QGIS Application - Bug report #19914
using the console and cloning a symbol from a graduated style layer crashes qgis 3

2018-09-21 03:43 PM - cris c

Status:	Closed		
Priority:	Normal		
Assignee:	Alessandro Pasotti		
Category:	PyQGIS Console		
Affected QGIS version:	3.3(master)	**Regression?:**	No
Operating System:		**Easy fix?:**	No
Pull Request or Patch supplied:	No	**Resolution:**	invalid
Crashes QGIS or corrupts data:	Yes	**Copied to github as #:**	27738

Description

User Feedback
Working on a project with quite a few layers loaded. I selected a graduated styled layer and I am attempting to change the sizes of the symbol via the python console. As soon as I issue the clone() command it crashes.

```
layer = qgis.utils.iface.mapCanvas().currentLayer()
renderer = layer.renderer()
current_symbol = renderer.ranges()[1].symbol()
new_symbol = current_symbol.clone()
```

I've tried it in the 3.0 version and also deleted all layers except one graduated layer with 5 ranges.

also the same effect with this sequence:

```
layer = qgis.utils.iface.mapCanvas().currentLayer()
renderer = layer.renderer()
current_symbol = renderer.ranges()[1].symbol()
current_symbol.size()
```

Report Details

I tried it with the data defined override for symbol size on and off with the same effect. Also with different index number in the range.

Crash ID: afdb5feb28fe4c13c1eaedd723b645ed4adbbf15

Stack Trace

```
meth_QgsMarkerSymbol_clone sip_corepart1.cpp:339792
PyCFunction_FastCallDict :
PyObject_GenericGetAttr :
PyEval_EvalFrameDefault :
PyErr_Occurred :
PyEval_EvalCode :
PyDict_SetItemId :
PyDict_SetItemId :
PyCFunction_FastCallDict :
PyObject_GenericGetAttr :
PyEval_EvalFrameDefault :
PyObject_GenericGetAttr :
PyEval_EvalFrameDefault :
PyErr_Occurred :
PyObject_GenericGetAttr :
PyEval_EvalFrameDefault :
PyErr_Occurred :
PyObject_GenericGetAttr :
PyEval_EvalFrameDefault :
PyObject_GenericGetAttr :
PyEval_EvalFrameDefault :
```

PyObject_GenericGetAttr :
PyEval_EvalFrameDefault :
PyFunction_FastCallDict :
PyObject_CallFunctionObjArgs :
PyObject_Call :
PyInit_sip :
PyInit_Qsci :
QWidget::event :
QFrame::event :
QAbstractScrollArea::event :
PyInit_Qsci :
QApplicationPrivate::notify_helper :
QApplication::notify :
QgsApplication::notify qgsapplication.cpp:380
QCoreApplication::notifyInternal2 :
QSizePolicy::QSizePolicy :
QSizePolicy::QSizePolicy :
QApplicationPrivate::notify_helper :
QApplication::notify :
QgsApplication::notify qgsapplication.cpp:380
QCoreApplication::notifyInternal2 :
QGuiApplicationPrivate::processKeyEvent :
QWindowSystemInterface::sendWindowSystemEvents :
QEventDispatcherWin32::processEvents :
CallWindowProcW :
DispatchMessageW :
QEventDispatcherWin32::processEvents :
qt_plugin_query_metadata :
QEventLoop::exec :
QCoreApplication::exec :
main main.cpp:1470
WinMain mainwin.cpp:170
__scrt_common_main_seh exe_common.inl:253
BaseThreadInitThunk :
RtlUserThreadStart :

QGIS Info
QGIS Version: 3.3.0-Master
QGIS code revision: commit:a2db44c383
Compiled against Qt: 5.9.2
Running against Qt: 5.9.2
Compiled against GDAL: 2.2.4
Running against GDAL: 2.2.4

System Info
CPU Type: x86_64
Kernel Type: winnt
Kernel Version: 10.0.17134

History

#1 - 2018-09-21 04:58 PM - Alessandro Pasotti

- Resolution set to invalid

- Assignee set to Alessandro Pasotti

- Status changed from Open to Closed

you need to store a reference to the range object or the symbol will be garbage collected before you can call clone() on it:

```
layer = qgis.utils.iface.mapCanvas().currentLayer()
renderer = layer.renderer()
```

```
renderer_range = renderer.ranges()[1]
current_symbol = renderer_range.symbol()
new_symbol = current_symbol.clone()
```