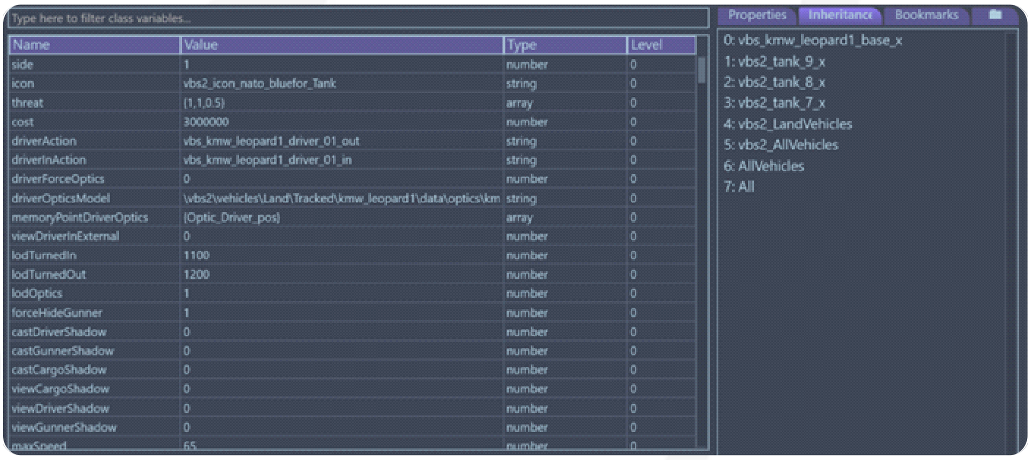
Config Patch Builder



Type here to filter class variables...

Name	Value	Type	Level
side	1	number	0
icon	vbs2_icon_nato_bluefor_Tank	string	0
threat	{1,1,0.5}	array	0
cost	3000000	number	0
driverAction	vbs_kmw_leopard1_driver_01_out	string	0
driverInAction	vbs_kmw_leopard1_driver_01_in	string	0
driverForceOptics	0	number	0
driverOpticsModel	\vbs2\vehicles\Land\Tracked\kmw_leopard1\data\optics\km	string	0
memoryPointDriverOptics	{Optic_Driver_pos}	array	0
viewDriverInExternal	0	number	0
lodTurnedIn	1100	number	0
lodTurnedOut	1200	number	0
lodOptics	1	number	0
forceHideGunner	1	number	0
castDriverShadow	0	number	0
castGunnerShadow	0	number	0
castCargoShadow	0	number	0
viewCargoShadow	0	number	0
viewDriverShadow	0	number	0
viewGunnerShadow	0	number	0
maxSpeed	65	number	0

0: vbs_kmw_leopard1_base_x
1: vbs2_tank_9_x
2: vbs2_tank_8_x
3: vbs2_tank_7_x
4: vbs2_LandVehicles
5: vbs2_AllVehicles
6: AllVehicles
7: All

CONFIG PATCH BUILDER

VBS4 24.1.1

Bohemia Interactive SIMULATIONS

Documentation Legal Notice

This Documentation, including any embedded help systems and electronically distributed materials, (hereinafter referred to as the “Documentation”) is for your informational purposes only and is subject to change or withdrawal by Bohemia Interactive Simulations (BISim) at any time. This Documentation and its contents are proprietary information of BISim, also protected by copyright, and may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of BISim.

If you are a licensed user of the software product(s) addressed in the Documentation, you may print or otherwise make available a reasonable number of copies of the Documentation for internal use by you and your employees in connection with that software, provided that all BISim copyright notices and legends are affixed to each reproduced copy.

The right to print or otherwise make available copies of the Documentation is limited to the period during which the applicable license for such software remains in full force and effect. Should the license terminate for any reason, it is your responsibility to certify in writing to BISim that all copies and partial copies of the Documentation have been returned to BISim or destroyed.

BISim has made every reasonable effort to ensure the accuracy of all the information contained in the Documentation. However, product specifications are subject to change without notice, and BISim makes no representations or warranties regarding the accuracy, completeness, or suitability of information contained in the Documentation. To the maximum extent permitted by law, BISim disclaims any and all liability for any loss, damage (direct or indirect) or other consequence which may arise from the use of or reliance upon any information contained in the Documentation.

The use of any software product referenced in the Documentation is governed by the applicable license agreement and such license agreement is not modified in any way by the terms of this notice.

Customer Support

The Bohemia Interactive Simulations Support page can be found at:

- http://www.bisimulations.com/support

For any type of assistance with Bohemia Interactive Simulations products, use the following support email and we will respond to your query with urgency.

- support@bisimulations.com

Our website contains a range of media and handouts relating to Bohemia Interactive Simulations products:

- http://www.bisimulations.com/

The BISim Wiki is the primary resource on VBS4 scripting:

https://sqf.bisimulations.com/display/SQF/VBS+Scripting+Reference

PhysX

VBS4 uses the PhysX physics engine. For more information on PhysX visit the Nvidia site.

https://gameworksdocs.nvidia.com/simulation.html

Contents

1. Config Patch Builder

The Config Patch Builder (CPB) allows users to generate patch files (`.pbo`) that contain parameter and value changes for assets in VBS. It does this without requiring the user to have an indepth understanding of configuration file writing, configuration syntax, or other technical knowledge.

Such simple changes could be:

- Changing the maximum speed of a vehicle.
- Changing the engine power of a vehicle.
- Increasing the damage and velocity of specific ammunition.
- Changing the default weapons of a tank.

Advanced Use Cases

The CPB is also capable of inserting new configuration parameters / name-value pairs, and even entire new classes. Using the Batch Import (on page 16) feature, it enables you to edit configurations from a spreadsheet environment, and then transfer the value changes automatically into a patch file that VBS can load and apply to alter a simulation. In addition, the CPB can serve as a quick and simple offline configuration browser.

This topic covers the following:

- Basic Concepts (on the next page)
- Basic Workflow (on page 10)
- Editing Values (on page 10)
- Adding Entries (on page 11)
- Adding Classes (on page 13)

Further sub-topics include:

- CPB Batch Editing (on page 16)
- CPB Example - M1A1 Tank (on page 21)

The CPB GUI is found at the following location:

`\VBS_Installation\optional\configpatchbuilder\`

1.1 Basic Concepts

The CPB has the following basic concepts and functions.

Config Cache

The CPB operates from a config cache that is generated by VBS4. You need to run VBS4 at least once and reach the Main Menu for a config cache to be built. The cache is saved at the following location:

```
\VBS4_Installation\cache\config.cache
```

When you launch the CPB, you have to first load a config cache using the **Load Config** button, and then select the cache file. On every subsequent launch of CPB, the last loaded config cache is automatically loaded again for you.

TIP

The config cache files do not necessarily need to be stored in the VBS4 installation directory. You can copy a config cache file from the VBS4 installation folder to an auxiliary location, and keep it there for version control.

It is recommended that you make a copy of the original VBS4 config cache file, store it somewhere safe, and then use it as a basis for any changes.

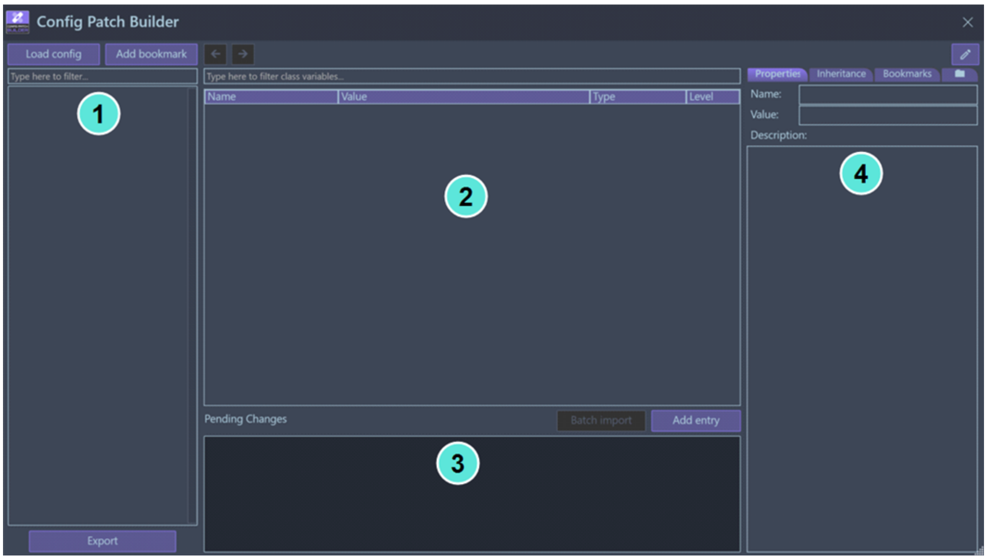
WARNING

If you apply a configuration patch file to VBS4, launch VBS4 and reload the CPB, it defaults to your previously made changes as part of the config cache. It no longer knows which changes were externally made, and which ones were default VBS4.

It is recommended that you export each patch also as a `.csv` file, so that you can re-load the patch into the CPB as a project at a later stage.

Be extremely careful that you keep track of your patch files, which changes they make, and which ones are currently applied and integrated into the currently loaded config cache. This has potential to cause extreme confusion, so work slowly and with great care.

Image-1: Config Patch Builder Layout

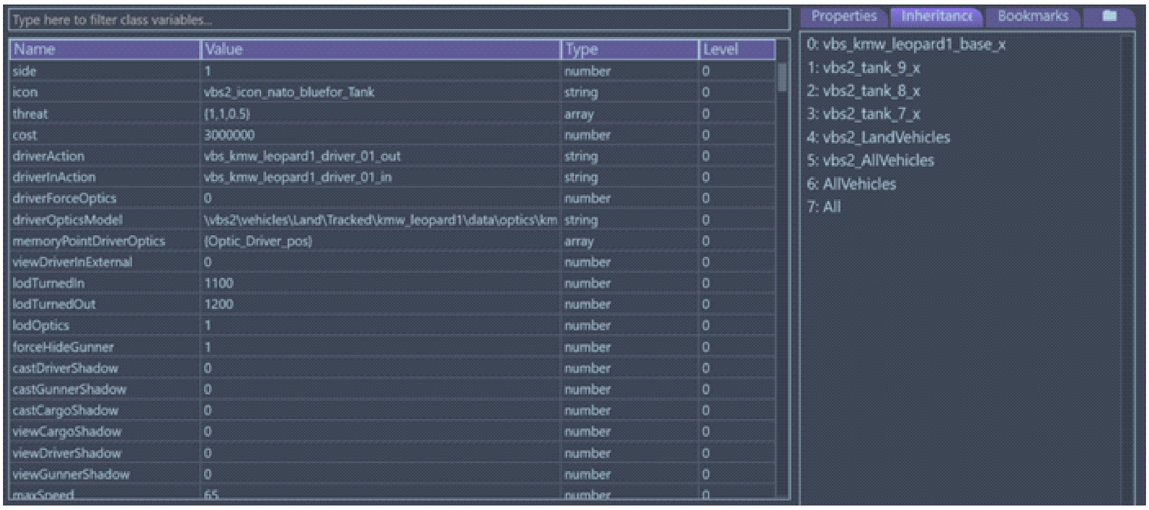


The CPB is divided into four windows.

1	**Config Class Tree**	Use to browse all available classes from the configuration cache. As you view different classes, the address bar above the Config Main View (below) window shows your current location and allows you to step back and forth (using the arrow buttons) between recently viewed classes. **NOTE** By default, the CPB only shows the classes that are white listed in the `AllowedClasses.csv` file, located in the settings folder, next to `CPB.exe`. Delete or edit this file to access further classes.
2	**Config Main View**	Displays all the parameters and values present in the currently selected class. Here, you can also edit parameter values. At the bottom of this window is the **Add Entry** button, which allows you to add new parameters and classes to the currently selected configuration class (see Adding Entries (on page 11)).
3	**Pending Changes**	Shows all currently made changes to the config cache. Changed entries are listed in the Config Main View (above) window and are shown in **red**. You can click on an entry or class here to directly jump to that class again. This makes working with a large amount of changes manageable.
4	**Context Information**	Additional help about the currently selected configuration parameter. This information is only available for the most common parameters, which are documented in the Configuration Manual in the VBS Developer Reference. In this window you can also view the key for the Inheritance Level (on the next page), handle Bookmarks (on page 9), and settings specific to the CPB.

Inheritance Level

The Inheritance Level shows from which base class a given parameter is inherited. Click the **Inheritance** tab in the Context Information (on the previous page) window to see the class name.



Name	Value	Type	Level
side	1	number	0
icon	vbs2_icon_nato_bluefor_Tank	string	0
threat	{1,1,0.5}	array	0
cost	3000000	number	0
driverAction	vbs_kmw_leopard1_driver_01_out	string	0
driverInAction	vbs_kmw_leopard1_driver_01_in	string	0
driverForceOptics	0	number	0
driverOpticsModel	\vbs2\vehicles\Land\Tracked\kmw_leopard1\data\optics\km	string	0
memoryPointDriverOptics	{Optic_Driver_pos}	array	0
viewDriverInExternal	0	number	0
lodTurnedIn	1100	number	0
lodTurnedOut	1200	number	0
lodOptics	1	number	0
forceHideGunner	1	number	0
castDriverShadow	0	number	0
castGunnerShadow	0	number	0
castCargoShadow	0	number	0
viewCargoShadow	0	number	0
viewDriverShadow	0	number	0
viewGunnerShadow	0	number	0
maxSpeed	65	number	0

0: vbs_kmw_leopard1_base_x
1: vbs2_tank_9_x
2: vbs2_tank_8_x
3: vbs2_tank_7_x
4: vbs2_LandVehicles
5: vbs2_AllVehicles
6: AllVehicles
7: All

A parameter with level (number) 0 assigned is defined directly in the class. A parameter with level 1 is defined in the first parent class of the asset. A parameter with level 5 is defined in the 5th-level inheritance for the asset, and so on.

You can click on a class in the inheritance list to directly jump to its base class, which is highlighted in the Config Class Tree (on the previous page) window.

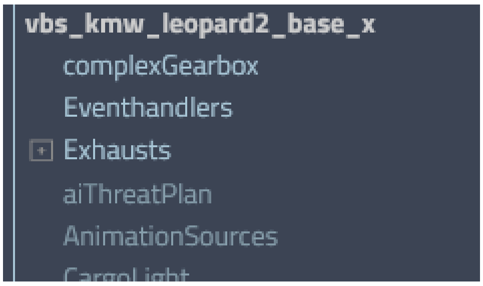
TIP

Work on base classes to ensure that all child classes receive the new values.

For example, instead of increasing the `enginePower` of a Desert Camouflaged M1A1 Tank and then making the same change to a Woodland Camouflaged M1A1 Tank, you can make this change only once to the base class that both the Desert and Woodland variants use.

Usually, this is the level 1, or rarely level 2, parent of a placeable asset from VBS Editor.

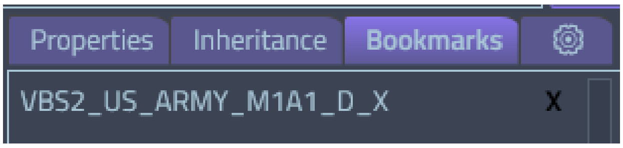
In the Config Class Tree (on the previous page) you see sub-classes that exist directly in a class highlighted. Classes shown in a darker tone only exist in the class using inheritance from other classes. They are not explicitly declared in the currently selected class. Hover the cursor over an inherited class to show the configuration path, where the sub-class originates.



vbs_kmw_leopard2_base_x
complexGearbox
Eventhandlers
Exhausts
aiThreatPlan
AnimationSources
CargoLight

Bookmarks

Classes you frequently work with can be saved as bookmarks in the **Bookmarks** tab of the **Context Information** window.



VBS2_US_ARMY_M1A1_D_X

To add a bookmark, navigate to the class you want to save, and click the **Add Bookmark** button at the top of the Config Class Tree (on page 7) window.

To skip to the bookmark, double-click the bookmark entry in the **Bookmarks** tab.

Applying Patches

Click **Export** to generate a patch as a `.pbo`. For more information, see Export Changes (on page 25).

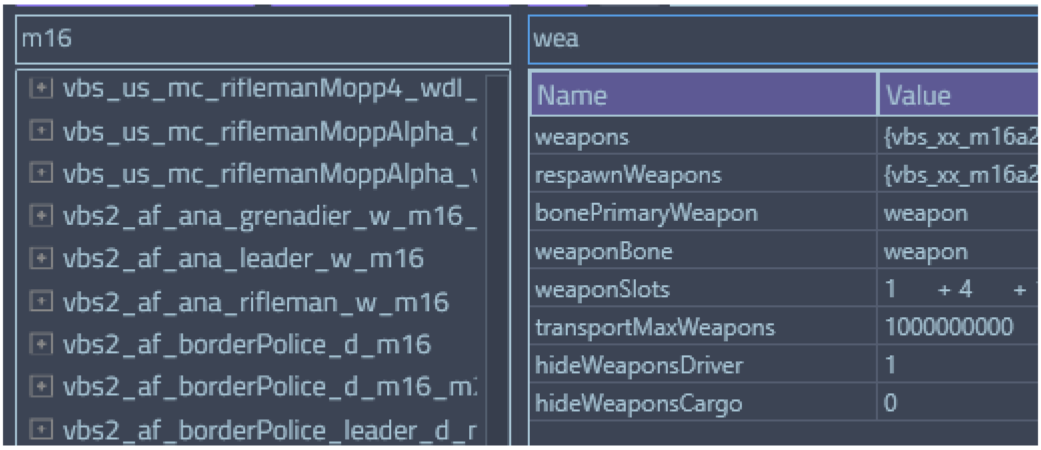
Copy the `.pbo` to your VBS4 installation folder, and restart VBS4 to activate your changes (the CPB can do this automatically for you).

`\VBS4_Installation\myData\Blue\content\`

Filtering

You can filter the shown items in the Config Class Tree (on page 7) by typing in the **Filter** box at the top, and pressing **Enter** on your keyboard to confirm. The same procedure applies to the filter of the Config Main View (on page 7), where it filters the shown parameters of the currently selected class.

To reset a filter, clear the text from the **Filter** box, and press **Enter** again.



m16

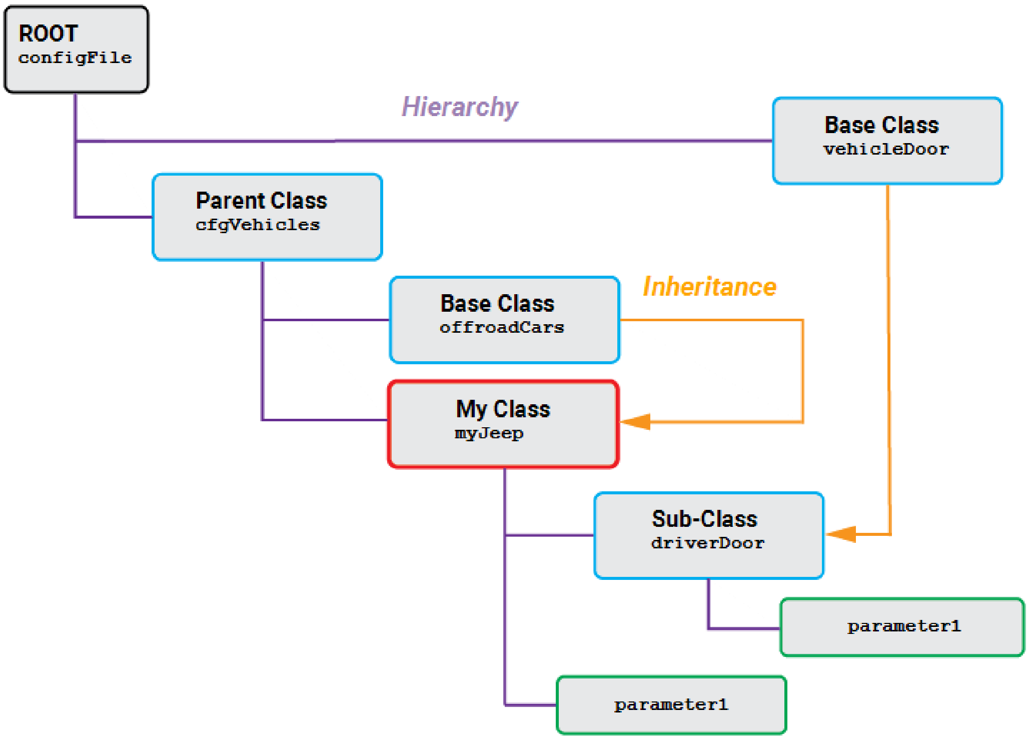
- vbs_us_mc_riflemanMopp4_wdl_
- vbs_us_mc_riflemanMoppAlpha_(
- vbs_us_mc_riflemanMoppAlpha_\
- vbs2_af_ana_grenadier_w_m16_
- vbs2_af_ana_leader_w_m16
- vbs2_af_ana_rifleman_w_m16
- vbs2_af_borderPolice_d_m16
- vbs2_af_borderPolice_d_m16_m
- vbs2_af_borderPolice_leader_d_r

wea

Name	Value
weapons	{vbs_xx_m16a2
respawnWeapons	{vbs_xx_m16a2
bonePrimaryWeapon	weapon
weaponBone	weapon
weaponSlots	1 + 4 +
transportMaxWeapons	1000000000
hideWeaponsDriver	1
hideWeaponsCargo	0

1.2 Basic Workflow

The following diagram illustrates the basic CPB workflow.

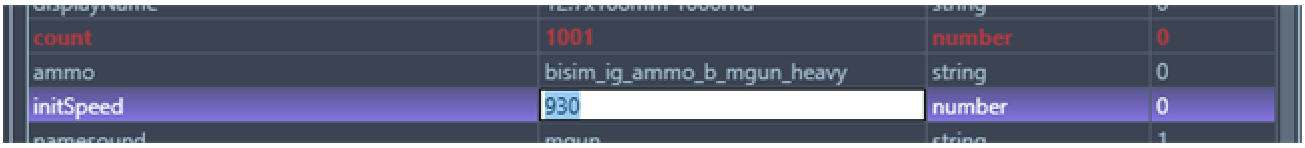


Terminology

- Root class
- Parent class
- Base class
- Sub-class
- Parameter
- **Terms are relative**
    - Viewed class as origin

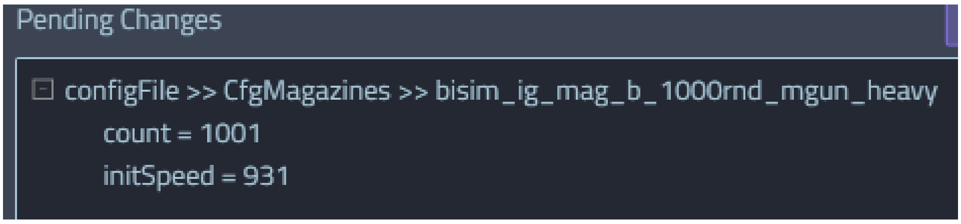
1.3 Editing Values

The most common use-case for the CPB is to make changes to existing values. This is done by clicking in the **Value** field of any currently shown configuration class, and typing in a new value.



displayName	12.7x108mm 1000rnd	string	0
count	1001	number	0
ammo	bisim_ig_ammo_b_mgun_heavy	string	0
initSpeed	930	number	0
[illegible]	mgun	string	1

Any changed value is indicated in **red** in the Config Main View (on page 7), while the change is also listed in the Pending Changes (on page 7) window.



```
⊟ configFile >> CfgMagazines >> bisim_ig_mag_b_1000rnd_mgun_heavy
    count = 1001
    initSpeed = 931
```

You can delete individual pending changes by selecting the **entry** in the Pending Changes (on page 7) window, and pressing **Delete** on your keyboard.

1.4 Adding Entries

You can introduce new name-value pairs / parameters, and new classes using the Config Patch Builder dialog (below).

Adding Parameters / Name-Value Pairs

Use the following procedure to add a new parameters / name-value pairs.

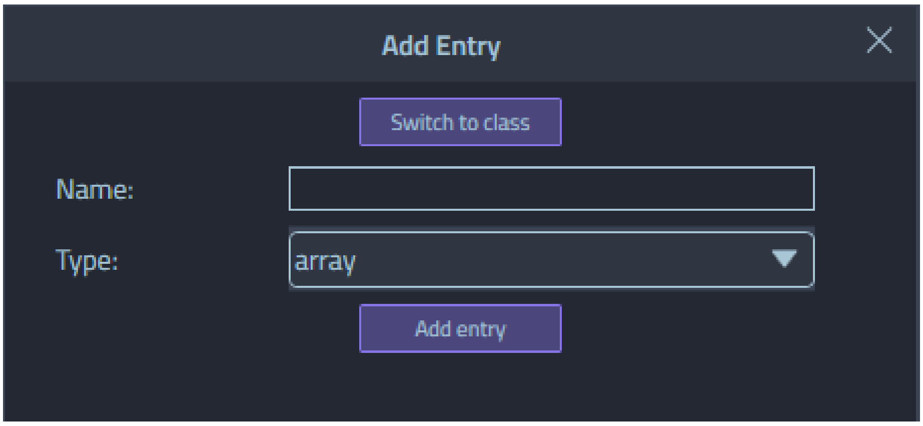
Follow these steps:

1. Navigate to the **class** that you want to modify, and click **Add Entry**.

This opens the Config Patch Builder dialog, which asks you to input the name of your new parameter / name-value pair, and the data type (array, bool, number, or string).

> **TIP**
>
> Use **Switch to Class** / **Switch to Variable** to switch between adding classes / parameters or name-value pairs. To add a class, see Adding Classes (on page 13).

Image-2: Config Patch Builder dialog



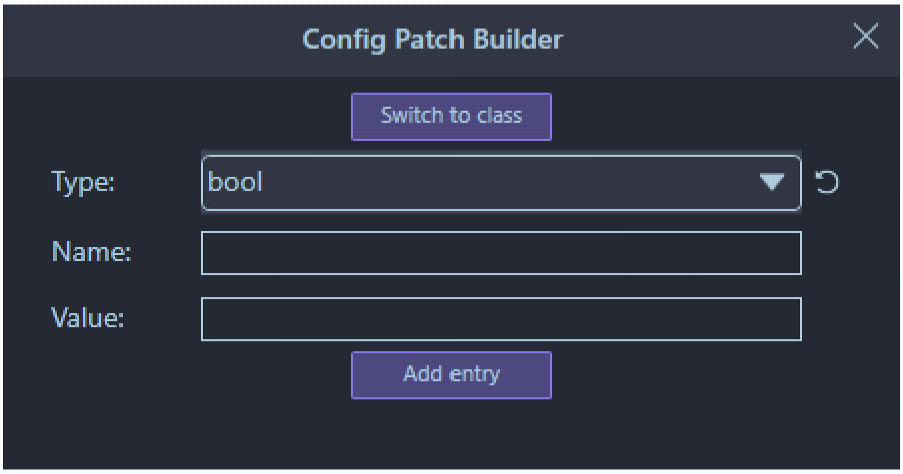
Name:

Type: array

2. Click the **Down Arrow** to expand the **Type** drop-down, and select the parameter type.

3. Enter a **Name** for your parameter.

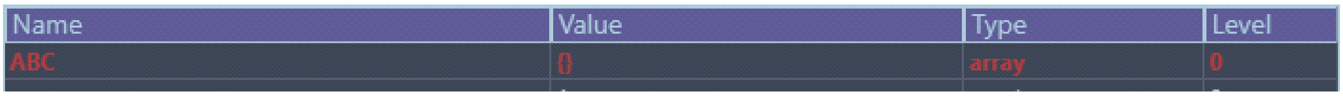
> **NOTE**
>
> If you select "bool", "number", or "string" as the parameter type, and want to add the parameters as a name-value pair, enter a value.



Type: bool

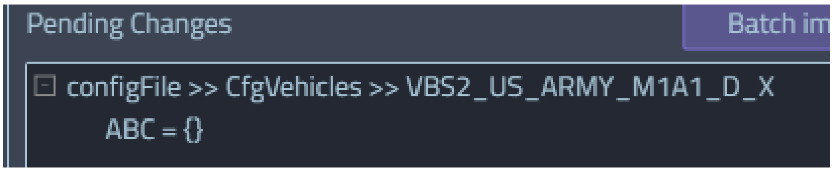
4. Click **Add Entry**.

   The dialog closes and you see the new parameter / name-value pair added to the class at the top of the Config Main View (on page 7) window. Since this is a pending change, it is marked **red**.



Name	Value	Type	Level
ABC	{}	array	0

   It is also listed in the Pending Changes (on page 7) window.



```
⊟ configFile >> CfgVehicles >> VBS2_US_ARMY_M1A1_D_X
    ABC = {}
```

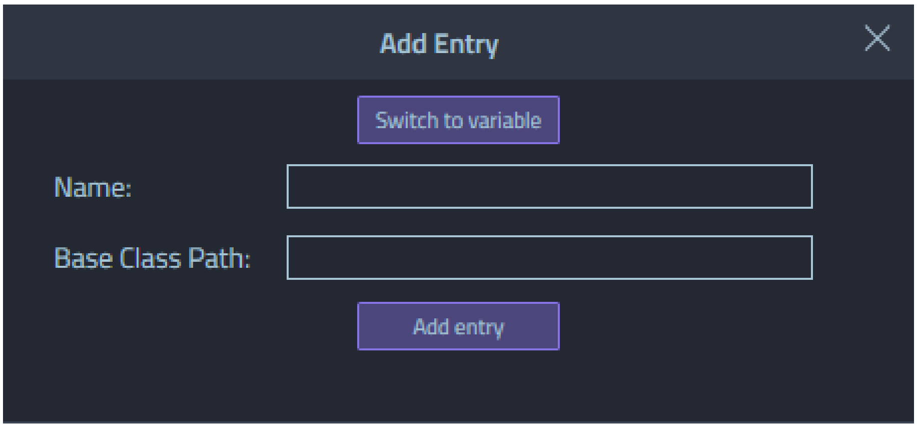
5. If necessary, double-click the parameter **value** in the Config Main View (on page 7) and begin editing.

1.5 Adding Classes

You can also add entire new classes. These always consist of a pair: `name` and `base_class_path`.

The `name` you enter becomes the name of the class, while the `base_class_path` must consist of a configuration path of the exact class you wish to inherit from.

> **NOTE**
>
> Base class paths are valid under the following conditions:
>
> - Leading to a class on the same level as you are currently working on.
> - Leading to a class higher up in the hierarchy tree.
> - Empty path.
>
> Base class paths are not valid under the following conditions:
>
> - Leading to a class on a deeper level than you are currently working on.
> - Leading to a class leading across and then deeper into the hierarchy tree of another class (for example, from `CfgVehicles` > `MyCar` you cannot inherit from `CfgWeapons` > `MyGun`).

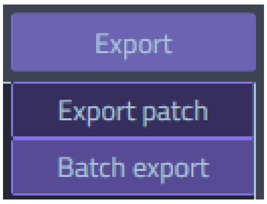


1.6 Exporting Changes

Once you have some changes ready, you can export them.

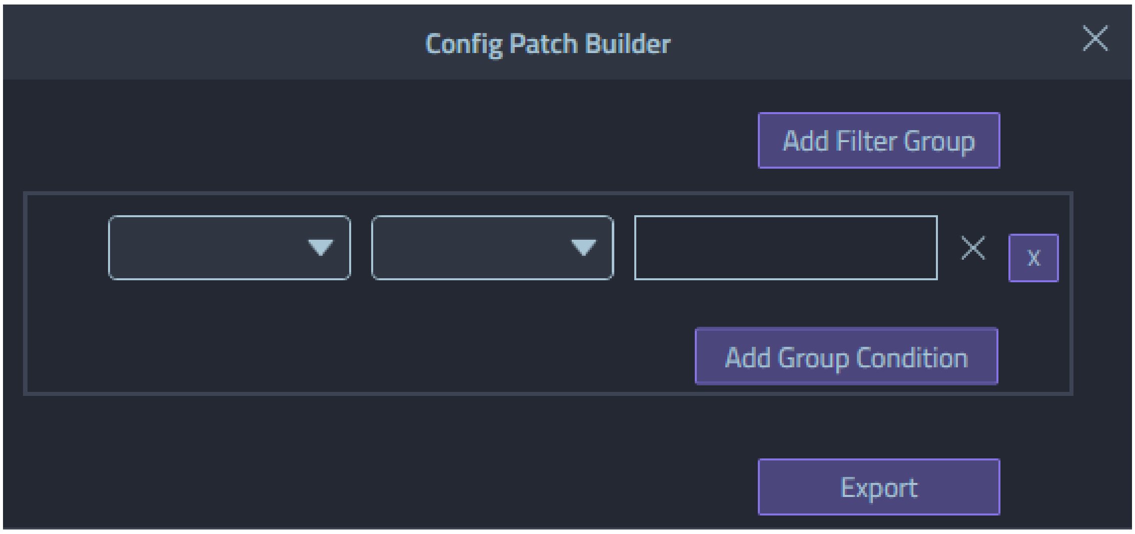
Follow these steps:

1. Click **Export** and select **Batch Export**.



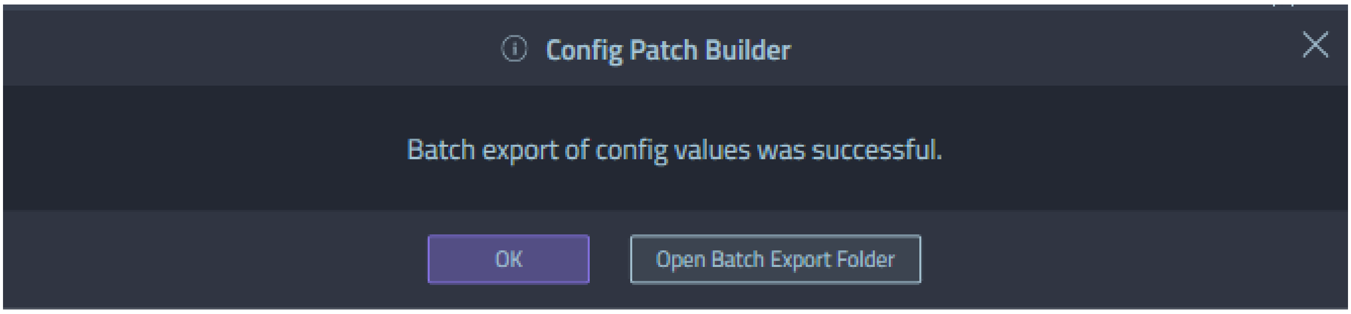
You see a filter dialog. This allows you filter all pending changes to only export those that you want to export to a `.csv` file (that is, only those that modify a specific parameter or base class).

2. If you want to export the entire contents of the Pending Changes (on page 7) window, leave the empty filter (default) dialog as it is.



3. Click **Export**.

Once completed, a dialog informs you of the successful export.

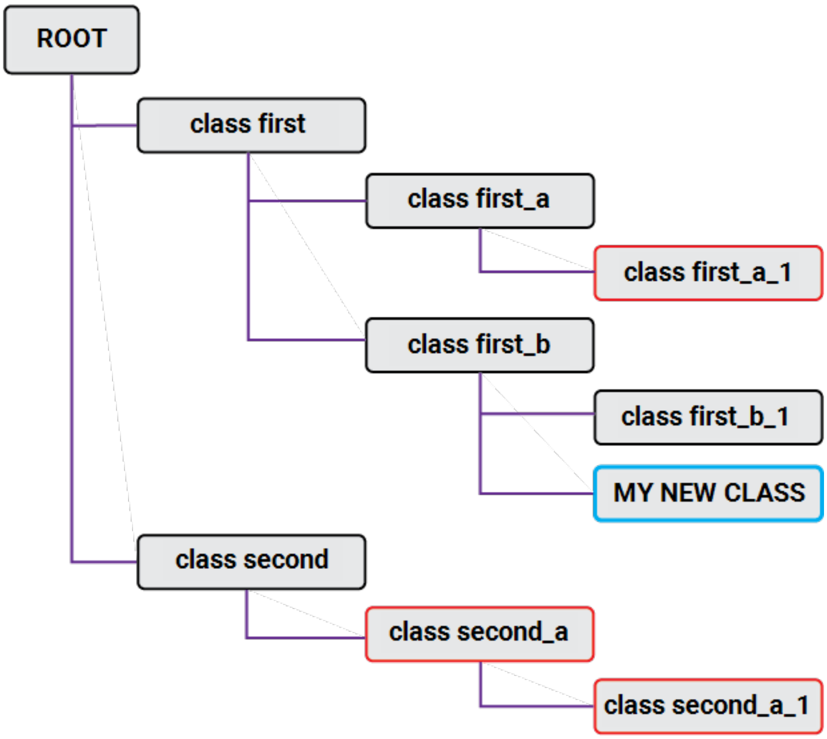


Batch export of config values was successful.

TIP

A shortcut button (**Open Batch Export Folder**) lets you directly navigate to the output folder, which is a new folder called `\BatchExport\` in the main output folder that you selected for the CPB.

The following diagram illustrates how classes are added.



WARNING

The following considerations apply:

- Classes may not inherit from:
  - Sub-classes
  - Themselves
  - Non-existent classes
- Only direct ancestry is permitted.

2. CPB Batch Editing

A very powerful feature of the CPB is the batch import / export functionality. Through the abstraction of a `.csv` spreadsheet file, you can make bulk changes to configurations, using a very efficient workflow.

2.1 Batch Import

If you have a properly formatted `.csv` file, containing configuration parameters (see CSV Export Syntax (on page 20)), you can import it into the CPB, to your Pending Changes (on page 7) window.

Follow these steps:

1. Click **Batch Import**.

2. Select your `.csv` file and confirm.

The changes from the `.csv` file are now ported to the Pending Changes (on page 7) window.

2.2 Batch Export

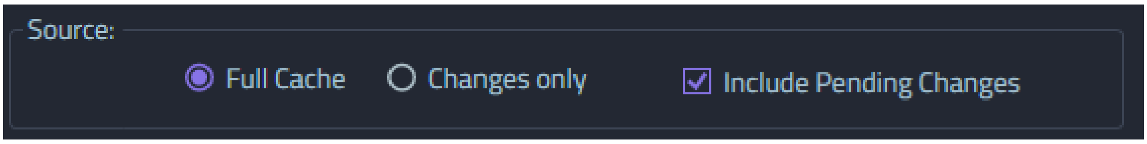
All changes that are prepared in your Pending Changes (on page 7) window can be exported to a spreadsheet file (`.csv`). This enables you to load a great number of configuration parameter changes into an easily manageable format. You can also add more entries there, and then re-import the spreadsheet to the CPB.

To begin, click the **Export** button.

A new window opens giving you control over the exact export you want to perform.

2.2.1 Source Selection

The CPB has two general sources of export:



Source: ● Full Cache ○ Changes only ☑ Include Pending Changes

Source	Description
Full Cache	Exports the entire configuration cache (including the option to also consider the changes that you made).
Changes Only	Exports only your changes made to the loaded cache file.

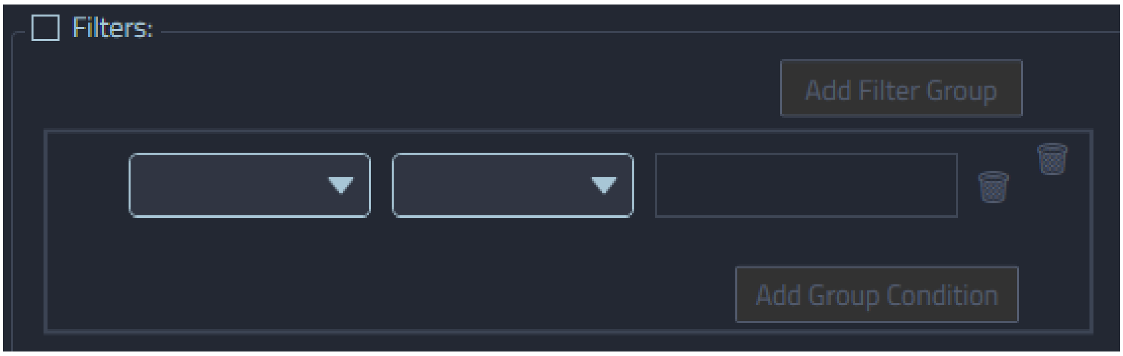
The source selector is very powerful, it allows you to use the CPB in multiple ways. The default setting in this menu is **Changes Only**. While the primary purpose of the CPB is to allow you to quickly and easily adjust entity simulation parameters, the source can also be selected to apply to the entire loaded cache (**Full Cache**). This can be useful when making a large batch of changes to classes by means of spreadsheet editing.

Alternatively, it can be used to take a quick inventory of specific assets with selected special parameters. For example, assembling a list of all the weapon classes that use a specific magazine type.

2.2.2 Export Filtering

With the source determined, you can then further restrict what the export file actually contains. If you wish to export the entire source, uncheck the filters, and proceed with the **Export** buttons.

Should you wish to only export certain parts of the cache, or your pending changes, activate the **Filters** section and begin setting up your filter conditions. Filters in the CPB work in groups. Each can contain multiple conditions.

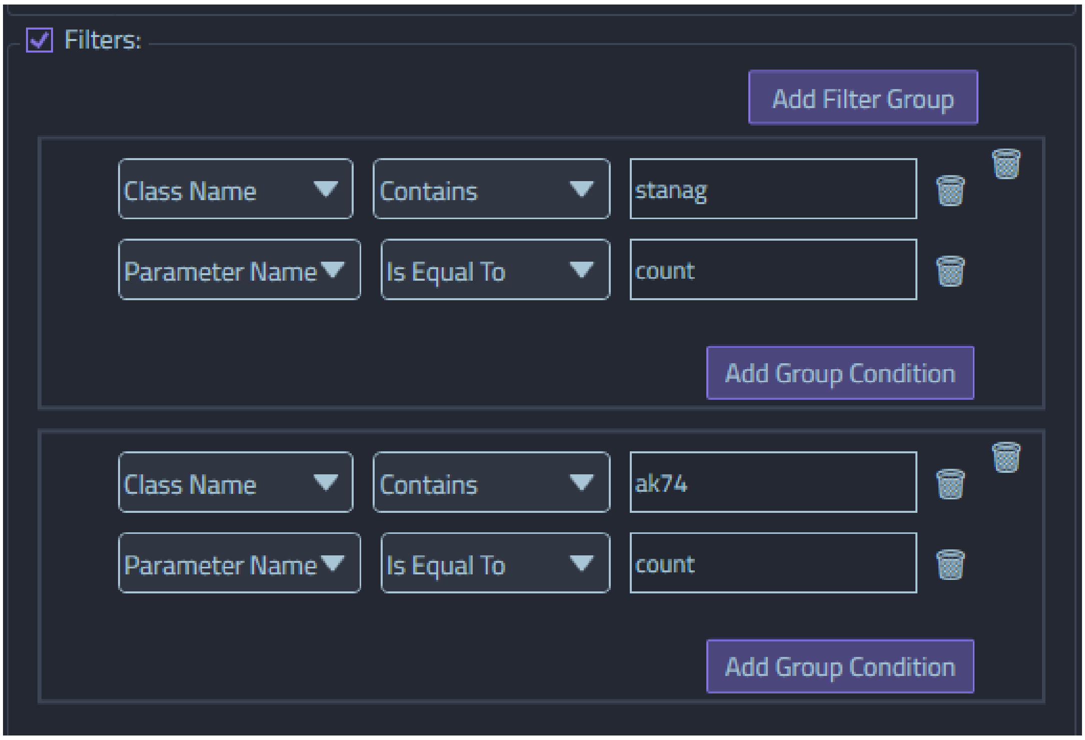


☐ Filters:

Each filter group is considered as an "or" condition during export, meaning that you can target multiple specific areas with separate filter groups.

Each filter condition inside a group is considered as an "and" condition, meaning that in order for the group to allow an entry through the filter, all filter conditions must be met inside the group.

Example: A filter set up to only export the `count` parameter values for items that contain either `stanag` or `ak74` would be constructed as follows:



☑ Filters:

Class Name ▼ | Contains ▼ | stanag

Parameter Name ▼ | Is Equal To ▼ | count

Class Name ▼ | Contains ▼ | ak74

Parameter Name ▼ | Is Equal To ▼ | count

You can read the filter conditions shown here as follows.

Only export:

- Where the Class Name contains `stanag`.
- And, it must also match where:
  - The Parameter Name is equal to `count`.

Or:

- Where the Class Name contains `ak74`.
- And, it must also match where:
  - The Parameter Name is equal to `count`.

This filter, when used to export to `.csv`, results in a `.csv` file that lists all magazines with their `count` parameters, and corresponding values, in the following syntax:

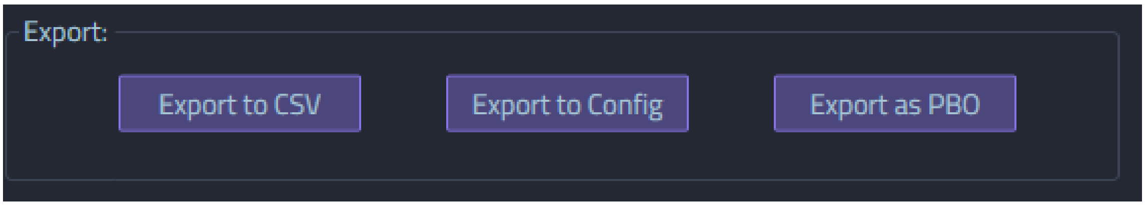
```
,ConfigFile >> CfgMagazines >> vbs2_mag_30rnd_545x39_Ball_ak74,count,30
,ConfigFile >> CfgMagazines >> vbs2_mag_30rnd_545x39_Trace_ak74,count,30
```

This filter (when used to export to configuration or as a `.pbo`) results in a patch file that only updates the filtered magazines with the `count` value. Here it makes sense to enable **Include Pending Changes**, or to select **Changes Only**, since there is no need to change the original magazine values to remain at their original values using a patch file.

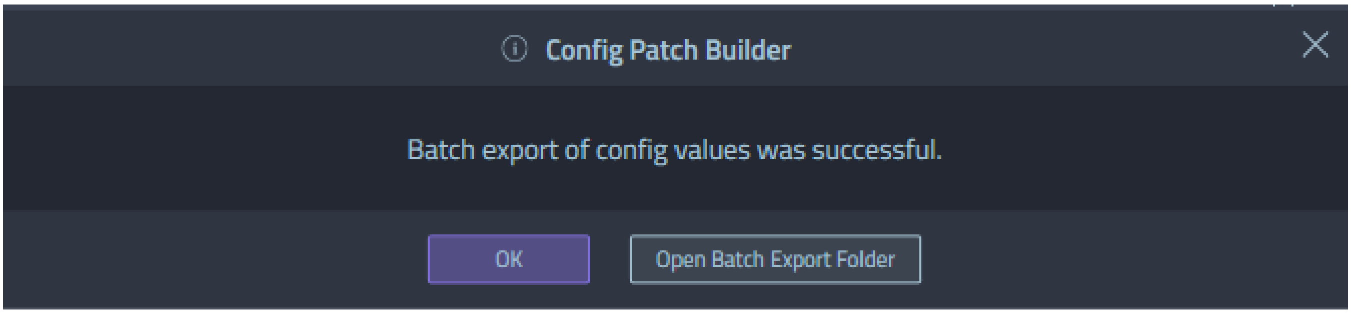
2.2.3 Export Modes

The CPB offers three separate export modes:

Export Mode	Description
Export to CSV	Generates a spreadsheet of your exported data.
Export to Config	Generates a `config.cpp` file that is able to apply your changes to VBS.
Export to PBO	Same as above, but with automatic `.pbo` packing and saving to VBS (this option requires you to set your VBS installation directory in the CPB settings).



Once completed, a message informs you of successful export. A shortcut button (**Open Batch Export Folder**) lets you directly navigate to the output folder.


Config Patch Builder

Batch export of config values was successful.

2.2.4 CSV Export Syntax

The syntax follows a very simple `.csv` (Comma Separated Value) table syntax. The first line declares the cell types, with every other line being data that is to be imported and interpreted.

Syntax	Description
Entry Type	Controls whether the imported entry is a name-value pair (parameter), or imported as a class. When importing a name-value pair, this cell is left empty. For class name-base class pairs this cell contains the letter `C`.
Config Path	Controls the location of the name-value pair or class name to be created.
Name	Controls the name of the parameter or class to be added.
Value	Controls the value of the parameter to be added, or the name of the base class for a class addition.

EXAMPLE

```
EntryType,ConfigPath,Name,Value
,configFile >> CfgVehicles >>
 vbs_nl_army_boxer_engineer_wdl_gmg_x,transportSoldier,8
,configFile >> CfgVehicles >>
 vbs_nl_army_boxer_engineer_wdl_gmg_x,NewArray,{item1,item2,item3}
C,configFile >> CfgVehicles >>
 vbs_nl_army_boxer_engineer_wdl_gmg_x , newClassWithoutBase,
C,configFile >> CfgVehicles >>
 vbs_nl_army_boxer_engineer_wdl_gmg_x , newEventhandlers,configFile >>
 CfgVehicles>>vbs_nl_army_boxer_engineer_wdl_gmg_x>>eventhandlers
,configFile >> CfgVehicles >>
 vbs_nl_army_boxer_engineer_wdl_gmg_x >> newEventhandlers,newEntry,123
```

3. CPB Example - M1A1 Tank

Perform the following task. For the US Army Tracked - Desert M1A1 tank, make three changes:

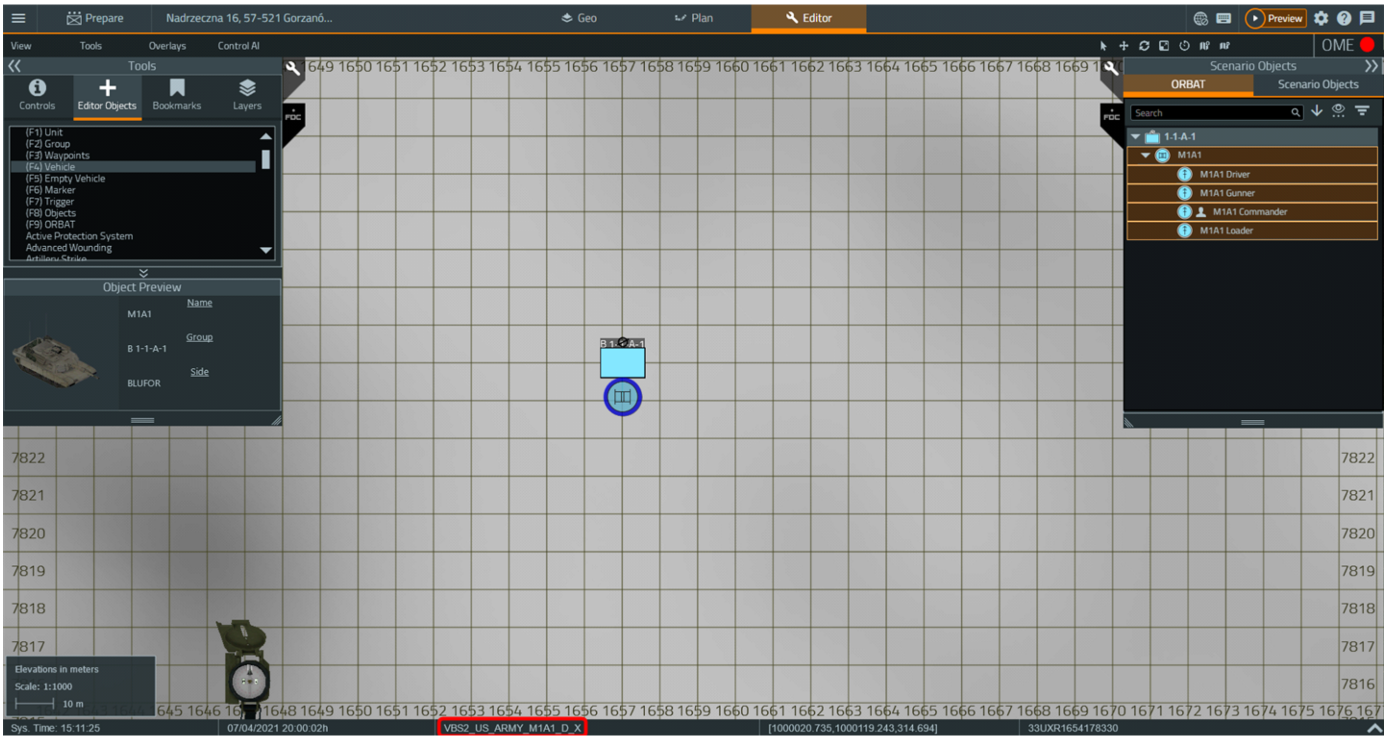
- Increase `enginePower`.
- Modify the `hiddenSelections` array values to contain an extra item.
- Introduce a new class called *`newClassName`*, with a base class using `turrets`.

First, you need to know the configuration class name of the asset that you wish to modify. If you are familiar with the asset, you probably have the class name readily available. If you are unsure about the exact asset class name, you can use the VBS Editor to help you find it.

Follow these steps:

1. Launch a Battlespace.
2. Enter the **VBS Editor**.
3. Place the **asset** you wish to edit (in this case, **US Army Tracked - Desert M1A1** tank).
4. Select the **asset**.

In the bottom bar of the VBS Editor interface, you see the class name displayed. The selected asset is an M1A1 tank, with the class name `VBS2_US_ARMY_M1A1_D_X`.



TIP

Generally, all interactable assets such as vehicles, objects, houses, and characters fall into the `CfgVehicles` class configuration domain.

Weapon classes are part of `CfgWeapons`. Magazine classes are part of `CfgMagazines`. Ammunition classes are part of `CfgAmmo`.

3.1 Loading a Config Cache

The CPB functions using the VBS internally generated config cache. This is automatically created and updated every time you launch VBS.

Follow these steps:

1. Launch the CPB and click **Load Config**.

2. Navigate to the **cache** folder of your VBS4 installation, and select the `config.cache` file.

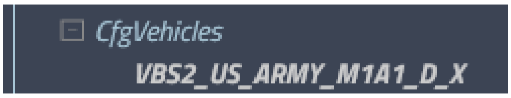
```
\VBS4_Installation\cache\
```

3.2 Simple Search

Perform a simple search for the M1A1 tank.

Follow these steps:

1. In the Config Class Tree (on page 7), click **+** to unroll the **CfgVehicles** category, and wait for the caching to complete.
2. Enter the **class name** (or any part of it) into the **Filter** box at the top of the Config Class Tree (on page 7) window.
3. From there, select the correct class name in the list.



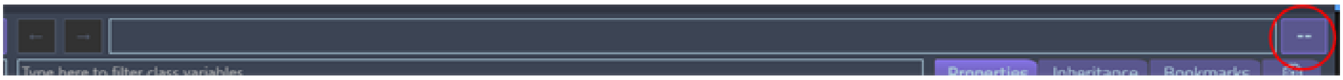
3.3 Manual Address Finding

As an alternative to just browsing for the class, you can also enter an exact configuration path, and have the CPB resolve and load the class immediately.

Follow these steps:

1. Ensure that you have the manual address bar enabled (do this by clicking the **Edit** button at the top-right of the CPB interface).

   It changes into a button showing **--**.

2. In the address bar, type:

```
configFile >> CfgVehicles >> VBS2_US_ARMY_M1A1_D_X
```

3. Press **Enter** to navigate directly to the entry.

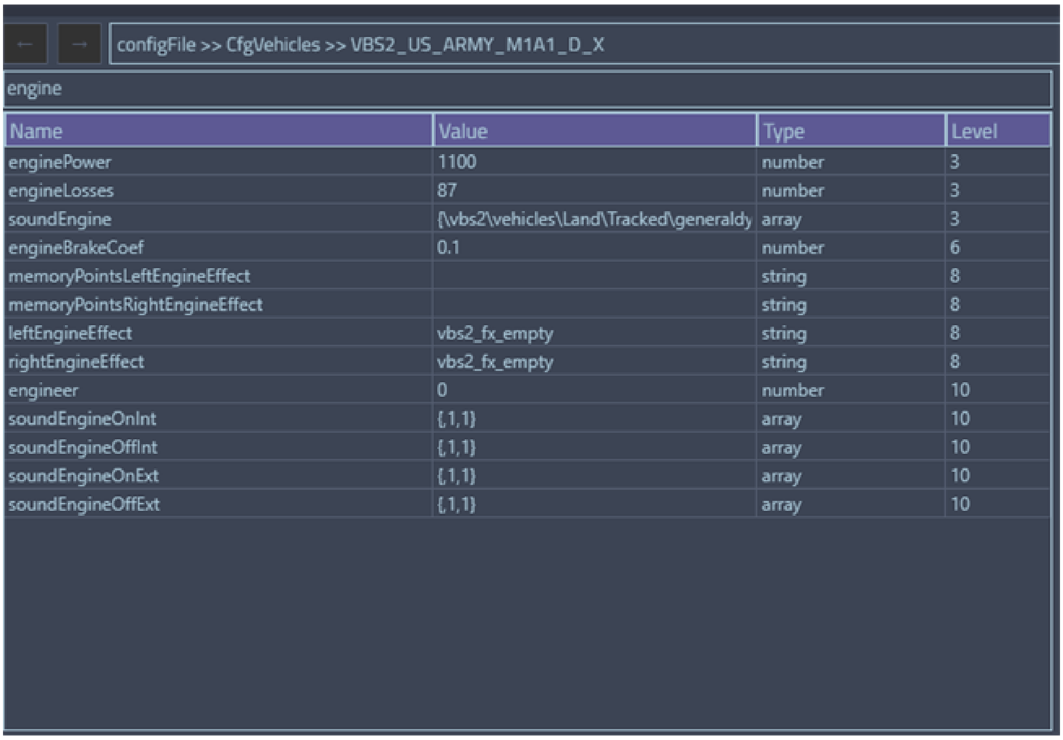
> **NOTE**
>
> The address bar also lets you copy SQF-formatted configuration paths that include quotation marks. Both of the following are valid inputs, and are accepted by the address bar.
>
> ```
> configFile >> CfgVehicles >> VBS2_US_ARMY_M1A1_D_X
> configFile >> "CfgVehicles" >> "VBS2_US_ARMY_M1A1_D_X"
> ```
>
> This simplifies working with SQF configuration paths and CPB.

3.4 Editing Values

With a class brought up in the Config Main View (on page 7) window, you can inspect its entire configuration class, including every inherited parameter.

If you want to make a vehicle engine more powerful, for example, you can either find the exact parameter name you are looking for, or use the search bar at the top of the Config Main View (on page 7) window.

Filtering for `engine` results in the following view. The parameter relevant for your intended change is the first one, `enginePower`.

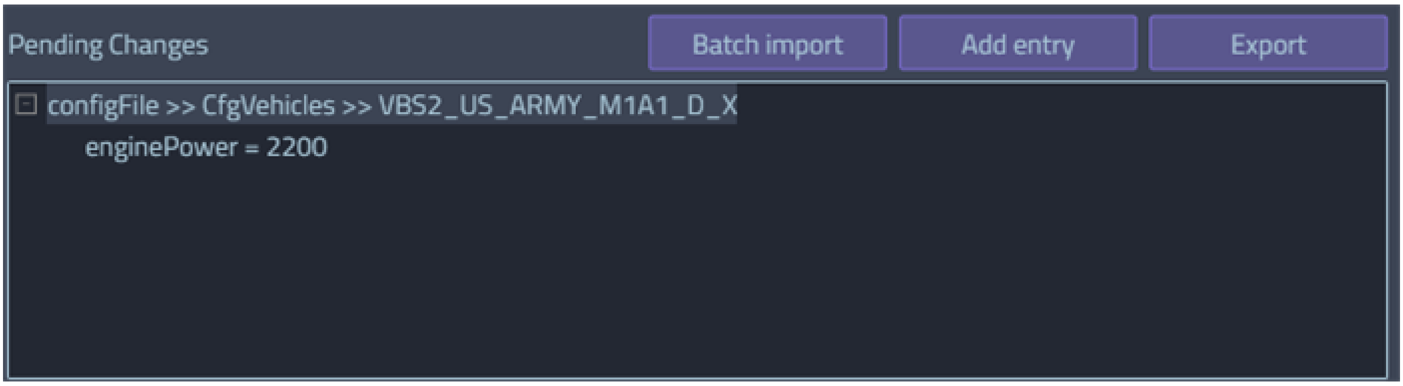


configFile >> CfgVehicles >> VBS2_US_ARMY_M1A1_D_X

engine

Name	Value	Type	Level
enginePower	1100	number	3
engineLosses	87	number	3
soundEngine	{\vbs2\vehicles\Land\Tracked\generaldy	array	3
engineBrakeCoef	0.1	number	6
memoryPointsLeftEngineEffect		string	8
memoryPointsRightEngineEffect		string	8
leftEngineEffect	vbs2_fx_empty	string	8
rightEngineEffect	vbs2_fx_empty	string	8
engineer	0	number	10
soundEngineOnInt	{1,1}	array	10
soundEngineOffInt	{1,1}	array	10
soundEngineOnExt	{1,1}	array	10
soundEngineOffExt	{1,1}	array	10

Double-click the current value (`1100`), and change it to `2200`. Your change is added to the Pending Changes (on page 7) window.



⊟ configFile >> CfgVehicles >> VBS2_US_ARMY_M1A1_D_X

enginePower = 2200

TIP

Every change you make to the entire configuration cache is listed in the Pending Changes (on page 7) window. You can click on entries here to quickly jump between them in order to bring up a specific class in the Config Main View (on page 7) window.

3.5 Deleting Pending Changes

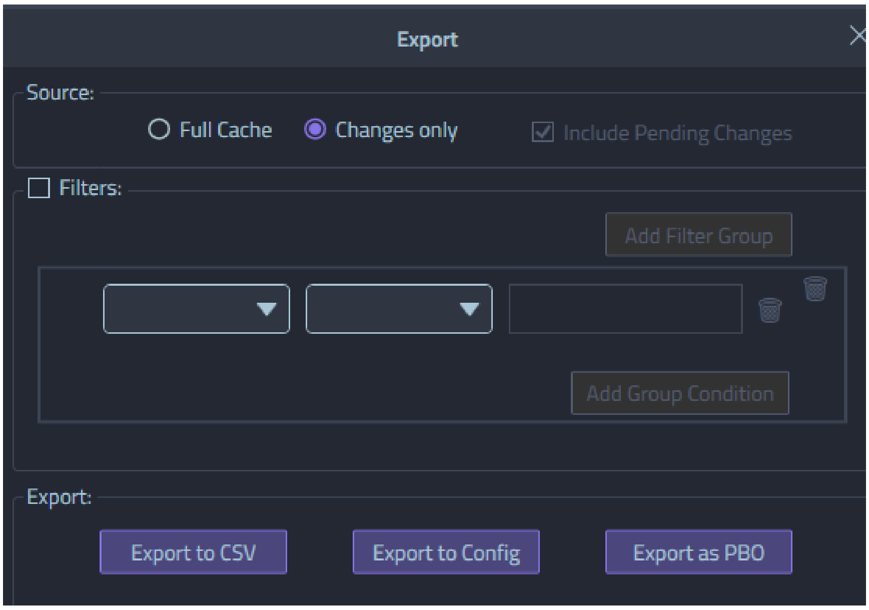
To delete a pending change, click the item in the Pending Changes (on page 7), and press **Delete** on your keyboard.

3.6 Export Changes

Once you have all of your configuration changes completed, you are ready to generate a patch file.

Follow these steps:

1. Click **Export**.
2. Select **Changes Only**.
3. Do not activate any filter.
4. Select **Export as PBO**.



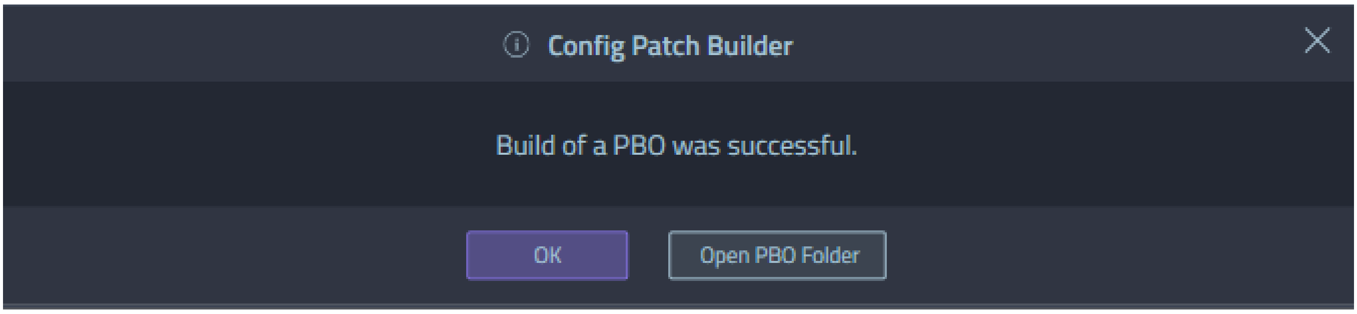
Export

Source: ○ Full Cache ● Changes only ☑ Include Pending Changes

□ Filters:

Export:

This generates a time stamped folder that contains the configuration patch with your changes. It also automatically creates a `.pbo` file and saves it to your VBS4 installation, so that it is loaded the next time you start VBS.



Config Patch Builder

Build of a PBO was successful.

Launch VBS4 to test the changes you made.

NOTE

Follow these steps:

1. Copy the generated `.pbo` file to:

   ```
   \VBS4_Installation\myData\Blue\content\
   ```

   This installs the patch for VBS4.

2. If VBS4 is currently running, restart it to refresh the config cache, and have the new changes from the file activate.

The CPB automatically generates patch files with a timestamp. While this avoids duplicate folders and files, it can lead to confusing file management.

To remedy this problem, you can rename the `.pbo` file from:

```
configpatch_20230821110615.pbo
```

To:

```
configpatch_m1a1engines.pbo
```

The name of the `.pbo` file is irrelevant for VBS to function. It is merely there to help you to organize your files efficiently.

3.7 Update the M1A1 Tank Using CSV

For more advanced users, you can save time and do the anticipated changes entirely using `.csv` files.

As a reminder, you intend to make the following changes:

- Increase `enginePower`.
- Modify the `hiddenSelections` array values to contain an extra item.
- Introduce a new class called *`newClassName`*, with a base class using `turrets`.

Constructing these changes using `.csv` requires the following file:

```
EntryType,ConfigPath,Name,Value
,ConfigFile >> CfgVehicles >>
 VBS2_US_ARMY_M1A1_D_X,enginePower,2200
,ConfigFile >> CfgVehicles >>
 VBS2_US_ARMY_M1A1_D_X,hiddenSelections,{swap_ext,swap_sprocket,thirdNewItem}
C,ConfigFile >> CfgVehicles >>
```

```
VBS2_US_ARMY_M1A1_D_X,newClassName,ConfigFile >> CfgVehicles >>
VBS2_US_ARMY_M1A1_D_X >> turrets
```

Follow these steps:

1. Create this file and call it `m1a1update.csv`, and save it to a location where you wish to keep your configuration patch `.csv` files (it is recommended to keep these files separate from the autogenerated `.csv` files that CPB can export).
2. Launch the CPB, and click **Batch Import**.

3. Navigate to the location of your `.csv` files and select `m1a1update.csv`.

The changes are applied, and listed in the Pending Changes (on page 7) window.