QGIS Application - Bug report #7900
Mass vertex removal with node tool leaks memory

2013-05-25 05:45 AM - Sandro Santilli

Status:	Closed		
Priority:	Normal		
Assignee:			
Category:	Digitising		
Affected QGIS version:	master	**Regression?:**	No
Operating System:		**Easy fix?:**	No
Pull Request or Patch supplied:	No	**Resolution:**	fixed
Crashes QGIS or corrupts data:	No	**Copied to github as #:**	16772

Description

I've been using qgis to simplify test cases by dropping vertices from input using the node tool.
The procedure is: click on a feature, shift-click-and-drag to select all vertices within an extent, hit the DEL key.

Doing so, I've noticed that the memory used by qgis grows incrementally to the number of vertices so removed.
The memory grows again as you delete the next chunk of vertices.

When exiting edit mode the memory is sometimes flushed, sometimes not.
Couldn't figure out what makes the difference. In any case it uses a lot of memory.

I think with 1.8.0 the memory was *never* released while with trunk it sometimes is,
but anyway this needs some inspection as the only way to effectively do that mass-vertex-removal with qgis is to periodically quit and restart the application to avoid a system lockup.

Related issues:

Related to QGIS Application - Bug report # 7929: Exiting edit mode withouth s...	**Closed**	**2013-05-28**

History

#1 - 2013-05-28 09:22 AM - Sandro Santilli

Gave valgrind a shot. This time I exited edit mode without saving.

11261 789,971 (32 direct, 789,939 indirect) bytes in 1 blocks are definitely lost in loss record 10,522 of 10,523
11261 at 0x4C2B1C7: operator new(unsigned long) (in /usr/lib/valgrind/vgpreload_memcheck-amd64-linux.so)
11261 by 0x6E1424: QgsSelectedFeature::updateGeometry(QgsGeometry*) (qgsselectedfeature.cpp:83)
11261 by 0x6E18D1: QgsSelectedFeature::geometryChanged(long long, QgsGeometry&) (qgsselectedfeature.cpp:157)
11261 by 0x897D0B: QgsSelectedFeature::qt_static_metacall(QObject*, QMetaObject::Call, int, void**) (moc_qgsselectedfeature.cxx:64)
11261 by 0x972D280: QMetaObject::activate(QObject*, QMetaObject const*, int, void**) (in /usr/lib/x86_64-linux-gnu/libQtCore.so.4.8.1)
11261 by 0x8544610: QgsVectorLayer::geometryChanged(long long, QgsGeometry&) (moc_qgsvectorlayer.cxx:337)
11261 by 0x8543E7A: QgsVectorLayer::qt_static_metacall(QObject*, QMetaObject::Call, int, void**) (moc_qgsvectorlayer.cxx:185)
11261 by 0x972D280: QMetaObject::activate(QObject*, QMetaObject const*, int, void**) (in /usr/lib/x86_64-linux-gnu/libQtCore.so.4.8.1)
11261 by 0x8544D52: QgsVectorLayerEditBuffer::geometryChanged(long long, QgsGeometry&) (moc_qgsvectorlayereditbuffer.cxx:164)
11261 by 0x81C52AA: QgsVectorLayerUndoCommandChangeGeometry::undo() (qgsvectorlayerundocommand.cpp:169)
11261 by 0xA2B05B0: QUndoCommand::undo() (in /usr/lib/x86_64-linux-gnu/libQtGui.so.4.8.1)
11261 by 0xA2B0EBE: QUndoStack::setIndex(int) (in /usr/lib/x86_64-linux-gnu/libQtGui.so.4.8.1)

#2 - 2013-05-28 09:37 AM - Sandro Santilli

The leak seems unrelated. Instead I found out that if you change selected tool before exiting edit mode, then the leak goes away
(but only after exiting edit mode)

#3 - 2013-05-28 09:38 AM - Sandro Santilli

Nope, I take it back, there's no direct corrispondence between deselecting the nodetool and the leak. It still sometimes happen and sometime not.

#4 - 2013-05-29 07:10 AM - Sandro Santilli

I believe the memory is used by the undo stack as QgsVectorLayerEditUtils only exposes a function to delete a single vertex so it makes a geometry clone for each removed vertex.

#5 - 2013-05-29 07:49 AM - Jürgen Fischer

Sandro Santilli wrote:

> *I believe the memory is used by the undo stack as QgsVectorLayerEditUtils only exposes a function to delete a single vertex so it makes a geometry clone for each removed vertex.*

On rollback it just reverses each operation on the undo stack one by one - and each vertex removal is one geometry change (QgsVectorLayerUndoCommandChangeGeometry). There doesn't seem to be any rollback optimization for consecutive geometry changes.

#6 - 2013-05-29 07:52 AM - Sandro Santilli

Yep, that's what I figured. Let's leave optimization (making multiple-vertices removal a single operation) for ticket #7929.
This ticket remains for the memory leak. Haven't checked if the memory in use is still in the undo stack or where else...

#7 - 2013-05-30 12:15 AM - Sandro Santilli

- Resolution set to fixed
- Status changed from Open to Closed

commit:59788cb8fcbe86a71397c3046981d00f06c40b00 made this unperceptible (can't see memory growing). So, good job !