Subject: More general Database handling
Posted by Sonarpulse on Sun, 08 Apr 2012 05:56:03 GMT
View Forum Message <> Reply to Message

Currently, Westwood3D can handle two "layers" of object databases: temps.ddb for map-specific changes, and objects.ddb for game-wide presets. This aproach I fine rather limiting. For example, if someone is working on two seperate mods that share some assets, or if one is making a map for a game like APB, whose presets are constantly changing, they must manually clone presets from one database.

Ideally, I think it would greatly enhance w3d if an arbitrary number databases could work as patches a là temps.ddb. So for example, somebody could make a mod pacalge that would inherit changes to the underlying game, and contain both mod-package-wide and map specific "temps.ddb". This would allow for presets to be managed in many more ways, giving much-needed flexibility to developers.

Naturally this same functionality could be also applied to string databases, allowing among other things translators to update their translations with much greater ease. (And partial/old translations to still work.)

Subject: Re: More general Database handling
Posted by StealthEye on Mon, 09 Apr 2012 22:53:44 GMT
View Forum Message <> Reply to Message

This would be nice in principle, but not a priority for TT to do imo.

Subject: Re: More general Database handling
Posted by Jerad2142 on Tue, 10 Apr 2012 05:00:18 GMT
View Forum Message <> Reply to Message

Maybe you guys should just use temps to ship updates back and forth, turn the temp files into objects presets and then delete the temp file, allowing you to create another new temp file for further updates.

Subject: Re: More general Database handling
Posted by Sonarpulse on Tue, 10 Apr 2012 05:14:22 GMT
View Forum Message <> Reply to Message

What happens if you drop a temps.ddb in /data? I have never tried it, but I assume it wouldn't work.

Subject: Re: More general Database handling

Posted by Jerad2142 on Tue, 10 Apr 2012 05:25:33 GMT
View Forum Message <> Reply to Message

Not sure if that works, but you could drop the presets in data after you update them; thus allowing you to bypass re-exporting always.dat and all that fun stuff.

Subject: Re: More general Database handling
Posted by danpaul88 on Tue, 10 Apr 2012 09:07:44 GMT
View Forum Message <> Reply to Message

Re-exporting always.dat isn't that bad really, only takes a minute or two and using the MIX patcher I developed for BHP (and any other mod team that wants it) patches to always.dat (and other .dat / .mix files) can be shrunk down to the size of the modified / new files within the MIX archive only.

AR have been doing this for our patches for years, typically our minor patches come out at about 30mb, even though they update the always.dat file (currently 400mb+).

Subject: Re: More general Database handling
Posted by Sonarpulse on Tue, 10 Apr 2012 14:23:56 GMT
View Forum Message <> Reply to Message

Well you can already use a always2.dat to make a dynamic patch (or your superior method to make a static patch). But I am just trying to patch objects.ddb. The basic reason I see patches as superior to overwrites is the "upstream" changes can be inherited with a patch.

In short, what your mix patcher and always2.ddb achieve on the archive level, I want to acceive on the (preset) file level.

Edit: But perhaps making a tool to combine to preset databases, as opposed to an engine modification to do that on he fly would be easier and safer. Is all the code for the TT preset and string database editors public? It would be an interesting feature to try to implement myself.

Subject: Re: More general Database handling
Posted by jonwil on Tue, 10 Apr 2012 16:22:56 GMT
View Forum Message <> Reply to Message

strings.tdb stuff is public.
objects.ddb code is not public (for various reasons that I cant go into) and isn't usable enough yet to be able to combine multiple preset files into a single one although its on the wishlist.

Subject: Re: More general Database handling

Posted by Sonarpulse on Tue, 10 Apr 2012 16:50:18 GMT
View Forum Message <> Reply to Message

Well maybe I wil just take a look at the string editor then. It won't help my project, but will help translators. And if there any similarities in the way strings and presets are handled, maybe it will help you guys.

Glad to hear it's on the wishlist.

Subject: Re: More general Database handling
Posted by Sonarpulse on Wed, 16 May 2012 08:15:06 GMT
View Forum Message <> Reply to Message

Not to bump my own topic, but I was just thinking, why don't you guys scrap the fancy binary object and string database formats, and just make the game read text files? hell the logic is already half there with the LE and tdbedit import and export functions.

Us modders would have the a benefit of a much easier-to-use format (especially because importing the text exports isn't always safe), and you guys would be spared from having to develop extra database editors.

Subject: Re: More general Database handling
Posted by saberhawk on Fri, 18 May 2012 22:08:27 GMT
View Forum Message <> Reply to Message

Sonarpulse wrote on Wed, 16 May 2012 01:15Not to bump my own topic, but I was just thinking, why don't you guys scrap the fancy binary object and string database formats, and just make the game read text files? hell the logic is already half there with the LE and tdbedit import and export functions.

Us modders would have the a benefit of a much easier-to-use format (especially because importing the text exports isn't always safe), and you guys would be spared from having to develop extra database editors.

That is never going to happen for many reasons. One of the big ones is the huge number of bugs that will be caused by replacing such a vital system, another is that *everything* would immediately be incompatible with 4.0.

Subject: Re: More general Database handling
Posted by Sonarpulse on Sat, 19 May 2012 07:01:05 GMT
View Forum Message <> Reply to Message

Well, as to incompatability, you could always keep the ability to read the binary formats and export their text counterparts. But, I can't really say anything to the issue of bugs. Thanks for letting me know.

Edit: While I've bumped this thread I guess I'll fire away with another question: Do any presets still need GDI or Nod in their name for hardcoded reasons, and if so can that easily be changed?

Subject: Re: More general Database handling
Posted by Sonarpulse on Mon, 18 Jun 2012 18:53:35 GMT
View Forum Message <> Reply to Message

Aha! the new ddbedit can now dump to a text file! Will RC 1 (or beta 6) bring the ability to re-import those dumps? Thanks so much, this new text file format is far superior than LE's tab delimited export (and LE import never seems to work). Having ddbedit import will effectively mean my dreams for workable text-based object handling are realized, as it doesn't matter what w3d actually uses if you can always mod presets as text.

On another note, I noticed tdbedit now has a menu item to import a preset database. Does that mean the tools are going to be combined?

EDIT:Comments: To make this a truly robust format it would be nice of have comments. It's basically an INI but with XML style end tags, so I think semi-colon comments would be sufficient.
ID references: It would be super convenient if preset numerical IDs were also in the ddb dump, that way presets that point to other presets can be easily determined. If it's too much of a pain to make them editable, just make them a comment at the beginning of a preset.
ID dereferences: This would be the icing on the cake: every time a preset points to another preset, put the name of the referenced preset as a comment next to the referring field (and ID).

Subject: Re: More general Database handling
Posted by Sonarpulse on Thu, 28 Jun 2012 03:22:44 GMT
View Forum Message <> Reply to Message

Ok, not to be a whiny, begging-for-features, post-bumping ass. But is there any chances that there will be any sort of working text import for presets? Even without all those fancy features I hoped for?

I am working on a mod: https://www.bluehellproductions.com/forum/index.php?showtopic=25411 that is mainly rigging, so the availability of text import greatly affects my workflow.

Subject: Re: More general Database handling
Posted by jonwil on Tue, 31 Jul 2012 07:42:44 GMT
View Forum Message <> Reply to Message

Adding numerical preset IDs to the ddb dump, that I will do for the next version. Resolving references is tricky due to how the dump code works.

Making a re-importer, that wont be happening anytime soon due to how complex such a thing would be.

And the reason tdbedit.exe reads objects.ddb is so that it can use the sound definitions in it to provide better editing for sounds attached to strings.

EDIT: Turns out our code already resolves numeric IDs to preset names e.g. KilledExplosion=Regular_Explosion or Engine Start Sound=Supply_Truck_Start