Cope's Dawn of War II Toolbox

User Guide

Copernicus

June 13, 2010

Contents

1 Setup

1.1 Installation and configuration

Installing the tool is simple: Extract all the files from the archive you downloaded to any destination and run *CopeModToolDoW2.exe*. Up on the first start of the application, it might ask you to specify your *Steam* directory.
You shall then first check your options: Click on *Tools->Options* to open the *Options* dialog (see figure 1). I recommend not changing anything besides unchecking the *Use Chaos Rising* checkbox if you do not own *Chaos Rising*. You should also check that the *Steam Executable* path points to your *steam.exe*. Both of the mentioned settings are relevant for quickly starting your mod from the tool.

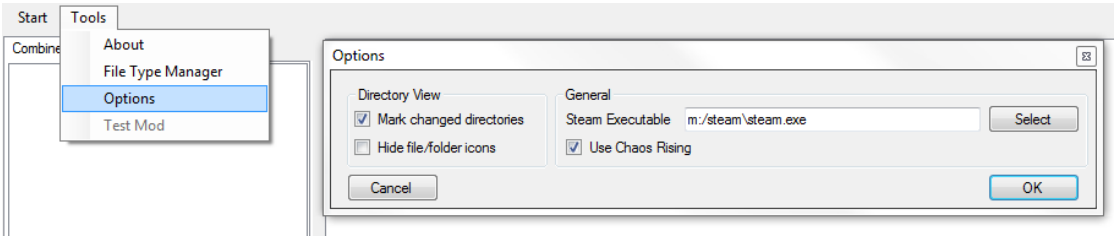

Figure 1: The *Options* dialog.

1.2 Installing file type plugins

To install a file type plugin you need to move the DLL of the plugin into the *plguins/filetypes* subfolder located in the directory of the tool itself. Start the tool and double check the *File Type Manager* (*Tools->File Type Manager*) and make sure that the tool is actually using the new plugin for the selected file type.

1.3 Uninstalling the tool

Uninstalling is as easy as installing: Simply delete all the files you extracted when you installed the tool.

2 Your First Mod

In this tutorial you will learn how to create a basic *Dawn of War II*modification. Our first goal is to make some simple changes: We will increase the squad size of the *Space Marines' Scout* squad while keeping it's original requisition cost.
This tutorial does not only aim at teaching you some basic modding skills but also at giving you some more insight into *Dawn of War II*and the use of my tool.

2.1 The right tool

Since the release of *Dawn of War II*the community created a variety of tools for modding the game. Starting with *Corsix' RBFConv* (which still works but is somewhat more complicated to use) people modded the game, later on *MartinNJ's DoW2 Mod Studio* was released. This tool allowed people to easily create new mods; it combines multiple other tools (like the afore mentioned *RBFConv*).
This guide will use my own tool as I'm most familiar with it. If you experience a crash make sure to post the *log.txt* from the tool's directory in the tool's thread on RelicNews. You can find a link to the most recent version of the tool in that thread as well.

2.2 Creating a new mod

Start the tool and click on *Start* in the upper left corner of the tool and select *New DoW2 Mod* from the menu. You'll see a new dialog (see figure 2 on the next page). First, click the *Browse* button and search for a file called *DoW2.module* which should be in your *Dawn of War II*directory. We'll get to so called *module* files in a second. Now enter your mod's name into the second text box. We'll just go with *z_MyFirstMod* this time. I like using the *z_* prefix to keep all my mods grouped but you really don't need to use it at all ;). The *Displayed Mod Name*, the *Description* and the *Mod Version* are optional, we will just leave them blank. Hit OK. The tool will now take some time to load all the files required to continue, this may take some time. For me it's usually a few seconds.

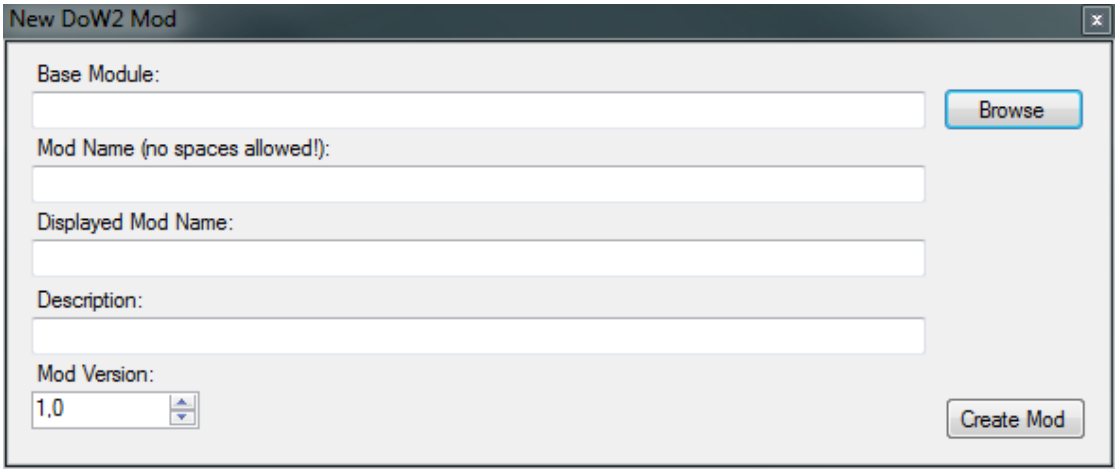

Figure 2: The *New DoW2 Mod* dialog.

2.3 Module files and file management

Every mod for *Dawn of War II*needs a so called *module* file. This file should be placed in the *Dawn of War II*directory. A *module* file provides basic information about a mod like which files to load, what this mod is called etc.. It's essential! *Module* files can easily be edited with notepad or something similar.
I recommend taking a look into your *module* file which the tool created for you when you used the *New DoW2 Mod* dialog (it also created a folder with your mod's name in the *Dawn of War II*directory). It's name is just the same as the name of your mod. Having opened it you'll see different sections, the most interesting ones for us are *global*, *attrib:common* and *data:common* (which usually is at the end of the file).
Global contains general information, it's pretty much self explanatory. *Attrib:common* and *data:common* both list archive files (*SGA archives*) and directories. The game will normally only load the archives (using the *-dev* command line parameter it will also load the files from the directories). Everything from the *attrib:common* section contains attribute information for the game, e.g. how strong a certain unit is, in which way an ability works etc.; things listed under *data:common* contain data like textures, models, scripts, maps, sound-containers and similar things.
If there are duplicate files in the archives of directories the one with the highest priority is used. Files from folders always have a higher priority than those from archives and the archive (or folder) entry with the lowest index has the highest priority among archives (or folders).

```
[data:common]
folder.01 = GameAssets\Data
archive.01 = GameAssets\Archives\221_data.sga
archive.02 = GameAssets\Archives\220_data.sga
archive.03 = GameAssets\Archives\210_data.sga
archive.04 = GameAssets\Archives\200_data.sga
archive.05 = GameAssets\Archives\GameArt_DELTA.sga
```

Figure 3: Excerpt from the *data:common* section from the *Dawn of War II*module file.

For the sake of simplicity your mod should only use one folder for the *data:common* section and one for the *attrib:common* section. My tool only loads up the first directory to simplify the whole process.
You might also notice that your mod has a seperate folder for *attrib* and *data* while the original *DoW2.module* does not. This is also a limit set by my tool which will help you to keep the overview at all times. Changing that might crash the tool, so always ensure to have a dedicated directory for *attrib* and *data*.

2.4 The user interface

By now the tool should have finished loading your newly created mod, it's time to take a closer look at the ***G**raphical**U**ser**I**nterface* (*GUI*, see figure 4 on the following page). At the left hand side there's the **Directory View**, you'll notice that there's a tree view with two main nodes: *ATTRIB* and *DATA*. These nodes contain all the directories and files specified in the *module* file. Blue nodes represent virtual files (files from the archives) whereas red nodes represent local files.
Using the **Mode Selector** right above the Directory View you can switch between a virtual only tree view and a combined tree view which shows both local and virtual files. The virtual only tree view comes in handy when there's a local file with the same name and position as a virtual file and you still want to access the virtual one. Right clicking a node other than *ATTRIB* and *DATA* will open up a drop down menu using which you can extract files.
Double clicking a file from the Directory View will try to open it. If there's a suitable plugin found for the file type you're trying to open a new tab will be created in the **File Application View**. The content of the tab depends on the file type. You can use the buttons from the **File Tab Control** to close a tab or even all tabs at once.
There also is a quick way for testing your mod: Click *Tools* in the main menu at the upper left and select *Test Mod*. The tool will then start *Dawn of War II*using your mod and *-dev* (so the game loads all the files from the directories in addition to those from the archives). The latter one is important because you can not edit any files from the archives; as soon as you save the file the tool will extract the file and save the modified version locally.

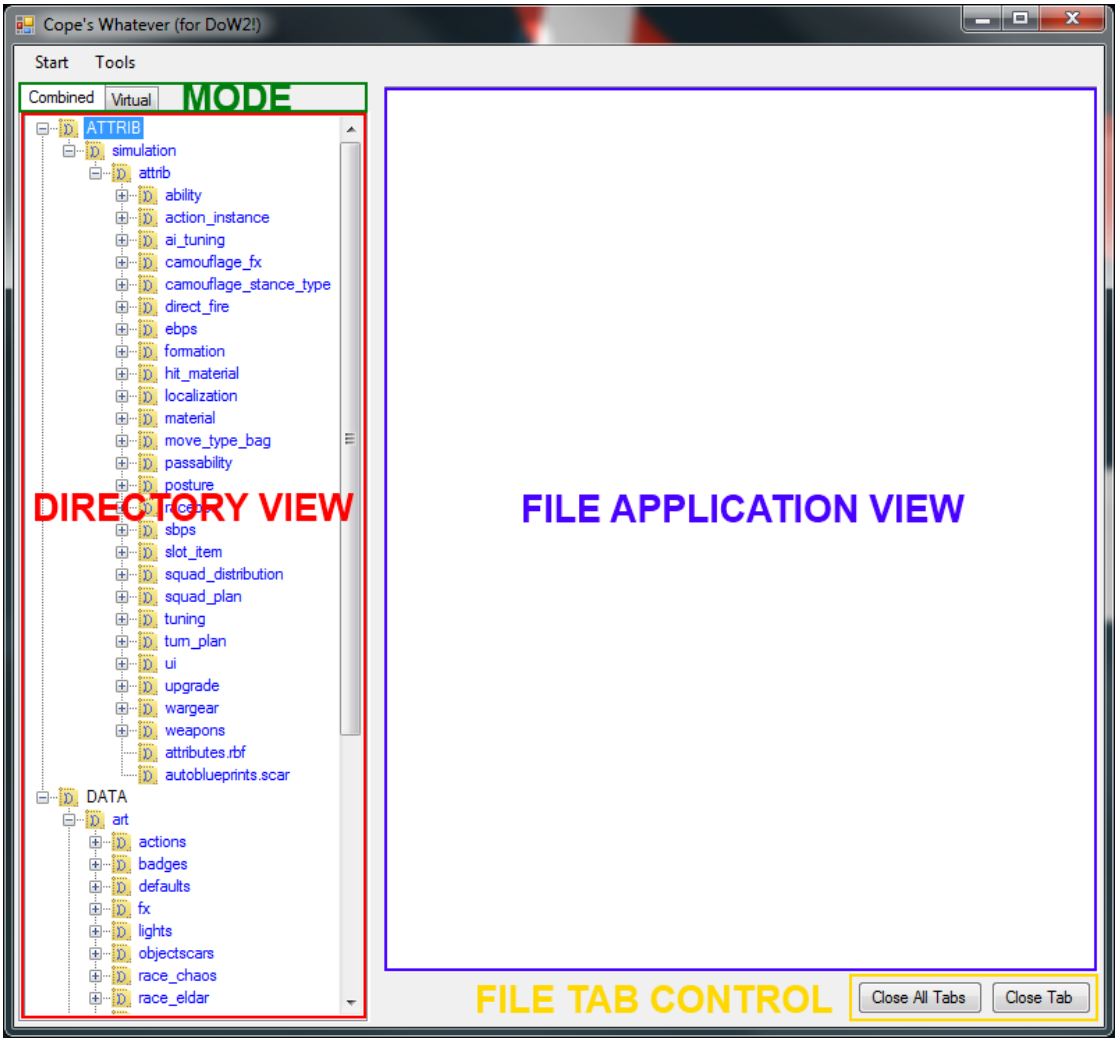

Figure 4: The main GUI of the tool.

2.5 Modding the Space Marines' Scout squad

Now that you know about the basic theory and the tool it's time to get modding!
We'll need to change some *attrib* files in order to achieve our goals. Those files' type is usually *RBF* (Relic Binary File) and fortunately my tool already ships with a custom made RBF-Editor. Navigate to *ATTRIB/simulation/attrib/spbs/pvp/race_marine/troops* using the Directory View and double click *sm_scout_marine.rbf* to open that file (see figure 5 on the next page). All the files in the *SBPS* folder represent ***S**quad**B**lue**P**rints*, the *PVP* folder indicates that the files it contains are used in PVP matches. As you can see on the right hand side of the tool, RBF files have a tree-like structure. They're basically *XML* files burnt to some binary format.

2.5.1 Changing the number of squad members

You'll need to change a few values in that file (marked in red in figure 5): *squad_loadout_ext* contains the information about what the squad consists of, it's got an entry called *unit_list* which itself has an entry for every different entity in that squad. *max* and *num* of that entry control how many instances of the specified entity (*type*, marked in green in figure 5) the squad starts with (*num*) and can have at maximum (*max*). Set both values to 4.

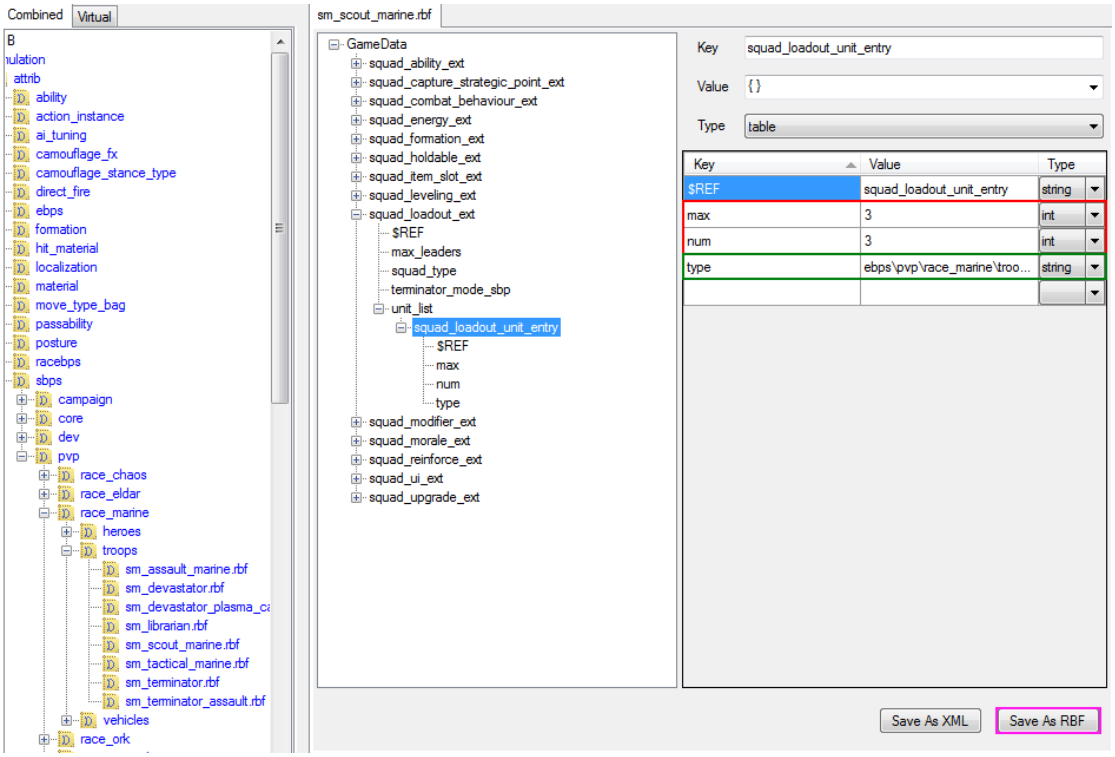

Figure 5: *sm_scout_marine.rbf* (SBP) opened with Cope's RBF-Editor plugin.

2.5.2 Modifying the squad cost

Searching the SquadBluePrint for any way to control the squad's cost we'll notice that there is none. Prices in *Dawn of War II*are calculated dynamically, each entity stores information about it's cost and the SquadBluePrint just sums it all up. Thus we need to edit the *EBP* (***E**ntity**B**lue**P**rint*) of the *Space Marines' Scout*. Navigate to the *EBP* specified by *type* (green marking in figure 5) and open it. Hit *Save As RBF* (marked in pink in figure 5) and close the *SBP* file.

In the *EBP* search for *cost_ext/time_cost/cost* and set *requisition* to 52.5 ($4 \times 52.5 = 3 \times 70 = 210$). Save and close the file.

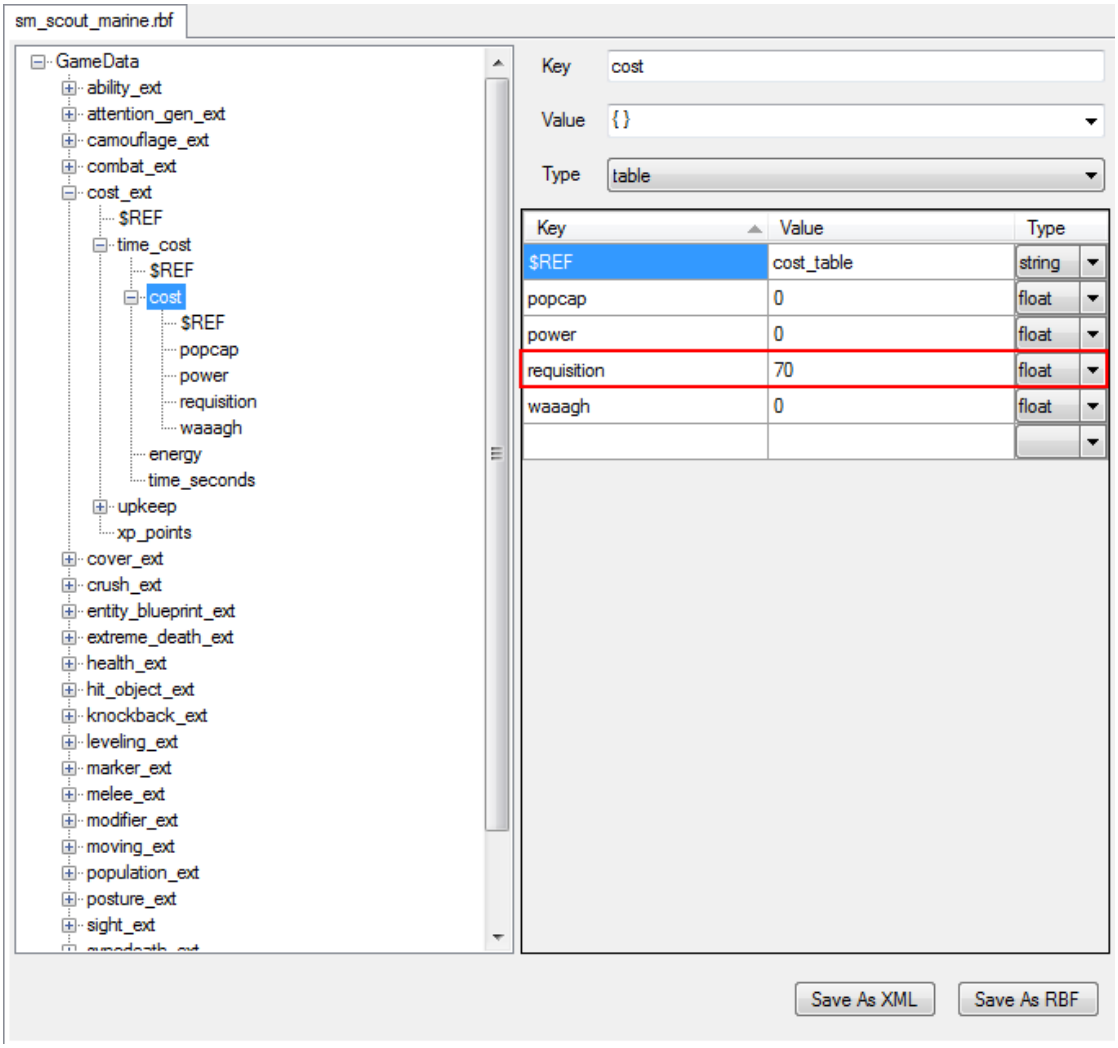

Figure 6: *sm_scout_marine.rbf* (EBP) opened with Cope's RBF-Editor plugin.

2.6 Testing the changes

Having made our exciting changes we'd certainly like to see them ingame. Select *Tool->Test Mod* to start the game usign your mod and start a new PVP match. I recommend playing against any AI enemy as you can only play with other players if they've got the exactly same mod installed and running. Let's see the changes ingame (figure 7 on the following page).

Figure 7: Our changes ingame.

2.7 Distributing your mod

Before even thinking about distribution you should ensure that (if you have any third-party files e.g. from other mods in your mod) you own the right to publish anything shipping with your mod.
Releasing your mod is quite easy for you: Select *Tools->Release Mod* to pack your mod into archives (one for *attrib* and one for *data*). In the new dialog (see figure 8 on the next page) make sure to check *Create Installation Instructions* and *Create Simple Launcher*. Select an output folder where the tool shall store the files. The tool will then create a new module file, a launcher and some instructions on how to install the mod.
Packing your mod might take some time depending on the amount of files it needs to pack. You'll be notified as soon as the tool has finished preparing your mod for distribution, you may then choose to open the output folder. In the output directory you will find a directory which contains your mod's archives and all the other files generated by the tool. The only thing left to you is creating a ZIP archive and uploading the mod to a website[1] so people can download it.

[1] e.g. http://www.filefront.com

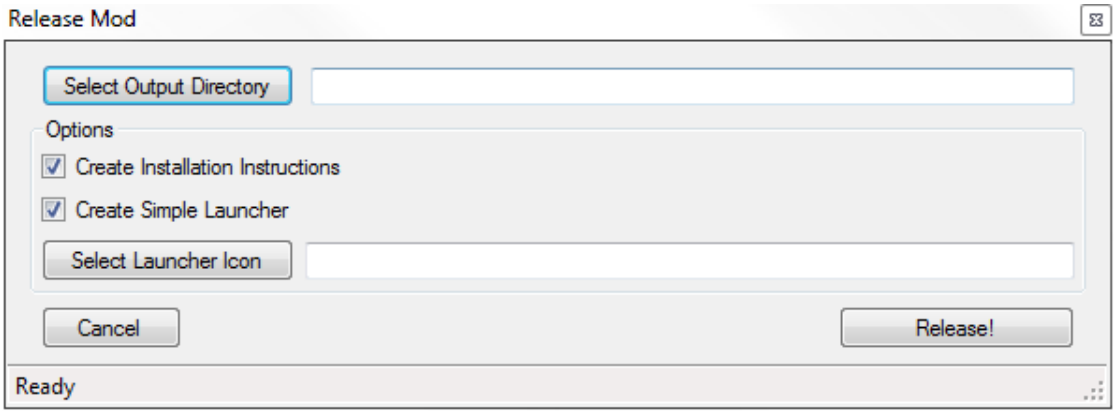

Figure 8: The *Release Mod* dialog.

2.8 Closing words

Congratulations, you've successfully created your first *Dawn of War II*modification - yet there still is a lot to learn, much more than I can cover in this tutorial. You should now try to learn about *UCS* as you'll certainly need it in the near future, advanced RBF-editing will come in handy as well. *SCAR* scripting is interesting for anyone who likes to change the concept of the game itself instead of just changing some numbers.

3 Misc Features

Loading Mods, SGAs and other files
The tool is capable of loading both mods (*Dawn of War II*module files) and SGA archives; it also supports some other file types. You can review which plugin is used for what file type by opening the *File Type Manager*(*Tools->File Type Manager*).

Drag and drop
There's support for drag and drop; you can simply drop module files etc. on the tool to open them. However, some features don't work for a file which is not associated with a mod.

Default tool for module files
Using *Open with...* from the Windows context menu for files you can set this tool as the default application for *Dawn of War II*module files.

Advanced RBF-Editor
In the past if often took a long time to convert and modify RBF files. With the RBF-Editor shipping with this tool it's easier than ever before to change any aspect of RBFs.

If the tool recognizes any filepaths in RBFs it will automatically present you a collection of other files from that directory so you can quickly modify relationships between RBFs.

RBF-dictionary

Using the RBF-dictionary you can specifiy default-values and search-paths for certain keys of RBF-values. Every release already includes a quite complete dictionary, but you may of course add your own entries. Right-click any value in the RBF-editor (in the table view) and select *Add Value To Dictionary* to add the selected value to the corresponding key. **Experienced users only**: Manually editing the dictionary allows for more control and grants you the ability to add custom searchpaths. Open the tool's directory and look for *rbf_dictionary.txt*. If that file does not exist, simply start and close the tool once. Inside it you'll find instructions on how to edit the dictionary manually.

RBF-library

The RBF-library allows you to store *snippets* (RBF-Values) from RBFs and tag them. Tagging your entries enables you to quickly insert them in the right places; e.g. tagging an entry from the RBF-library with *actions* will include it in the dropdown list of *Insert From Library Into Table* everytime you right-click on a table called *actions*. RBFs can of course have multiple tags. Adding values to the RBF-library is easy: Right-click any value in the RBF-editor's treeview and select *Copy Into Library*. You'll be prompted to enter a unique name for your entry and at least one tag. In the RBF-editor click the *Open RBF-library* button to open the RBF-library in a seperate window. You can copy values from the library as you'd do it in the RBF-editor. **Experienced users only:** Editing the RBF-library from outside the tool is just as easy: Open the directory of the tool and search for *rbf_library.txt*. If that file does not exist, simply start and close the tool once. Inside it you'll find instructions on how to edit the library manually. But beware and doublecheck everything, the tool isn't forgiving about mistakes and you can pretty easily screw up your whole library.

UCS support

It's easy to quickly add new UCS strings and insert their indices right away, the RBF-Editor even looks up indices for you! The tool will automatically create a new UCS file for your mod starting with an index of your choice.

Fully compatible to *Corsix' RBFConv*

The included RBF-editor is fully compatible to *Corsix' RBFConv*, it allows you to copy any value as a string readable by *Corsix' RBFConv*and it is capable of converting strings created by *Corsix' RBFConv*to RBF values.

RelicChunky Viewer/Editor

You can view *RelicChunky* files using this tool. A lot of files used in *Dawn of War II*actually are *RelicChunky* files. Just double-click any file, if it's a *RelicChunky* the *RelicChunky* Viewer/Editor will open, otherwise it will be opened using a very basic text-editor. If you don't know what *RelicChunky* is then you'll probably never need this

feature.

Plugin support
If the tool does not support a specific file type, feel free to write a plugin for it on your own. You can find a base file for plugins in the *filetype plugin example* subfolder. As the tool is *.NET* based you can use any .NET language; the base file however is in C#.

DDS/TGA preview
The tool ships with a plugin which allows you to view DDS and TGA files from within the tool.

4 FAQ

I'm trying to load a mod but the tool tells me that the mod's *data:common* and *attrib:common* directories overlap. What shall I do?
Mods created by the tool itself will never have overlapping *data:common* and *attrib:common* directories, so it seems like you're trying to edit another mod or the original *Dawn of War II*module file (*DoW2.module*). If you're trying to edit the original module file, better create a new mod using the tool and select the original module file as a base module. Otherwise you need to manually edit the module file and change the first directory entry of the *attrib:common* section so it differs from the first directory entry of the *data:common* section. See section 2.3 of this document for further help.

The tool crashed!
Post the logfile (*log.txt*) from the tool's directory in the tool's thread at RelicNews.

What are tables and values?
Every RBF file consists of lots of *RBF-values* which in terms consist of a *key* and a *value*. There are five different types of values: *integers*, *floating point values*, *strings*, *booleans* and *tables*. The *key* is always a string though. A *table* value is a value which can hold other *RBF-values*.

What is a *Corsix' style* string?
A *Corsix' style* string is a string which is compatible with *Corsix' RBFConv*. It's another representation for the content of RBF files and usually looks like this:

```
ability_ext: {
| $REF: "entity_extensions\ability_ext";
| abilities: {
| | ability: "ability\pvp\race_marine\sm_repair_pvp";
| | ability: "ability\pvp\race_marine\scout\sm_damage_aoe_grenade";
| | ability: "ability\pvp\race_marine\scout\sm_stun_aoe_knockback";
| };
```

```
};
```

Corsix' style is useful for posting parts of a RBF file (or the whole file) in forums or any other text-based way of communication. Using the included RBF-Editor you can convert any value to *Corsix' style*: Just right-click the value and select *Copy As Corsix-String*. You can also convert any *Corsix' style* string to RBF using the inbuilt RBF-Editor. Right-click any table and select *Insert Value From Corsix-String Into Table*, then insert your string and hit *Done*.

What is a *UCS* file?

A UCS file contains localized text for every language. You can find the original UCS file for your language in *GameAssets/Locale/<your_language>/DOW2.ucs*. Your mod however should use an own UCS file. The tool will automatically create a UCS file as soon as you add a UCS string (*CTRL + U* or *Tools/Add UCS String*). Every UCS string has an index, everything which uses UCS strings will refer to a specific string using it's index.

How can I add maps to my mod?

This depends on whether the map is already packed into an SGA or not. If it is, use *Tools/Mod Settings* to add it. You'll need to select the proper section (which is *data:common*), click the *Add SGAs* button and select the maps you'd like to add. The tool will then copy these maps into your mod's archive directory. If the map(s) you'd like to add are not packed, just copy them to *<your_mod's_directory>/Data/maps/pvp* or *<your_mod's_directory>/Data/maps/ffa* (depending on whether the map is FFA or not).

I'd like to change the name/version/description of my mod.

Use *Tools/Mod Settings* to change various global aspects of your mod.

I'd like to learn more about modding. Where can I get more information?

You should visit the modding forums on RelicNews.