

Advanced Artillery Request System

Script for ArmA2

by Bon_Inf*

10th-community.com

Installation / Integration	2
Customization	2
<i>Availability Restrictions</i>	3
Usage	4
<i>The Overall Interface</i>	4
<i>Call in a (spread) fire mission</i>	7
<i>Call in a precise fire mission using adjustment fire</i>	9
<i>Call in precise fire mission using a laser designator</i>	10
For Scripters	11
References	11



Installation / Integration

Assume you already created a mission in the editor, and you want this feature to be integrated in your mission. Then unzip the archive, namely *artillery.utes.zip*, and do the following steps.

1. Copy the enclosed folder *bon_artillery* into your mission folder.
2. Merge the enclosed *Description.ext* with your own *Description.ext* extension file:
 - a. In case you don't have such file yet, simply copy the *Description.ext* into the root of your mission folder as it is.
 - b. In case you have such *Description.ext* already, copy and paste the content of the enclosed *Description.ext* into it. Now, you might probably have a *CfgSounds* class defined already, then only copy the line

```
#include "bon_artillery\Description.ext"
```

INTO your *CfgSounds* class.
3. Same play for the *Init.sqf* file. If you have no *Init.sqf* in the mission folder's root yet, simply copy the enclosed one. Otherwise, make sure your *Init.sqf* contains the line

```
[ ] execVM "bon_artillery\bon_arti_init.sqf".
```
4. very important now: Start ArmA2, open the editor and load your mission into it.
 - (1) Press F1, double-click somewhere on the map.
 - (2) Choose *Game Logic* as Side,
 - (3) *Objects* as Class,
 - (4) and write *Server* into the *Name-field*. Then click *OK*.

The integration step is complete.

Customization

There are many things you can customize in this component. For customization, open the file *bon_arti_init.sqf* located in the *bon_artillery* folder.

Call Sign

HW_Arti_CallSign can be any string that then will appear in the radio chatter when using the artillery. (And yes, I am a big fan of Generation Kill...)

Number Cannons

HW_Arti_CannonNumber must be an integer ≥ 1 and sets the number of cannons to appear in the list of cannons when opening the artillery menu. A high number of cannons is more effective than a high number of shells that can be fired for each cannon on a single request - Firing two shells from the same cannon implies a short reload time between the splashdown, where two cannons can fire in parallel.

Number Rounds / h

HW_arti_number_shells_per_hour must be an integer ≥ 1 and sets how many shells can be ordered to fire every hour. For each shell being fired this value decreases by 1, and resets after a full hour has past during the game. Players get notified of reset by a sidechat message.



Artillery Types

HW_arti_types is an array / a list consisting of elements in the form

[“Displayname”, “Round Type”].

- “Displayname” is a string that appears in the combobox of the menu when using it.
- “Round Type” is the string referring to the classname of the type that gets created over the designated zone. The classnames are defined in ArmA2’s *CfgVehicle* configfile.

Max. Number Shells

HW_arti_maxnrshells must be an integer ≥ 1 and sets how many rounds can be fired by each cannon in a single fire mission request.

Artillery Spread Types

HW_arti_spreads is an array / a list consisting of elements in the form

[“Displayname”, <Dispersion in meters>].

- “Displayname” is a string that appears in the combobox of the menu when using it.
- <Dispersion in meters> is an integer that sets the radius in meters shells will go down with the designated zone as center. Note: the feature to designate a target with a laser designator, the dispersion must be an integer < 0 .

Displaying Position

HW_arti_show_pos_on_map set to *true* enables every player to read out his/her polar coordinates when opening the ingame map. This allows players who aren’t even Artillery Observers to transmit their position to those who are. Set to *false* disables this feature.

Availability Restrictions

Of course most of the time you want this feature to be restricted to certain players.

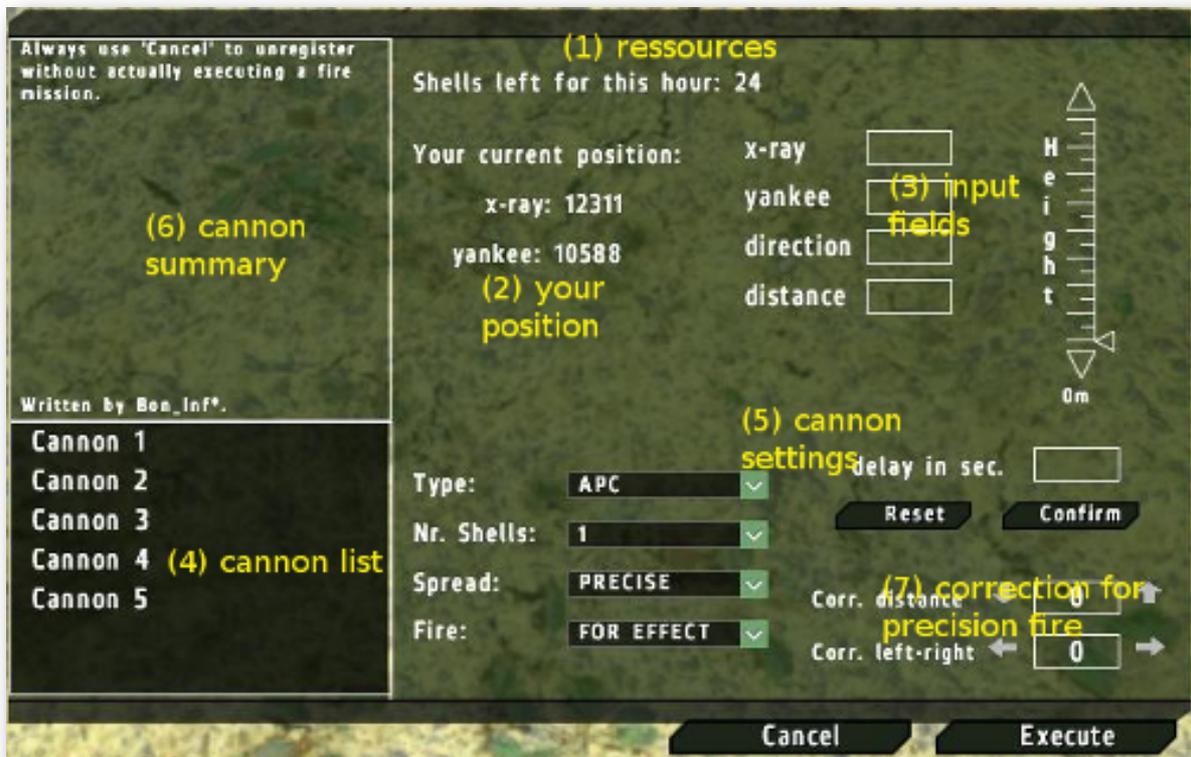
This script offers you to easily restrict it to:

1. the player’s class: Extend the list *_arti_cond_classes* by the classes which should be able to use this system, or leave it empty for no such restriction.
2. the player’s side: Extend the list *_arti_cond_sides* by the sides which should be able to use this system, or leave it empty for no such restriction. Possible sides are: *WEST, EAST, CIVILIAN, GUER*.
3. weapons the player must be equipped with: Extend the list *_arti_cond_weapons* by the weapons classnames defined in ArmA2’s *CfgWeapons* configfile.
4. own restrictions: Replace the *true* in the string *_arti_cond_other* by any statement that either returns *FALSE* or *TRUE* when calling it.



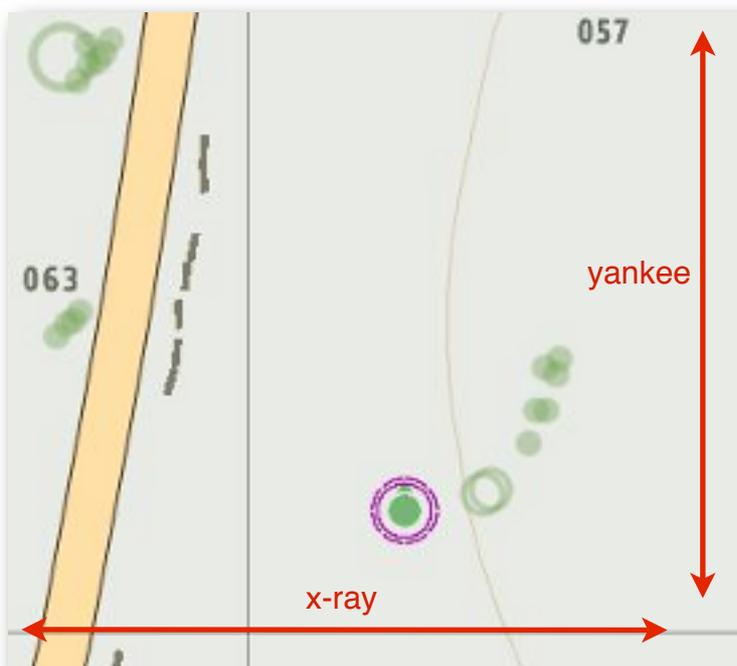
Usage

The Overall Interface



Ok. lets take a look at what all these elements mean:

(1) resources: shows how many shells can be ordered to fire in the remaining hour for the particular side.



(2) your own position in cartesian coordinates:

The position on the map/the three-dimensional space can be determined by three coordinates, namely
x-ray for the point on the x-axis (west to east),
yankee for the point on the y-axis (south to north)
zulu for the height, where the latter one is of special interest for us, as we will discuss in the next section.

There is also another, more rough representation of a map's

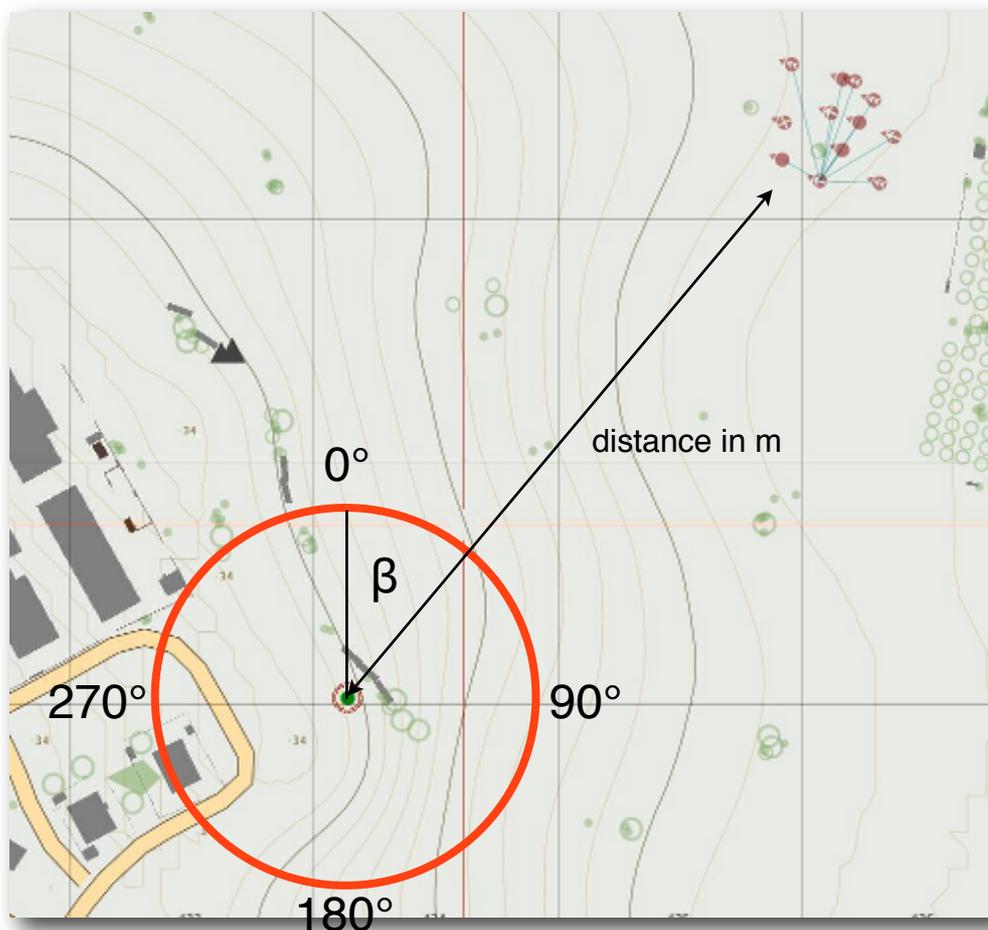


position, i.e. the grid partitioning: in the picture above the unit is placed in grid 057-063, and each grid is of size 100x100m, means one grid represents all cartesian coordinates within a 100x100m range. However, the grid partition is not the positional representation we are working with here.

(3) *Input fields*: Here you have to insert the values of the positional vector in form we discussed one point above. Note: The values to enter are not necessarily the ones representing your own position. It might be possible another player transmits his/her coordinates and direction/distance information to you so you can call artillery on a target another player has designated.

Apart from your positional data you have to enter the direction in degrees to the target and the distance between the given position and the target.

For most of you it is clear, I am using this opportunity to go into detail:



Obtaining direction:
North is always considered 0° ,
E = 90° ,
S = 180° ,
SW = $180^\circ + 45^\circ = 225^\circ$
and so on.

In the example to the left what is asked for is the angle called " β ".

You can always retrieve this β direction angle using your compass.

For obtaining distance in ArMA2 Vanilla simply press SPACEBAR. Using ACE2 you might need to be equipped with a specific [range-finding device](#).

Last but not least you need to set the height difference between the spotter and the target using the slider on the right-hand side of the dialog.

Why do we need to do that? Simply said, the polar coordinates, consisting of a two-dimensional position, an angle and a distance value are only defined in two-dimensional space. That causes one problem since we are working in a three-dimensional vector space. The problem is illustrated in the following picture.



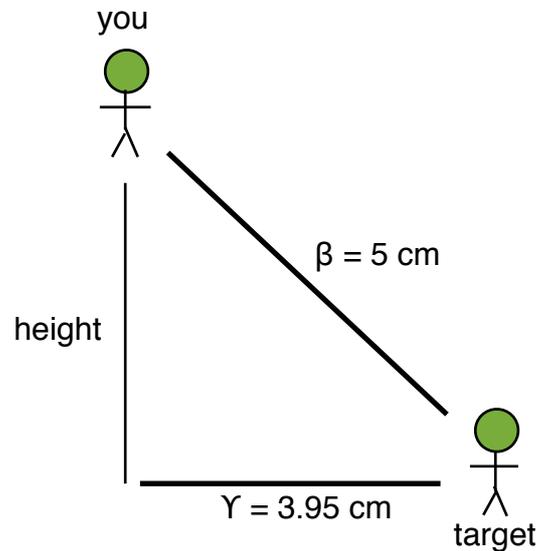
We have: β .

We need: γ .

Unfortunately,

$\beta \neq \gamma$.

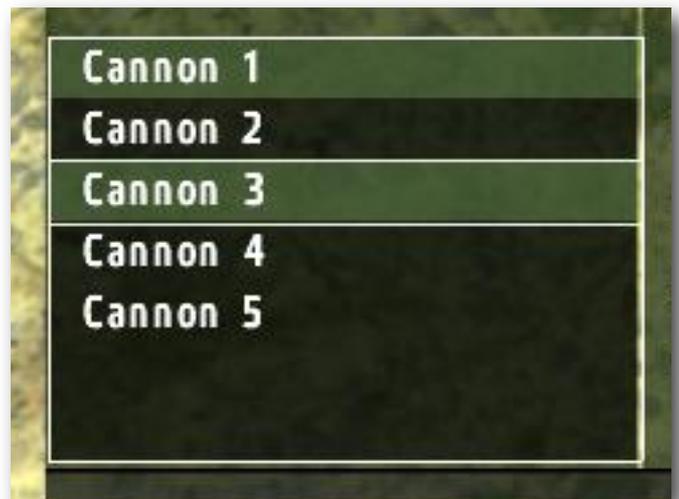
The system uses the height value to compensate this difference.



(4) Cannon list: The list of cannons that can be ordered to fire. Each cannon needs reload time after a fire mission, thus is not available due to this time. The duration of this reload time depends on the number shells that were fired in the previous fire mission. A cannon unavailable appears colored red in this list.

You can select a single cannon by pressing the left mouse button on it, or you can select any set of cannons by pressing and holding LEFT CTRL while clicking the left mouse button, as it is done in the picture on the right.

Or also can select a set of consecutive entries using the LEFT SHIFT button in combination with the left mouse button.



(5) Cannon Settings: Several options might be applied to a particular cannon, whose are:

- the type of fire mission,
- the number shells the particular cannon has to fire for this mission,
- the dispersion or spread of the shots,
- and either it should be fire for effect or for adjustment. Both these mission types are explained in the following sections.

(6) Cannon Summary: Once you applied settings to cannon(s), your settings are summarized in this window, when marked in the cannon list. Thus you can check the settings for every cannon afterwards.



(7) Correction for precise fire missions: Precise fire missions require adjustment. In case you order adjustment fire and actually have to correct the splashdown position, this area of menu is used.

Call in a (spread) fire mission

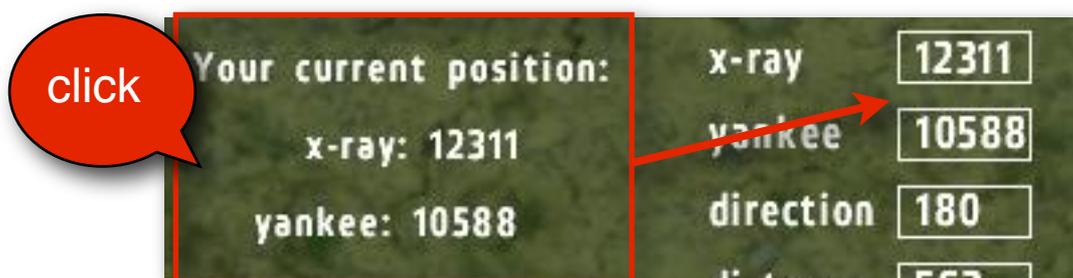
Open the artillery dialog from your action menu. You now either gets registered for fire mission. Then you will stay registered unless you press the *Cancel* button or *Execute* button. No one else will be able to call in artillery while you are registered.

This enables you to close the dialog by pressing ESC without losing any data entered so far, change your position or direction you are looking at. This way you can designate more than one area or target with one single fire mission, as it is also demonstrated in the first video (see References).

Or another player already registered himself, in this case the dialog won't open and you have to wait until the currently registered player unregisters.

Now let us define a checklist to call in artillery. Should be easy to remember the procedure when having a routine to follow.

Apply your / the spotters position by writing them into the editable fields labeled with *x-ray* and *yankee*. In case its your own position, you can simply click somewhere on the area in the menu labeled with *Your current position* and the positional values below:



Apply direction and distance to target by writing them into the editable fields labeled with *direction* and *distance*.

With all these information we have the *polar coordinates* of the designated position, which then, again, are converted into cartesian coordinates by the system.



- Set height dispersion** to your target.

- Select the cannons to fire**, by highlighting them in the cannon list (4).

- Select Type of Mission**, regarding to the type of target that has to be suppressed.

- Select Number Shells** each cannon should fire. As already mentioned, consider that one cannon firing two shots need about 15 seconds, two cannons firing single shot each can fire in parallel. The drawback in the second option is that you occupy two cannons for one target.

- Select the Splashdown Dispersion**, namely spread. This can be completely customized, the default values are:
 - TIGHT, with a spread radius of 25m,
 - SCATTERED, with a spread radius of 50m,
 - WIDE, with a spread radius of 100m.

- Set a fire delay** for the selected cannons. Apart from the time the mission data is transmitted and the shell travel time (which all in all might be something about 15 seconds) this value in seconds will be added.

- Apply your settings** to the selected cannons by clicking on the *confirm* button. The cannon summary window (6) will be updated to show the setup of the currently selected cannon. You can browse through the cannons in the cannon list, the summary window will always update corresponding to the selection. On the same way you can clear cannons from any setup clicking on the *Reset* button.

- Once you have confirmed all settings, **press *Execute*** to call the artillery in.



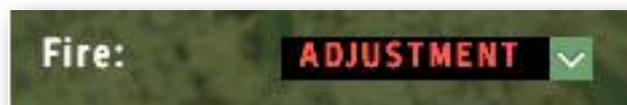
Call in a precise fire mission using adjustment fire

Besides the spread types TIGHT, SCATTERED and WIDE there is also the type PRECISE. Of course it is unlikely artillery fires with an accuracy better than 25 meters (in reality, even 25m of accuracy is hardly achievable), BUT, we have the option to adjust fire, means, firing a single shot, i.e., a red smoke shell, to see where the shells would go down and correct this position relative to our / the spotter's position.

That works as simple as follows:

Enter Positional Data as we did it in the previous section.

Now instead of selecting cannons **select ADJUSTMENT FIRE** from the mission type text box.



Press Execute and await red smoke.

The red smoke indicates where the shells would go down when firing for effect. If this position **is okay**, **proceed the same way as in the previous section**, except, that you **select PRECISE** spread type.



Otherwise use the arrow buttons located in the bottom right corner of the dialog to adjust the splashdown position relative to where the spotter stands. In the picture above the spotter might adjust the position 10 meters to the right, and perhaps 5 meters more far away. That would be clicking the "right-arrow" 10 times and the "forward-arrow" 5 times.

Once the position is settled, confirm the settings to the cannons as explained in the previous section, and press *Execute*.



Call in precise fire mission using a laser designator

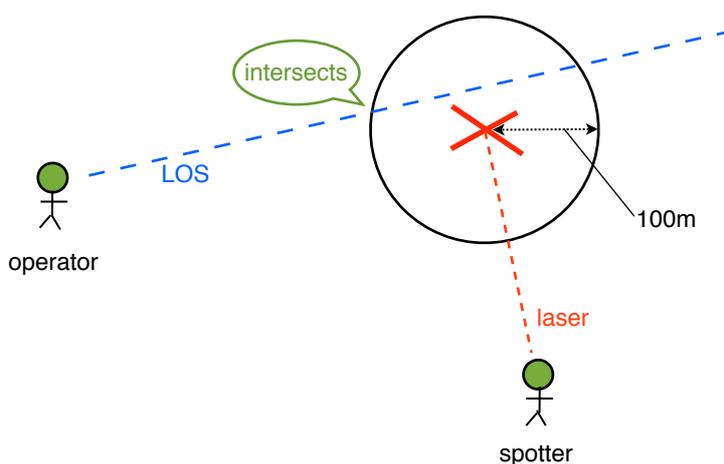
Using the laser designator to mark the target is the most simple and most reliable way of calling artillery on a target, and there is not much to explain, since no more positional data is required.

- Select the cannons to fire**, by highlighting them in the cannon list (4).
- Select Type of Mission**, regarding to the type of target that has to be suppressed.
- Select Number Shells** each cannon should fire.
- Select LASER as spread**.
- Confirm the settings, press *Execute***.
- Lase the target**.

It is also possible for another player to lase the target, such that the system uses his/her lasertarget to drop artillery on.

Important here:

- The player using this system must have visual to the lasered target.
- The player using this system must also look into the lasered targets direction, more specifically, the operator's line of sight has to intersect the imaginary circle around the lasertarget and a radius of 100m.



For Scripters

This system focusses on REQUESTING a fire mission. It's main intention is to read in user data and to compute the position for splashdown from it. However, the actual *firing* is nothing but creating shells about 150m above this position.

To combine it with a system that focusses more on FIRING artillery upon a designated position or area, I prepared the script *bon_arti_fire_advanced.sqf* located in the *bon_artillery* folder. In this script, there is an area remarked which can be modified by those who have the competence. In the original script this area is responsible for creating the shells, executed for each single shot fired by each available cannon. It provides

1. the 3D-position computed by the system,
2. the type that was selected in the menu and is assigned to the particular cannon, respectively,
3. and the radius of dispersion in meters, with the 3D-position as its center.

Information is also written in the script itself.

For this concern expect me to provide as many support as I can, any cooperation would be appreciated. Use the Bohemia Interactive Forum as platform to contact me - the link to the particular thread you can find in the last section.

References

Bohemia Interactive Forum:

<http://forums.bistudio.com/showthread.php?t=93200>

Youtube demonstration videos:

<http://www.youtube.com/watch?v=y5OvVv8K6SI>

<http://www.youtube.com/watch?v=0uYhsKYo0GQ>

