QGIS Application - Feature request #3447
Grouping "add layer" toolbar buttons

2011-01-28 04:38 AM - Alexander Bruy

Status:	Closed		
Priority:	Low		
Assignee:	Tim Sutton		
Category:	GUI		
Pull Request or Patch supplied:		**Resolution:**	wontfix
Easy fix?:	No	**Copied to github as #:**	13507

Description

Proposed patch replaced five "add layer" buttons with one dropdown button. See also discussion in qgis-developer mailing list [http://osgeo-org.1803224.n2.nabble.com/Grouping-toolbar-buttons-tp5967127p5967127.html]

Related issues:

Duplicated by QGIS Application - Feature request # 7487: Load database layer ...	**Closed**	**2013-04-01**

History

#1 - 2011-01-28 06:44 AM - Tim Sutton

- Status changed from Open to In Progress

#2 - 2011-01-28 07:06 AM - Tim Sutton

Applied with commit:58bd7acd (SVN r15095). However I think we should have all add layers except Add vector and Add raster in the group. There are still a number of icons not in the group that could go in - sqlanywhere, wfs etc. I was thinking we could have a [[QgisApp]]::addLayerLoaderAction(QAction *) that providers could call to have their icon dropped into the group.

Regards

Tim

#3 - 2011-01-30 11:47 PM - Jürgen Fischer

Replying to [comment:3 timlinux]:

> *I was thinking we could have a [[QgisApp]]::addLayerLoaderAction(QAction *) that providers could call to have their icon dropped into the group.*

And also drop the "providers don't have a UI" rule?

#4 - 2011-01-31 12:11 AM - Martin Dobias

Replying to [comment:4 jef]:

> *Replying to [comment:3 timlinux]:*
>
> > *I was thinking we could have a [[QgisApp]]::addLayerLoaderAction(QAction *) that providers could call to have their icon dropped into the group.*
>
> *And also drop the "providers don't have a UI" rule?*

I remember we started this rule long time ago when it was normal that provider constructor or nextFeature call showed a message box :-)

IMO we should move the "add layer" dialogs (which are specific to each provider) to the providers and add a new API method to provider modules that would create provider's add layer dialog.

A next step could be an unified add layer dialog: main window would contain just one "add layer" action that would trigger a dialog having all the provider dialogs inside (dialogs would be switched with a stacked widget).

(Pure imagination begins now) At some point, we could reach a situation where we have an integrated dialog with a file browser, database/web connections, "favorite" layers, "last used" layers and similar stuff, with good usability and minimized amount of necessary clicks per one layer.

#5 - 2011-01-31 12:19 AM - Tim Sutton

Replying to [comment:5 wonder]:

> *Replying to [comment:4 jef]:*
>
> > *Replying to [comment:3 timlinux]:*
> >
> > > *I was thinking we could have a [[QgisApp]]::addLayerLoaderAction(QAction *) that providers could call to have their icon dropped into the group.*
> >
> > *And also drop the "providers don't have a UI" rule?*

Sorry I had actually meant to write 'that a provider's gui plugin could call'.

> *I remember we started this rule long time ago when it was normal that provider constructor or nextFeature call showed a message box :-)*
>
> *IMO we should move the "add layer" dialogs (which are specific to each provider) to the providers and add a new API method to provider modules that would create provider's add layer dialog.*
>
> *A next step could be an unified add layer dialog: main window would contain just one "add layer" action that would trigger a dialog having all the provider dialogs inside (dialogs would be switched with a stacked widget).*
>
> *(Pure imagination begins now) At some point, we could reach a situation where we have an integrated dialog with a file browser, database/web connections, "favorite" layers, "last used" layers and similar stuff, with good usability and minimized amount of necessary clicks per one layer.*

I think there is still a good reason to avoid GUI interaction at the provider level. For example if we are creating an embedded (e.g. android) version of QGIS it may be that we want a completely different UI for adding layers than the desktop one. Then it may be better / easier to simply replace / tweak the plugin implementations themselves and leave the providers untouched.

@wonder Regards your other thoughts on a grand central station for provders, +1 from me - its quite cumbersome our current system (both to use and to explain to novices) and it would be nice to get them all together into a drawer etc paradigm.

Regards

Tim

#6 - 2011-01-31 02:19 AM - Jürgen Fischer

Replying to [comment:5 wonder]:

> *(Pure imagination begins now) At some point, we could reach a situation where we have an integrated dialog with a file browser, database/web connections, "favorite" layers, "last used" layers and similar stuff, with good usability and minimized amount of necessary clicks per one layer.*

Yes and being at it, we shouldn't stop there and also add a way to show provider specific layer properties (probably very useful for raster providers especially WMS).

#7 - 2011-01-31 02:59 PM - Martin Dobias

Replying to [comment:6 timlinux]:

> *I think there is still a good reason to avoid GUI interaction at the provider level. For example if we are creating an embedded (e.g. android) version of QGIS it may be that we want a completely different UI for adding layers than the desktop one. Then it may be better / easier to simply replace / tweak the plugin implementations themselves and leave the providers untouched.*

What I had in my mind is that provider code would be untouched, just the code from their accompanying plugins would be added to the provider - instead of having it in a separate module. Now it is a mess: some providers have their dialogs directly in the src/app (ogr, postgres), while others have a plugin that only shows gui for loading layers from that provider. In order to use that provider, the user has to explicitly enable the plugin. Is that necessary?

Regarding the mobile interface, I don't see a reason why an alternative gui couldn't be added to a provider.

> *@wonder Regards your other thoughts on a grand central station for provders, +1 from me - its quite cumbersome our current system (both to use and to explain to novices) and it would be nice to get them all together into a drawer etc paradigm.*

Yeah. Looking at the changes by the patch attached here, I have the impression that grouping the action in the way it is done now is merely a step back rather than a progress. It saves some space but sacrifices the usability as the plugins still create ungrouped actions. (I would suggest a revert until we have a better solution)

Martin

#8 - 2011-02-01 12:22 AM - Marco Hugentobler

A great benefit of bringing the dialogs into the provider libs is that dialogs could link to provider classes and using specific provider methods. E.g. the WMS dialog could pass the WMS capabilities document directly to the provider. Currently, WMS dialog fetches the capabilities and the WMS provider needs to fetch it again because the dialog has no possibility to communicate with the WMS provider directly (only through the provider interface that is common for all providers).

Regarding grouping the addLayer buttons: I like it because it because of the additional space I get. Couldn't we just export the toolbutton such that plugins could add their buttons there too? Additionally, we talked about bringing wfs, gps plugins into app (or into providers depending on the outcome of above discussion).

Regards,
Marco

#9 - 2011-02-01 01:46 AM - Martin Dobias

Replying to [comment:9 mhugent]:

> *A great benefit of bringing the dialogs into the provider libs is that dialogs could link to provider classes and using specific provider methods. E.g. the WMS dialog could pass the WMS capabilities document directly to the provider. Currently, WMS dialog fetches the capabilities and the WMS provider needs to fetch it again because the dialog has no possibility to communicate with the WMS provider directly (only through the provider interface that is common for all providers).*

Right. I see one more benefit: the dependencies (e.g. postgresql library for postgis) will be only inside the provider module. Therefore if the user has a mismatch in the libraries, there will be a lower chance that QGIS will fail to start (only the affected provider won't be loaded).

While we're at it, I would also suggest to move "new layer" functionality to the providers. Nowadays we can create new layer with ogr and spatialite backends, though the implementation is in src/app. If we had that functionality within providers, it would be possible to create new layers seamlessly also in plugins and implement "new layer" feature for further providers.

> *Regarding grouping the addLayer buttons: I like it because it because of the additional space I get. Couldn't we just export the toolbutton such that plugins could add their buttons there too? Additionally, we talked about bringing wfs, gps plugins into app (or into providers depending on the outcome of above discussion).*

Yes, putting the other "add layer" buttons into the group is necessary. Exporting the toolbutton group can be a temporary solution.

Martin

#10 - 2011-02-01 10:29 AM - Jürgen Fischer

Replying to [comment:10 wonder]:

> *Right. I see one more benefit: the dependencies (e.g. postgresql library for postgis) will be only inside the provider module. Therefore if the user has a mismatch in the libraries, there will be a lower chance that QGIS will fail to start (only the affected provider won't be loaded).*
>
> *While we're at it, I would also suggest to move "new layer" functionality to the providers. Nowadays we can create new layer with ogr and spatialite backends, though the implementation is in src/app. If we had that functionality within providers, it would be possible to create new layers seamlessly also in plugins and implement "new layer" feature for further providers.*

+1

Wait a minute - dejavu:
http://lists.osgeo.org/pipermail/qgis-developer/2008-April/003647.html ff

Jürgen

#11 - 2011-02-04 02:33 AM - Tim Sutton

- Resolution set to wontfix

- Status changed from In Progress to Closed

As per the discussion above, I have rolled back this commit as I agree it does not improve usability.

Side note: How nice is git! I just did this:

```
git revert d1dfeae396b244bcb76076ef9ab80bfc42d05587
```

And it removed the patch for me. Cool bananas....

I'm going to mark this one as 'wont fix' and close it...

Tim

Files

group_toolbuttons.diff	3 KB	2011-01-28	Alexander Bruy