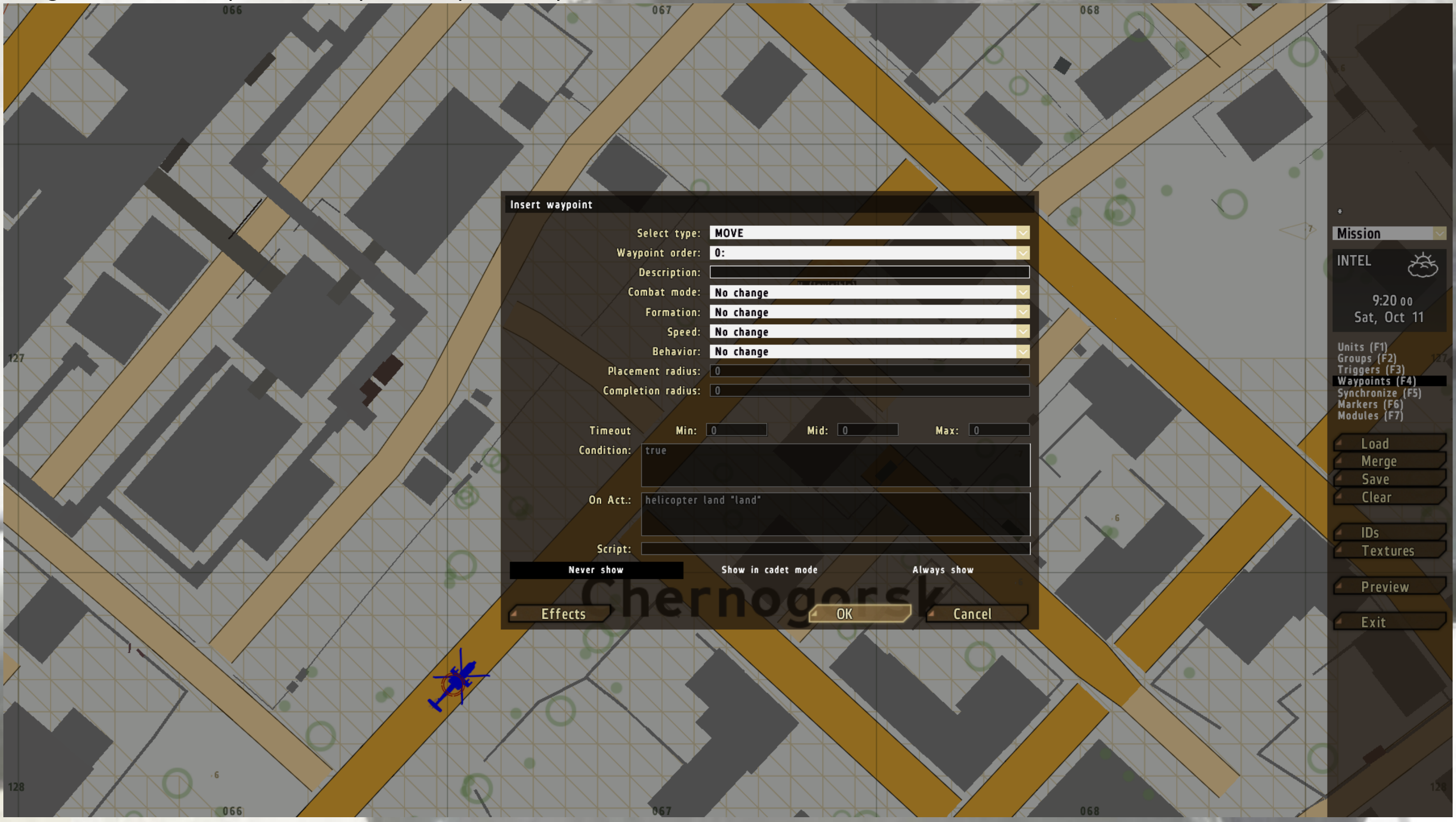


Image 2.33 – A complete helicopter/helipad setup



```
Insert waypoint

        Select type: MOVE
     Waypoint order: 0:
        Description:
        Combat mode: No change
          Formation: No change
              Speed: No change
           Behavior: No change
   Placement radius: 0
  Completion radius: 0

Timeout     Min: 0        Mid: 0        Max: 0
Condition:  true
On Act.:    helicopter land "land"
Script:

Never show        Show in cadet mode        Always show

Effects                          OK          Cancel
```

Image 2.34 – Creating a Move waypoint with the On Act. paramaters to make it land

So what the hell have I just done?

helicopter land "land", or in the Editor's language: **helicopter land landing mode**

The **helicopter** variable is our chopper's name, **land** issues a landing command, and **"land"** is a landing mode where the helicopter touches down and turns off it's engines. There are also **"get in"** and **"get out"** landing modes, which allow him to remain with the rotors turning for quick take-off if in need

Air vehicles control – sum up and results

There, we've set up our settings as follows:

AH-64D:

Name: **helicopter**

Init: **this flyInHeight 40** *NOTE: FOR TROOP LANDINGS, USE 85, HERE I USED 40 BECAUSE IT'S AN ATTACK HELI*

Special: **Flying**

Waypoint 0:

Type: **Move**

On Act.: **helicopter land "land"**

And here are screenshots of the results

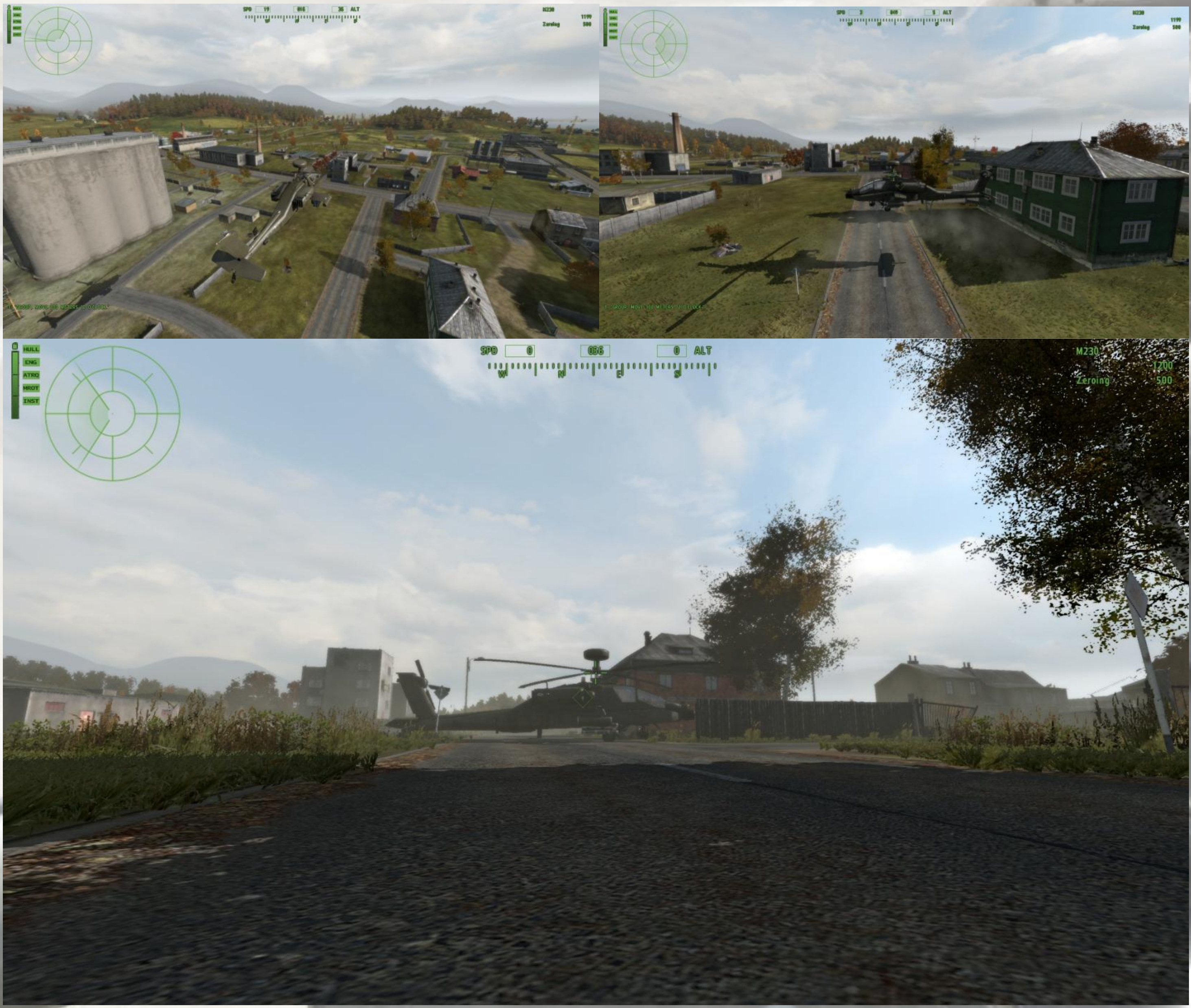
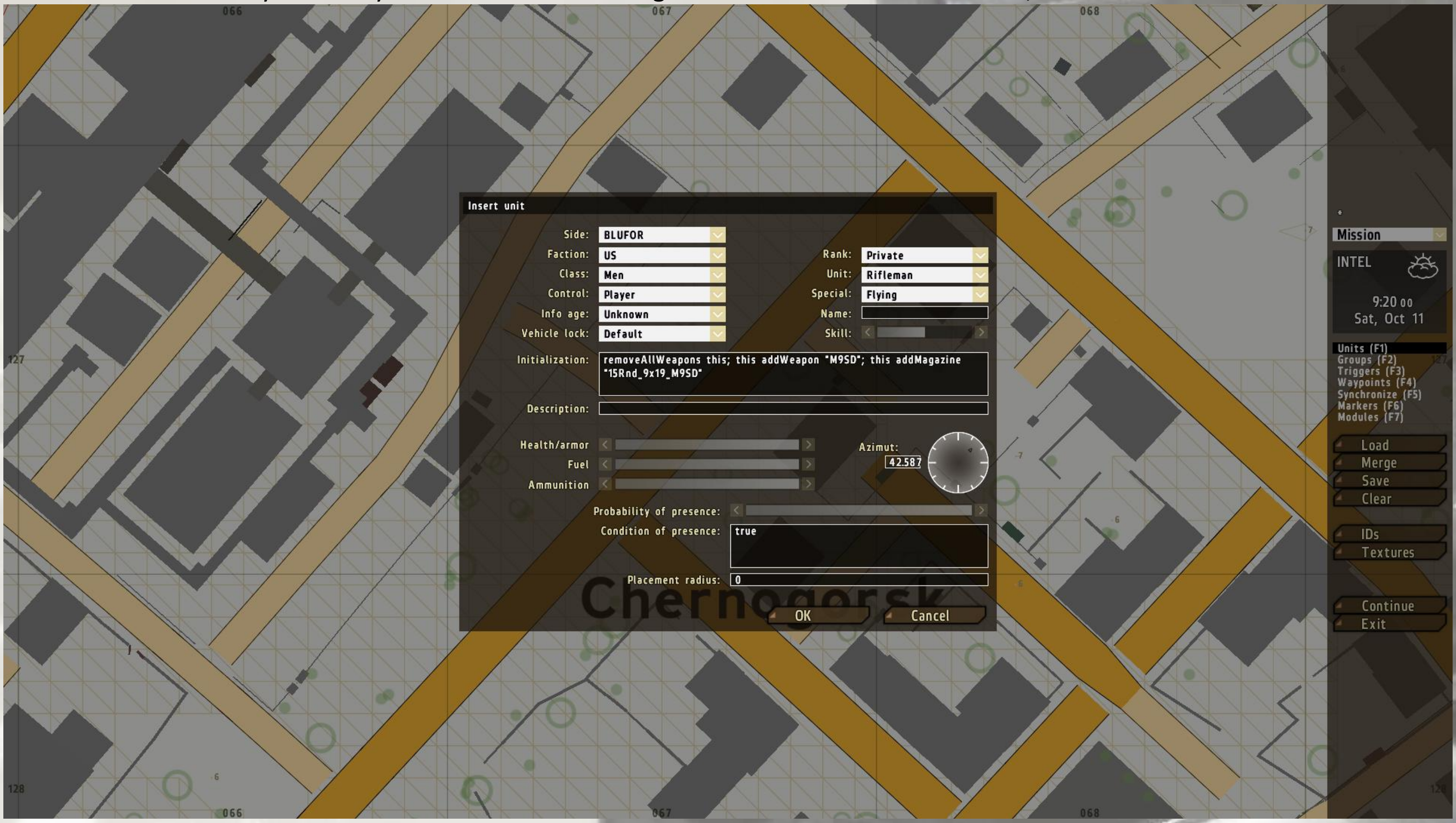


Image 2.4 – The helicopter is (1) in the air (2) performing landing maneuvers (3) on the ground with it's engines off

Unit gear control

Ever been frustrated by the fact you can't customize the gear for a unit in a mission? Well, here's how:



Insert unit

Side:	BLUFOR		
Faction:	US	Rank:	Private
Class:	Men	Unit:	Rifleman
Control:	Player	Special:	Flying
Info age:	Unknown	Name:	
Vehicle lock:	Default	Skill:	

Initialization: removeAllWeapons this; this addWeapon "M9SD"; this addMagazine "15Rnd_9x19_M9SD"

Description:

Azimut: 42.587

Condition of presence: true

Placement radius: 0

Image 2.5 – Placing a Rifleman armed only with his M9 Silenced sidearm and only one magazine

An explanation of these factors is like this:

removeAllWeapons this; this addWeapon "M9SD"; this addMagazine "15Rnd_9x19_M9SD"

or in the Editor's help boxes: **removeAllWeapons unit; unit addWeapon weapon; unit addMagazine magazine**

The **removeAllWeapons** command is used to strip the unit of all it's weapons to make room for your custom loadout

The **addWeapon** command adds another weapon to the current gear, and is followed by the weapon's classname in quotes

The **addMagazine** command works in a similar way as the previous one, but it gives ammo to the unit

NOTE: EVEN IF THE UNIT'S WEAPON DOES NOT USE MAGAZINES (A ROCKET LAUNCHER, E.G.), THE COMMAND IS THE SAME

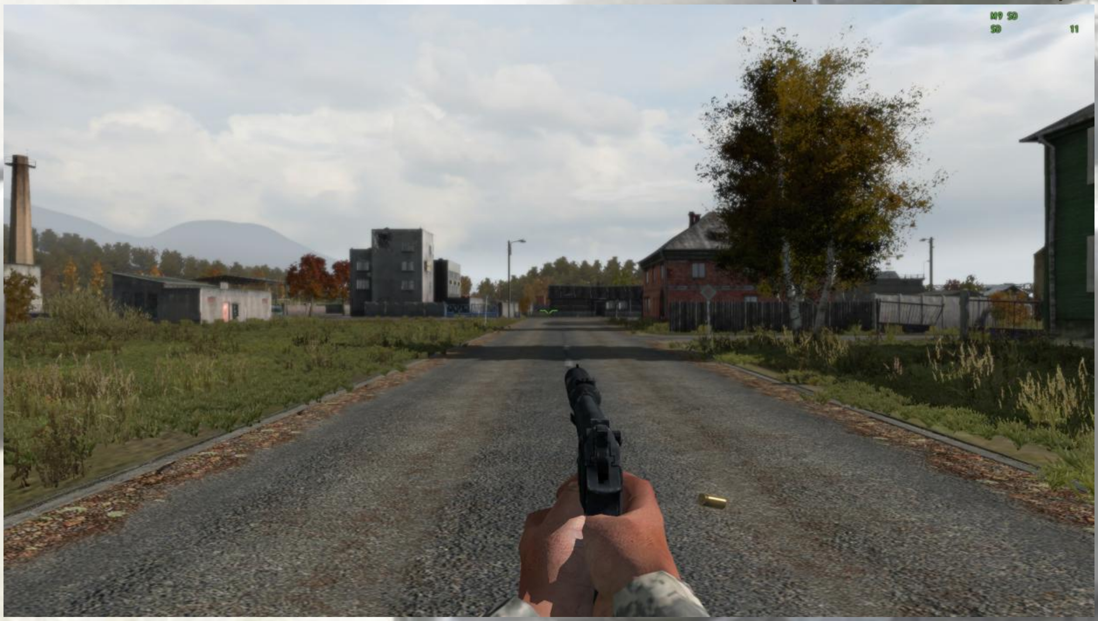
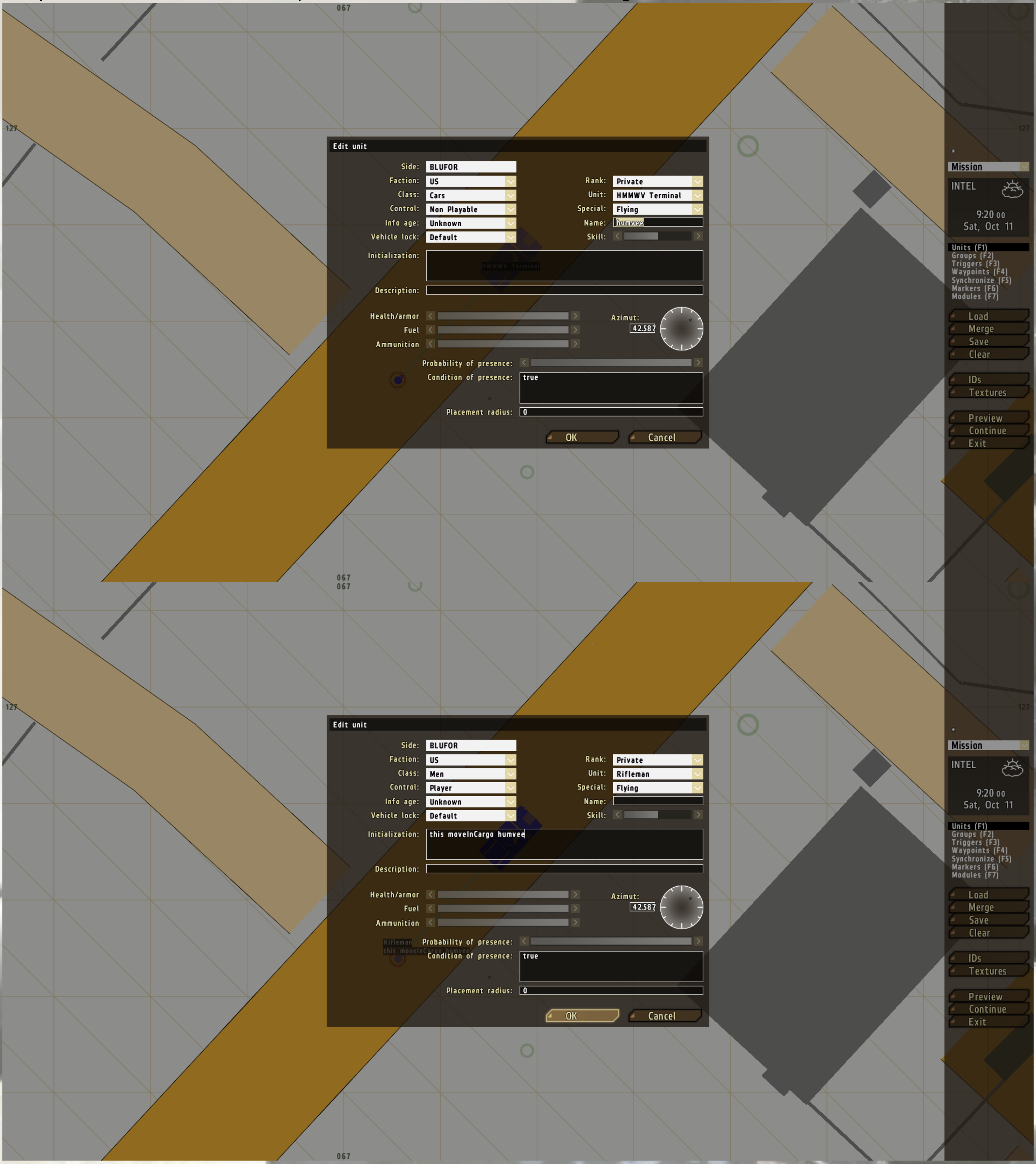


Image 2.51 – Here you see the results of the M9SD + 1 Magazine command. Look in the top-right corner to check the amount of ammunition for the current weapon. Only 1 15-round magazine! There are 11 rounds in currently because I fired some :P

Boarding vehicles

Boarding vehicles is not tough in-game, but when you want to place units inside some vehicle's cargo space it might get a bit nasty… and brutal. So, in order to explain how it works, we could use an image first



Edit unit

Side: BLUFOR
Faction: US
Class: Cars
Control: Non Playable
Info age: Unknown
Vehicle lock: Default
Rank: Private
Unit: HMMWV Terminal
Special: Flying
Name: humvee
Skill:
Initialization:
Description:
Health/armor
Fuel
Ammunition
Azimut: 42.587
Probability of presence:
Condition of presence: true
Placement radius: 0
OK Cancel

Edit unit

Side: BLUFOR
Faction: US
Class: Men
Control: Player
Info age: Unknown
Vehicle lock: Default
Rank: Private
Unit: Rifleman
Special: Flying
Name:
Skill:
Initialization: this moveInCargo humvee
Description:
Health/armor
Fuel
Ammunition
Azimut: 42.587
Probability of presence:
Condition of presence: true
Placement radius: 0
OK Cancel

Image 2.6 – Placing a Rifleman unit inside a HMMMV Terminal car

Give your desired vehicle a **name**. In this example I used the name humvee, then insert this command into your unit's Init field: **this moveInCargo vehicle**. You can also use the **moveInGunner** and **moveInDriver** codes

Mission editing modes

To select a **mission editing mode**, click on the **Mission** field in the side menu and select the desired mode from **Mission**, **Intro**, and **Outro win/lose** modes

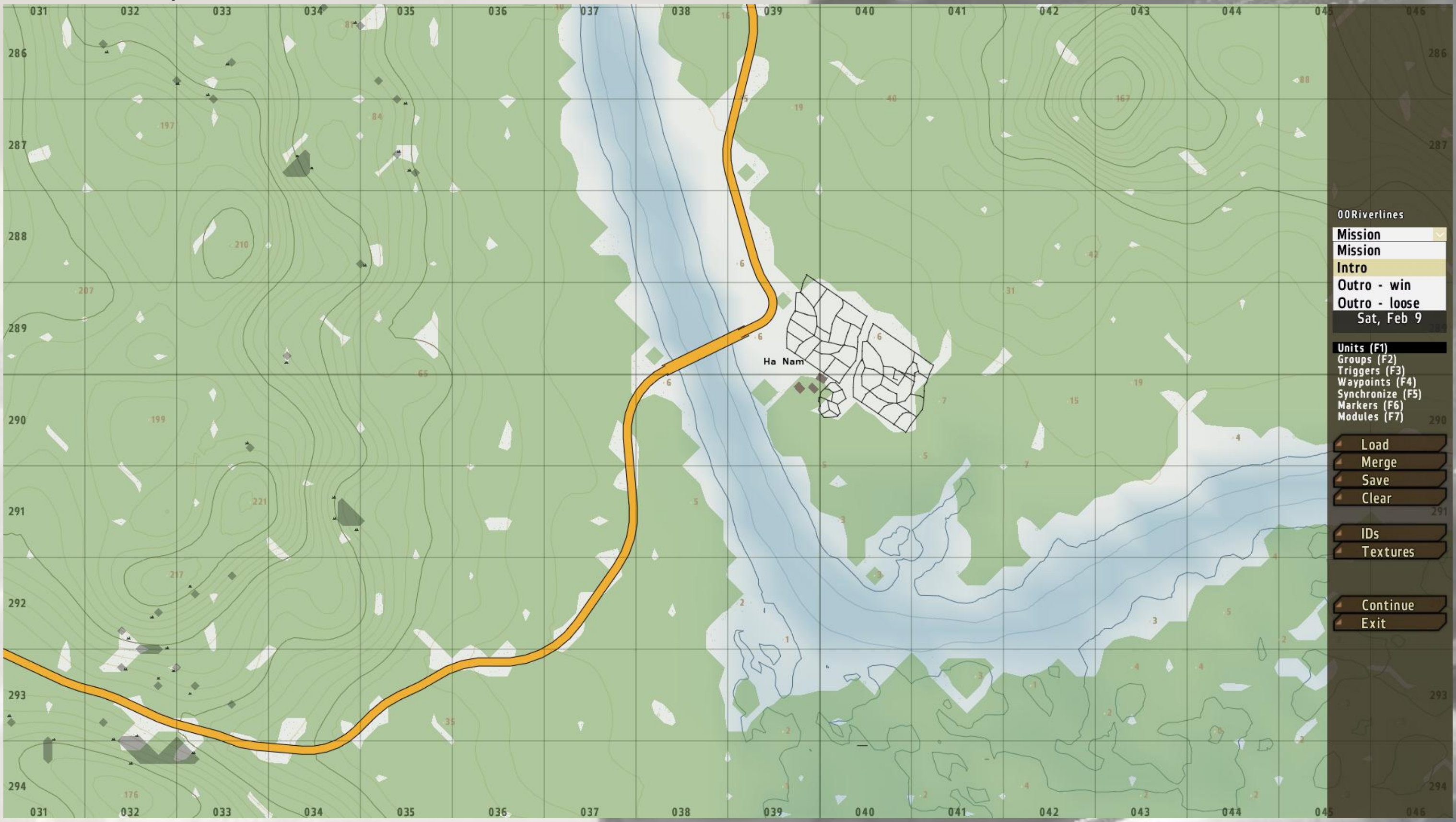


*Image 2.7 – Selecting the Intro editing mode. If you noticed you don't have this map in your game, don't worry – it's an **addon** map called **MBG_Vietnam***

A LIST OF EDITING MODES

Mission In this editing mode, you edit your Mission, during which the player character is human controlled
Intro In this editing mode, you edit your Intro, during which the player character is CPU controlled and mostly is a cutscene
Outro Outros are the same thing as the Intro, but are played after the mission ends and have two types

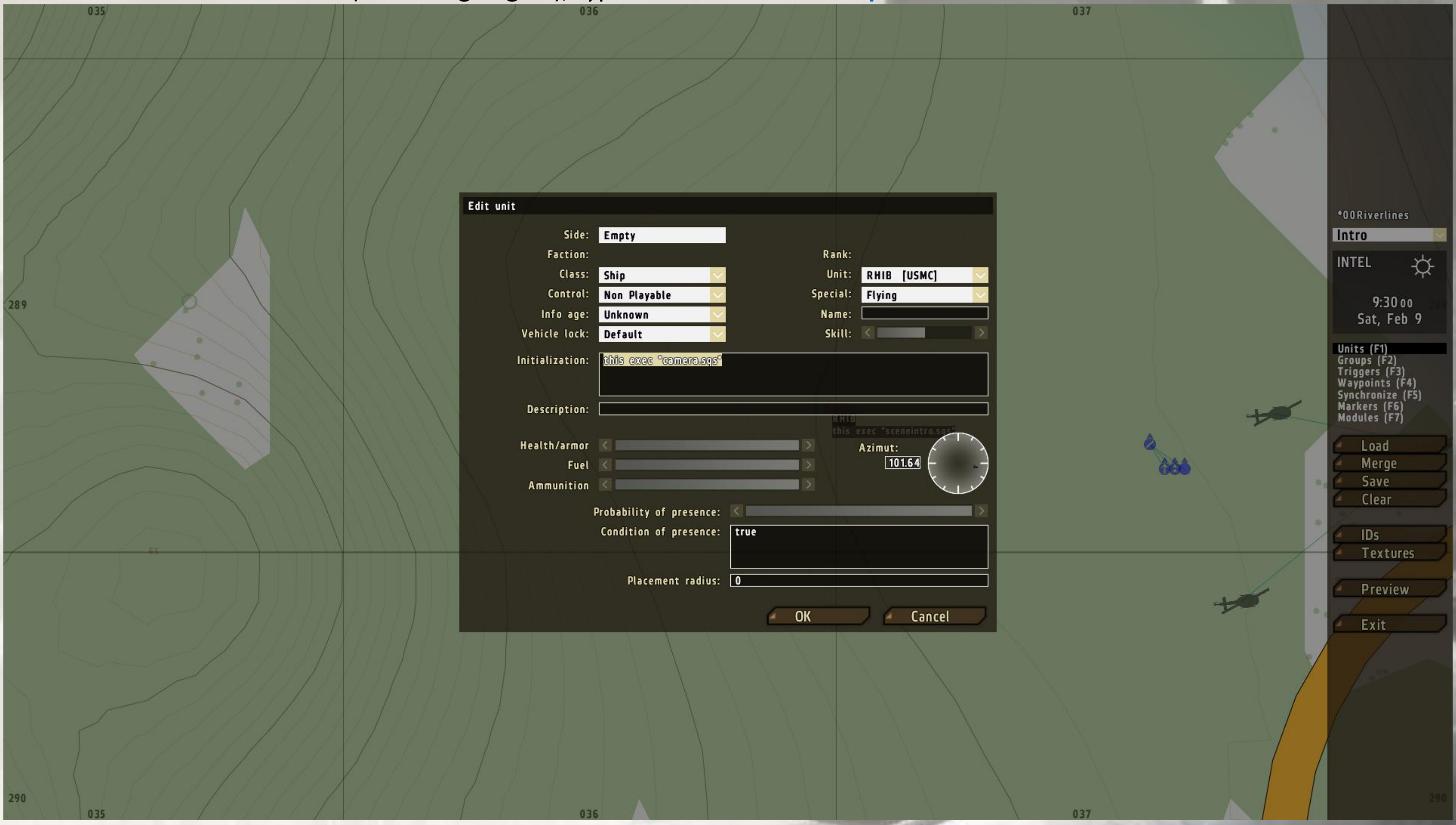
Outro – win This Outro type is the type played after you've **won** the mission (completed enough tasks)
Outro – lose This Outro type is the type played after you've **lost** the mission (did not accomplish enough objectives)

In order to create cutscenes for Intro and Outro modes, you must use scripts and other external text files that will be covered in the next section

Camera scripting

In the Intro and Outro modes you will want to create cutscenes to familiarize players with your mission's story, or just for showing off, but in any case you will need some extra text files and a bit of scripting.

1. In your mission's Intro or Outro mode, create a unit ***NOTE: YOU WILL NEED A PLAYER ASWELL***
2. In that unit's **Init** field (here we go again), type **this exec "camera.sqs"**



Edit unit

Side:	Empty		
Faction:		Rank:	
Class:	Ship	Unit:	RHIB [USMC]
Control:	Non Playable	Special:	Flying
Info age:	Unknown	Name:	
Vehicle lock:	Default	Skill:	

Initialization: this exec "camera.sqs"

Description:

Health/armor

Fuel

Ammunition

Azimut: 101.64

Probability of presence:

Condition of presence: true

Placement radius: 0

3. Preview the mission. You will notice that you are not playing the player. That is correct – you're "playing" the camera

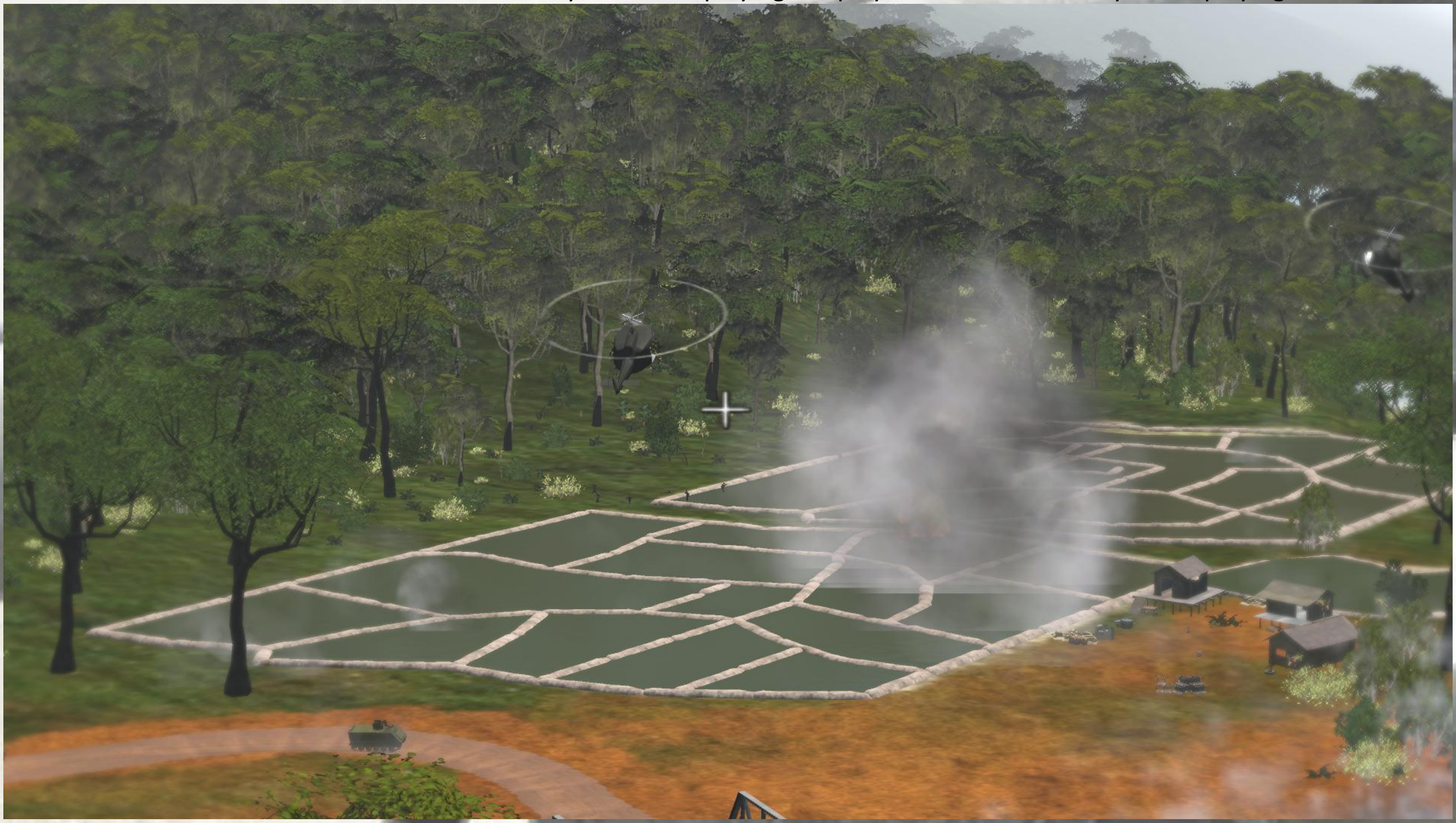


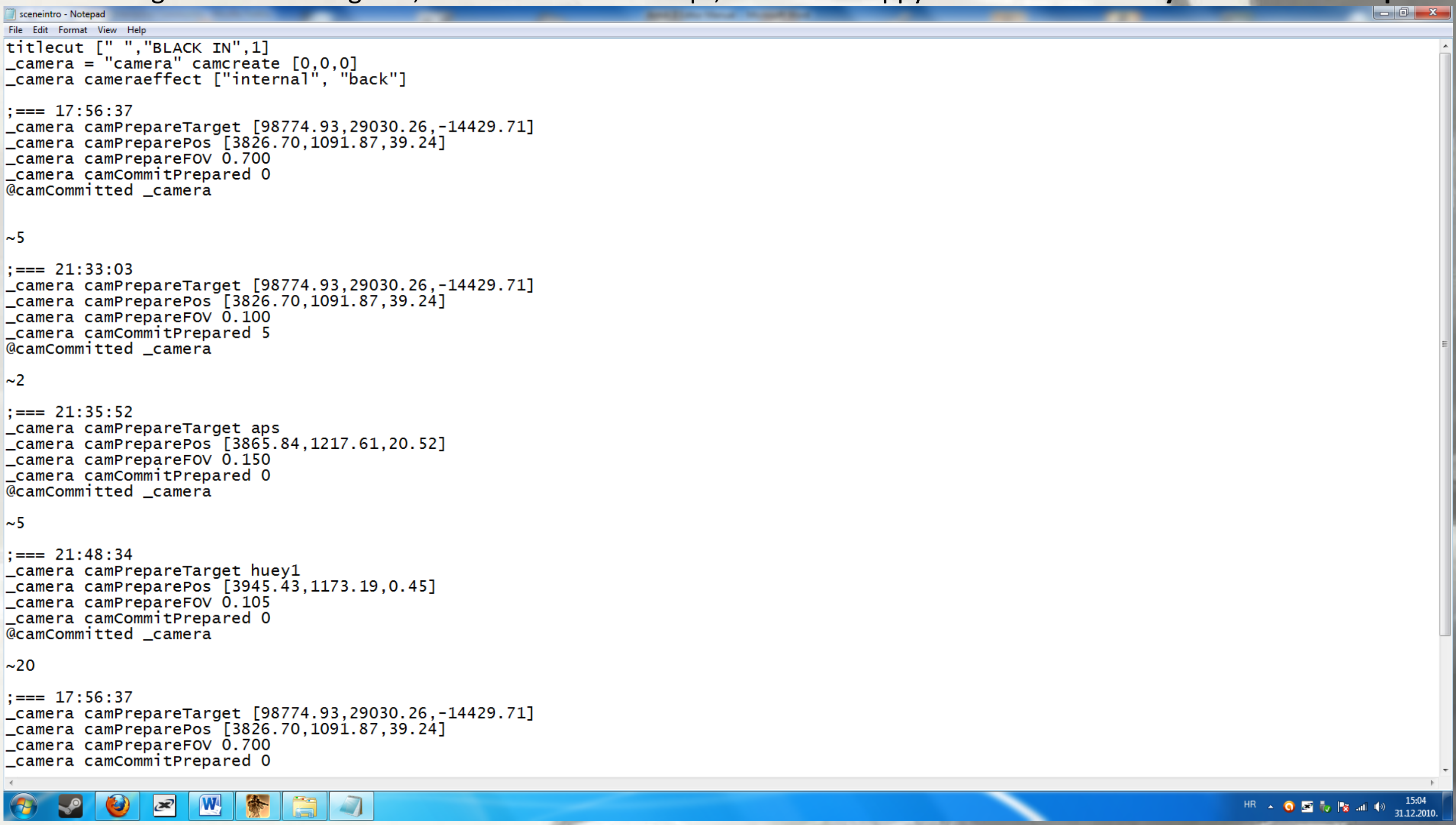
Image 2.8 – Placing a unit with the [this exec "camera.sqs"] command (upper slide), and camera.sqs in action (lower slide)

4. Use this table of camera controls to move the camera around, and when finally statisfied with the shot, press the **LMB**

ARMA II CAMERA.SQS CONTROLS

Movement	**W, A, S, D** camera movement	**Q, Z** increase/decrease height	**Shift+WASD** fast moving
Zooming and tilting/panning	**Num +, Num -** zoom in/out	**Num 4, 6** pan left/right	**Num 8, 2** tilt up/down
Other controls	**F** target nearest unit	**L** toggle crosshair	**Del** floating mode on/off
ArmA II: OA only controls	**N** swap between normal vision/NV/TI WB/TI BW	**1-9** select post-processing for camera	**Mousewheel up/down** modify focal lenght

5. Then use **Alt+Tab** to minimize the game, open **Notepad** and use **Ctrl+V** to paste the camera information
6. You can go back into the game, or further edit the script, and when happy save it as **All files** – **yourscenename.sqs**



```
titlecut [" ","BLACK IN",1]
_camera = "camera" camcreate [0,0,0]
_camera cameraeffect ["internal", "back"]

;=== 17:56:37
_camera camPrepareTarget [98774.93,29030.26,-14429.71]
_camera camPreparePos [3826.70,1091.87,39.24]
_camera camPrepareFOV 0.700
_camera camCommitPrepared 0
@camCommitted _camera


~5

;=== 21:33:03
_camera camPrepareTarget [98774.93,29030.26,-14429.71]
_camera camPreparePos [3826.70,1091.87,39.24]
_camera camPrepareFOV 0.100
_camera camCommitPrepared 5
@camCommitted _camera

~2

;=== 21:35:52
_camera camPrepareTarget aps
_camera camPreparePos [3865.84,1217.61,20.52]
_camera camPrepareFOV 0.150
_camera camCommitPrepared 0
@camCommitted _camera

~5

;=== 21:48:34
_camera camPrepareTarget huey1
_camera camPreparePos [3945.43,1173.19,0.45]
_camera camPrepareFOV 0.105
_camera camCommitPrepared 0
@camCommitted _camera

~20

;=== 17:56:37
_camera camPrepareTarget [98774.93,29030.26,-14429.71]
_camera camPreparePos [3826.70,1091.87,39.24]
_camera camPrepareFOV 0.700
_camera camCommitPrepared 0
```

Image 2.81 – A finished scene script, called "sceneintro.sqs" in my case

How to use the ArmA II .sqs scripting language?

titlecut [" ","BLACK IN",1]
_camera = "camera" camcreate [0,0,0]
_camera cameraeffect ["internal", "back"]

This is the beginning of your script. It issues the command for a fade-in from black and prepares you for the camera commands

;=== 17:56:37	
_camera camPrepareTarget [98774.93,29030.26,-14429.71]	YOUR CAMERA TARGET'S POSITION
_camera camPreparePos [3826.70,1091.87,39.24]	YOUR CAMERA'S POSITION
_camera camPrepareFOV 0.700	THE AMOUNT OF ZOOM (SMALLER NUMBER = MORE ZOOM)
_camera camCommitPrepared 0	THIS IS THE LENGTH OF THE CAMERA'S PAN INTO THE NEXT SCENE. 0 = NO PAN
@camCommitted _camera	LEAVE THIS AS IS

This is what you will paste from the camera.sqs from in-game. It contains information about the cam's position and zoom

~5

This is the command that says to the game "wait this long before you cross to the next camera position"
When you save the script, you will want to move it to (**Documents → ArmA 2 → missions → YourMissionName.Map)**
You will see the effects of my "sceneintro.sqs" script in the following video

„scene.sqs“ demonstration video

*Video 1 - In this video you could see the Intro part of my mission. It took me only a couple of hours to create and fine-tune it so it shouldn't take any more time from you, except when creating more complex stuff. You can also hear the background music is customized (see **section 3**, **chapter 2**: Custom Background Music). For those who want to know what song that was, it's **„Fortunate Son“ by Creedence Clearwater Revival***

Advanced use of the Editor

Practical use of triggers and syncing

You've heard enough about all this in theory, but you have to put it into practice. In order to make two things happen at once, you will need to synchronize them. This enables you (in combinations with triggers) to create special waypoints which are essentialy **Move** type waypoints, but in-game are displayed with a **WAIT** message under the marker. Here's how you do it:

1. Create a trigger and set it to **OPFOR** and with **Present** conditions
2. Create a player unit and a **Move** waypoint about 5 meters from it
3. Create another **Move** waypoint somewhere else and **synchronize** that waypoint with the trigger
4. Create an **OPFOR** unit and set a **Move** waypoint for it inside the trigger's are of effect
5. Preview the mission

You will probably notice that you first had to **Wait** at the first waypoint until the **OPFOR** unit came up to the trigger, and just then your other waypoint got online. Try another example of practical synchronization. Let's say you need some guys to move somewhere and when that, two guys join them. How do we make it? Here are the steps

1. Create two **Men**, one a player, rank **Private**, one non-playable, rank **Sergeant**
2. Create two **Move** waypoints for them
3. Create two more **Men**, one rank **Private**, one **Sergeant**, both non-playable and set them as a **separate group**
4. Create two **Move** waypoints for them, too
5. **Synchronize** their first waypoint with the first group's last waypoint
6. Preview the mission

Notice that the "waiting" group (the second one) will wait until the first group accomplishes their second objective. They will then begin to move out to whatever their objective is

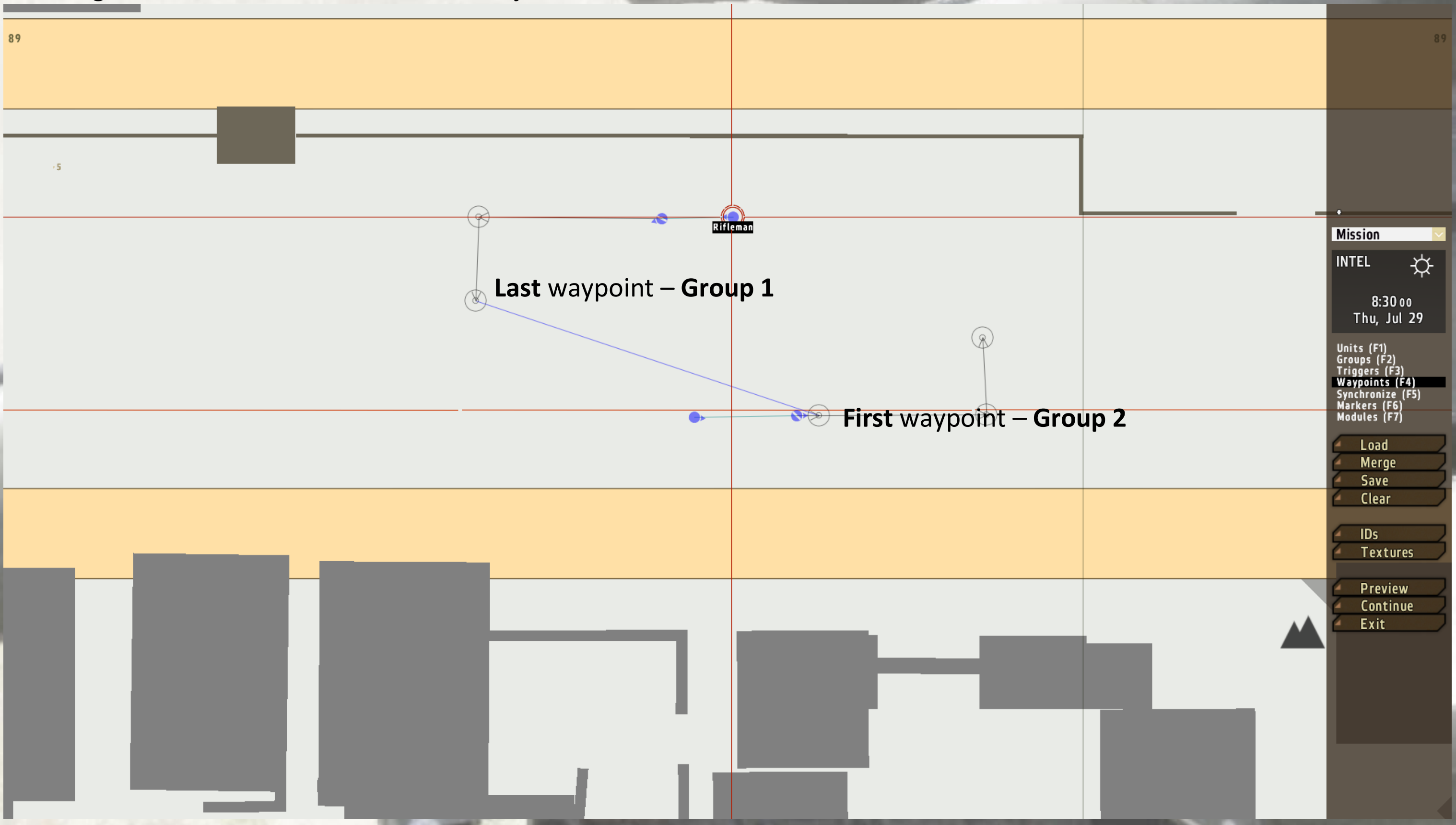


Image 3 – Here you can see the editor view of syncing the first and last waypoints of two different groups

Inserting custom background music

As you've seen in Video 1 on page 22, it is possible to insert custom music and sounds into ArmA II using the game's editor and some more supporting text files

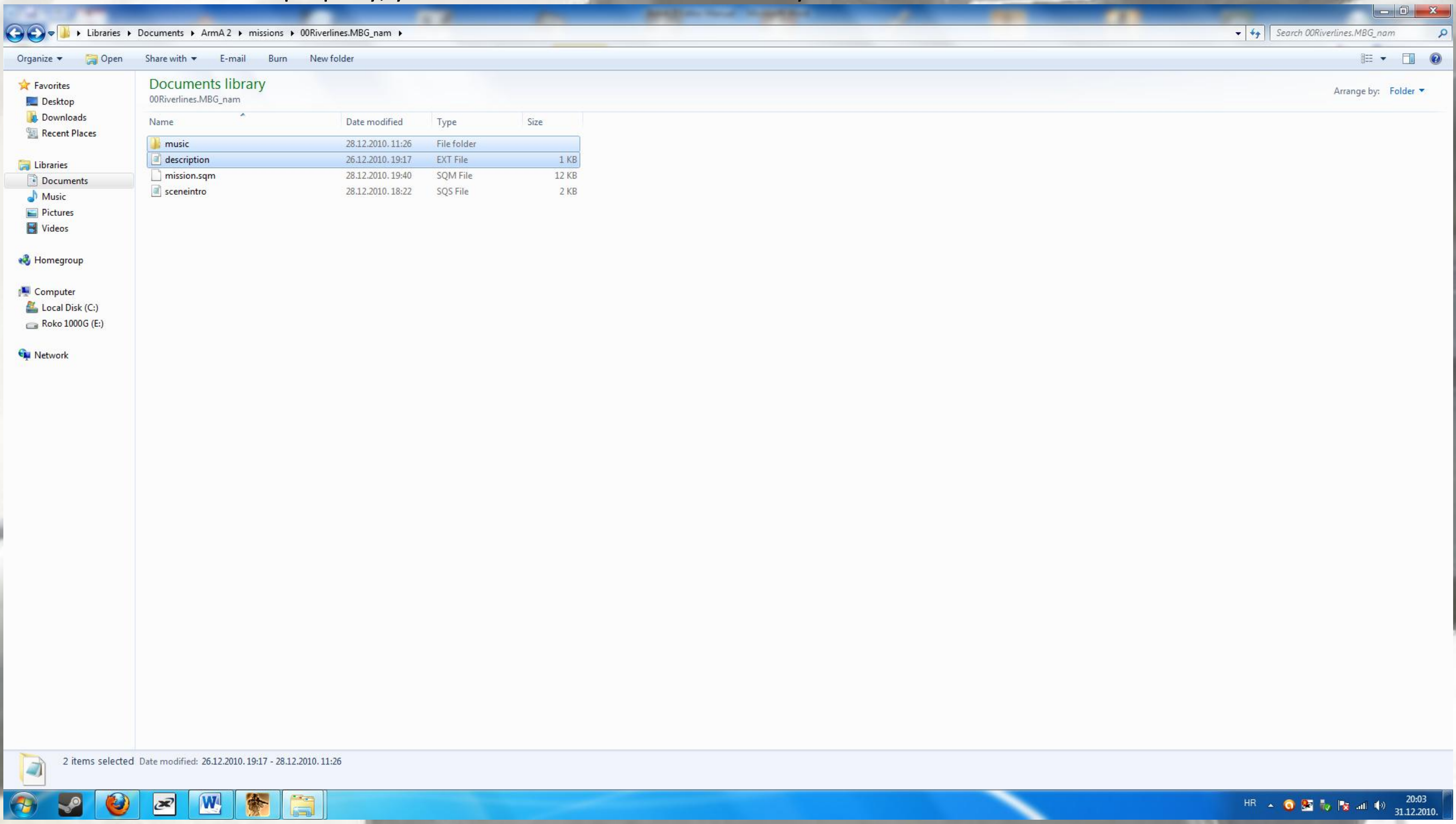
1. Inside your mission folder, create a new text file
2. In that text file, type

```
class CfgMusic
{
tracks[]={yourmusicname };

class yourmusicname
{
name = "yourmusicname"; //The Name seen in the Editor
sound[] = {"\music\yourmusicname.ogg", db+0, 1.0};
};


};
```

This will make your music track
NOTE: THE EDITOR ONLY ACCEPTS .OGG FILES (you can convert files to .ogg using format factory, it's free) appear in your **Effects box** of triggers and waypoints under the name **yourmusicname** defined by the **name = "yourmusicname";** line in the code (the **7th line** in the code above).
In order for this to work properly, you will need a **music** folder inside your mission folder


Libraries ▸ Documents ▸ ArmA 2 ▸ missions ▸ 00Riverlines.MBG_nam ▸

Documents library
00Riverlines.MBG_nam

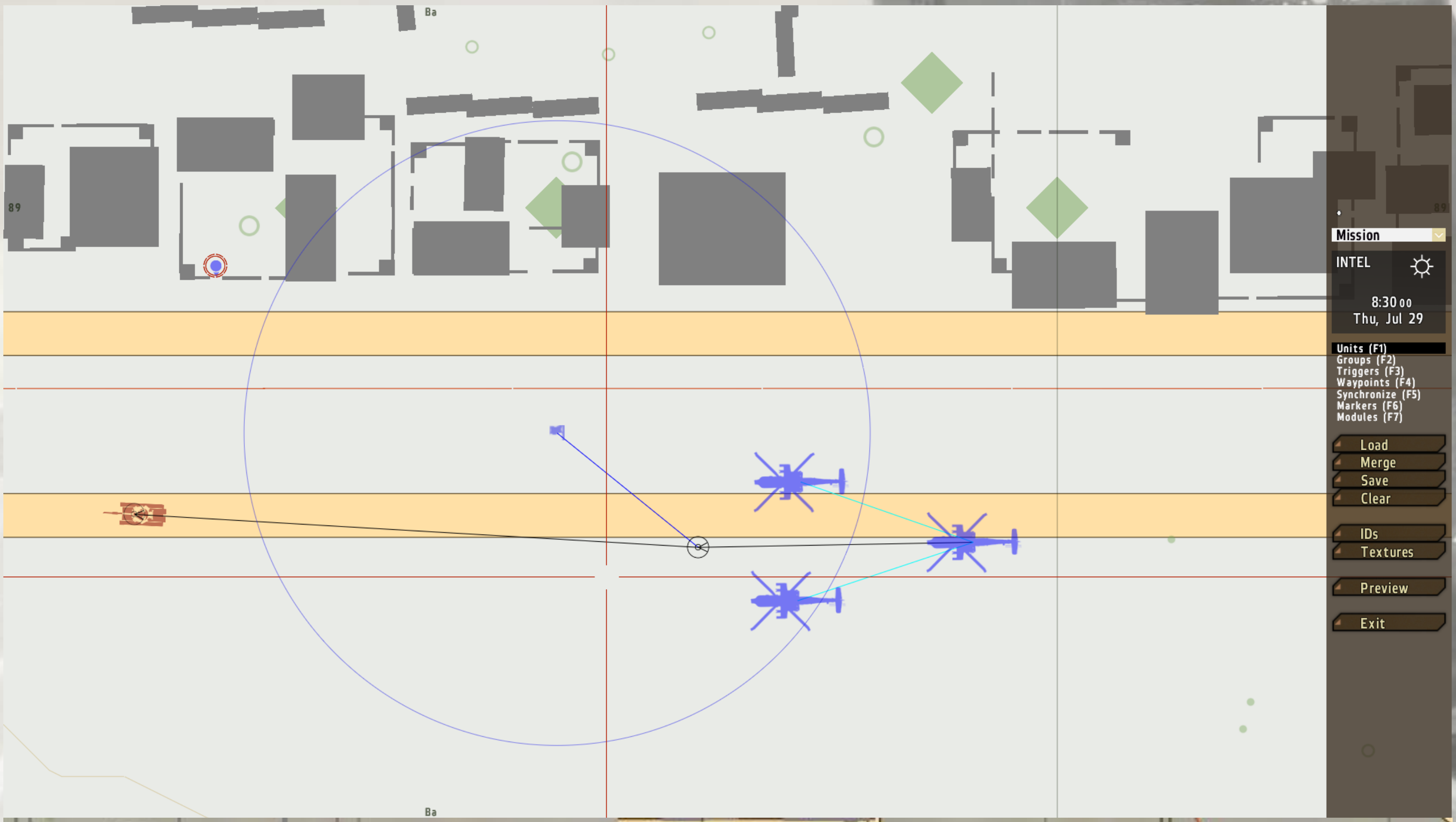
Name	Date modified	Type	Size
music	28.12.2010. 11:26	File folder	
description	26.12.2010. 19:17	EXT File	1 KB
mission.sqm	28.12.2010. 19:40	SQM File	12 KB
sceneintro	28.12.2010. 18:22	SQS File	2 KB

2 items selected Date modified: 26.12.2010. 19:17 - 28.12.2010. 11:26

Image 3.1 – My mission folder (00Riverlines.MBG_nam) with the music folder and description.ext file included

3. Save that file as file type: **All files**, file name: **description.ext**

Calling in predefined CAS



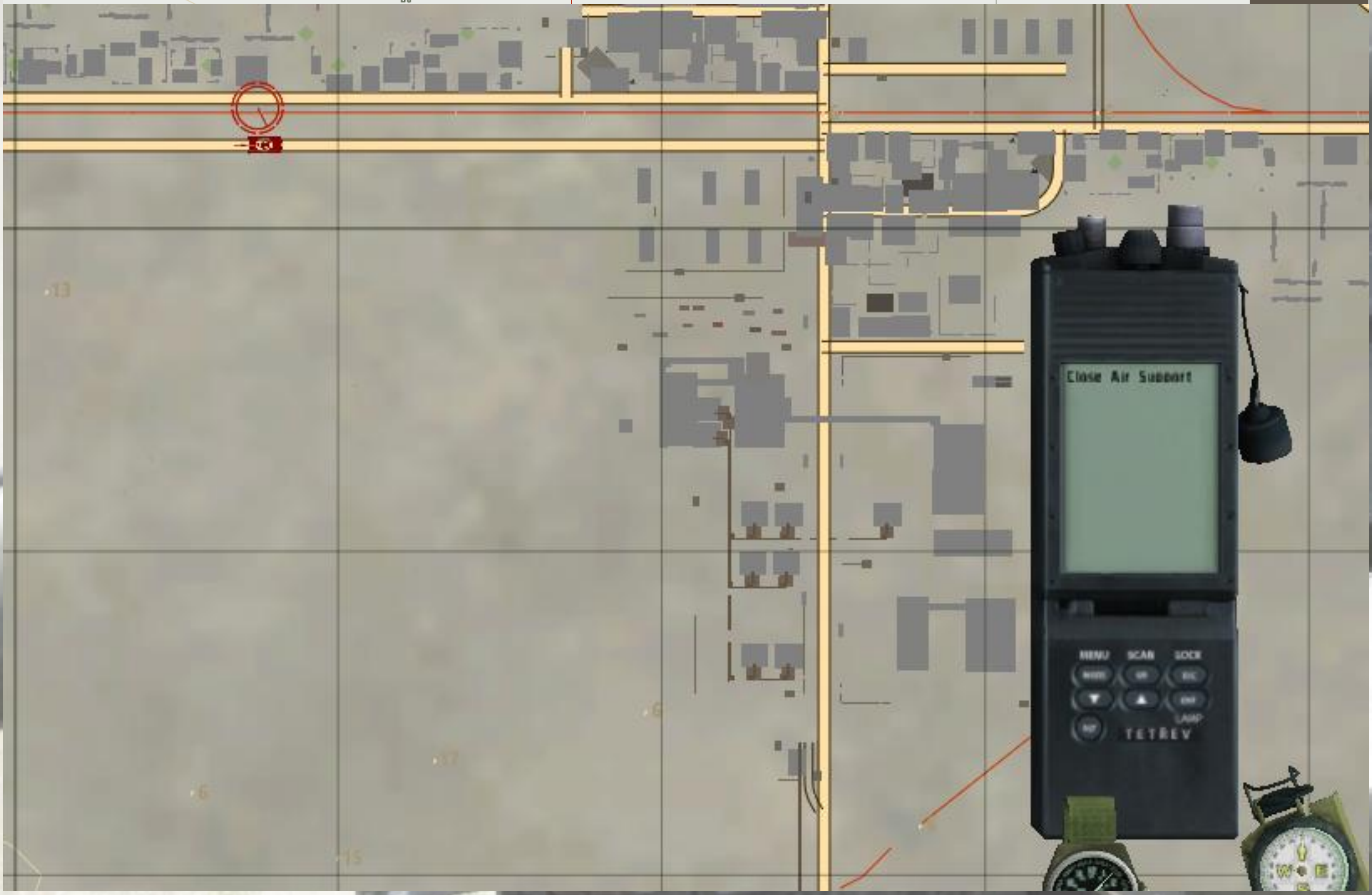
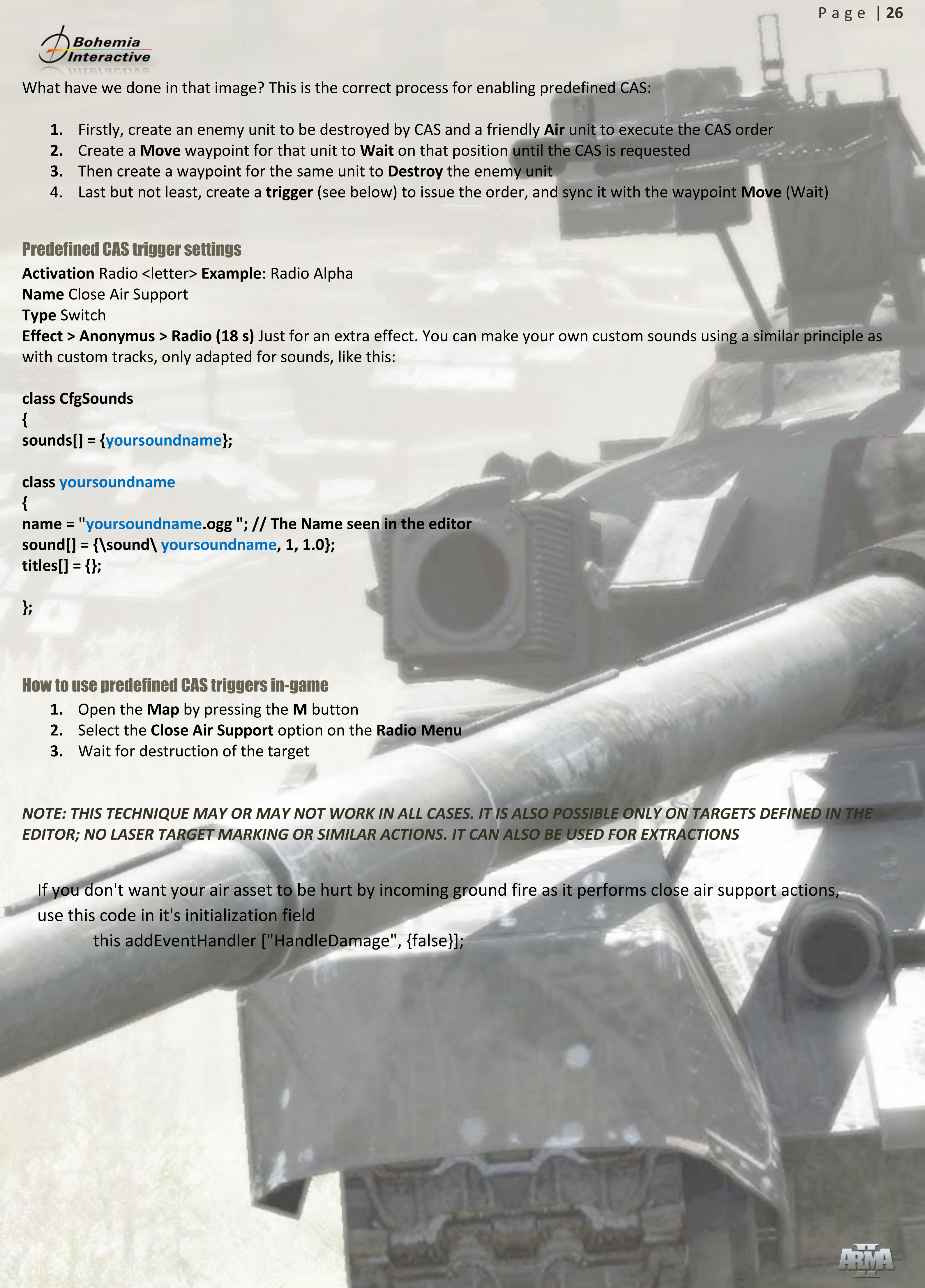


Image 3.2 – The upper slide shows the correct unit/trigger/waypoint setup for predefined CAS. The lower one displays the Map view when in-game after that kind of setup



What have we done in that image? This is the correct process for enabling predefined CAS:

1. Firstly, create an enemy unit to be destroyed by CAS and a friendly **Air** unit to execute the CAS order
2. Create a **Move** waypoint for that unit to **Wait** on that position until the CAS is requested
3. Then create a waypoint for the same unit to **Destroy** the enemy unit
4. Last but not least, create a **trigger** (see below) to issue the order, and sync it with the waypoint **Move** (Wait)

Predefined CAS trigger settings

Activation Radio <letter> **Example**: Radio Alpha
Name Close Air Support
Type Switch
Effect > Anonymus > Radio (18 s) Just for an extra effect. You can make your own custom sounds using a similar principle as with custom tracks, only adapted for sounds, like this:

```
class CfgSounds
{
sounds[] = {yoursoundname};

class yoursoundname
{
name = "yoursoundname.ogg "; // The Name seen in the editor
sound[] = {\sound\ yoursoundname, 1, 1.0};
titles[] = {};

};
```

How to use predefined CAS triggers in-game

1. Open the **Map** by pressing the **M** button
2. Select the **Close Air Support** option on the **Radio Menu**
3. Wait for destruction of the target

NOTE: THIS TECHNIQUE MAY OR MAY NOT WORK IN ALL CASES. IT IS ALSO POSSIBLE ONLY ON TARGETS DEFINED IN THE EDITOR; NO LASER TARGET MARKING OR SIMILAR ACTIONS. IT CAN ALSO BE USED FOR EXTRACTIONS

If you don't want your air asset to be hurt by incoming ground fire as it performs close air support actions, use this code in it's initialization field

```
        this addEventHandler ["HandleDamage", {false}];
```

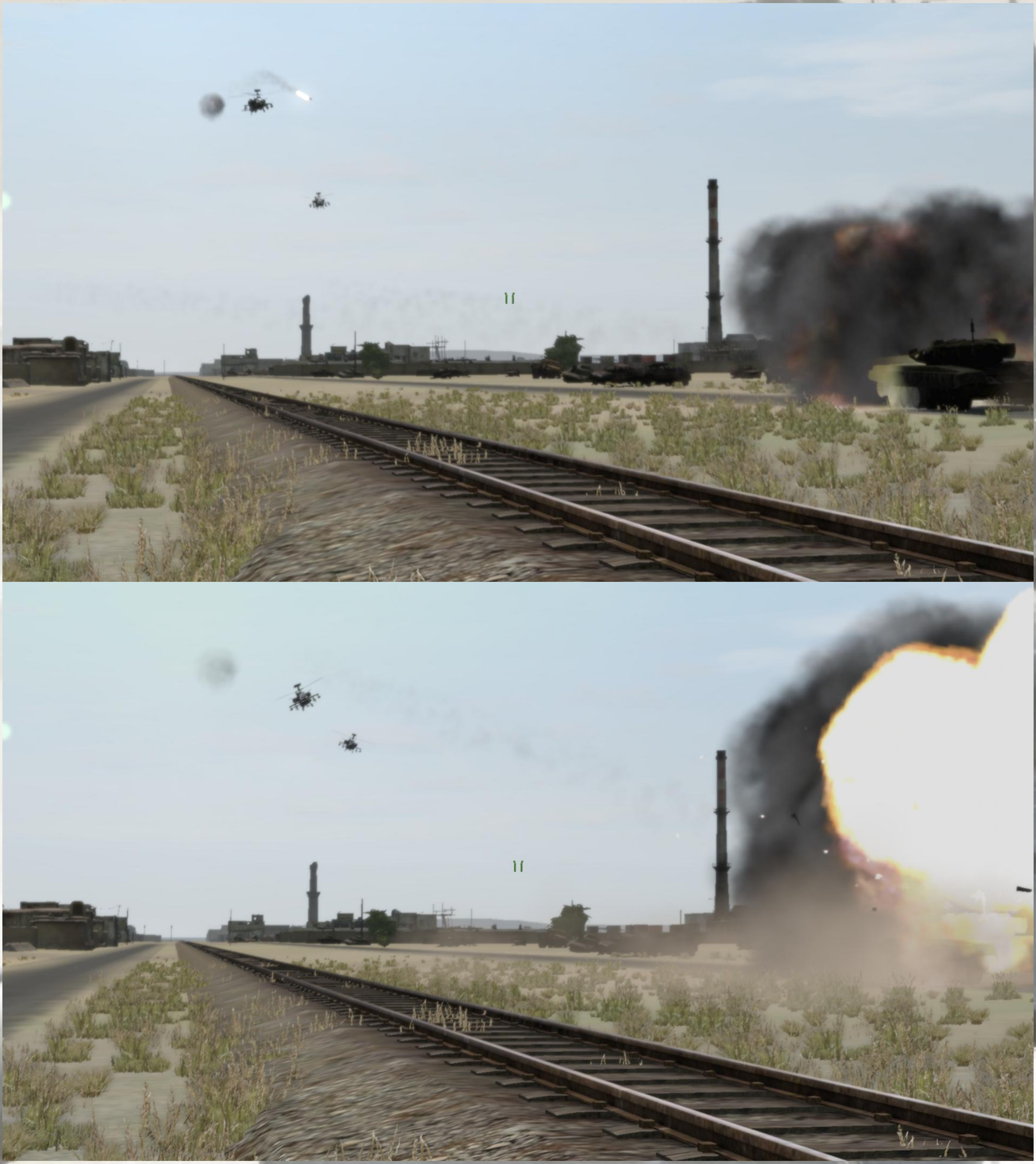
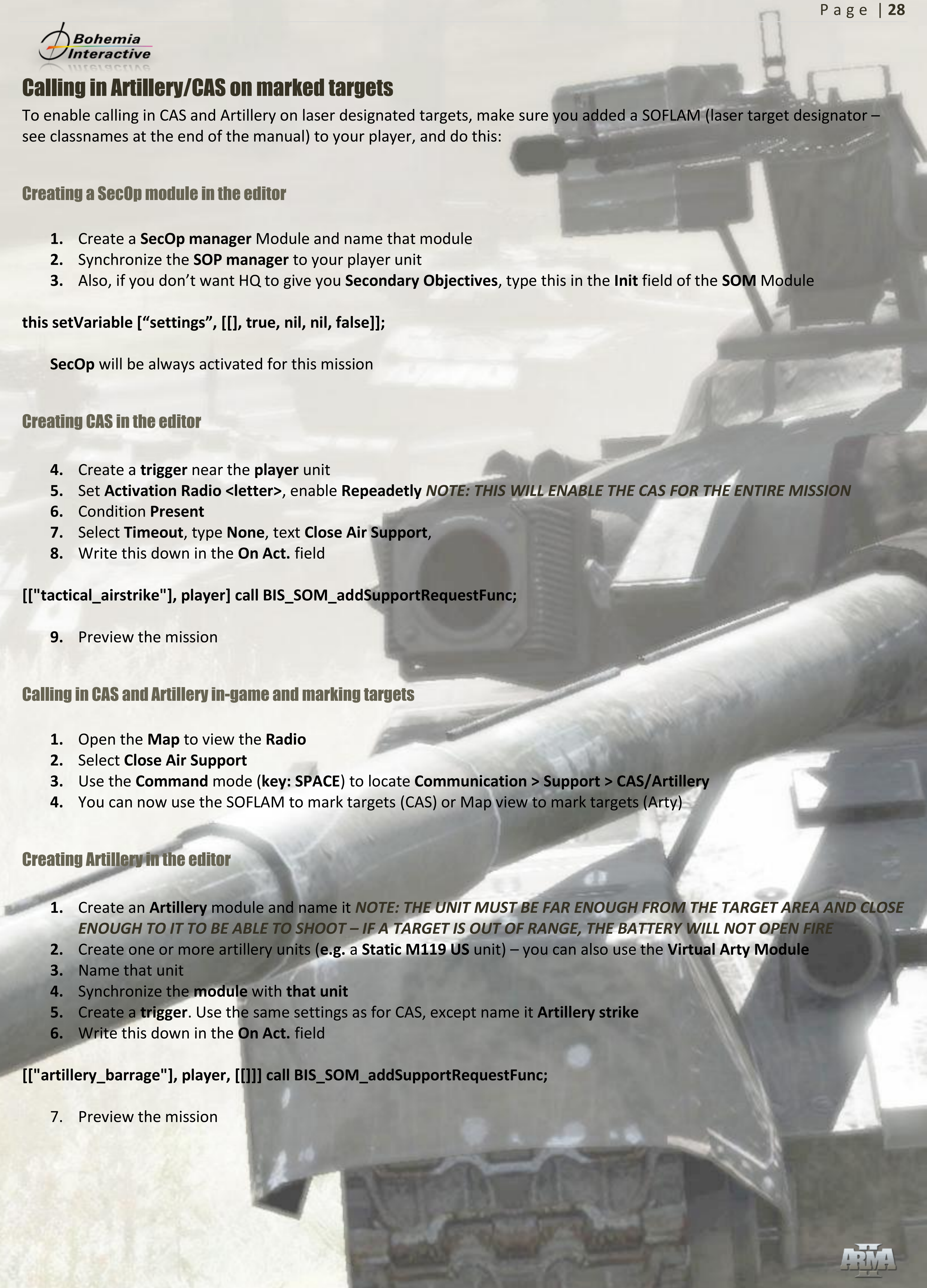


Image 3.21 – This is how predefined CAS looks in-game



Calling in Artillery/CAS on marked targets

To enable calling in CAS and Artillery on laser designated targets, make sure you added a SOFLAM (laser target designator – see classnames at the end of the manual) to your player, and do this:

Creating a SecOp module in the editor

1. Create a **SecOp manager** Module and name that module
2. Synchronize the **SOP manager** to your player unit
3. Also, if you don't want HQ to give you **Secondary Objectives**, type this in the **Init** field of the **SOM** Module

```
this setVariable [“settings”, [[], true, nil, nil, false]];
```

SecOp will be always activated for this mission

Creating CAS in the editor

4. Create a **trigger** near the **player** unit
5. Set **Activation Radio <letter>**, enable **Repeadetly** *NOTE: THIS WILL ENABLE THE CAS FOR THE ENTIRE MISSION*
6. Condition **Present**
7. Select **Timeout**, type **None**, text **Close Air Support**,
8. Write this down in the **On Act.** field

```
[["tactical_airstrike"], player] call BIS_SOM_addSupportRequestFunc;
```

9. Preview the mission

Calling in CAS and Artillery in-game and marking targets

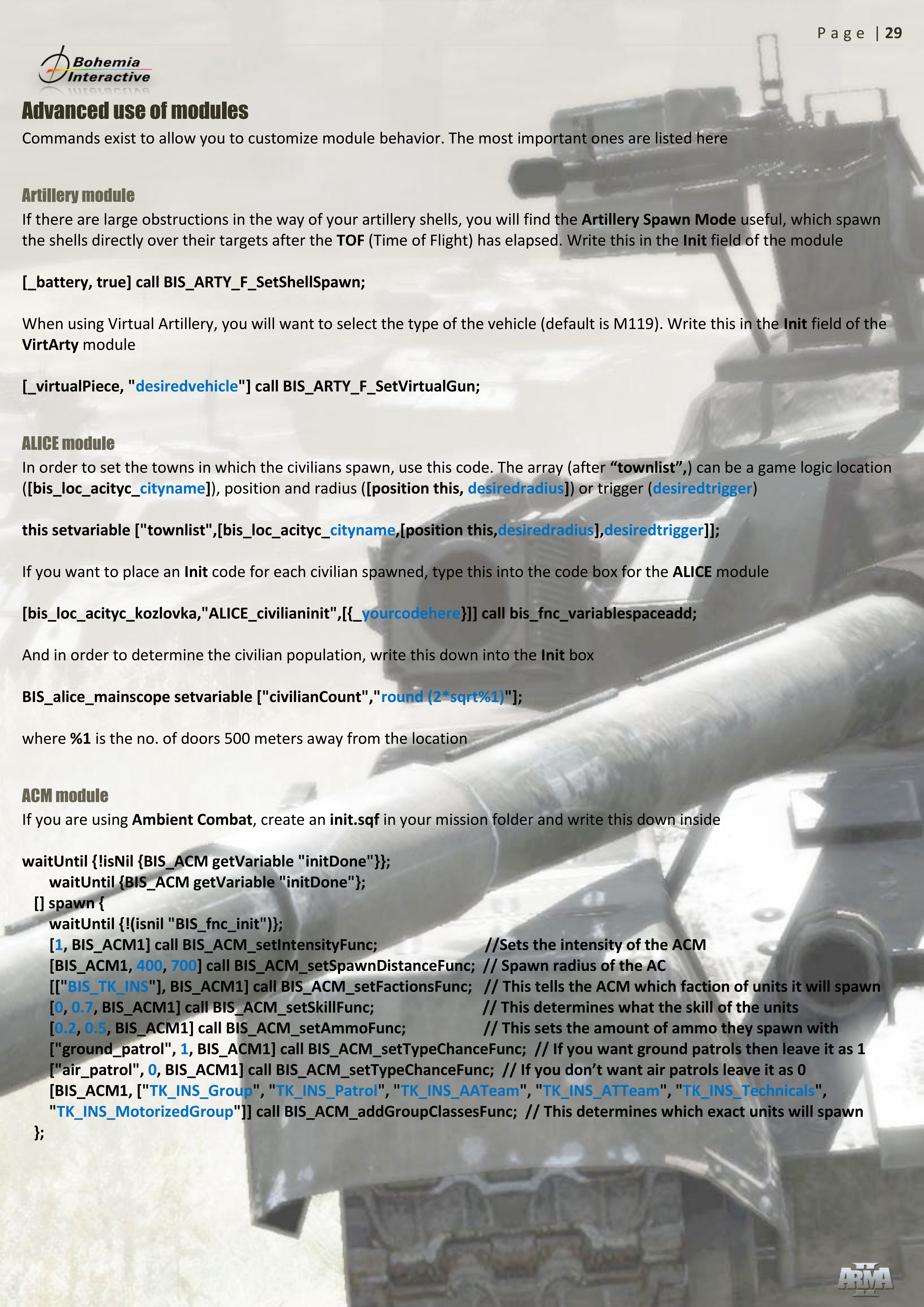
1. Open the **Map** to view the **Radio**
2. Select **Close Air Support**
3. Use the **Command** mode (**key: SPACE**) to locate **Communication > Support > CAS/Artillery**
4. You can now use the SOFLAM to mark targets (CAS) or Map view to mark targets (Arty)

Creating Artillery in the editor

1. Create an **Artillery** module and name it *NOTE: THE UNIT MUST BE FAR ENOUGH FROM THE TARGET AREA AND CLOSE ENOUGH TO IT TO BE ABLE TO SHOOT – IF A TARGET IS OUT OF RANGE, THE BATTERY WILL NOT OPEN FIRE*
2. Create one or more artillery units (**e.g.** a **Static M119 US** unit) – you can also use the **Virtual Arty Module**
3. Name that unit
4. Synchronize the **module** with **that unit**
5. Create a **trigger**. Use the same settings as for CAS, except name it **Artillery strike**
6. Write this down in the **On Act.** field

```
[["artillery_barrage"], player, [[]]] call BIS_SOM_addSupportRequestFunc;
```

7. Preview the mission



Advanced use of modules

Commands exist to allow you to customize module behavior. The most important ones are listed here

Artillery module

If there are large obstructions in the way of your artillery shells, you will find the **Artillery Spawn Mode** useful, which spawn the shells directly over their targets after the **TOF** (Time of Flight) has elapsed. Write this in the **Init** field of the module

```
[_battery, true] call BIS_ARTY_F_SetShellSpawn;
```

When using Virtual Artillery, you will want to select the type of the vehicle (default is M119). Write this in the **Init** field of the **VirtArty** module

```
[_virtualPiece, "desiredvehicle"] call BIS_ARTY_F_SetVirtualGun;
```

ALICE module

In order to set the towns in which the civilians spawn, use this code. The array (after **"townlist",**) can be a game logic location (**[bis_loc_acityc_cityname]**), position and radius (**[position this, desiredradius]**) or trigger (**desiredtrigger**)

```
this setvariable ["townlist",[bis_loc_acityc_cityname,[position this,desiredradius],desiredtrigger]];
```

If you want to place an **Init** code for each civilian spawned, type this into the code box for the **ALICE** module

```
[bis_loc_acityc_kozlovka,"ALICE_civilianinit",[{_yourcodehere}]] call bis_fnc_variablespaceadd;
```

And in order to determine the civilian population, write this down into the **Init** box

```
BIS_alice_mainscope setvariable ["civilianCount","round (2*sqrt%1)"];
```

where **%1** is the no. of doors 500 meters away from the location

ACM module

If you are using **Ambient Combat**, create an **init.sqf** in your mission folder and write this down inside

```
waitUntil {!isNil {BIS_ACM getVariable "initDone"}};
    waitUntil {BIS_ACM getVariable "initDone"};
  [] spawn {
    waitUntil {!(isnil "BIS_fnc_init")};
    [1, BIS_ACM1] call BIS_ACM_setIntensityFunc;                    //Sets the intensity of the ACM
    [BIS_ACM1, 400, 700] call BIS_ACM_setSpawnDistanceFunc;  // Spawn radius of the AC
    [["BIS_TK_INS"], BIS_ACM1] call BIS_ACM_setFactionsFunc;   // This tells the ACM which faction of units it will spawn
    [0, 0.7, BIS_ACM1] call BIS_ACM_setSkillFunc;                   // This determines what the skill of the units
    [0.2, 0.5, BIS_ACM1] call BIS_ACM_setAmmoFunc;                 // This sets the amount of ammo they spawn with
    ["ground_patrol", 1, BIS_ACM1] call BIS_ACM_setTypeChanceFunc;  // If you want ground patrols then leave it as 1
    ["air_patrol", 0, BIS_ACM1] call BIS_ACM_setTypeChanceFunc;  // If you don't want air patrols leave it as 0
    [BIS_ACM1, ["TK_INS_Group", "TK_INS_Patrol", "TK_INS_AATeam", "TK_INS_ATTeam", "TK_INS_Technicals",
    "TK_INS_MotorizedGroup"]] call BIS_ACM_addGroupClassesFunc;  // This determines which exact units will spawn
  };
```

HALO jumps

Write this down in the **Init** field of a unit or **On Act.** field of a trigger/waypoint

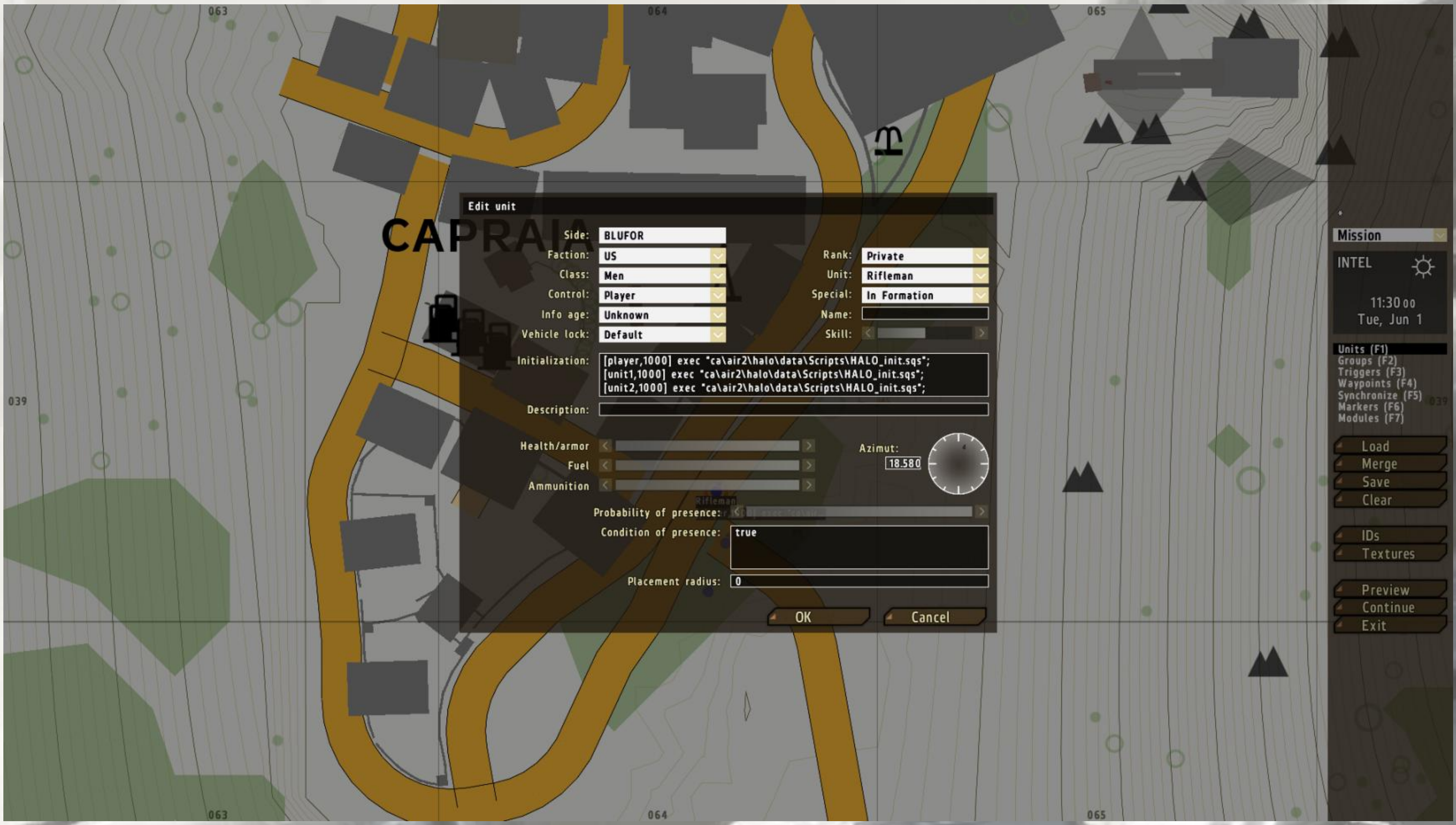
[player,halojumpheight] exec "ca\air2\halo\data\Scripts\HALO_init.sqs";

and if deploying from an aircraft, write this down into the aircraft's **Init** field

this setpos [getpos this select 0, getpos this select 1,(getpos this select 2) +halojumpheight]; this flyInHeight halojumpheight

where **player** can also be the name of an **AI unit** that will deploy the chute at 200 meters.

NOTE: IF DEPLOYING FROM AN AIRCRAFT, THERE IS A HIGH POSSIBILITY THAT THE UNITS WILL ISSUE A "BOARD THAT AIRPLANE" ORDER ON EXIT. CREATE A TRIGGER TO SWITCH THEM BACK TO THEIR ORIGINAL OBJECTIVE. ALSO NOTE THAT AI UNITS WILL PERFORM UNEXPECTED MANOUVRES WHILE IN FLIGHT, AND BE PREPARED TO USE WASD WHILE FLYING



Side:	BLUFOR		
Faction:	US	Rank:	Private
Class:	Men	Unit:	Rifleman
Control:	Player	Special:	In Formation
Info age:	Unknown	Name:	
Vehicle lock:	Default	Skill:	

Initialization:
```
[player,1000] exec "ca\air2\halo\data\Scripts\HALO_init.sqs";
[unit1,1000] exec "ca\air2\halo\data\Scripts\HALO_init.sqs";
[unit2,1000] exec "ca\air2\halo\data\Scripts\HALO_init.sqs";
```

Description:

Health/armor

Fuel

Ammunition

Azimut: 18.580

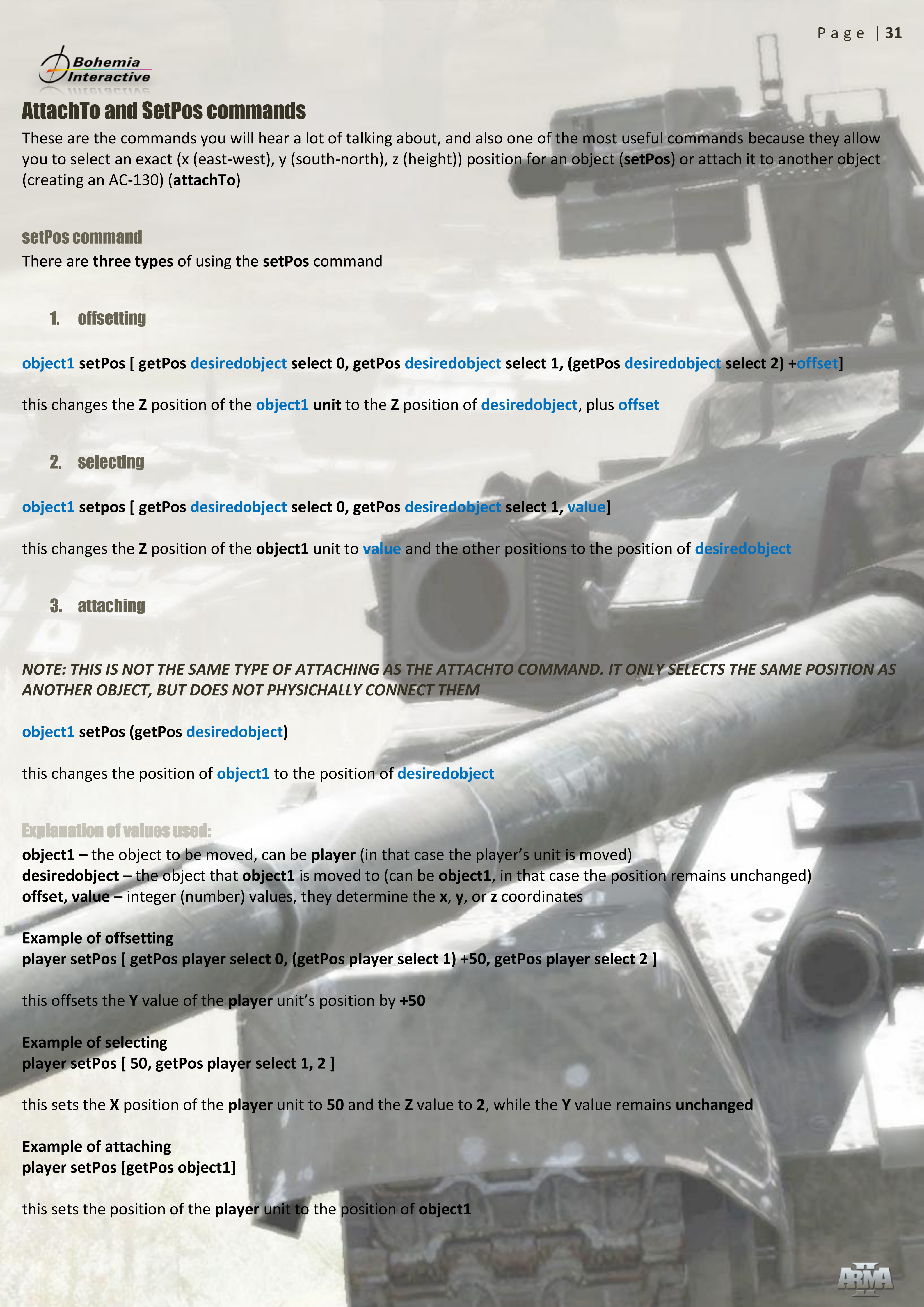
Probability of presence:

Condition of presence: true

Placement radius: 0

Image 3.5 – A HALO script execute code in a unit's Init field. This sends the player, and unit1 and unit2 on a HALO jump

Image 3.51 – HALO jump landing with AI units



AttachTo and SetPos commands

These are the commands you will hear a lot of talking about, and also one of the most useful commands because they allow you to select an exact (x (east-west), y (south-north), z (height)) position for an object (**setPos**) or attach it to another object (creating an AC-130) (**attachTo**)

setPos command

There are **three types** of using the **setPos** command

1. offsetting

```
object1 setPos [ getPos desiredobject select 0, getPos desiredobject select 1, (getPos desiredobject select 2) +offset]
```

this changes the **Z** position of the **object1 unit** to the **Z** position of **desiredobject**, plus **offset**

2. selecting

```
object1 setpos [ getPos desiredobject select 0, getPos desiredobject select 1, value]
```

this changes the **Z** position of the **object1** unit to **value** and the other positions to the position of **desiredobject**

3. attaching

NOTE: THIS IS NOT THE SAME TYPE OF ATTACHING AS THE ATTACHTO COMMAND. IT ONLY SELECTS THE SAME POSITION AS ANOTHER OBJECT, BUT DOES NOT PHYSICHALLY CONNECT THEM

```
object1 setPos (getPos desiredobject)
```

this changes the position of **object1** to the position of **desiredobject**

Explanation of values used:

object1 – the object to be moved, can be **player** (in that case the player's unit is moved)
desiredobject – the object that **object1** is moved to (can be **object1**, in that case the position remains unchanged)
offset, value – integer (number) values, they determine the **x**, **y**, or **z** coordinates

Example of offsetting

```
player setPos [ getPos player select 0, (getPos player select 1) +50, getPos player select 2 ]
```

this offsets the **Y** value of the **player** unit's position by **+50**

Example of selecting

```
player setPos [ 50, getPos player select 1, 2 ]
```

this sets the **X** position of the **player** unit to **50** and the **Z** value to **2**, while the **Y** value remains **unchanged**

Example of attaching

```
player setPos [getPos object1]
```

this sets the position of the **player** unit to the position of **object1**

attachTo command

This command may seem to be simpler to use, but it has an instance of the **setPos** command implemented in the shape of the **offset box** ([value, value, value]) to gain some interesting effects. You can use the **setDir** command to rotate the attached object relative to the attached-to object's heading

```
object1 attachTo [desiredobject,[ value, value, value]];
```

NOTE: THE POSITION ARRAY WORKS SAME AS IN THE SETPOS COMMAND (NOT TESTED)

this attaches **object1** to **desiredobject** at positions **X** = **value**, **Y** = value, **Z** = value relative to the position of **desiredobject**

detach command

object1 can be detached from **desiredobject** using the **detach** command like this

```
detach object1
```

NOTE: WHEN USING THE ATTACHTO COMMAND YOU WILL GO THROUGH A LOT OF TRIAL AND ERROR BEFORE YOU FIND OUT THE CORRECT OFFSET FOR PLACEMENT OF THE OBJECT WITHOUT GETTING STRANGE SHAPES, AS SEEN IN IMAGE 3.61

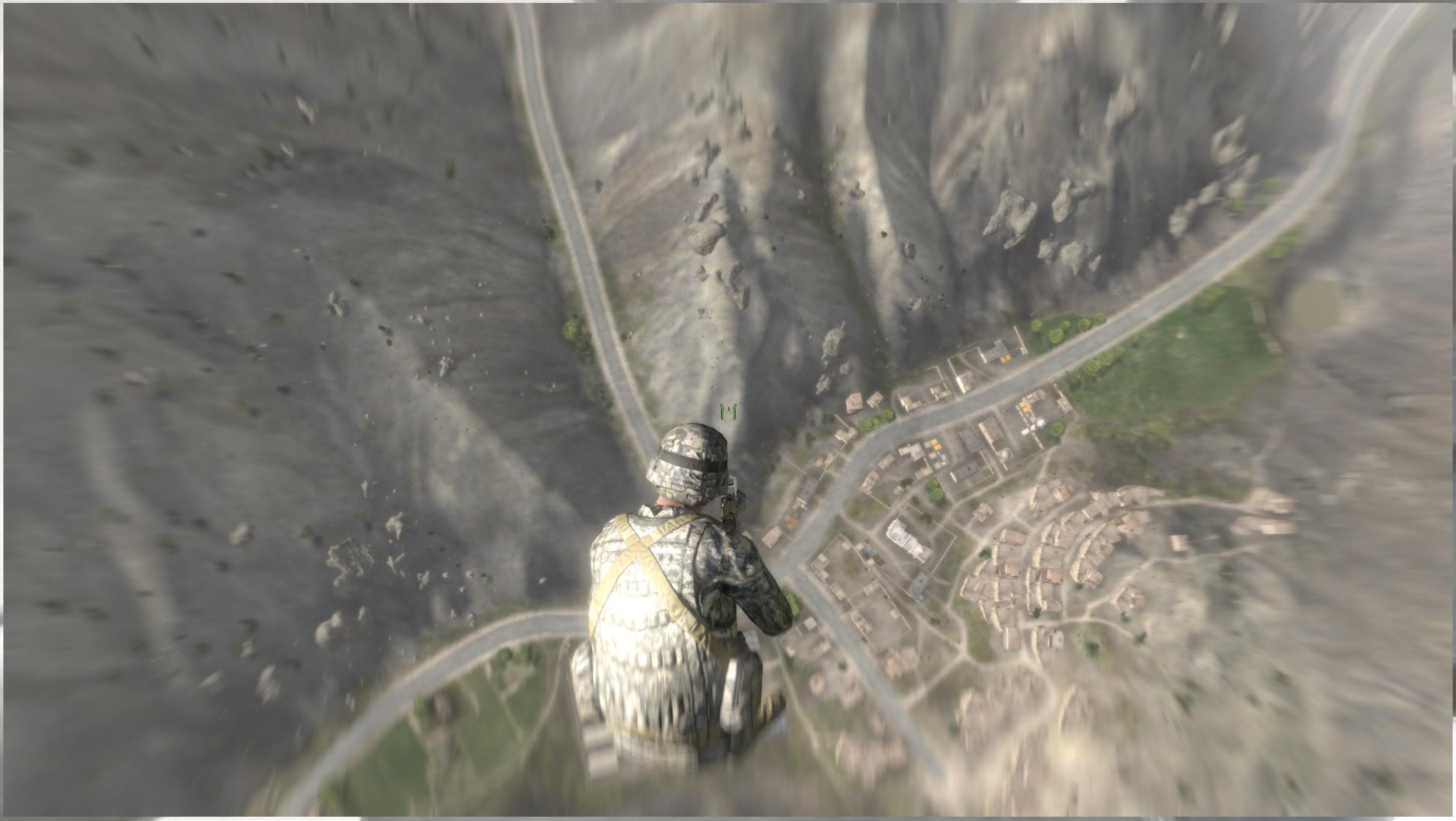
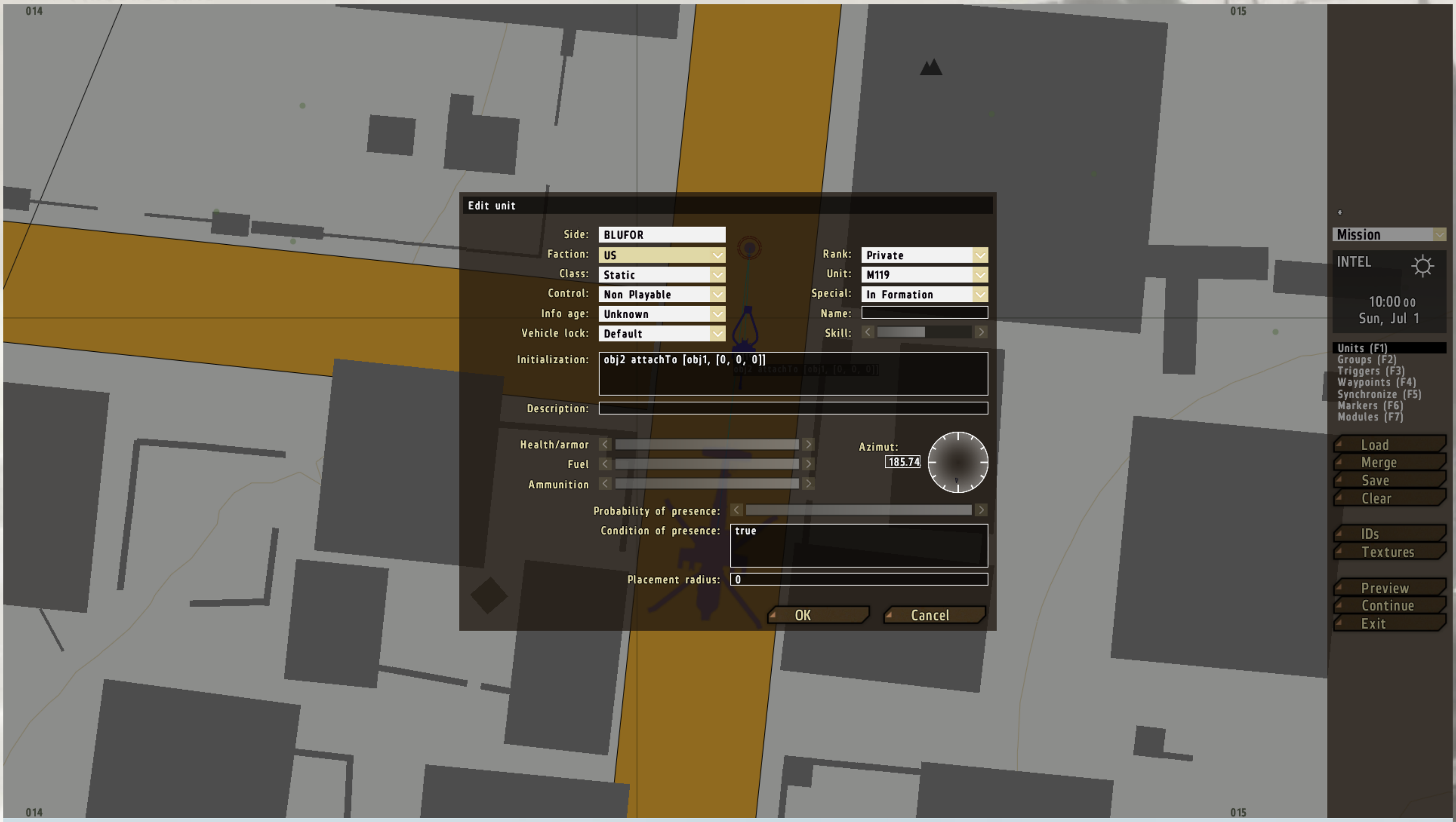


Image 3.6 – The setPos command enables you to spawn units in various locations like rooftops, buildings, balconies, even thin air, as can be seen here

We can see in Image 3.6 that the player is falling from a high altitude. This was achieved with the **setPos** command. This is how it looked like

```
player setPos [getPos player select 0, getPos player select 1, (getPos player select 2) +1000]
```

that way the player was spawned on an altitude of 1000 meters


Edit unit

Side:	BLUFOR		
Faction:	US	Rank:	Private
Class:	Static	Unit:	M119
Control:	Non Playable	Special:	In Formation
Info age:	Unknown	Name:	
Vehicle lock:	Default	Skill:	

Initialization:
```
obj2 attachTo [obj1, [0, 0, 0]]
```

Description:

Health/armor

Fuel

Ammunition

Azimut: 185.74

Probability of presence:

Condition of presence: true

Placement radius: 0

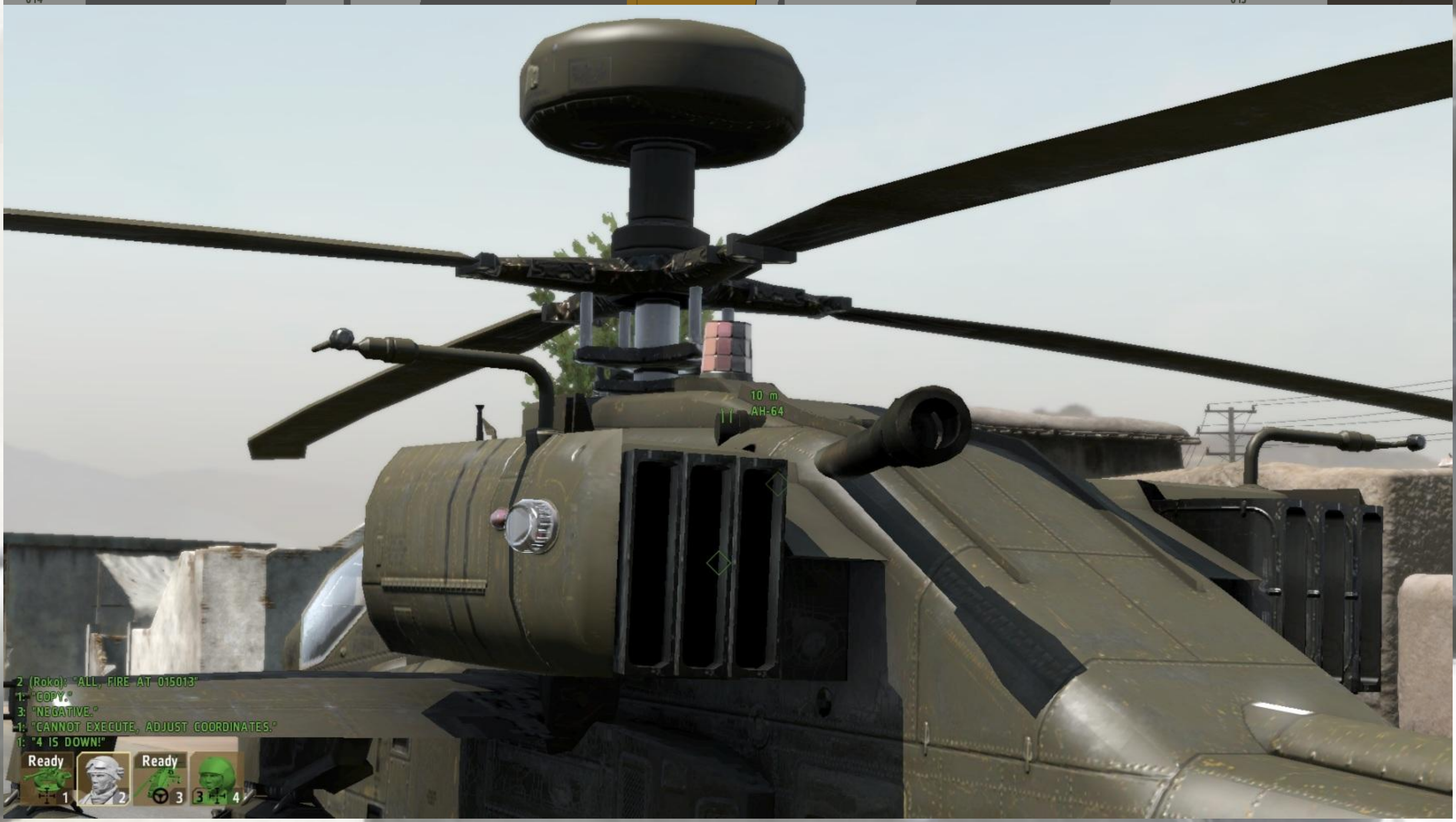

```
2 (Roko): "ALL, FIRE AT 015013"
1: "COPY."
3: "NEGATIVE."
1: "CANNOT EXECUTE, ADJUST COORDINATES."
1: "4 IS DOWN!"
```

Image 3.61 – The attachTo command used on an M119 artillery piece and an AH-64D Apache helicopter

Briefings

In order to have a complete mission, you'll need a briefing. To create a briefing you will need the following external text files

- **init.sqf**
- **briefing.sqf**

Init.sqf

In **init.sqf**, type

execVM "briefing.sqf";

if(true) exitWith ();

this executes the **briefing.sqf** file once the mission is ran

Briefing.sqf

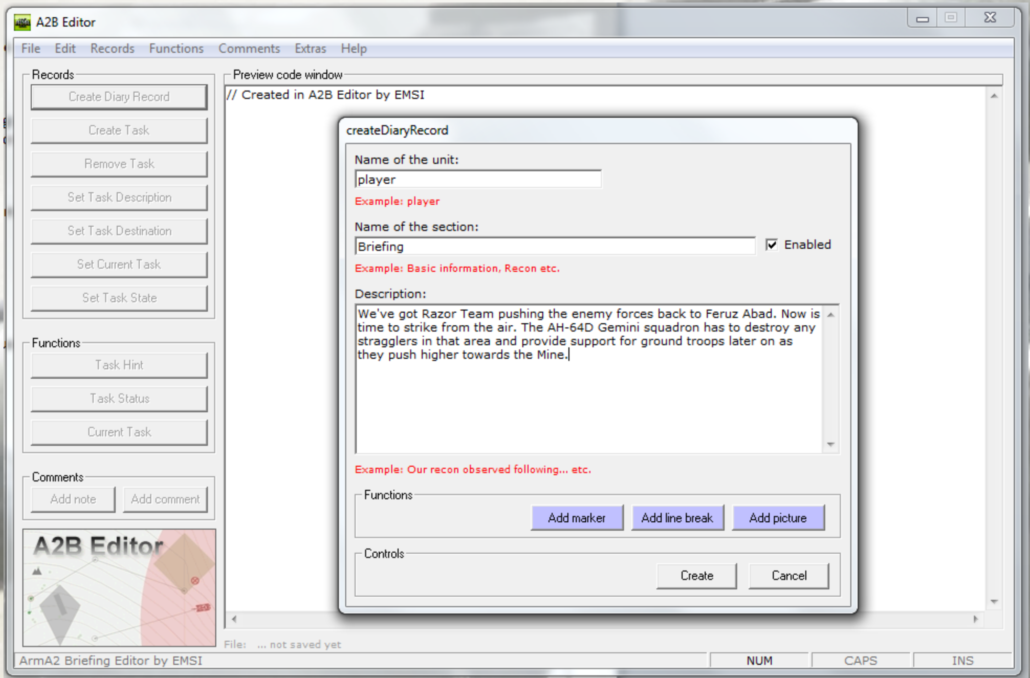
The easier way to create briefings is to download **A2B Editor** (available on armaholic) and use it to create the **briefing.sqf** with all the information you need (it creates **templates** to edit). That's what we'll cover here

Creating the basic entries

The **diary entries** the campaign missions have are as follows

- **Briefing**
- **Situation**
- **Mission**
- **Execution**
- **Support**

In order to add them into **A2B**, use the **Create Diary Record** button



// Created in A2B Editor by EMSI

Name of the unit: player

Example: player

Name of the section: Briefing ☑ Enabled

Example: Basic information, Recon etc.

Description:

We've got Razor Team pushing the enemy forces back to Feruz Abad. Now is time to strike from the air. The AH-64D Gemini squadron has to destroy any stragglers in that area and provide support for ground troops later on as they push higher towards the Mine.

Example: Our recon observed following... etc.

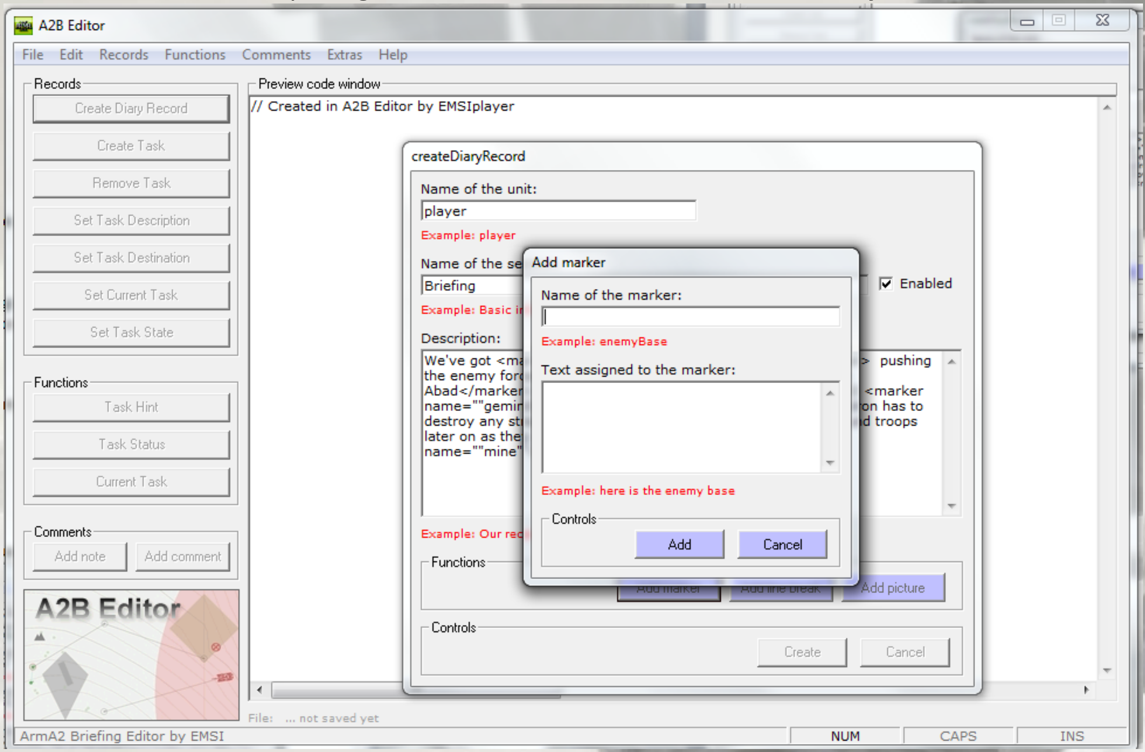
File: ... not saved yet

ArmA2 Briefing Editor by EMSI

Image 3.7 – Adding a Briefing diary record that will be displayed in the Notes section of the Map view and mission briefing

After that you have to change any key points in that text into **Markers** that will show the positions of objectives when pressed. Those are the markers you've set up in the **Editor** while creating your mission. Also, create **Markers** for your **Tasks** (**objectives**) so that you can determine whether the mission is **complete**.

Add markers into **A2B** by using the **Add marker** button in the **createDiaryRecord** window



// Created in A2B Editor by EMSIplayer

Name of the unit: player

Example: player

Name of the marker:

Example: enemyBase

Text assigned to the marker:

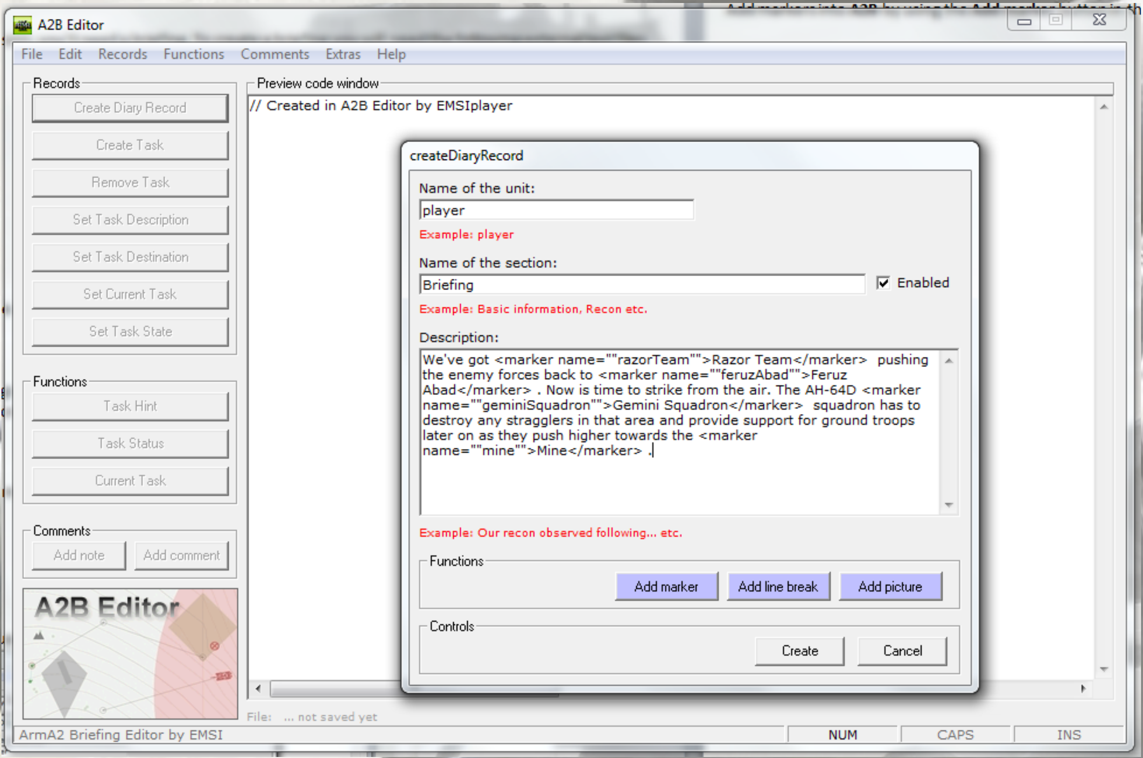
Example: here is the enemy base

File: ... not saved yet

ArmA2 Briefing Editor by EMSI

Image 3.71 – The Add marker dialogue

To successfully add the marker, you will have to input the **Name** of the **Marker** that is stated in the **Editor**. If you haven't placed any markers in the mission, **now would be a good time**. In the **Name** field of the marker, write down a name. Then, in the **A2B** editor under **Name of the marker:** type your marker's name from the **Name** field



// Created in A2B Editor by EMSIplayer

Name of the unit: player

Example: player

Name of the section: Briefing

Example: Basic information, Recon etc.

Description:

```
We've got <marker name=""razorTeam"">Razor Team</marker>  pushing
the enemy forces back to <marker name=""feruzAbad"">Feruz
Abad</marker> . Now is time to strike from the air. The AH-64D <marker
name=""geminiSquadron"">Gemini Squadron</marker>  squadron has to
destroy any stragglers in that area and provide support for ground troops
later on as they push higher towards the <marker
name=""mine"">Mine</marker> .
```

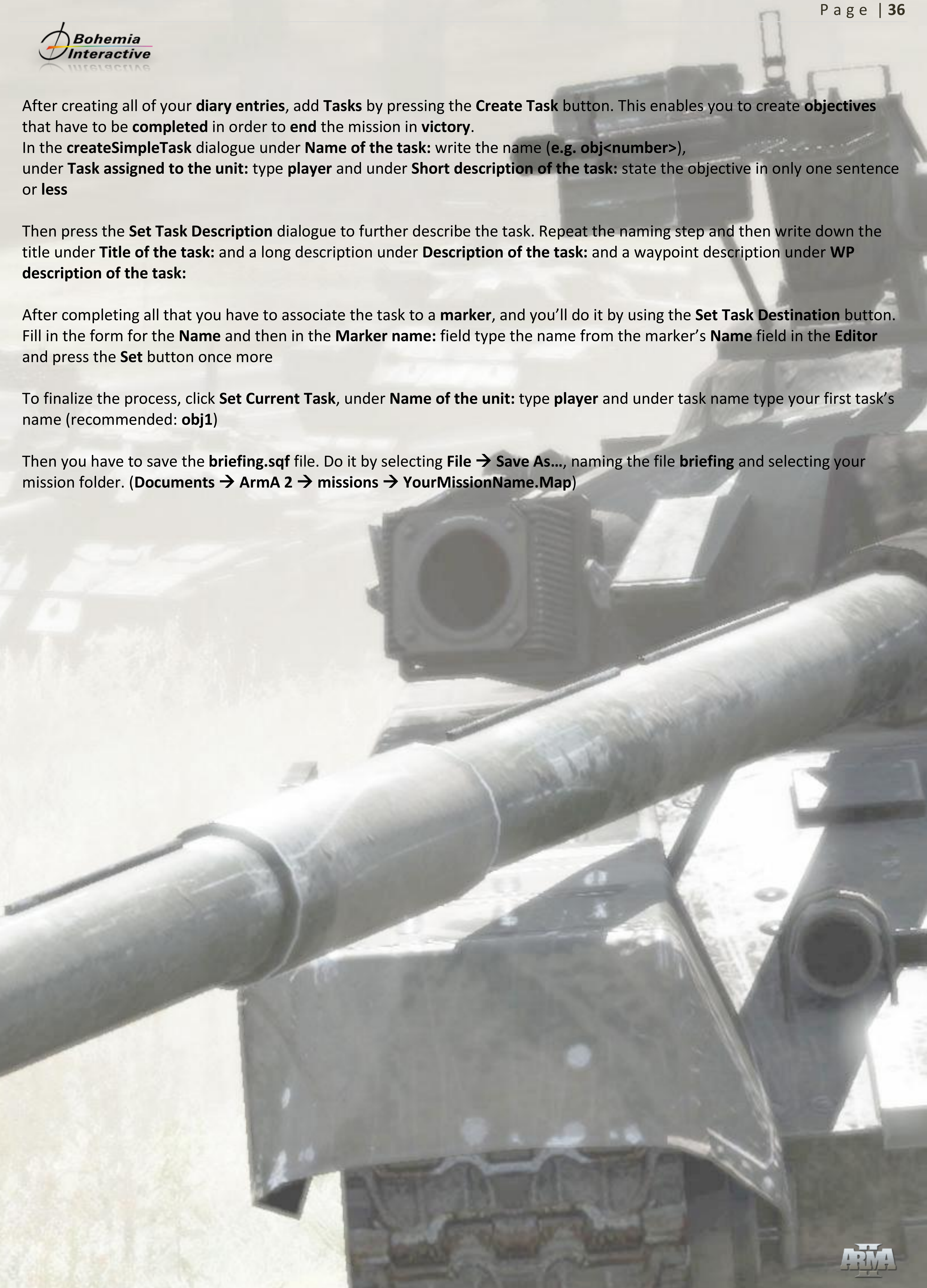
Example: Our recon observed following... etc.

File: ... not saved yet

ArmA2 Briefing Editor by EMSI

Image 3.72 – The Briefing entry with the markers added

In order to keep things organized, add line breaks (basically like pressing **Enter** (**Return**) on the keyboard) by pressing the **Add line break** button



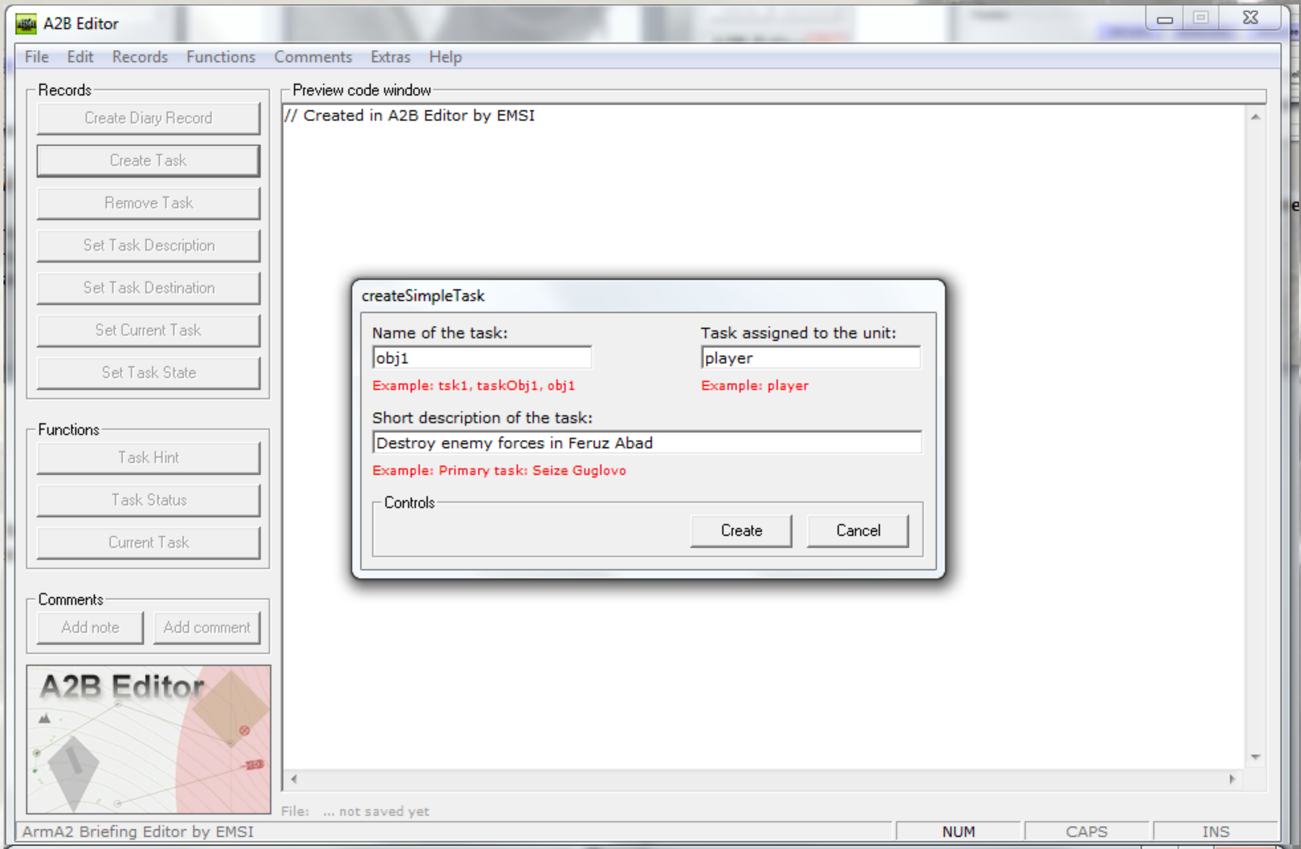
After creating all of your **diary entries**, add **Tasks** by pressing the **Create Task** button. This enables you to create **objectives** that have to be **completed** in order to **end** the mission in **victory**.
In the **createSimpleTask** dialogue under **Name of the task:** write the name (**e.g. obj<number>**),
under **Task assigned to the unit:** type **player** and under **Short description of the task:** state the objective in only one sentence or **less**

Then press the **Set Task Description** dialogue to further describe the task. Repeat the naming step and then write down the title under **Title of the task:** and a long description under **Description of the task:** and a waypoint description under **WP description of the task:**

After completing all that you have to associate the task to a **marker**, and you'll do it by using the **Set Task Destination** button. Fill in the form for the **Name** and then in the **Marker name:** field type the name from the marker's **Name** field in the **Editor** and press the **Set** button once more

To finalize the process, click **Set Current Task**, under **Name of the unit:** type **player** and under task name type your first task's name (recommended: **obj1**)

Then you have to save the **briefing.sqf** file. Do it by selecting **File → Save As…**, naming the file **briefing** and selecting your mission folder. (**Documents → ArmA 2 → missions → YourMissionName.Map**)

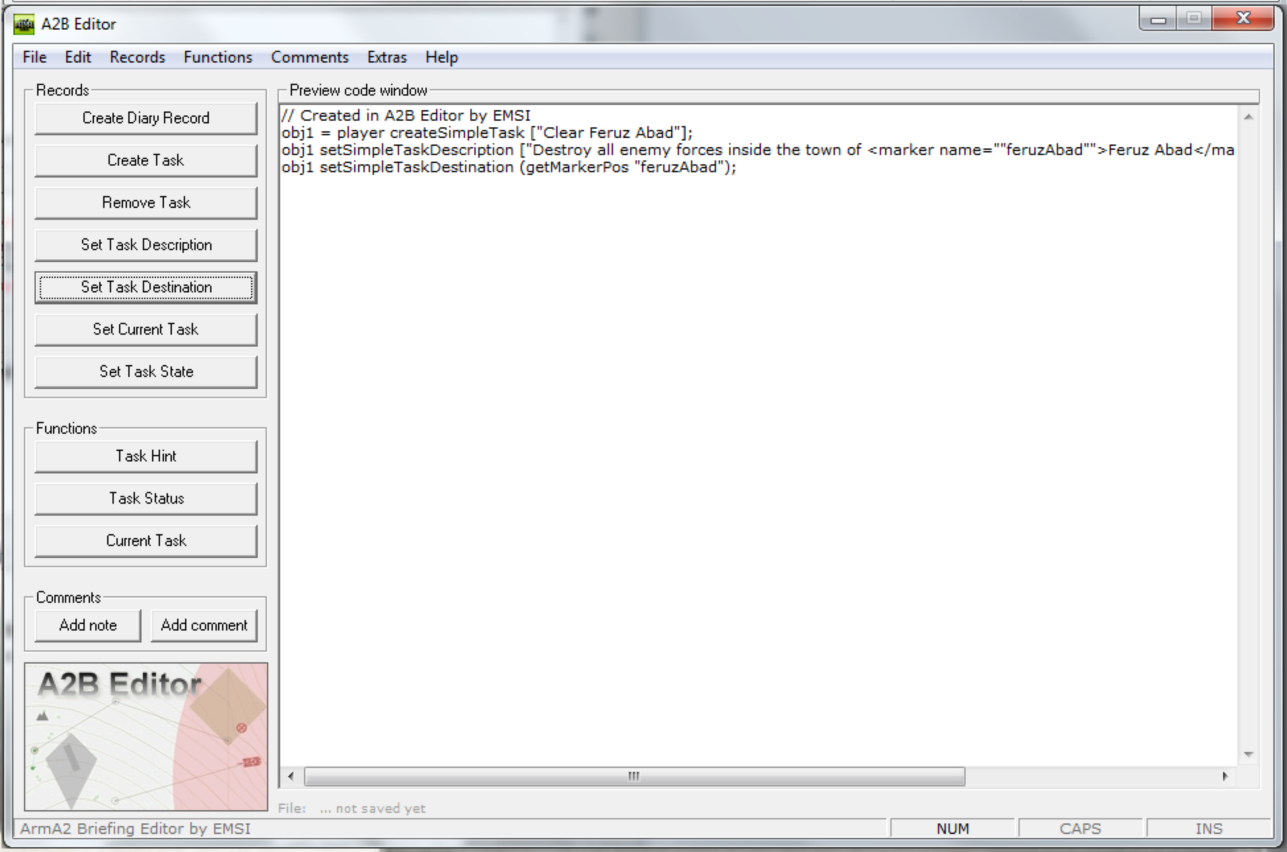


```
// Created in A2B Editor by EMSI
```

Name of the task: obj1
Example: tsk1, taskObj1, obj1

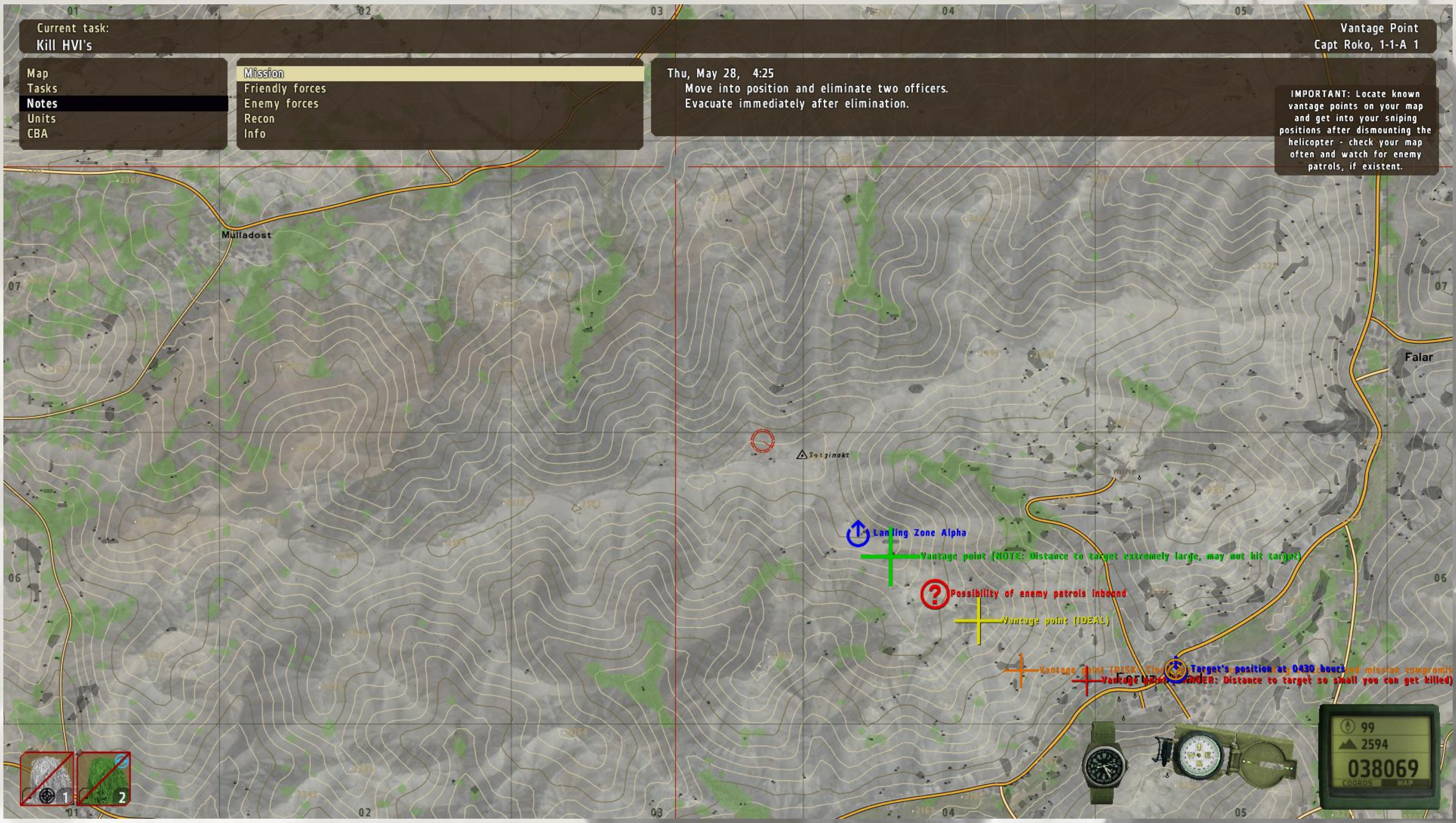
Task assigned to the unit: player
Example: player

Short description of the task: Destroy enemy forces in Feruz Abad
Example: Primary task: Seize Guglovo



```
// Created in A2B Editor by EMSI
obj1 = player createSimpleTask ["Clear Feruz Abad"];
obj1 setSimpleTaskDescription ["Destroy all enemy forces inside the town of <marker name=""feruzAbad"">Feruz Abad</ma
obj1 setSimpleTaskDestination (getMarkerPos "feruzAbad");
```

Image 3.73 – Creating a task


Current task:
Kill HVI's

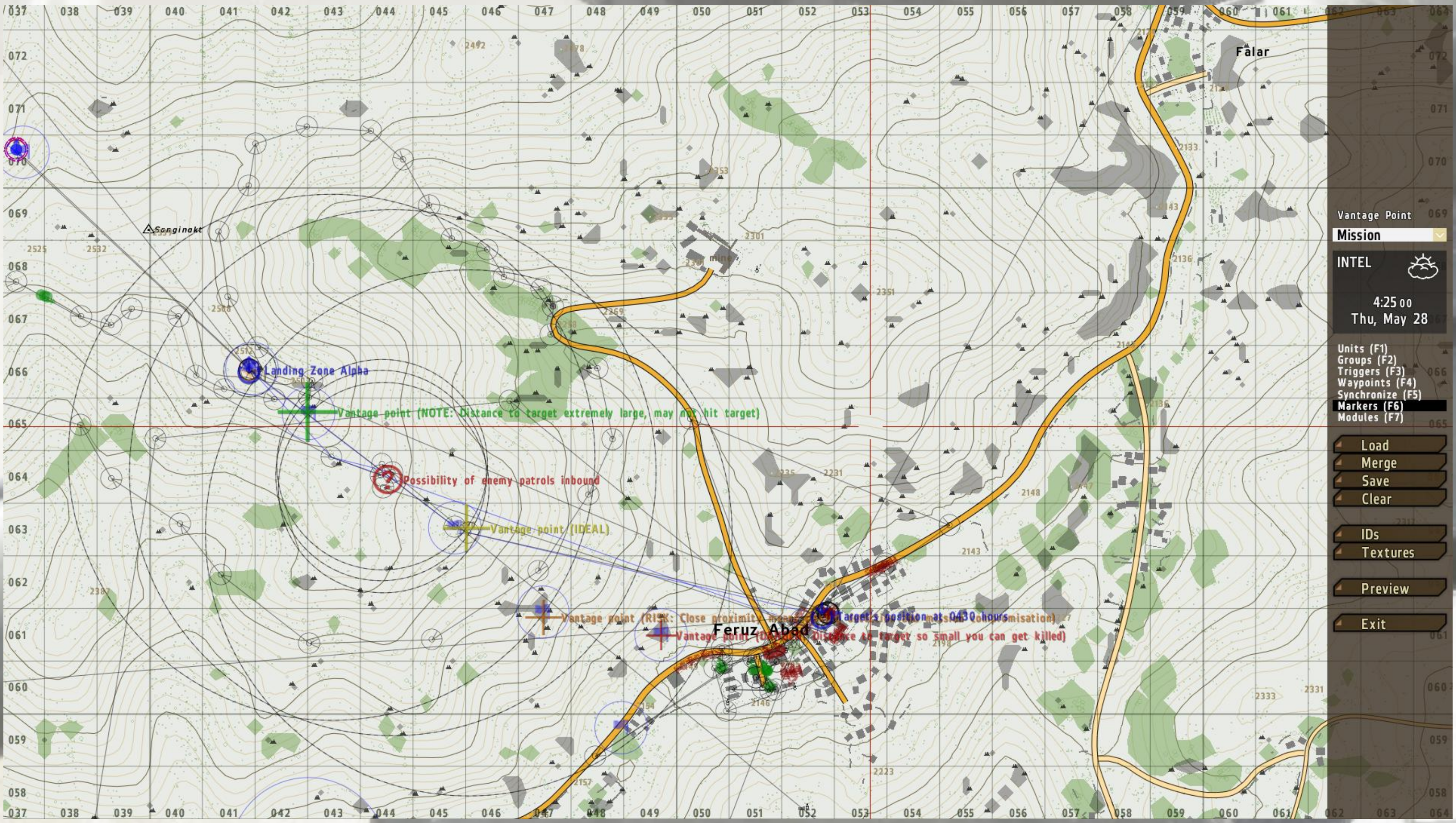
Vantage Point
Capt Roko, 1-1-A 1

Thu, May 28, 4:25
Move into position and eliminate two officers.
Evacuate immediately after elimination.

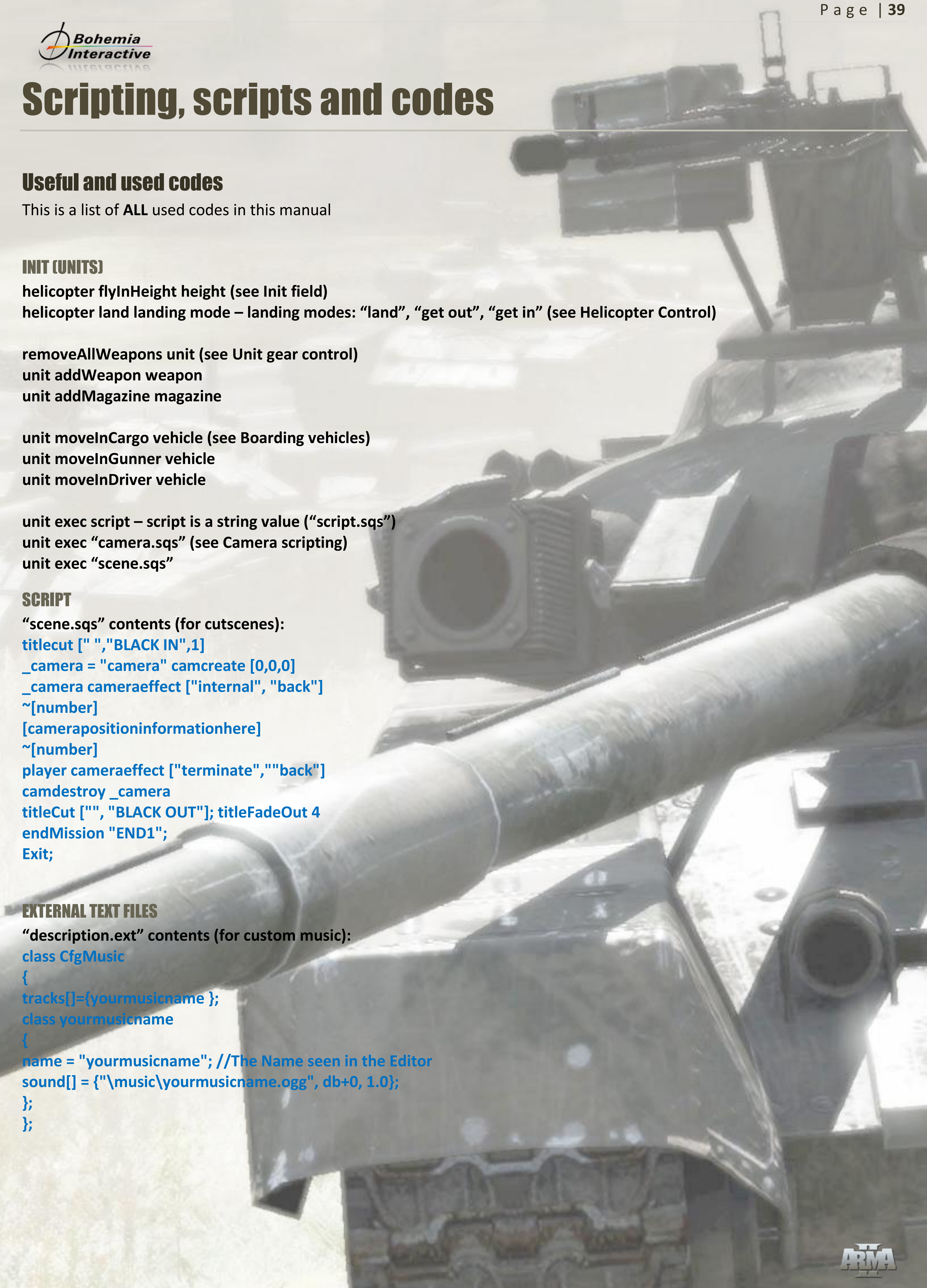
IMPORTANT: Locate known vantage points on your map and get into your sniping positions after dismounting the helicopter - check your map often and watch for enemy patrols, if existent.

Map
Tasks
Notes
Units
CBA
Mission
Friendly forces
Enemy forces
Recon
Info
Mulladost
Falar
Sanginakt
mine
Landing Zone Alpha
Vantage point (NOTE: Distance to target extremely large, may not hit target)
Possibility of enemy patrols inbound
Vantage point (IDEAL)
Vantage point (RISK: Close proximity means ... and mission compromis
Target's position at 0430 hours
Vantage point (DANGER: Distance to target so small you can get killed)
99
2594
038069
COORDS
MAP

This is how the Map view of a mission with a completed briefing looks like (mission: ***Vantage Point*** *by AFOF_Murdock (****WIP****))*


Falar
Sanginakt
mine
Landing Zone Alpha
Vantage point (NOTE: Distance to target extremely large, may not hit target)
Possibility of enemy patrols inbound
Vantage point (IDEAL)
Vantage point (RISK: Close proximity means ... mission compromisation)
Target's position at 0430 hours
Vantage point (DANGER: Distance to target so small you can get killed)
Feruz Abad
Vantage Point
Mission
INTEL
4:25 00
Thu, May 28
Units (F1)
Groups (F2)
Triggers (F3)
Waypoints (F4)
Synchronize (F5)
Markers (F6)
Modules (F7)
Load
Merge
Save
Clear
IDs
Textures
Preview
Exit

This is how a finished mission looks like when viewed from the Editor



Scripting, scripts and codes

Useful and used codes

This is a list of **ALL** used codes in this manual

INIT (UNITS)

helicopter flyInHeight height (see Init field)
helicopter land landing mode – landing modes: “land”, “get out”, “get in” (see Helicopter Control)

removeAllWeapons unit (see Unit gear control)
unit addWeapon weapon
unit addMagazine magazine

unit moveInCargo vehicle (see Boarding vehicles)
unit moveInGunner vehicle
unit moveInDriver vehicle

unit exec script – script is a string value (“script.sqs”)
unit exec “camera.sqs” (see Camera scripting)
unit exec “scene.sqs”

SCRIPT

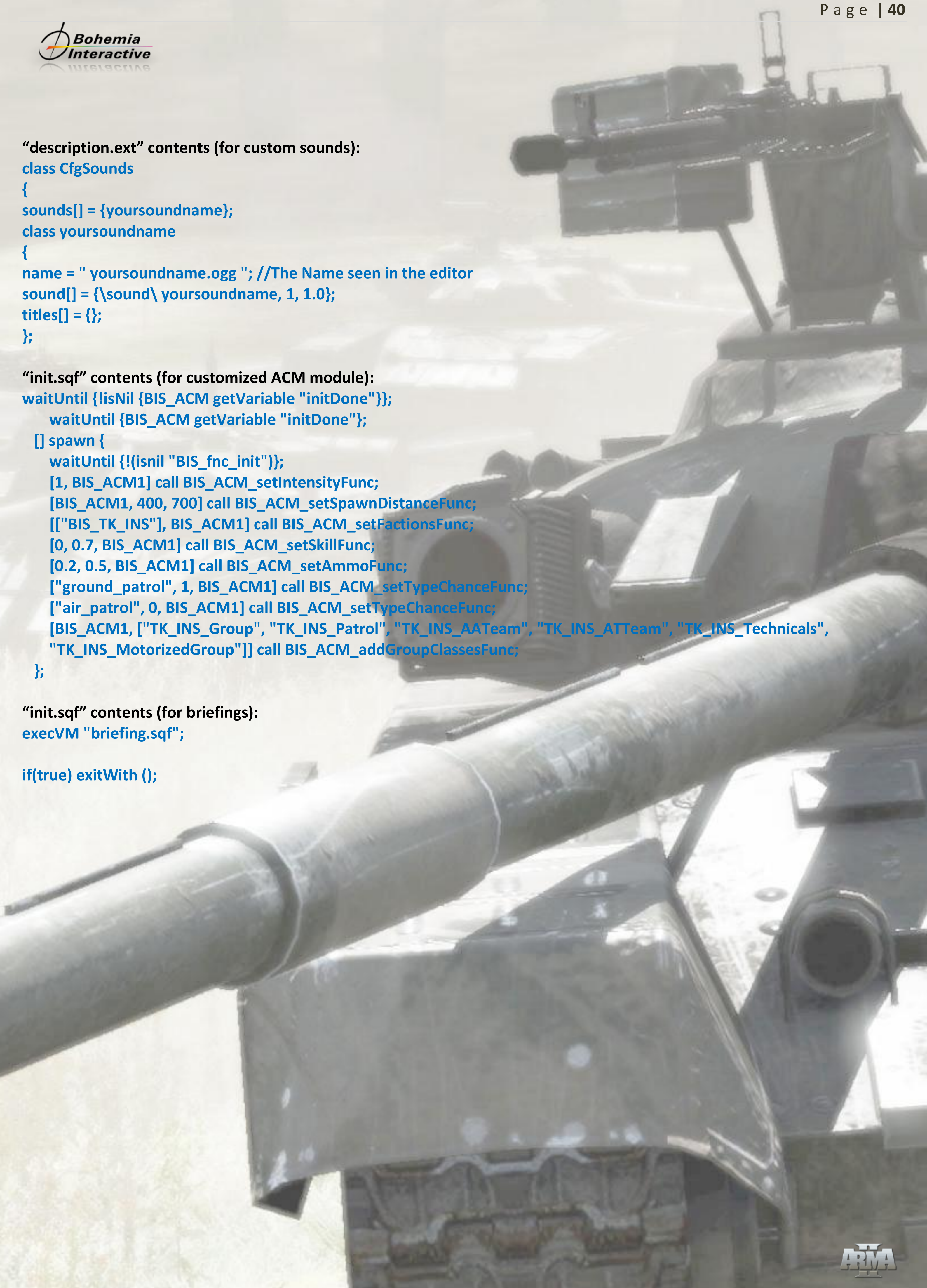
“scene.sqs” contents (for cutscenes):

```
titlecut [" ","BLACK IN",1]
_camera = "camera" camcreate [0,0,0]
_camera cameraeffect ["internal", "back"]
~[number]
[camerapositioninformationhere]
~[number]
player cameraeffect ["terminate",""back"]
camdestroy _camera
titleCut ["", "BLACK OUT"]; titleFadeOut 4
endMission "END1";
Exit;
```

EXTERNAL TEXT FILES

“description.ext” contents (for custom music):

```
class CfgMusic
{
tracks[]={yourmusicname };
class yourmusicname
{
name = "yourmusicname"; //The Name seen in the Editor
sound[] = {"\music\yourmusicname.ogg", db+0, 1.0};
};
};
```



“description.ext” contents (for custom sounds):

```
class CfgSounds
{
sounds[] = {yoursoundname};
class yoursoundname
{
name = " yoursoundname.ogg "; //The Name seen in the editor
sound[] = {\sound\ yoursoundname, 1, 1.0};
titles[] = {};
};
```

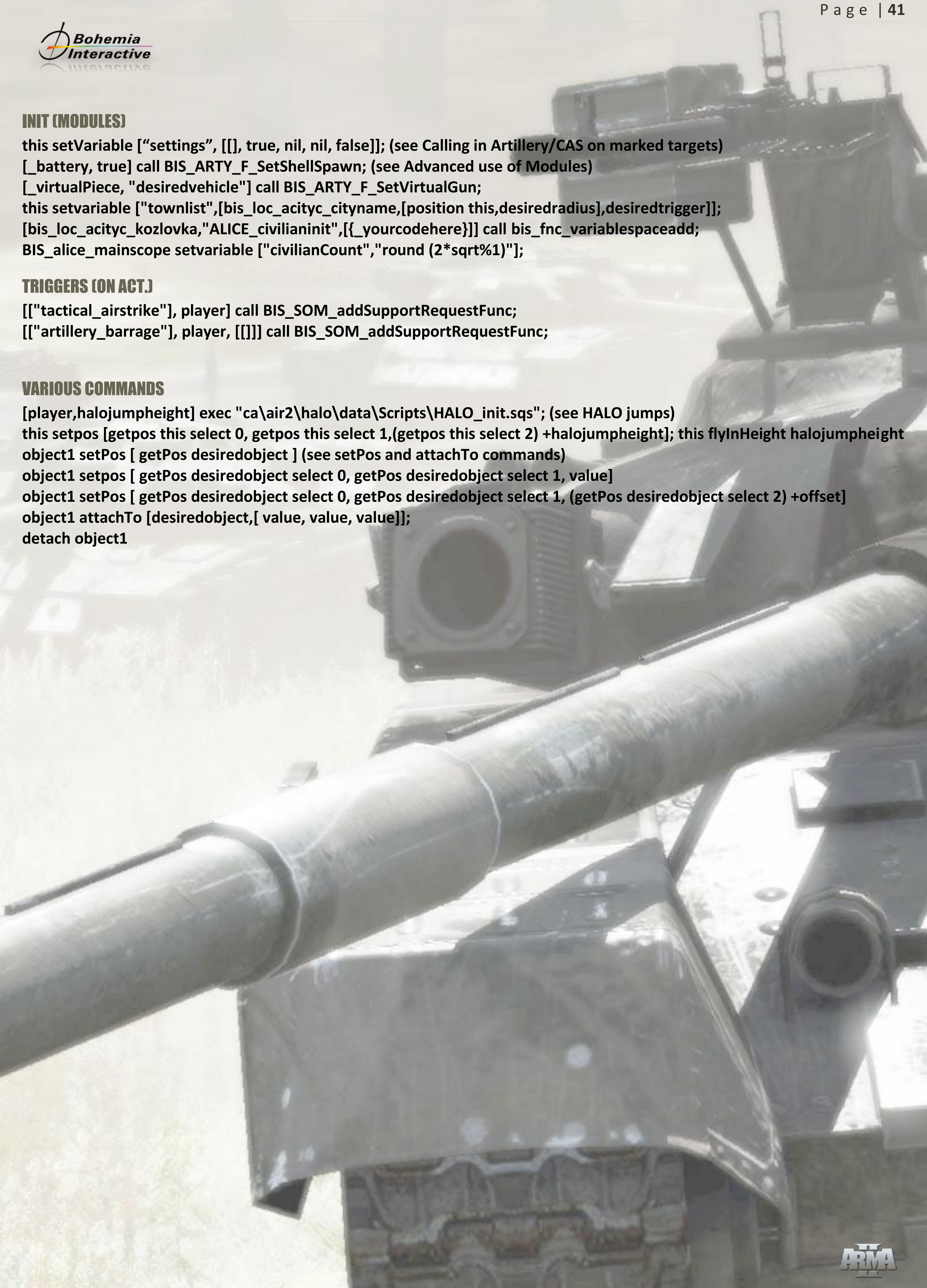
“init.sqf” contents (for customized ACM module):

```
waitUntil {!isNil {BIS_ACM getVariable "initDone"}};
    waitUntil {BIS_ACM getVariable "initDone"};
  [] spawn {
    waitUntil {!(isnil "BIS_fnc_init")};
    [1, BIS_ACM1] call BIS_ACM_setIntensityFunc;
    [BIS_ACM1, 400, 700] call BIS_ACM_setSpawnDistanceFunc;
    [["BIS_TK_INS"], BIS_ACM1] call BIS_ACM_setFactionsFunc;
    [0, 0.7, BIS_ACM1] call BIS_ACM_setSkillFunc;
    [0.2, 0.5, BIS_ACM1] call BIS_ACM_setAmmoFunc;
    ["ground_patrol", 1, BIS_ACM1] call BIS_ACM_setTypeChanceFunc;
    ["air_patrol", 0, BIS_ACM1] call BIS_ACM_setTypeChanceFunc;
    [BIS_ACM1, ["TK_INS_Group", "TK_INS_Patrol", "TK_INS_AATeam", "TK_INS_ATTeam", "TK_INS_Technicals",
    "TK_INS_MotorizedGroup"]] call BIS_ACM_addGroupClassesFunc;
  };
```

“init.sqf” contents (for briefings):

```
execVM "briefing.sqf";

if(true) exitWith ();
```



INIT (MODULES)

```
this setVariable [“settings”, [[], true, nil, nil, false]]; (see Calling in Artillery/CAS on marked targets)
[_battery, true] call BIS_ARTY_F_SetShellSpawn; (see Advanced use of Modules)
[_virtualPiece, "desiredvehicle"] call BIS_ARTY_F_SetVirtualGun;
this setvariable ["townlist",[bis_loc_acityc_cityname,[position this,desiredradius],desiredtrigger]];
[bis_loc_acityc_kozlovka,"ALICE_civilianinit",[{_yourcodehere}]] call bis_fnc_variablespaceadd;
BIS_alice_mainscope setvariable ["civilianCount","round (2*sqrt%1)"];
```

TRIGGERS (ON ACT.)

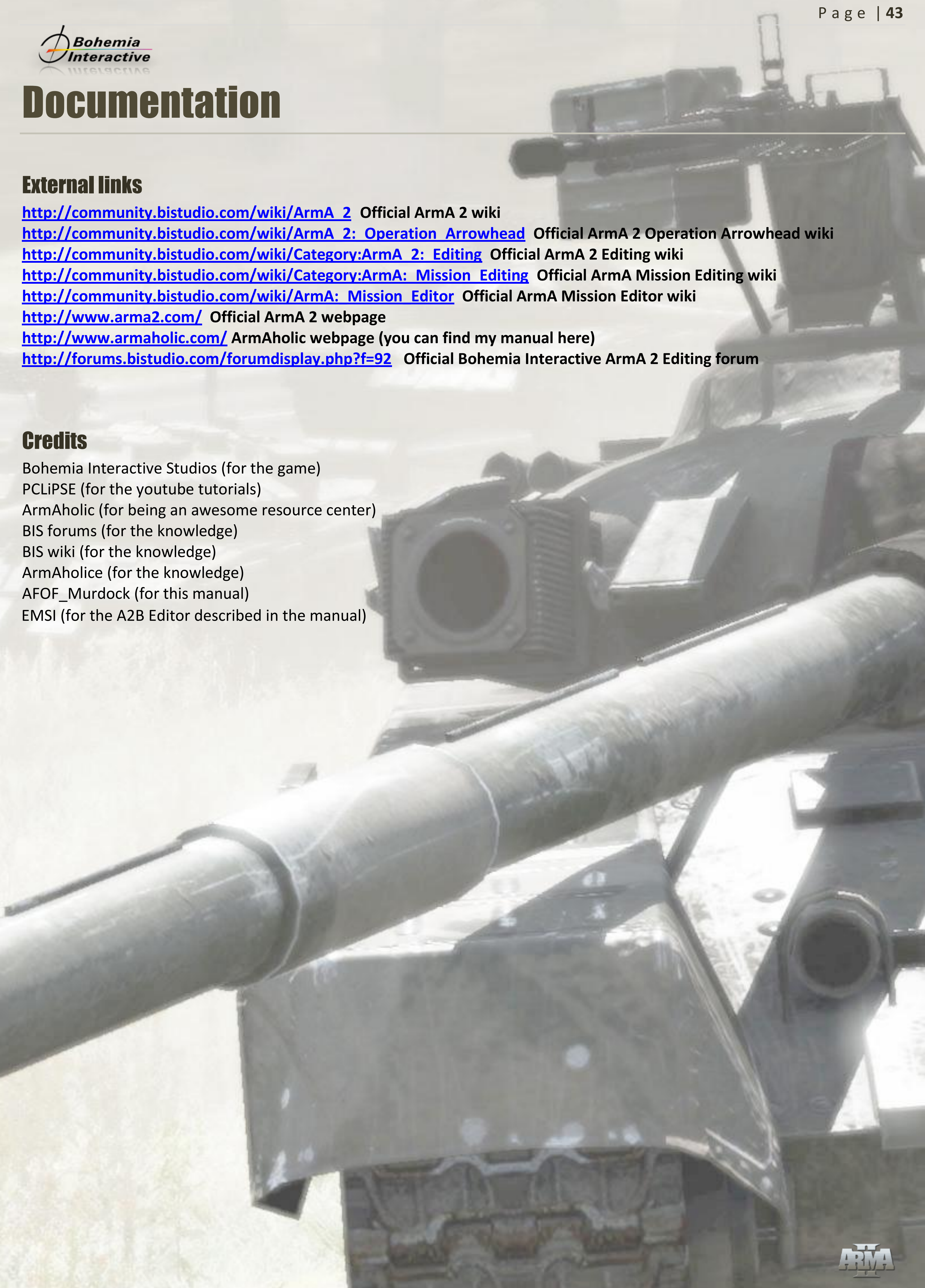
```
[["tactical_airstrike"], player] call BIS_SOM_addSupportRequestFunc;
[["artillery_barrage"], player, [[]]] call BIS_SOM_addSupportRequestFunc;
```

VARIOUS COMMANDS

```
[player,halojumpheight] exec "ca\air2\halo\data\Scripts\HALO_init.sqs"; (see HALO jumps)
this setpos [getpos this select 0, getpos this select 1,(getpos this select 2) +halojumpheight]; this flyInHeight halojumpheight
object1 setPos [ getPos desiredobject ] (see setPos and attachTo commands)
object1 setpos [ getPos desiredobject select 0, getPos desiredobject select 1, value]
object1 setPos [ getPos desiredobject select 0, getPos desiredobject select 1, (getPos desiredobject select 2) +offset]
object1 attachTo [desiredobject,[ value, value, value]];
detach object1
```

NOTE: ALTHOUGH ONCE PLANNED, THE CLASSNAMES LISTS HAVE BEEN REMOVED BECAUSE OF THE ENORMUS SPACE CONSUMPTION. YOU CAN STILL FIND THEM ON THE INTERNET. THEY MAY BE IMPLEMENTED LATER ON

Documentation



External links

http://community.bistudio.com/wiki/ArmA_2 Official ArmA 2 wiki
http://community.bistudio.com/wiki/ArmA_2:_Operation_Arrowhead Official ArmA 2 Operation Arrowhead wiki
http://community.bistudio.com/wiki/Category:ArmA_2:_Editing Official ArmA 2 Editing wiki
http://community.bistudio.com/wiki/Category:ArmA:_Mission_Editing Official ArmA Mission Editing wiki
http://community.bistudio.com/wiki/ArmA:_Mission_Editor Official ArmA Mission Editor wiki
http://www.arma2.com/ Official ArmA 2 webpage
http://www.armaholic.com/ ArmAholic webpage (you can find my manual here)
http://forums.bistudio.com/forumdisplay.php?f=92 Official Bohemia Interactive ArmA 2 Editing forum

Credits

Bohemia Interactive Studios (for the game)
PCLiPSE (for the youtube tutorials)
ArmAholic (for being an awesome resource center)
BIS forums (for the knowledge)
BIS wiki (for the knowledge)
ArmAholice (for the knowledge)
AFOF_Murdock (for this manual)
EMSI (for the A2B Editor described in the manual)

Please read this appeal

This is an appeal from AFOFMurdock, the creator of this manual, to all of those out there who read this manual and found it useful; and to those who know all this already, but wanted to see my work:

Please, if you have the time, patience or resources, and think this manual could be expanded, made better, etc., help me out with the further development of this manual by writing your own short chapters or tutorials to be implemented in this manual. MS Word (.docx or .doc) format files would be appreciated, but a notepad text (.txt) file will do just fine. I will also be glad if you've noticed some errors or something is missing to PM me or post about it in the armaholic forums.

Thank you,

yours sincerely,

AFOF_Murdock