QGIS Animation Workbench

The QGIS Animation Plugin

Bringing your QGIS maps to life!

Tim Sutton, Nyall Dawson

Table of contents

1. Welcome to the Animation Workbench

This is a great plugin!

2. Quickstart

2.1 Installing the QGIS Animation Workshop plugin

2.2 Initial Configuration

3. Frequently Asked Questions

4. Tutorials

4.1 Tutorials

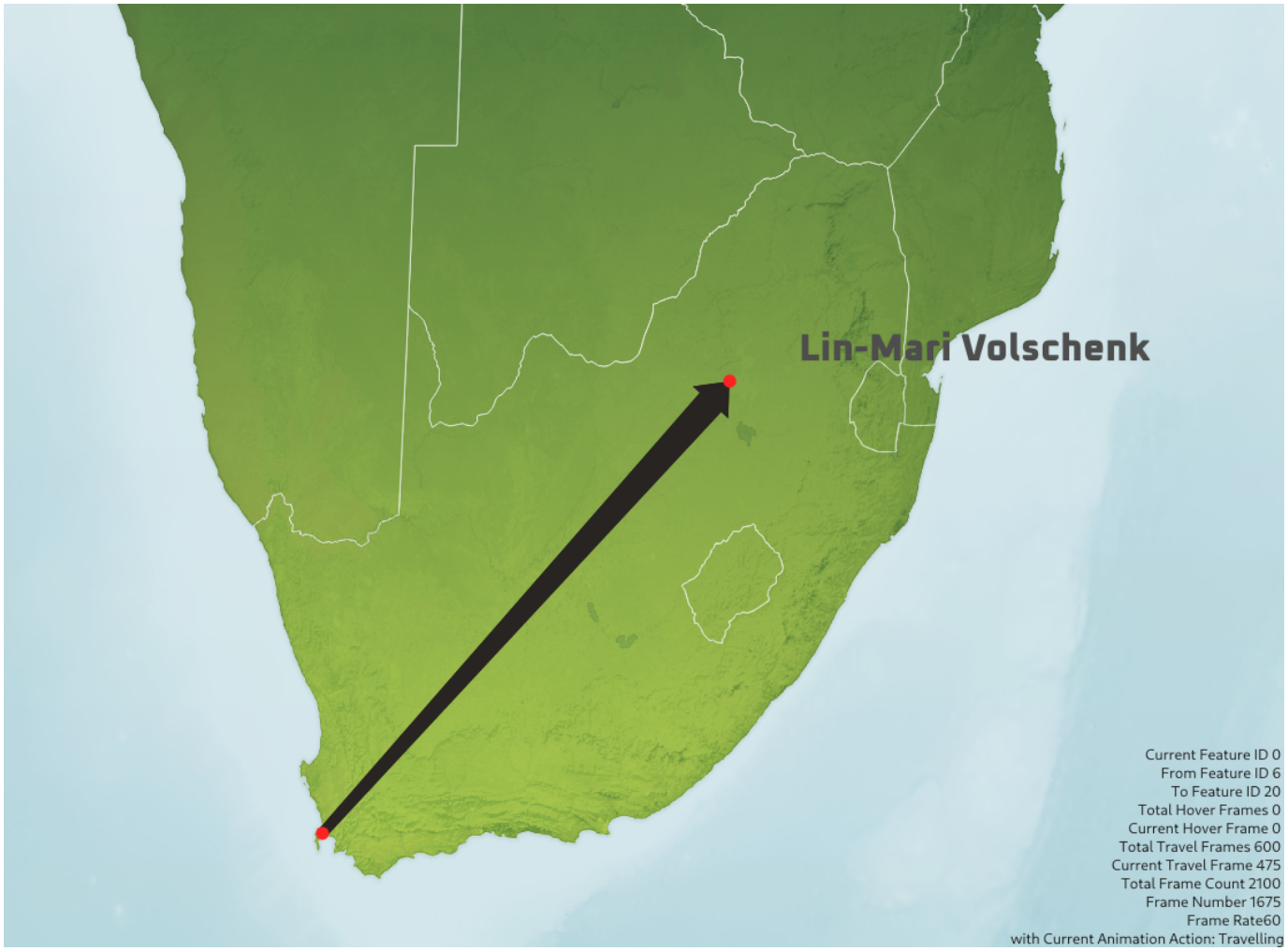
5. Library

5.1 Snippets

5.1.1 Line of travel

In this example we use a geometry generator to create a line between the origin point and the destination point:

```
if (@from_feature_id = $id OR @to_feature_id = $id,
 -- read this from inside to out so
 -- last tranform the geometry back to the map crs
 transform(
  -- densify the geometry so that when we transform
  -- back it makes a great circle
  densify_by_count(
   -- move the geometry into a crs that
   -- shows a great circle as a straight line
   transform(
    -- make a line from the previous pont to the next point
    make_line(
     geometry(@from_feature),
     geometry(@to_feature)
    ),
    @map_crs, 'EPSG:4326'),
   99),
  'EPSG:4326',  @map_crs),
 None)
```

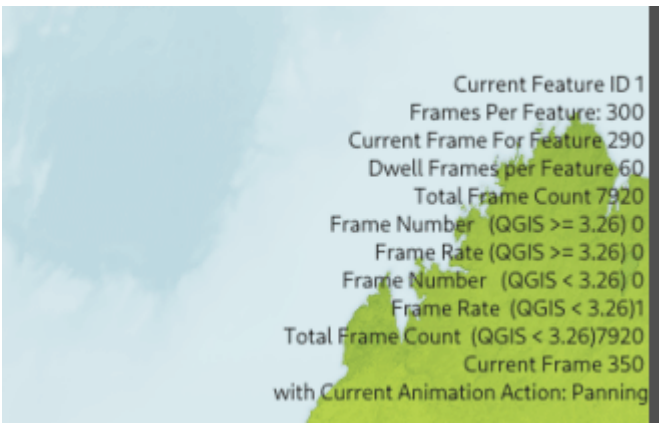


5.1.2 Showing diagnostic info as a copyright label

Showing diagnostic information in the QGIS copyright label:

```
[%
'Feature Variables:' ||
' \n------------------------' ||
' \nPrevious Feature ' || to_string(coalesce(attribute(@previous_feature, 'name'), '-'))  ||
' \nPrevious Feature ID ' || to_string(coalesce(@previous_feature_id, '-'))  ||
' \n' ||
' \nNext Feature ' || to_string(coalesce(attribute(@next_feature, 'name'), '-'))  ||
' \nNext Feature ID ' || to_string(coalesce(@next_feature_id, '-'))  ||
' \n' ||
' \nHover Feature ' || to_string(coalesce(attribute(@hover_feature, 'name'), '-'))  ||
' \nHover Feature ID ' || to_string(coalesce(@hover_feature_id, '-'))  ||
' \n' ||
' \nFrom Feature ' || to_string(coalesce(attribute(@from_feature, 'name'), '-'))  ||
' \nFrom Feature ID ' || to_string(coalesce(@from_feature_id, '-'))  ||
' \n' ||
' \nTo Feature ' || to_string(coalesce(attribute(@to_feature, 'name'), '-'))  ||
' \nTo Feature ID ' || to_string(coalesce(@to_feature_id, '-'))  ||
' \n' ||
' \nTotal Hover Frames ' || to_string(coalesce(@hover_frames, 0))  ||
' \nCurrent Hover Frame ' || to_string(coalesce(@current_hover_frame, 0))  ||
' \nTotal Travel Frames ' || to_string(coalesce(@travel_frames, 0))  ||
' \nCurrent Travel Frame ' || to_string(coalesce(@current_travel_frame, 0))  ||
' \nTotal Frame Count ' || to_string(coalesce(@total_frame_count, 0))  ||
' \nFrame Number ' || to_string(coalesce(@frame_number, 0))  ||
' \nFrame Rate ' || to_string(coalesce(@frame_rate, 0))  ||
' \nwith Current Animation Action: ' || @current_animation_action ||
' \nTo Direction ' ||  coalesce(format_number(degrees(azimuth( geometry(@hover_feature), geometry(@previous_feature) ) ) ), 0) ||
' \nFrom Direction ' ||  coalesce(format_number(degrees( azimuth( geometry(@hover_feature), geometry(@next_feature) ) ) ), 0)
%]
```

Example output:



5.1.3 Variable size of labels

Variably changing the size on a label as we approach it in the animation:

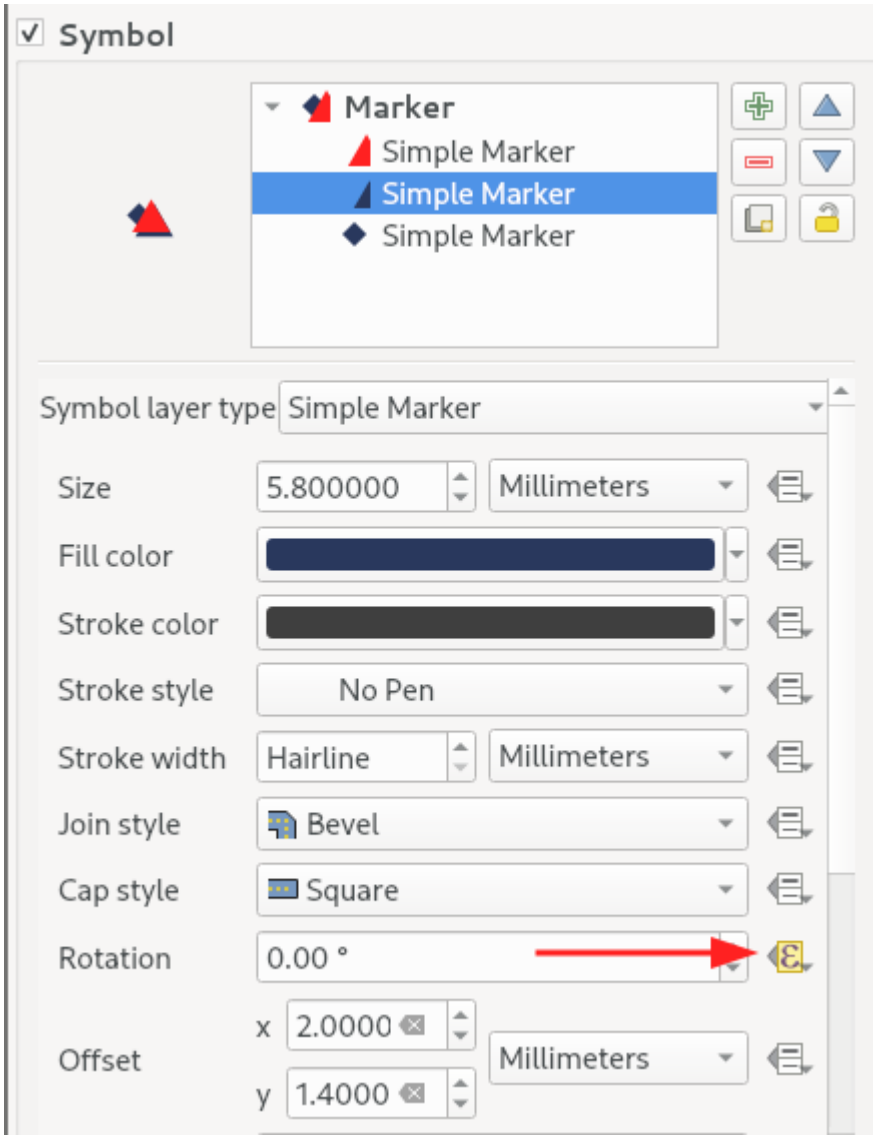
```40 * ((@frame_number % @hover_frames) / @hover_frames)

```
Calculating the angle between points

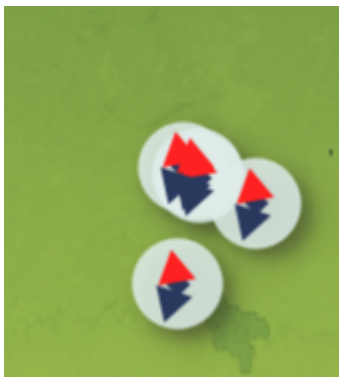
You can calculate the angle between the hover point and the previous point like this:

```python
coalesce(
 format_number(
  degrees(
   azimuth(
    geometry(@hover_feature),
    geometry(@previous_feature)
   )
  )
 ), 0)
```

5.1.4 Rotation

You can set the angle of rotation for a symbol using this expression:





Using this technique you can also create an animation effect showing the source direction of travel and the new destination.

```
scale_linear (
 @current_hover_frame,
 0,
 @hover_frames,
 degrees(
  azimuth(
```

```
    geometry(@hover_feature),
    geometry(@previous_feature)
   )
  ),
  degrees(
   azimuth(
    geometry(@hover_feature),
    geometry(@next_feature)
   )
  )
)
```

Will produce something like this:

6. Develop

6.1 Setup

6.2 Design

6.3 Working with documentation

Documentation is written using mkdocs.

6.3.1 Building documentation PDF

You can build a copy of the documentation as a PDF file using the following steps:

```
pip install mkdocs-with-pdf
pip install mkdocs-material
pip install qrcode
mkdocs build
xdg-open pdfs/QGISAnimationPlugin.pdf
```

https://github.com/timlinux/QGISAnimationWorkbench