QGIS Application - Bug report #8775
PyQGIS QgsPoint has a __hash__ function, even though it is mutable

2013-10-05 05:01 AM - J. Dugge

Status:	Closed		
Priority:	Normal		
Assignee:			
Category:	Python plugins		
Affected QGIS version:	master	**Regression?:**	No
Operating System:		**Easy fix?:**	No
Pull Request or Patch supplied:	No	**Resolution:**	end of life
Crashes QGIS or corrupts data:	No	**Copied to github as #:**	17481

Description

The QgsPoint class in PyQGIS has a (automatically generated?) __hash__ function, which returns a hash value that does not depend on the coordinates of the point. This leads to the inconsistent behaviour that two points that are equal according to QgsPoint.__eq__ do not have the same hash value, which causes problems with functions that rely on proper __hash__ behaviour, like set.

To reproduce this, load a polygon layer and run the following in the Python console:

```
provider = iface.activeLayer().dataProvider()
for f in provider.getFeatures():
  feature = f
points = f.geometry().asPolygon()[0]

points[0] == points[1]
# Returns True, the first and last points in a polygon are identical

set(points)
# The first/last point appears twice in the set, even though it should only appear once according to the equality
```

To fix this, the __hash__ function in QgsPoint should be removed (e.g. by setting QgsPoint.__hash__ = None), which will raise TypeError: unhashable type: 'QgsPoint' when set is used with a list of QgsPoint objects.

Associated revisions

Revision 44b77671 - 2013-10-12 12:26 PM - Matthias Kuhn

Create hash method for QgsPoint (Fix #8775)

History

#1 - 2013-10-06 04:48 AM - Matthias Kuhn

For reference:

http://www.mail-archive.com/pyqt@riverbankcomputing.com/msg15114.html

#2 - 2013-10-12 03:26 AM - Matthias Kuhn

- Status changed from Open to Closed

Fixed in changeset commit:"44b7767134e442b95b6d99a1cbe612d2aeb856c7".

#3 - 2013-10-12 05:15 AM - J. Dugge

Thanks for the quick reaction!

I think the changeset doesn't actually fix the issue though: QgsPoint is a mutable type (its value can be changed using `setX()`, for instance), and as such, it *mustn't* have a __hash__() function (see http://docs.python.org/2/glossary.html#term-hashable)

Consider the following to see why the new implementation is problematic:

```
a = QgsPoint(0,0)
b = QgsPoint(1,1)
c = set([a,b])

print c
# correctly returns [(0,0),(1,1)]

a.set(1,1)
print c
# returns [(1,1),(1,1)], which is incorrect
```

The proper way to fix this is to *remove* the __hash__ function by setting __hash__ = None, which explicitly marks the class as the mutable and unhashable type it is (http://docs.python.org/2/reference/datamodel.html#object.__hash__) so set operations (which don't work with mutable types) are disabled.

#4 - 2013-10-13 01:50 AM - J. Dugge

- Status changed from Closed to Reopened

#5 - 2017-05-01 01:09 AM - Giovanni Manghi

- Regression? set to No

- Easy fix? set to No

#6 - 2019-03-09 04:04 PM - Giovanni Manghi

- Resolution set to end of life

- Status changed from Reopened to Closed

End of life notice: QGIS 2.18 LTR

Source:

http://blog.qgis.org/2019/03/09/end-of-life-notice-qgis-2-18-ltr/