QGIS Application - Bug report #16068
getFeatures() as well as spartial indices are inaccurate with memory layers

2017-01-12 07:10 AM - Georg Wicke

Status:	Closed		
Priority:	Normal		
Assignee:			
Category:	Python plugins		
Affected QGIS version:	2.18.0	**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 #:**	23983

Description

This bug report is essentially the question I asked on StackExchange already: [[http://gis.stackexchange.com/questions/222481/getfeatures-is-inaccurate-with-memory-layers]]

Steps to reproduce:

1. Create new in-memory VectorLayer with the same CRS as an existing vector layer
2. Copy all fields and attributes into the new layer
3. Use a spartial index or a getFeatures(QgsFeatureRequest) call to query for the features at a specific point on the map
Result: The features returned are very inaccurate. Much more features than actually covered by the feature request are returned.

Environment

- QGIS 2.18.0
- Windows 10
- Tested with a QgsMapTool-derived class (see also StackOverflow link at the top on more information how to get this bug or contact me in case that you have problems reproducing it
- Map geometry type was Polygon
- Map was a set of non-intersecting, but touching areas (tested with multiple maps of this type)
- WKT string of the map projection:

```
PROJCS["unnamed",GEOGCS["Bessel
1841",DATUM["unknown",SPHEROID["bessel",6377397.155,299.1528128]],PRIMEM["Greenwich",0],UNIT["degree",0.0174532925199433]
925199433]],PROJECTION["Lambert_Conformal_Conic_2SP"],PARAMETER["standard_parallel_1",48.66666666666666],PARAMETER["st
METER["standard_parallel_2",53.66666666666666],PARAMETER["latitude_of_origin",0],PARAMETER["central_meridian",10],PARAMETER
ARAMETER["false_easting",0],PARAMETER["false_northing",0],UNIT["Meter",1]]
```

Related issues:

Related to QGIS Application - Bug report # 16544: Memory Layers do not seem t...	**Closed**	**2017-05-12**

History

#1 - 2017-01-12 07:11 AM - Georg Wicke

Damn it, can I change the formatting afterwards? I thought the .h1 command would just affect the current line... sorry...

#2 - 2017-01-28 08:01 PM - Martin Dobias

- Status changed from Open to Feedback

By default feature request with filtering based on an extent picks all features that have their bounding box intersecting the given extent, so you may get more features. This is done for performance reasons - true intersection test with geometries is a more expensive operation. If you want true intersection,

add ExactIntersect flag to your feature request (this is e.g. what identify tool in QGIS does).

Please close this issue if the above suggestion solves your problem - or post a script that reproduces the problem if it is still a valid problem.

#3 - 2017-02-02 07:32 AM - Georg Wicke

Okay, but then the point is that the default behaviour for OGR layers is the exact pick, while the default behaviour for memory layers is bounding box pick. In summary, there are three problems with this:

1. It is not intuitive.
2. It is not documented.
3. It is not consistent with the standard for OGR layers.

I found nothing in the docs, nothing on google and on StackOverflow noone could answer me this question for multiple weeks. I haven't tried your fix yet due to the fact that I've already written a workaround. I would expect it to work, but even if it did, I wouldn't like to close this ticket, because I'm pretty sure that future users will have the same problem. If you can't change it back because it will break everything, please at least document it properly. I think selling this bug as a feature is pretty cheap, especially because it's just some words of documentation or a changed default value that would need to be changed, which is really not that much work. Heck, I would even do that myself if you guys are busy with other things. (I would need to set up a development environment for QGIS though, which could be a bit disproportional.)

Don't get me wrong on this, I am grateful that you took the time to answer me and I'm grateful to everyone involved in this project for creating such a great and useful piece of software. But I think that this is a bit sub-standard.

#4 - 2017-02-02 08:26 AM - Georg Wicke

Okay, I tested it with memory layers and added the ExactIntersect flag to my request. Now I'm not getting any results any more. I did the sanity check; without the flag, it works, but has the described inaccuracies. So it looks like there's another bug there, too.

#5 - 2017-02-02 09:06 AM - Georg Wicke

Oh, and by the way: if you try to check containment with `feature.geometry().contains(point)`, you seem to always get False as a result (at least with this setup with the memory layer), while `feature.geometry().intersects(QgsRectangle(point, point))` seems to work. Should I open another bug report?

#6 - 2017-02-02 10:48 AM - Matthias Kuhn

Can you paste the output of

```
feature.geometry().exportToWkt()
point.exportToWkt()
```

It should be easy to add a test to https://github.com/qgis/QGIS/blob/master/tests/src/python/test_qgsgeometry.py with this information (If you can already make a pull request with a failing test, that would be really nice).

I also added a request for an API adjustment for QGIS 3.0, feel free to add your input there:
https://github.com/qgis/qgis3.0_api/issues/74

FYI; The behaviour is already documented in http://qgis.org/api/classQgsFeatureRequest.html -> Detailed Description, but it would certainly be nice to also

add a note for this in setFilterRect and the QgsRectangle constructur.

#7 - 2017-03-06 10:19 AM - Giovanni Manghi

- Category set to Python plugins

- Status changed from Feedback to Open

#8 - 2017-05-01 01:01 AM - Giovanni Manghi

- Regression? set to No

- Easy fix? set to No

#9 - 2017-05-19 08:58 AM - Jürgen Fischer

- Related to Bug report #16544: Memory Layers do not seem to support the ExactIntersect flag for getFeatures(QgsMapRequest) added

#10 - 2019-03-09 03:09 PM - Giovanni Manghi

- Resolution set to end of life

- Status changed from Open 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/

QGIS 3.4 has recently become our new Long Term Release (LTR) version. This is a major step in our history – a long term release version based on the massive updates, library upgrades and improvements that we carried out in the course of the 2.x to 3x upgrade cycle.

We strongly encourage all users who are currently using QGIS 2.18 LTR as their preferred QGIS release to migrate to QGIS 3.4. This new LTR version will receive regular bugfixes for at least one year. It also includes hundreds of new functions, usability improvements, bugfixes, and other goodies. See the relevant changelogs for a good sampling of all the new features that have gone into version 3.4

Most plugins have been either migrated or incorporated into the core QGIS code base.

We strongly discourage the continued use of QGIS 2.18 LTR as it is now officially unsupported, which means we'll not provide any bug fix releases for it.

You should also note that we intend to close all bug tickets referring to the now obsolete LTR version. Original reporters will receive a notification of the ticket closure and are encouraged to check whether the issue persists in the new LTR, **in which case they should reopen the ticket**.

If you would like to better understand the QGIS release roadmap, check out our roadmap page! It outlines the schedule for upcoming releases and will help you plan your deployment of QGIS into an operational environment.

The development of QGIS 3.4 LTR has been made possible by the work of hundreds of volunteers, by the investments of companies, professionals, and administrations, and by continuous donations and financial support from many of you. We sincerely thank you all and encourage you to collaborate and support the project even more, for the long term improvement and sustainability of the QGIS project.