

QGIS Application - Bug report #8258

QgsVectorDataProvider.uniqueValues returns not None but QPyNullVariant

2013-07-10 07:17 PM - Minoru Akagi

Status:	Closed	
Priority:	Normal	
Assignee:		
Category:	Python plugins	
Affected QGIS version:	master	Regression?: No
Operating System:		Easy fix?: No
Pull Request or Patch supplied:		Resolution: fixed
Crashes QGIS or corrupts data:		Copied to github as #: 17062

Description

Open polygons.shp (testdata on #8219). The second field of the attribute table is a string type and includes NULL values. Then, in the python console:

```
>>> layer = iface.activeLayer()
>>> provider = layer.dataProvider()
>>> provider.uniqueValues(1)
[<PyQt4.QtCore.QPyNullVariant object at 0x10CABFB8>, u'pentagon', u'quadrangle', u'triangle']

>>> f = QgsFeature()
>>> provider.getFeatures( QgsFeatureRequest().setFilterFid( 0 ) ).nextFeature( f )
True
>>> f.attributes()
[u'f', None, 10]

>>> provider.uniqueValues(1)[0] == None
False
```

QGIS version:
1.9.0-Master Master, 630427f

History

#1 - 2013-08-28 04:35 AM - Matthias Kuhn

Will now be:

```
>>> layer = iface.activeLayer()
>>> provider = layer.dataProvider()
>>> provider.uniqueValues(1)
[NULL, u'pentagon', u'quadrangle', u'triangle']

>>> f = provider.getFeatures( QgsFeatureRequest().setFilterFid( 0 ) ).next()
True
>>> f.attributes()
[u'f', NULL, 10]

>>> provider.uniqueValues(1)[0] == None
```

```

False
>>> provider.uniqueValues(1)[0] is None
False
>>> provider.uniqueValues(1)[0].isNull()
True
>>> if not provider.uniqueValues(1)[0]:
>>>     'it is NOT'
'it is NOT'

```

Now everything returning a NULL attribute returns a QPyNullVariant (by design) but will be represented as "NULL" on the console

#2 - 2013-08-28 06:26 PM - Minoru Akagi

The evaluation of (None == None) is True. But the evaluation of the following conditional expression is False.

```

>>> provider.uniqueValues(1)[0] == f.attributes()[1]
False

```

It makes sense to me that (QPyNullVariant == QPyNullVariant) is evaluated as True.

#3 - 2013-08-28 06:29 PM - Nathan Woodrow

You would think so but it's not the case.

We should be able to fix that by overloading the *eq* method. Hold tight.

#4 - 2013-08-28 06:35 PM - Nathan Woodrow

We can just do this:

```

from types import MethodType
from PyQt4.QtCore import QPyNullVariant

def __nonzero__(self):
    return False

def __eq__(self, other):
    return True

QPyNullVariant.__nonzero__ = MethodType(__nonzero__, None, QPyNullVariant)
QPyNullVariant.__eq__ = MethodType(__eq__, None, QPyNullVariant)

```

which will work like so:

```

>>> n = QPyNullVariant('QString')
>>> n2 = QPyNullVariant('QString')
>>> n == n2

```

True

Just be aware this doesn't compare the internal type of QPyNullVariant so maybe we should just compare that rather than returning True. In practice should this work?

```
>>> n = QPyNullVariant('QString')
>>> n2 = QPyNullVariant('int')
>>> n == n2
True
```

or do we not really care in this case? Because we are just trying to do: NULL == NULL

#5 - 2013-08-29 08:32 PM - Nathan Woodrow

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

This is as fixed as we are going to be able to do.

```
>>> n = QPyNullVariant('QString')
>>> n2 = QPyNullVariant('QString')
>>> n == n2
True
>>> n = QPyNullVariant('QString')
>>> n2 = QPyNullVariant('int')
>>> n == n2
True
>>> n = QPyNullVariant('QString')
>>> n == None
True
>>> n = QPyNullVariant('QString')
>>> n is None
False
>>> n = 100
>>> n2 = QPyNullVariant('int')
>>> n == n2
False
```

There is also a new shortcut NULL type in qgis.core which is a QPyNullVariant. So you can just do:

```
>>> from qgis.core import NULL
>>> provider.uniqueValues(1)[0] == NULL
True
```

#6 - 2013-08-29 09:02 PM - Minoru Akagi

- Status changed from Closed to Reopened

Great works! Thanks, Matthias and Nathan!!

Now the evaluation of the following conditional expression is True.

```
>>> provider.uniqueValues(1)[0] == f.attributes()[1]
True
```

However, the following is also True!

```
>>> provider.uniqueValues(1)[0] != f.attributes()[1]
True
```

Probably, we should override *ne* method too as below:

```
def __ne__(self, other):
    return not self.__eq__(other)

QPyNullVariant.__ne__ = MethodType(__ne__, None, QPyNullVariant)
```

#7 - 2013-08-29 09:07 PM - Nathan Woodrow

Good catch!

Will fix that now.

#8 - 2013-08-30 06:00 AM - Minoru Akagi

- % Done changed from 0 to 100

- Status changed from Reopened to Closed