Audacious - Feature #602

Restoring XMMS "AUTO" (entry-specific) equalizer settings (patch/suggestion included).

December 09, 2015 08:16 - Jim Turner

Status:	New	**Start date:**	December 09, 2015
Priority:	Minor	**Due date:**	
Assignee:		**% Done:**	0%
Category:	core	**Estimated time:**	0.00 hour
Target version:	3.8		
Affects version:	3.7		

Description

Would you consider restoring the "Auto" button in the equalizer to allow for saving and using preset config files tied to specific songs / streams? I've implemented a patch that does this for the WinAmp skins for now (I use the WinAmp interface exclusively). This patch restores functionality to the Equalizer window's "Auto" button. When something's playing and one attempts to do "Export.Preset File", it prefills the default filename to save to <file/streamname>.preset instead of "<name>.preset" and sets the current/default save path to "~/.config/audacious/". If the user saves the .preset file under that name there, then, if the "Auto" button is active when the file/stream is opened to play, and a .preset file with that song's filename exists in the config directory, the song's preset file is then loaded for that song. Care is taken to file the "default" (eq settings from startup or that were in effect from the last entry that did NOT have an "auto-preset") so that they are properly restored when the song ends and a non auto-preset song begins. This works even if multiple entries with "auto-presets" are played in a row. This seems to be working pretty well for me, though you may have a better way or different idea for how to implement this feature, but this is a start! It does not implement XMMS's "directory-specific" presets are anything like that, just a simple song/stream-specific "auto-preset" option. I think I got all the patch files attched.

Regards,

Jim

History

#1 - December 10, 2015 04:37 - John Lindgren

Can you use "git diff" to generate your patches, or at least use unified diff format ("diff -u")?

#2 - December 10, 2015 04:40 - John Lindgren

Problem is, I see a block of code like this ...

```
358a412,413
>     bool prev_songautoeq = aud_get_bool("audacious", "equalizer_songauto");
>     s_songautoeq = prev_songautoeq;
```

... and I don't even know what function it goes in, without looking at the exact version of the file you modified and matching up line numbers.

#3 - December 10, 2015 06:01 - John Lindgren

What do you think about making the user interface for this feature much simpler?

1. Turning the AUTO button ON would immediately save the current equalizer settings to an auto-preset for the current song.
2. Turning the AUTO button OFF again would delete that auto-preset and return to the previous equalizer settings.
3. When a new song plays, the auto-preset for that song will always be loaded, if it exists, and the AUTO button would light up. This means there would be no way to disable the feature entirely, only on a per-song basis by deleting individual presets. (Edit: You could, of course, still disable the equalizer entirely.)

The first time a user clicks on the AUTO button, we'd want to pop up a message dialog explaining what it does, with a "Don't show this again" checkbox. The biggest problem with the old implementation of this feature (aside from it not working at all for several releases!) was that no one

knew how it worked.

Also, a couple of implementation ideas:

1. Let's keep the config format compatible with Audacious 3.6.x; that is, store all the auto-presets in ~/.config/audacious/eq.auto_preset.
2. I don't think we need a separate "default" preset, as long as we don't overwrite the settings stored in the main config file. It's probably better for the UI code to stop accessing the "equalizer_xxx" config values directly, anyway.

#4 - December 10, 2015 07:55 - Jim Turner

1: Good idea!

2: ditto, except if the prev. song also had a preset file, we need to be sure to revert to the previous "non-auto-preset" values as opposed to the prev. song's values! I was playing w/XMMS the other day (to see how this shtuff worked) and it did not handle this properly either (kept last song's auto preset after switching to a non-auto song - losing your non-auto settings).
Also, by "delete" I hope you mean just reset the equalizer, not delete the song's preset file?

3: I disagree. In XMMS, AUTO just permits song-specific presets to be used (and the equalizer to be changed). I like this (XMMS way) idea better. In my patch, I light up the [Preset] button on the equalizer as the indicator for this purpose when a song-specific preset is loaded. The skins contain 2 states for this button. XMMS doesn't seem to ever light it up, but only uses the 2 states as "pressed-and-held" and "not-pressed", but I simply toggle to the opposite state whilst pressed & held, then back to the proper indicator state when released.

1(paragraph b): Is eq.auto_preset to be a directory or file? Directory: cool, single file: could become large & will make it harder to remove song presets w/o having to create a separate GUI for that purpose? At least, maybe "eq_auto.preset" or "eq.auto_presets", since more than one.

2(b): Sounds great as long as we can keep the last-known non-song-specific settings somewhere.
One other thought a/b files - Some streams (and stdin? - haven't tested) don't seem to always have "names". I think my patch makes sure it falls thru to something like "unnamed", which sortta sux, but not sure best way to handle that.

My 2¢

Jim

#5 - December 10, 2015 08:23 - Jim Turner

- File libaudcore.output.cc.patch added

- File libaudgui.eq-preset.cc.patch added

- File libaudgui.preset-browser.cc.patch added

- File skins.equalizer.cc.patch added

Sorry I missed addressing your 1st 2 comments in my prev. reply.
Here are the patches redone w/diff -u:

Regards,

Jim

#6 - December 10, 2015 15:22 - John Lindgren

Jim Turner wrote:

> ... Also, by "delete" I hope you mean just reset the equalizer, not delete the song's preset file?

I actually meant, delete the preset for that song. But maybe this is not a good idea.

> ... In my patch, I light up the [Preset] button on the equalizer as the indicator for this purpose when a song-specific preset is loaded. The skins contain 2 states for this button. ...

It would be a novel use of the Preset button, but I don't think I agree with it. The Preset button isn't meant to be a toggle; toggle buttons have four states in the skins, not two. How about making the AUTO button access a small drop-down menu with three options: Enable auto-presets (checkbox), Save auto-preset, Delete auto-preset? Then make it light up only when auto-presets are enabled AND a preset has been loaded for the current song.

Making changes to the equalizer settings while an auto-preset is loaded should save those changes automatically, I think. Otherwise those changes would just get lost without warning when the next song plays.

> ... Is eq.auto_preset to be a directory or file? ...

Single file; that's the way it was in 3.6. Yes, it could become large, but the alternative--creating hundreds of tiny (around 150 bytes) files--is rather wasteful of disk space, so that's not ideal either.

#7 - December 10, 2015 18:35 - Jim Turner

That (small dd menu w/3 options on [Auto] button) sounds like a a good plan. Single file should in practice work fine too (I don't use very many auto-presets, though some abUsers might). Idk how it worked in 3.6 since I never used it until recently when I noticed that a cpl. of streams I started using sounded awful w/my normal eq. settings and I figured out that they must be already "equalized" and I switch between 'em regularly. My patch used separate files b/c that was the quickest and easiest way for me to make it work by piggybacking off the existing "Export Preset" and "Load Preset options.

The reason I like my idea of toggling the [Preset] button is that then I can see both whether I have "Auto Preset switching" ([Auto] button) on and, if so, whether or not the current song has (altered the EQ with) it's own auto-preset.

So, let me see if I got this straight: User brings up EQ., IF [Auto] is on and abUser changes a slider, it immediately saves it to/as that song's auto-preset and EQ settings revert when song is finished/changed; ELSE it immediately saves as the current (non-auto) EQ settings as normal. If that's the case, then the menu only needs 2 settings: 1:[Activate Auto Preset] or [Deactivate Auto Preset] (depending on current state); and 2:(iff preset exists for this song) [Delete Auto Preset]. The top choice (to toggle) should be under the mouse when menu pops up allowing a double-click of [Auto] to just toggle.

If user brings up EQ, turns Auto on in song A, then changes EQ settings whilst song A is playing (saves changes as song A's auto-settings), then plays song B (which has already has it's own auto-settings) - EQ settings then change to B's settings, then plays song C (which does NOT have it's own settings), EQ settings should revert back to what they were BEFORE he changed 'em whilst song A was playing (Is your head hurtin' as much as mine yet - LOL?!) That's why my patch creates 2 aud variables: "equalizer_autoload" saves the state of the [Auto] button (on|off) and "equalizer_song_auto" which saves the state of whether the EQ was last set to a song's auto-settings or not, so that when changing songs or Audacious is restarted, we know whether or not we need to revert to last known "non-auto" settings or not (output.cc) and why it creates a "_nonauto.preset" preset.

Regards,

Jim

#8 - December 15, 2015 03:44 - Jim Turner

After finding and fixing a bug in my patch causing the improper switching / restoring presets when first creating a new auto-preset during play, I decided to document it a bit better. Here's what I've found:

I see 4 possible scenarios when opening an entry (either by song-advance or startup): (N=entry has no auto-presets, P=entry has presets, shown as [Prev. Entry]->[Current Entry])
based on prev. entry played: (NOTE: we handle P->N regardless of [Auto] state!)
1) N->N: Do nothing with equalizer.
2) N->P: (IF [Autoload]) Save current EQ settings as "default", then load entry's auto-preset.
3) P->P: (IF [Autoload]) Load entry's auto-preset file.
4) P->N: Load the prev. saved "default" preset file.

If [Autoload] is OFF, then the only 2 possible states are N->N and P->N since no
song-specific preset is checked for, much less loaded.

Hope this is helpful.

Jim

#9 - December 25, 2018 06:29 - Waky II Sr

Jim Turner wrote:

After finding and fixing a bug in my patch causing the improper switching / restoring presets when first creating a new auto-preset during play, I decided to document it a bit better. Here's what I've found:

I see 4 possible scenarios when opening an entry (either by song-advance or startup): (N=entry has no auto-presets, P=entry has presets, shown as [Prev. Entry]->[Current Entry])
based on prev. entry played: (NOTE: we handle P->N regardless of [Auto] state!)
1) N->N: Do nothing with equalizer.
2) N->P: (IF [Autoload]) Save current EQ settings as "default", then load entry's auto-preset.
3) P->P: (IF [Autoload]) Load entry's auto-preset file.
4) P->N: Load the prev. saved "default" preset file.

If [Autoload] is OFF, then the only 2 possible states are N->N and P->N since no
song-specific preset is checked for, much less loaded.

Hope this is helpful.

Jim

Hi,

I downloaded Audacious 3.8.2 for Windows, and noticed the AUTO button is not clickable.
I tried the same button on Ubuntu many months ago and it worked to some degree.
I was a Winamp heavy user decades ago and value greatly this Audacious port.

Appreciate some help.

#10 - December 26, 2018 15:15 - Jim Turner

@Waky: That's b/c this is a proposed patch, which has not been implemented in Audacious as of this time. It IS implemented in my forked version. Email me directly (see my profile) or search Github for Audacious fork as this is a separate, competing project that I don't feel right about advertising for or discussing here.

Regards,

Jim

#11 - December 02, 2019 20:53 - John Lindgren

- Category changed from libaudgui to core

This needs to be re-implemented in a way that doesn't depend on a particular UI or toolkit (beyond the toggle to enable/disable the feature).

#12 - December 03, 2019 05:31 - Jim Turner

- File equalizer-qt.cc added

- File equalizer.cc added

- File eq-preset-qt.cc added

At the time (4 years ago), I had not implemented this patch on the QT side as I was waiting for y'all to implement the eq-preset screen in QT (Audacious commit# 7dc6148). I've since done it so it's no longer UI-dependent. I'm going ahead and including my current patched QT files, though our code is significantly different enough that it may not be that useful to you at this point, but maybe can save some time implementing this by using them as a reference.

Jim

#13 - December 03, 2019 14:40 - John Lindgren

Thanks Jim. I don't have time to work on porting your changes myself, but I would review a pull request if someone has time to make one.

Files

libaudcore.output.cc.patch	3.65 KB	December 09, 2015	Jim Turner
libaudgui.eq-preset.cc.patch	253 Bytes	December 09, 2015	Jim Turner
libaudgui.preset-browser.cc.patch	2.06 KB	December 09, 2015	Jim Turner
skins.equalizer.cc.patch	1.69 KB	December 09, 2015	Jim Turner
libaudcore.output.cc.patch	4.5 KB	December 10, 2015	Jim Turner
libaudgui.eq-preset.cc.patch	651 Bytes	December 10, 2015	Jim Turner
libaudgui.preset-browser.cc.patch	3.04 KB	December 10, 2015	Jim Turner
skins.equalizer.cc.patch	3.52 KB	December 10, 2015	Jim Turner
eq-preset-qt.cc	26 KB	December 03, 2019	Jim Turner
equalizer.cc	10.7 KB	December 03, 2019	Jim Turner
equalizer-qt.cc	7.7 KB	December 03, 2019	Jim Turner