QGIS Application - Bug report #9777
Simplification of polygons may return invalid polygons

2014-03-13 04:31 AM - Hugo Mercier

Status:	Closed		
Priority:	Low		
Assignee:			
Category:	Vectors		
Affected QGIS version:	master	**Regression?:**	No
Operating System:		**Easy fix?:**	No
Pull Request or Patch supplied:	Yes	**Resolution:**	
Crashes QGIS or corrupts data:	No	**Copied to github as #:**	18319

Description

Even after #9655 fix, some polygons may still be invalid after simplification.

Here is an example of a self-intersected multipolygon.

I am not sure this has to be considered a bug, since the user is warned that simplification may results in inconsistencies. But, shouldn't the simplifier returns the original geometry in case of an invalid simplified one ? (because, except using some topology-oriented geometry fixer, I cannot see any good solution to that).

History

#1 - 2014-03-13 09:04 AM - Alvaro Huarte

The on-the-fly-simplification capability of QGIS 2.2 was developed to get fast rendering of geometries. It can draw self-intersected polygons (The simplification only simplify the geometries loaded to use in one rendering task) especially if the threshold is very high.

These geometries to draw could be fixed checking its topology after the simplification works, but then the performance will be much worse than original. For layers where it is not acceptable show self-intersections you can disable it, or even disable it by default for all layers.

Alvaro

#2 - 2014-03-13 09:14 AM - Alvaro Huarte

My previous comment is about rendering simplification, if you're talking about an tool to simplify vectors then, of course, it will be possible fix the geometries after run the simplification. Now, there is an easy trick, execute a buffer of 0, it fixes the most of invalid geometries.

Alvaro

#3 - 2014-03-13 09:24 AM - Hugo Mercier

Yes, I was talking about rendering simplification (MapToPixelGeometrySimplifier).
My problem is when using $geometry in a labelling expression and spatial predicates, like contains().
It is prone to failure because $geometry may be invalid.
I did circumvent this by creating a user-function where the geometry is tested for validity then buffer(0) is called in case of invalidity ... it works, but it is not straightforward.

I have to think about it a bit, but a solution would be to have something like "$exact_geometry" available in order to access the current geometry before simplification

#4 - 2014-03-13 09:33 AM - Alvaro Huarte

Hugo Mercier wrote:

> *I have to think about it a bit, but a solution would be to have something like "$exact_geometry" available in order to access the current geometry before simplification*

The original geometry can not fetched in all cases (e.g. postgis provider) the simplification runs before fetch the geometry from the feature provider.

Could be the solution to add a new checkbox to the simplification configuration panel to force the geometry validation?

#5 - 2014-03-13 09:46 AM - Hugo Mercier

Alvaro Huarte wrote:

> *The original geometry can not fetched in all cases (e.g. postgis provider) the simplification runs before fetch the geometry from the feature provider.*

Not sure to understand this. simplify() is done in nextFeature() just after fetching the geometry, right ? The simplification is currently done in place (the geometry is replaced by its simplified version). My proposal here is to store two versions of the geometry : the original one and the simplified one, not sure exactly how however.

> *Could be the solution to add a new checkbox to the simplification configuration panel to force the geometry validation?*

That could be an alternative, but we would loose the benefits we gained from simplification as you stated earlier : testing the validity of each geometry takes time ...

#6 - 2014-03-13 09:59 AM - Alvaro Huarte

Hugo Mercier wrote:

> *Not sure to understand this. simplify() is done in nextFeature() just after fetching the geometry, right ? The simplification is currently done in place (the geometry is replaced by its simplified version). My proposal here is to store two versions of the geometry : the original one and the simplified one, not sure exactly how however.*

For postgis layers, the simplification can run on provider side, it add the "ST_SnapToGrid" method to the SQL request to fetch geometries:
"SELECT a,b,c,ST_SnapToGrid(the_geom) FROM table", then the original geometry is not fetched,

#7 - 2014-03-13 10:22 AM - Hugo Mercier

Ok, I see.
So it would need to fetch the two geometries. And we probably do not want that ...
Thanks for clarifying this.

#8 - 2014-03-13 01:03 PM - Alvaro Huarte

Hi Hugo, I am going to study if the simplification code can avoid create invalid geometries.
All comments are welcome!

Alvaro

#9 - 2014-03-14 04:26 AM - Alvaro Huarte

- Pull Request or Patch supplied changed from No to Yes

Hi Hugo, I proposed a new pull request to fix this issue:
https://github.com/qgis/QGIS/pull/1241

I added a new method to QgsGeometry class to enable the validation of a geometry when it is required export it to GEOS geometry. It ensures the simplified geometries are valid. The simplification code enables this new flag, disabled by default, when a geometry is simplified (No generalized by its bounding box).

The validation code is only executed when needed, drawing labels of a simplified geometry, or any code requires the GEOS geometry and this flag is enabled. If a simplified geometry is drawed without internally call to 'exportWkbToGeos()', as usual, then the validation is not executed and there is no penalty in the performance.

Best Regards
Alvaro

#10 - 2014-03-14 05:27 AM - Hugo Mercier

Hi Alvaro,
Thanks for your work ! Putting the validation (0-buffer) in exportWkbToGeos() seems indeed a good idea.

Do you know if there could still be cases where a buffer of 0 is not enough to fix a simplified geometry ?

#11 - 2014-03-14 05:42 AM - Alvaro Huarte

Hi Hugo, thanks to you for tests and ideas, I think that it is needed to write a ST_MakeValid implementation for QgsGeometry class.
There are other issues where it is commented.

IMHO, now, using buffer_0 seems work sufficiently fine, but certainly can appear bad geometries.

Alvaro

#12 - 2014-04-02 03:16 PM - Alvaro Huarte

I have improved the fix implementing the **ST_MakeValid** function for QgsGeometry class. The code is a fork from the original code in postgis implementation with slight changes:

https://github.com/postgis/postgis/blob/svn-trunk/liblwgeom/lwgeom_geos_clean.c

Also, the user can configure automatic validation of the fetched geometries of a layer and convert them to valid when needed.

https://github.com/qgis/QGIS/pull/1241

Best Regards

Alvaro

#13 - 2014-06-23 07:27 AM - Regis Haubourg

Is the ticket still opened? In a labeling use case, i do not have anymore troubles.

#14 - 2014-06-23 08:12 AM - Alvaro Huarte

Regis Haubourg wrote:

> *Is the ticket still opened? In a labeling use case, i do not have anymore troubles.*

Really simplification can generate self-crossing polygons, the PR (https://github.com/qgis/QGIS/pull/1241) not merged allows convert invalid geometries, from simplification or not, to valid with a ST_MakeValid implementation such postgis does.

#15 - 2015-05-21 02:03 AM - Saber Razmjooei

- Status changed from Open to Closed

- Priority changed from Normal to Low

Tried in 2.8.2 and can't reproduce.

Files

cap.jpg	172 KB	2014-03-13	Hugo Mercier
invalid_simplification.sqlite	4.08 MB	2014-03-13	Hugo Mercier