

QGIS Application - Bug report #7129

QgsCollapsibleGroupBox auto saves its collapsed state

2013-02-09 08:34 AM - Olivier Dalang

Status:	Closed	
Priority:	Normal	
Assignee:		
Category:	GUI	
Affected QGIS version:	master	Regression?: No
Operating System:	Windows	Easy fix?: No
Pull Request or Patch applied:	Yes	Resolution:
Crashes QGIS or corrupts data:	No	Copied to github as #: 16198
Description		
Hi ! QgsCollapsibleGroupBox is by default auto-saving it's folded state. The feature is nice, but this has the following problems : - It leads to a strange behaviour when there are several instances of the same QgsCollapsibleGroupBox : while folding/unfolding group boxes, each instance has it own folded state, but upon reload, each instance will have the last expanded state (see screenshot). - It is actually quite unexpected for a QWidget to auto-store settings. It seems to me that it's not implemented in the right place. All other widgets (checkboxes...) need ad-hoc implementation to save their state. - It overrides the ability to set the initial folded state from within QtDesigner (which you can do by adding the property as dynamic properties) Actually, I think the feature was developped to be used in global options. But since those CollapsibleBoxes are very practical and good looking, it would be better to consider them as normal QWidget, to be used everywhere. In the provided patch, the saveCollapsedState is set to false by default. Additionally, collapsed, saveCollapsedState and saveCheckedStateproperties are declared so they can be edited in QtDesigner. All existing QgsCollapsibleGroupBox in the UI files are set to collapsed=false, saveCollapsedState=true and saveCheckedStateproperties=false. So this should lead to no change of behaviour in existing code. The main drawback is that DEVs will have to remember they can add those properties as "dynamic properties" in QtCreator. What do you think ? Provided : - patch - plugin to test functionality		

History

#1 - 2013-02-09 10:17 AM - Larry Shaffer

- Category set to GUI

- Target version set to Future Release - Nice to have

QgsCollapsibleGroupBox is by default auto-saving it's folded state.

The feature is nice, but this has the following problems :

- It leads to a strange behaviour when there are several instances of the same QgsCollapsibleGroupBox : while folding/unfolding group boxes, each instance has it own folded state, but upon reload, each instance will have the last expanded state (see screenshot).

This only happens when the group boxes have the same object name and the same parent window name. It is bad form to name two group boxes with the same object name, but I don't think you are implying that's the case. Please provide a functional use case example of this issue (you only show a screen shot of the result) so the class can be fixed, if needed.

An additional class feature that might fix this (assuming I understand the issue) is an accessible property to set the QSettings key string, as an option to having the key built off of the parent window's object name (default). This way, the setting string could be generated base upon the current composer map item, instead of the window name, if say, the group box doesn't change, but its contents do. This assumes the group box is destroyed between map item selections.

- It is actually quite unexpected for a QWidget to auto-store settings. It seems to me that it's not implemented in the right place. All other widgets (checkboxes...) need ad-hoc implementation to save their state.

There is a big difference between auto-saving the collapsed state of this custom widget and the state of an actual form control, e.g. checkbox. Form controls usually directly relate to setting options, and as such should *not* generally be auto-saved (though many are with named slots); however, the collapsed state of this group box class (not a form control widget) is **purely cosmetic** and completely independent of any functional application setting.

There is nothing 'wrong' with auto-saving the collapsed state of this custom widget on its destruction. There is no 'expected' behavior for a custom widget. They work as needed. It was designed this way so developers do not have to write get/set code for every use just to save a cosmetic attribute of the widget. Its auto-saving of the collapsed state *is a feature* IMO.

- It overrides the ability to set the initial folded state from within QtDesigner (which you can do by adding the property as dynamic properties)

Actually, I think the feature was developped to be used in global options. But since those CollapsibleBoxes are very practical and good looking, it would be better to consider them as normal QWidgetts, to be used everywhere.

In the provided patch, the saveCollapsedState is set to false by default.

Additionnaly, collapsed, saveCollapsedState and saveCheckedStateproperties are declared so they can be edited in QtDesigner.

All existing QgsCollapsibleGroupBox in the UI files are set to collapsed=false, saveCollapsedState=true and saveCheckedStateproperties=false.

So this should lead to no change of behaviour in existing code.

The main drawback is that DEVs will have to remember they can add those properties as "dynamic properties" in QtCreator.

The Q_PROPERTY solution is a decent workaround for setting properties in Designer. **So why not leave the initial class property values as is (e.g. mSaveCollapsedState = true;) and have the dynamic property for *not* auto-saving be the one added?** Really no different than setting it in code, and nothing needs changed in the existing .ui files.

Another issue with this approach: if the class ever get refactored, all of the .ui files' dynamic properties would need to be as well. In my test, QtCreator did not locate any properties in .ui files during a refactor search/replace command, or in a general search across the project (default CMake project). Errors would be thrown during compilation, though.

If you want Designer integration, [creating a custom widget for Designer](#) is the best approach.

IMO, the class and functionality should stay as is, but any work to make it better and any work on creating a Designer plugin for it would certainly be welcome.

#2 - 2013-02-10 12:39 PM - Etienne Tourigny

I mostly agree with Larry. Existing functionality should not be changed, we have worked quite a bit on this.

Another option would be to have 2 classes, one that is the widget itself (ay QgsCollapsobleGroupBoxBasic) and a sub-class which has default auto-saving and restore.

for Designer integration, it would be more handy for some devs to have custom widgets, but much easier to implement and setup using Olivier's solution.

Olivier : can you provide a patch without the changes to default state (including mods to ui files) and a test case (with code) for the bug you mention? I'm not sure I understand why this happens.

#3 - 2013-02-10 01:07 PM - Larry Shaffer

Etienne Tourigny wrote:

...
Another option would be to have 2 classes, one that is the widget itself (ay QgsCollapsobleGroupBoxBasic) and a sub-class which has default auto-saving and restore.

I'm in favor of this option. Thanks for suggesting it. Seems to address the concerns of everyone involved and doesn't require a retrofit to existing code.

for Designer integration, it would be more handy for some devs to have custom widgets, but much easier to implement and setup using Olivier's solution.

I like the simplicity of Oliver's Q_PROPERTY setup over the work of making a custom Designer widget as well. A potential issue is that it requires the dev to correctly type in the property name and set up a compatible value. However, if there are two classes like you suggested, typing in those custom properties becomes totally optional, and they can always be set in code.

#4 - 2013-02-10 02:40 PM - Olivier Dalang

- File qgsCollapsedBug.qgs.zip added

Thanks for you answers.

I like the two classes solution too ! I'd suggest calling the complete one QgsPersistentCollapsibleGroupBox (a bit long, but at least it's very clear of what the class does)

An additional class feature that might fix this (assuming I understand the issue) is an accessible property to set the QSettings key string, as an option to having the key built off of the parent window's object name (default).

Yes that is a good idea too !

Olivier : can you provide a patch without the changes to default state (including mods to ui files) and a test case (with code) for the bug you mention?

I'm not sure I understand why this happens.

About the test case, I'm quite new to QGIS developpement, so I still have to learn about testing and won't be able to provide a test case soon. But here's a simple file explaining the bug.

I think I should be able to provide a patch with this in the next few days if we agree on this solution.

#5 - 2013-02-10 04:00 PM - Etienne Tourigny

I think that the composer bug should be treated separately from the main issue, which is having a means to set the original expanded state in designer.

IMHO the base class (without save/restore) should have a new name, while derived class should keep same name, to lessen impact on existing code.

I've managed to replicate the problem, which is due to the fact that there are the same controls for different objects. I believe Larry has run into that when working with the advanced labelling. Solution would be to set the settings key to a unique value, possibly `<frame id>/widget/parent`.

#6 - 2013-02-10 05:29 PM - Larry Shaffer

Etienne Tourigny wrote:

I've managed to replicate the problem, which is due to the fact that there are the same controls for different objects. I believe Larry has run into that when working with the advanced labelling. Solution would be to set the settings key to a unique value, possibly `<frame id>/widget/parent`.

The problem I ran into was kinda the opposite: the labeling group boxes needed to persist their collapsed state when shown in multiple places, i.e. under different parents. To that end we added the ability to set the QSettings group, [setSettingGroup\(\)](#), and I [used it here](#). While this may work to solve Olivier's issue (which I could reproduce), developing a base class for the collapsible group box would allow for a more flexible approach.

IMO the collapsed state of the group boxes for composer items should not be saved in the app's QSettings. At that point the collapsed state becomes a property of the composer item, like color, etc., and as such should be probably be stored in the project file. Otherwise, the the app settings will become filled (polluted?) with many settings relative to project-based items.

Alternatively, `setSettingGroup()` could be used to craft a custom string and the new `setSettings(QSettings* settings)` could be used to define a separate QSettings file in the user's .qgis directory to store those states. This would not be portable like a project file, though.

#7 - 2013-02-10 05:54 PM - Olivier Dalang

- File 0001-Declare-properties-of-QgsCollapsibleGroupBox-so-they.patch added

Solution would be to set the settings key to a unique value, possibly `<frame id>/widget/parent`.

Actually, you'd also need to store the settings in the project file rather than in the global settings (since two different projects can have objects with same IDs). But for global options, I think the goal was exactly to save the settings as being global to all projects. So it's not that simple.

In fact, this illustrates well my frustration about this class : it has some too specific features that makes it unsuitable to use as a normal widget.

For an inattentive developer, it could even lead to severe bugs : the use of persistence of non-cosmetic nature would induce modifications across all projects ! (and this seems even encouraged by the `saveCheckedState` property !)

IMO, if we want to have QgsCollapsibleGroupBox used more generally across QGis and plugins, it would be better to provide a simple and robust class. And if this means change a bit of the existing code, it's better to do it now than later. (right now, the class is used only in few places of QGis, and I doubt that many plugin developers already use it).

Besides, it seems to me that every developer (even for small plugins) has to deal with persistence. He has to implement it for every other aspects of his work. Providing auto-persistence for one specific widget will only make the code more obscure by doing two very similar tasks in two very different ways. Take for example QgsProjectProperties and compare how it's QTabWidget's state and its QgsCollapsibleGroupBox's states are stored.

But anyways, the problem is not that important for now... One simply has to remember to disable the feature in case the implementation does not suit his needs.

I join another patch which does only add the Q_PROPERTY(), comments and the properties in the .ui files.

#8 - 2013-02-11 06:02 AM - Etienne Tourigny

It might be easier to just disable saving the state for the groupboxes in the composer, to avoid any bugs and make life easier for everybody. I fear a "complete" fix for this would be quite complex, not really worth the effort IMHO.

I'm also now not against making the default not auto-save, based on Olivier's concerns, but I'll leave that up to Larry.

I've checked your patch, and opened one of the widgets in designer, which has the dynamic properties. However, this is only slightly useful as they are not added to new widgets when promoted to QgsCollapsibleGroupBox.
It's better than before though.

cheers

#9 - 2013-02-11 09:38 AM - Etienne Tourigny

- File patch2.txt added

Here is a patch with 2 classes, including Q_PROPERTY stuff. I haven't updated nor tested sip though.

Sorry I'll be away for a few days, won't be able to look at this until next week. So Larry - go ahead as you deem best, thanks.

#10 - 2013-02-20 10:32 AM - Larry Shaffer

- Status changed from Open to Closed

Files			
CollapsibleGroupBoxTester.zip	28.4 KB	2013-02-09	Olivier Dalang
qtdesignerproperties.png	204 KB	2013-02-09	Olivier Dalang
foldBug.png	52.4 KB	2013-02-09	Olivier Dalang
Set-QgsCollapsibleGroupBox-autosave-to-false.patch	21.9 KB	2013-02-09	Olivier Dalang
qgsCollapsedBug.qgs.zip	2.07 KB	2013-02-10	Olivier Dalang
0001-Declare-properties-of-QgsCollapsibleGroupBox-so-they.patch	24.5 KB	2013-02-10	Olivier Dalang
patch2.txt	38 KB	2013-02-11	Etienne Tourigny