



Geoc

Jared Erickson

Version 0.12.0-SNAPSHOT

# Table of Contents

|                                |    |
|--------------------------------|----|
| Core Commands .....            | 1  |
| List .....                     | 1  |
| Version .....                  | 2  |
| Help .....                     | 2  |
| Pipe .....                     | 3  |
| Shell .....                    | 4  |
| Carto Commands .....           | 8  |
| Map .....                      | 8  |
| Filter Commands .....          | 38 |
| CQL to XML .....               | 38 |
| Geometry Commands .....        | 39 |
| Convert .....                  | 39 |
| Decimal Degrees to Point ..... | 39 |
| GeoHash Bounds .....           | 40 |
| GeoHash Decode .....           | 40 |
| GeoHash Encode .....           | 41 |
| GeoHash Neighbors .....        | 41 |
| Great Circle Arc .....         | 42 |
| Offset .....                   | 43 |
| orthodromicDistance .....      | 44 |
| Plot .....                     | 44 |
| Point to Decimal Degrees ..... | 45 |
| Map Commands .....             | 46 |
| Map Layers .....               | 46 |
| Draw .....                     | 47 |
| Map Cube .....                 | 48 |
| Projection Commands .....      | 50 |
| Envelope .....                 | 50 |
| WKT .....                      | 51 |
| Raster Commands .....          | 52 |
| Envelope .....                 | 52 |
| Info .....                     | 53 |
| Get Size .....                 | 54 |
| Style Commands .....           | 54 |
| Create .....                   | 55 |
| CSS to SLD .....               | 56 |
| SLD to Ysld .....              | 58 |
| Ysld to SLD .....              | 60 |

|                                  |     |
|----------------------------------|-----|
| Unique Values from Text .....    | 61  |
| Tile Commands .....              | 66  |
| Tile Layers .....                | 66  |
| Delete .....                     | 67  |
| Generate .....                   | 68  |
| Tile Bounds .....                | 77  |
| List Tiles .....                 | 77  |
| Pyramid .....                    | 78  |
| Stitch Raster .....              | 86  |
| Stitch Vector .....              | 87  |
| Vector Grid .....                | 87  |
| Vector Commands .....            | 88  |
| Add .....                        | 89  |
| Add Fields .....                 | 89  |
| Add Area Field .....             | 90  |
| Add Length Field .....           | 91  |
| Add XY Fields .....              | 92  |
| Append .....                     | 93  |
| Buffer .....                     | 93  |
| Centroid .....                   | 94  |
| Convexhull .....                 | 95  |
| Convexhulls .....                | 96  |
| Count .....                      | 97  |
| Create .....                     | 97  |
| Delaunay .....                   | 98  |
| Ellipse .....                    | 99  |
| Envelope .....                   | 99  |
| Envelopes .....                  | 100 |
| From .....                       | 101 |
| Graticule - Hexagon .....        | 103 |
| Graticule - Line .....           | 103 |
| Graticule - Oval .....           | 104 |
| Graticule - Rectangle .....      | 105 |
| Graticule - Square .....         | 106 |
| Info .....                       | 107 |
| Interior Point .....             | 108 |
| Layer List .....                 | 109 |
| Minimum Bounding Circle .....    | 109 |
| Minimum Bounding Circles .....   | 110 |
| Minimum Bounding Rectangle ..... | 111 |
| Minimum Bounding rects .....     | 112 |

|                     |     |
|---------------------|-----|
| Project.....        | 112 |
| Random Points ..... | 113 |
| To .....            | 114 |
| Schema .....        | 115 |
| Voronoi.....        | 119 |

geoc is a geospatial command line application that follows the unix philosophy. Each command does one thing well (buffer a layer, crop a raster) by reading a vector layer as a CSV text stream or a raster layer as an ASCII grid, processing the layer or raster, and then writing out the vector layer as a CSV or a raster layer as an ASCII grid. Individual commands can be chained together with unix pipes.

## Core Commands

### List

List all command names.

| Short Name | Long Name     | Description             |
|------------|---------------|-------------------------|
| -d         | --description | Include the description |
|            | --help        | Print the help message  |
|            | --web-help    | Open help in a browser  |

```
geoc list
```

```
carto map
filter cql2xml
geometry convert
geometry dd2pt
geometry geohash bounds
geometry geohash decode
geometry geohash encode
geometry geohash neighbors
geometry greatcirclearc
geometry offset
...
```

List all commands names with a short description.

```
geoc list -d
```

```
carto map = Create a cartographic map
filter cql2xml = Convert a CQL statement to an OCG XML Filter
geometry convert = Convert a geometry from one format to another
geometry dd2pt = Convert a decimal degrees formatted string into a Point
geometry geohash bounds = Calculate the geohashes for the given bounds
geometry geohash decode = Decode a GeoHash to a Geometry.
geometry geohash encode = Encode a Geometry as a GeoHash
geometry geohash neighbors = Get a geohash's neighbors
geometry greatcirclearc = Create a great circle arc.
geometry offset = Create a Geometry offset from the input Geometry
...
...
```

## Version

Get the current version.

| Short Name | Long Name  | Description            |
|------------|------------|------------------------|
|            | --help     | Print the help message |
|            | --web-help | Open help in a browser |

```
geoc version
```

```
0.20.0-SNAPSHOT
```

## Help

You can get help from any subcommand.

```
geoc vector buffer --help
```

`geoc vector buffer`: Buffer the features of the input Layer and save them to the output Layer

|  |  |
|--|--|
| <code>--help</code>                      | : Print the help message (default: true)                           |
| <code>--web-help</code>                  | : Open help in a browser (default: false)                          |
| <code>-c (--capstyle) VAL</code>         | : The cap style (default: round)                                   |
| <code>-d (--distance) VAL</code>         | : The buffer distance  |
| <code>-i (--input-workspace) VAL</code>  | : The input workspace  |
| <code>-l (--input-layer) VAL</code>      | : The input layer  |
| <code>-o (--output-workspace) VAL</code> | : The output workspace   |
| <code>-q (--quadrantsegments) N</code>   | : The number of quadrant segments (default: 8)                     |
| <code>-r (--output-layer) VAL</code>     | : The output layer   |
| <code>-s (--singlesided)</code>          | : Whether buffer should be single sided or not<br>(default: false) |

## Pipe

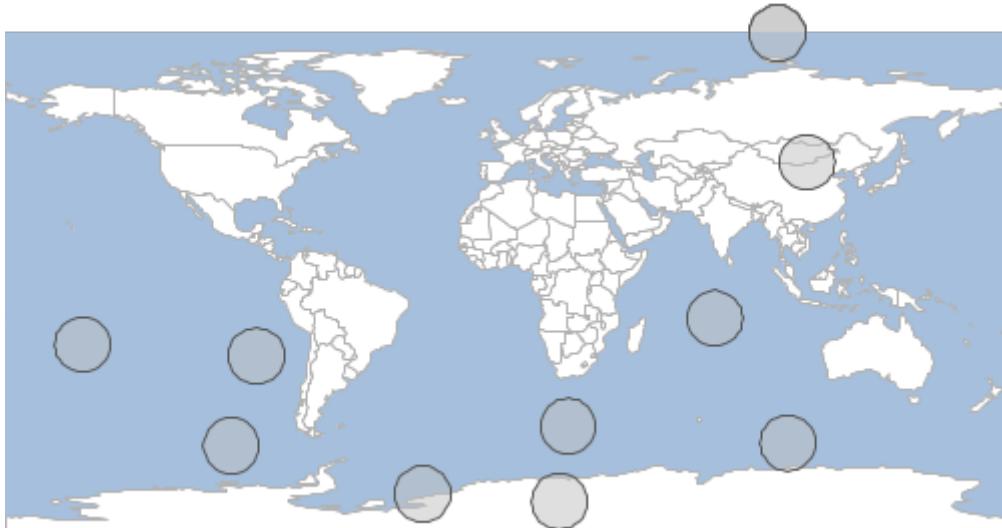
Combine multiple commands together with a pipe.

| Short Name      | Long Name               | Description               |
|-----------------|-------------------------|---------------------------|
| <code>-c</code> | <code>--commands</code> | Commands separate by pipe |
|                 | <code>--help</code>     | Print the help message    |
|                 | <code>--web-help</code> | Open help in a browser    |

```
geoc pipe -c vector randompoints -n 10 -g -180,-90,180,90 | vector buffer -d 10
```

```
"id:Integer","the_geom:Polygon:EPSG:4326"
"0","POLYGON ((-19.947715091207186 -75.99635909537186, -20.139862287174882
-77.94726231553314, -20.70891976609432 -79.82319341902276, -21.633018968181734
-81.55206142556789, -22.87664727934171 -83.06742690723733, -24.392012761011163
-84.31105521839731, -26.120880767556287 -85.23515442048472, -27.996811871045903
-85.80421189940417, -29.947715091207186 -85.99635909537186, -31.89861831136847
-85.80421189940417, -33.774549414858086 -85.23515442048472, -35.503417421403206
-84.31105521839731, -37.01878290307266 -83.06742690723733, -38.26241121423264
-81.55206142556789, -39.186510416320054 -79.82319341902276, -39.755567895239494
-77.94726231553314, -39.947715091207186 -75.99635909537186, -39.755567895239494
-74.04545587521058, -39.186510416320054 -72.16952477172096, -38.26241121423264
-70.44065676517585, -37.01878290307266 -68.92529128350638, -35.503417421403206
-67.6816629723464, -33.774549414858086 -66.757563770259, -31.898618311368473
-66.18850629133955, -29.94771509120719 -65.99635909537186, -27.996811871045903
-66.18850629133955, -26.120880767556287 -66.757563770259, -24.392012761011166
-67.6816629723464, -22.87664727934171 -68.92529128350638, -21.633018968181734
-70.44065676517583, -20.708919766094322 -72.16952477172096, -20.139862287174886
-74.04545587521056, -19.947715091207186 -75.99635909537186))"
```

...



## Shell

Run commands in an interactive shell.

| Short Name | Long Name  | Description            |
|------------|------------|------------------------|
|            | --help     | Print the help message |
|            | --web-help | Open help in a browser |

```
geoc shell
```

The screenshot shows a terminal window with the following details:

- Title Bar:** jericks — java < geoc shell — 80x24
- Current Directory:** ~ — java < geoc shell
- Last Login:** Wed Oct 5 18:18:54 on ttys002
- User:** [jericks@Jareds-MacBook-Pro-2 ~ % geoc shell
- Content:** A large, complex ASCII art logo consisting of various symbols like slashes, parentheses, and underscores, forming a stylized tree or forest.
- Prompt:** geoc> [cursor]

You can now type commands in the interactive shell.

If you hit the **tab** key you can get command line completion.

You can use the tab key again to cycle through the suggested values and hit the **return** key to select one.

```
Last login: Wed Oct  5 18:18:54 on ttys002
[jericks@Jareds-MacBook-Pro-2 ~ % geoc shell

[geoc> [carto      geometry    map        proj       shell       tile       version
filter     list        pipe       raster      style

```

In this example, we are looking for the vector contains command, so after typing vector c and hitting tab, we get a list of all vector commands that begin with the letter c.

The screenshot shows a macOS terminal window with the title bar "jericks — java - geoc shell — 80x24". The main pane displays a command-line interface for a tool named "geoc". The user has typed "[geoc> vector c]" and is currently completing a command. A list of available options is shown in blue text:

```
[geoc> vector c]
centroid      contains      coordinates      create
clip          convexhull    copy
compareschemas  convexhulls  count
```

Once we have found our command, the shell will also provide completion for options.

```
[geoc> vector buffer -]
--capstyle           --output-workspace   -i
--distance          --quadrantsegments -l
--help              --singlesided        -o
--input-layer       --web-help          -q
--input-workspace   -c                  -r
--output-layer      -d                  -s
```

# Carto Commands

## Map

Create a cartographic map from a JSON or XML definition file.

| Short Name | Long Name     | Description                            |
|------------|---------------|--|
| -t         | --type        | The type of the carto file (json, xml) |
| -c         | --carto-file  | The input carto definition file        |
| -o         | --output-file | The output file                        |
|            | --help        | Print the help message                 |
|            | --web-help    | Open help in a browser                 |

## JSON

Create a cartographic map from a JSON definition file.

```
geoc carto map -t json -c src/test/resources/carto/simple.json -o  
target/carto_simple_json.png
```

```
{
  "type": "png",
  "width": 792,
  "height": 612,
  "items": [
    {
      "x": 0,
      "y": 0,
      "width": 792,
      "height": 612,
      "type": "rectangle",
      "fillColor": "white",
      "strokeColor": "black"
    },
    {
      "x": 30,
      "y": 50,
      "width": 742,
      "height": 20,
      "type": "text",
      "text": "Map Title",
      "color": "Black",
      "horizontalAlign": "center",
      "verticalAlign": "middle",
      "font": {
        "name": "Arial",
        "style": "Bold",
        "size": 36
      }
    },
    {
      "x": 30,
      "y": 120,
      "width": 742,
      "height": 470,
      "type": "map",
      "name": "mainMap",
      "layers": [
        {"layertype": "layer", "file": "src/test/resources/data.gpkg", "layername": "ocean", "style": "src/test/resources/ocean.sld"},  

        {"layertype": "layer", "file": "src/test/resources/data.gpkg", "layername": "countries", "style": "src/test/resources/countries.sld"}
      ]
    }
  ]
}
```

# Map Title



## XML

Create a cartographic map from an XML definition file.

```
geoc carto map -t xml -c src/test/resources/carto/simple.xml -o  
target/carto_simple_xml.png
```

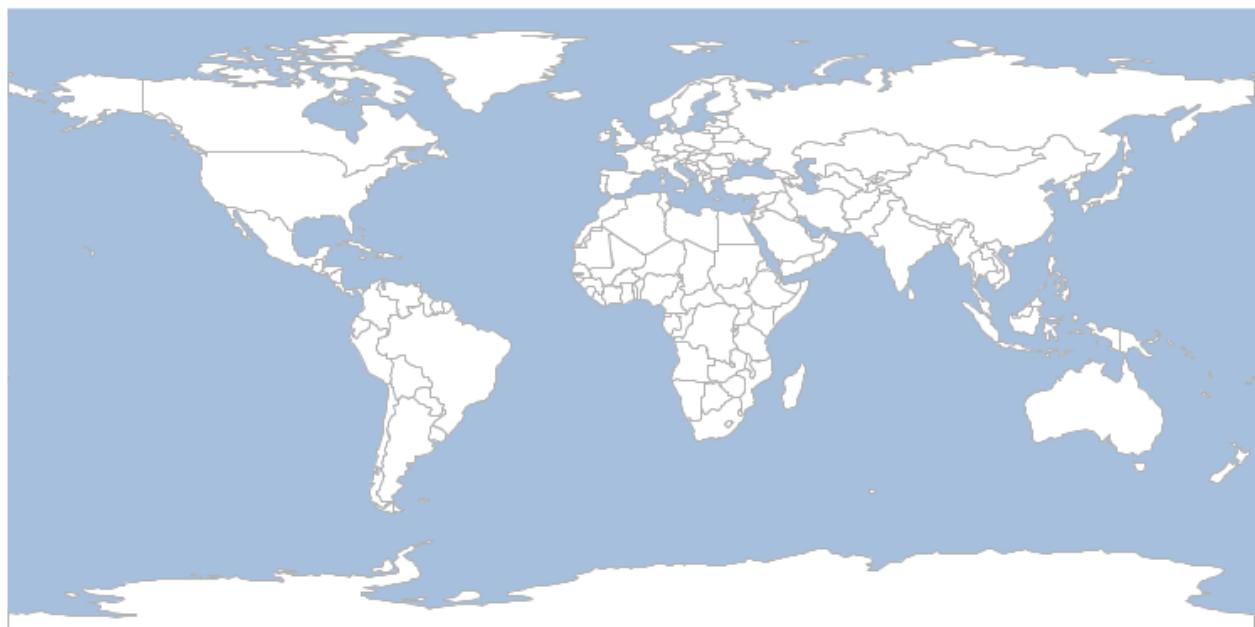
```
<carto>  
  <type>png</type>  
  <width>792</width>  
  <height>612</height>  
  <items>  
    <item>  
      <x>0</x>  
      <y>0</y>  
      <width>792</width>  
      <height>612</height>  
      <type>rectangle</type>  
      <fillColor>white</fillColor>  
      <strokeColor>black</strokeColor>  
    </item>  
    <item>
```

```

<x>30</x>
<y>50</y>
<width>742</width>
<height>20</height>
<type>text</type>
<text>Map Title</text>
<color>black</color>
<horizontalAlign>center</horizontalAlign>
<verticalAlign>middle</verticalAlign>
<font>
    <name>Arial</name>
    <style>bold</style>
    <size>36</size>
</font>
</item>
<item>
    <x>30</x>
    <y>120</y>
    <width>742</width>
    <height>470</height>
    <type>map</type>
    <name>mainMap</name>
    <layers>
        <layer>
            <layertype>layer</layertype>
            <file>src/test/resources/data.gpkg</file>
            <layername>ocean</layername>
            <style>src/test/resources/ocean.sld</style>
        </layer>
        <layer>
            <layertype>layer</layertype>
            <file>src/test/resources/data.gpkg</file>
            <layername>countries</layername>
            <style>src/test/resources/countries.sld</style>
        </layer>
    </layers>
</item>
</items>
</carto>

```

# Map Title



## Elements

The geocarto map command takes either a JSON or XML document made up of one or more elements.

### Map

Draw a map.

### JSON

```
{  
    "x": 10,  
    "y": 10,  
    "width": 380,  
    "height": 280,  
    "type": "map",  
    "name": "mainMap",  
    "imageType": "png",  
    "backgroundColor": "white",  
    "fixAspectRatio": true,  
    "proj": "EPSG:4326",  
    "bounds": {  
        "minX": -180,  
        "minY": -90,  
        "maxX": 180,  
        "maxY": 90  
    },  
    "layers": [  
        {"layertype": "layer", "file": "src/test/resources/data.gpkg", "layername": "ocean", "style": "src/test/resources/ocean.sld"},  
        {"layertype": "layer", "file": "src/test/resources/data.gpkg", "layername": "countries", "style": "src/test/resources/countries.sld"}  
    ]  
}
```



XML

```
<item>
    <x>10</x>
    <y>10</y>
    <width>380</width>
    <height>280</height>
    <type>map</type>
    <name>mainMap</name>
    <imageType>png</imageType>
    <backgroundColor>white</backgroundColor>
    <fixAspectRatio>true</fixAspectRatio>
    <proj>EPSG:4326</proj>
    <bounds>
        <minX>-180</minX>
        <minY>-90</minY>
        <maxX>180</maxX>
        <maxY>90</maxY>
        <proj>EPSG:4326</proj>
    </bounds>
    <layers>
        <layer>
            <layertype>layer</layertype>
            <file>src/test/resources/data.gpkg</file>
            <layername>ocean</layername>
            <style>src/test/resources/ocean.sld</style>
        </layer>
        <layer>
            <layertype>layer</layertype>
            <file>src/test/resources/data.gpkg</file>
            <layername>countries</layername>
            <style>src/test/resources/countries.sld</style>
        </layer>
    </layers>
</item>
```



## Overview Map

Draw a overview map.

JSON

```
{
  "x": 10,
  "y": 10,
  "width": 580,
  "height": 240,
  "type": "map",
  "name": "mainMap",
  "fixAspectRatio": false,
  "bounds": {
    "minX": -108.917446,
    "minY": 43.519820,
    "maxX": -89.229946,
    "maxY": 50.137433
  },
  "layers": [
    {"layertype": "layer", "file": "src/test/resources/data.gpkg", "layername": "ocean", "style": "src/test/resources/ocean.sld"},
    {"layertype": "layer", "file": "src/test/resources/data.gpkg", "layername": "countries", "style": "src/test/resources/countries.sld"},
    {"layertype": "layer", "file": "src/test/resources/data.gpkg", "layername": "states", "style": "src/test/resources/states.sld"}
  ]
},
{
  "x": 10,
  "y": 260,
  "width": 580,
  "height": 240,
  "type": "overViewMap",
  "zoomIntoBounds": false,
  "scaleFactor": 2.0,
  "linkedMap": "mainMap",
  "layers": [
    {"layertype": "layer", "file": "src/test/resources/data.gpkg", "layername": "ocean", "style": "src/test/resources/ocean.sld"},
    {"layertype": "layer", "file": "src/test/resources/data.gpkg", "layername": "countries", "style": "src/test/resources/countries.sld"}
  ]
}
```



## XML

```
<item>
  <x>10</x>
  <y>10</y>
  <width>580</width>
  <height>240</height>
  <type>map</type>
  <name>mainMap</name>
  <imageType>png</imageType>
  <backgroundColor>white</backgroundColor>
  <fixAspectRatio>true</fixAspectRatio>
  <proj>EPSG:4326</proj>
  <bounds>
    <minX>-108.917446</minX>
    <minY>43.519820</minY>
    <maxX>-89.229946</maxX>
    <maxY>50.137433</maxY>
    <proj>EPSG:4326</proj>
  </bounds>
  <layers>
    <layer>
      <layertype>layer</layertype>
      <file>src/test/resources/data.gpkg</file>
    </layer>
  </layers>
</item>
```

```

<layername>ocean</layername>
<style>src/test/resources/ocean.sld</style>
</layer>
<layer>
    <layertype>layer</layertype>
    <file>src/test/resources/data.gpkg</file>
    <layername>countries</layername>
    <style>src/test/resources/countries.sld</style>
</layer>
<layer>
    <layertype>layer</layertype>
    <file>src/test/resources/data.gpkg</file>
    <layername>states</layername>
    <style>src/test/resources/states.sld</style>
</layer>
</layers>
</item>
<item>
    <x>10</x>
    <y>260</y>
    <width>580</width>
    <height>240</height>
    <type>overviewMap</type>
    <zoomToBounds>false</zoomToBounds>
    <scaleFactor>2.0</scaleFactor>
    <linkedMap>mainMap</linkedMap>
    <layers>
        <layer>
            <layertype>layer</layertype>
            <file>src/test/resources/data.gpkg</file>
            <layername>ocean</layername>
            <style>src/test/resources/ocean.sld</style>
        </layer>
        <layer>
            <layertype>layer</layertype>
            <file>src/test/resources/data.gpkg</file>
            <layername>countries</layername>
            <style>src/test/resources/countries.sld</style>
        </layer>
    </layers>
</item>

```



## Text

Draw text.

## JSON

```
{  
    "x": 10,  
    "y": 10,  
    "width": 140,  
    "height": 30,  
    "type": "text",  
    "text": "Map Text",  
    "horizontalAlign": "center",  
    "verticalAlign": "middle",  
    "color": "black",  
    "font": {  
        "name": "Arial",  
        "style": "plain",  
        "size": 14  
    }  
}
```

Map Text

XML

```
<item>
    <x>10</x>
    <y>10</y>
    <width>140</width>
    <height>30</height>
    <type>text</type>
    <text>Map Text</text>
    <horizontalAlign>center</horizontalAlign>
    <verticalAlign>middle</verticalAlign>
    <color>black</color>
    <font>
        <name>Arial</name>
        <style>plain</style>
        <size>14</size>
    </font>
</item>
```

Map Text

## Rectangle

Draw a rectangle.

JSON

```
{
    "x": 10,
    "y": 10,
    "width": 30,
    "height": 30,
    "type": "rectangle",
    "fillColor": "white",
    "strokeColor": "black"
}
```



XML

```
<item>
  <x>10</x>
  <y>10</y>
  <width>30</width>
  <height>30</height>
  <type>rectangle</type>
  <fillColor>white</fillColor>
  <strokeColor>black</strokeColor>
</item>
```



## North Arrow

Draw a north arrow.

JSON

```
{
  "x": 10,
  "y": 10,
  "width": 130,
  "height": 130,
  "type": "northarrow",
  "style": "North",
  "fillColor1": "black",
  "fillColor2": "white",
  "strokeColor1": "black",
  "strokeColor2": "black",
  "strokeWidth": 1,
  "drawText": true,
  "textColor": "black",
  "font": {
    "name": "Arial",
    "style": "plain",
    "size": 24
  }
}
```



## XML

```
<item>
  <x>10</x>
  <y>10</y>
  <width>130</width>
  <height>130</height>
  <type>northarrow</type>
  <style>NorthEastSouthWest</style>
  <fillColor1>black</fillColor1>
  <fillColor2>white</fillColor2>
  <strokeColor1>black</strokeColor1>
  <strokeColor2>black</strokeColor2>
  <strokeWidth>1</strokeWidth>
  <drawText>true</drawText>
  <textColor>black</textColor>
  <font>
    <name>Arial</name>
    <style>plain</style>
    <size>24</size>
  </font>
</item>
```



## Legend

Draw a legend.

JSON

```
{
  "x": 10,
  "y": 10,
  "width": 380,
  "height": 190,
  "type": "map",
  "name": "mainMap",
  "layers": [
    {"layertype": "layer", "file": "src/test/resources/data.gpkg", "layername": "ocean", "style": "src/test/resources/ocean.sld"},
    {"layertype": "layer", "file": "src/test/resources/data.gpkg", "layername": "countries", "style": "src/test/resources/countries.sld"}
  ]
}, {
  "x": 10,
  "y": 210,
  "width": 380,
  "height": 70,
  "type": "legend",
  "map": "mainMap",
  "backgroundColor": "white",
  "title": "Legend",
  "titleFont": {
    "name": "Arial",
    "style": "bold",
    "size": 18
  },
  "titleColor": "black",
  "textColor": "black",
  "textFont": {
    "name": "Arial",
    "style": "plain",
    "size": 12
  },
  "numberFormat": "#.##",
  "legendEntryWidth": "50",
  "legendEntryHeight": "30",
  "gapBetweenEntries": "10"
}
```



## Legend

Ocean   Countries

XML

```
<item>
    <x>10</x>
    <y>10</y>
    <width>380</width>
    <height>190</height>
    <type>map</type>
    <name>mainMap</name>
    <layers>
        <layer>
            <layertype>layer</layertype>
            <file>src/test/resources/data.gpkg</file>
            <layername>ocean</layername>
            <style>src/test/resources/ocean.sld</style>
        </layer>
        <layer>
            <layertype>layer</layertype>
            <file>src/test/resources/data.gpkg</file>
            <layername>countries</layername>
            <style>src/test/resources/countries.sld</style>
        </layer>
        <layer>
            <layertype>layer</layertype>
            <file>src/test/resources/data.gpkg</file>
            <layername>states</layername>
            <style>src/test/resources/states.sld</style>
        </layer>
    </layers>
</item>
<item>
    <x>10</x>
    <y>210</y>
    <width>380</width>
```

```
<height>70</height>
<type>legend</type>
<map>mainMap</map>
<backgroundColor>white</backgroundColor>
<title>Legend</title>
<titleFont>
    <name>Arial</name>
    <style>bold</style>
    <size>14</size>
</titleFont>
<titleColor>black</titleColor>
<textColor>black</textColor>
<textFont>
    <name>Arial</name>
    <style>plain</style>
    <size>12</size>
</textFont>
<numberFormat># .##</numberFormat>
<legendEntryWidth>50</legendEntryWidth>
<legendEntryHeight>30</legendEntryHeight>
<gapBetweenEntries>10</gapBetweenEntries>
</item>
```



### Legend

|  |       |  |           |  |        |
|--|-------|--|-----------|--|--------|
| <span style="background-color: #4682B4; width: 15px; height: 15px; display: inline-block;"></span> | Ocean | <span style="width: 15px; height: 15px; border: 1px solid black; display: inline-block;"></span> | Countries | <span style="width: 15px; height: 15px; border: 1px solid black; display: inline-block;"></span> | States |
|--|-------|--|-----------|--|--------|

### Date

Draw a date.

JSON

```
{  
    "x": 10,  
    "y": 10,  
    "width": 140,  
    "height": 30,  
    "type": "datetext",  
    "date": "1/22/2022",  
    "format": "MM/dd/yyyy",  
    "horizontalAlign": "center",  
    "verticalAlign": "middle",  
    "color": "black",  
    "font": {  
        "name": "Arial",  
        "style": "plain",  
        "size": 14  
    }  
}
```

01/22/2022

XML

```
<item>  
    <x>10</x>  
    <y>10</y>  
    <width>140</width>  
    <height>30</height>  
    <type>dateText</type>  
    <date>1/22/2022</date>  
    <format>MM/dd/yyyy</format>  
    <horizontalAlign>center</horizontalAlign>  
    <verticalAlign>middle</verticalAlign>  
    <color>black</color>  
    <font>  
        <name>Arial</name>  
        <style>plain</style>  
        <size>14</size>  
    </font>  
</item>
```

01/22/2022

## Scale Text

Draw scale text.

## JSON

```
{  
    "x": 10,  
    "y": 10,  
    "width": 380,  
    "height": 280,  
    "type": "map",  
    "name": "mainMap",  
    "layers": [  
        {"layertype": "layer", "file": "src/test/resources/data.gpkg", "layername": "ocean", "style": "src/test/resources/ocean.sld"},  
        {"layertype": "layer", "file": "src/test/resources/data.gpkg", "layername": "countries", "style": "src/test/resources/countries.sld"}  
    ]  
}, {  
    "x": 10,  
    "y": 250,  
    "width": 380,  
    "height": 40,  
    "type": "scaletext",  
    "map": "mainMap",  
    "format": "#",  
    "prefixText": "Scale: ",  
    "horizontalAlign": "center",  
    "verticalAlign": "middle",  
    "color": "black",  
    "font": {  
        "name": "Arial",  
        "style": "plain",  
        "size": 14  
    }  
}
```



Scale: 1:376644894

## XML

```
<item>
    <x>10</x>
    <y>10</y>
    <width>580</width>
    <height>240</height>
    <type>map</type>
    <name>mainMap</name>
    <layers>
        <layer>
            <layertype>layer</layertype>
            <file>src/test/resources/data.gpkg</file>
            <layername>ocean</layername>
            <style>src/test/resources/ocean.sld</style>
        </layer>
        <layer>
            <layertype>layer</layertype>
            <file>src/test/resources/data.gpkg</file>
            <layername>countries</layername>
            <style>src/test/resources/countries.sld</style>
        </layer>
        <layer>
            <layertype>layer</layertype>
            <file>src/test/resources/data.gpkg</file>
            <layername>states</layername>
            <style>src/test/resources/states.sld</style>
        </layer>
    </layers>
</item>
<item>
    <x>10</x>
    <y>250</y>
    <width>380</width>
    <height>40</height>
    <type>scaletext</type>
    <map>mainMap</map>
    <format>#</format>
    <prefixText>Scale :</prefixText>
    <horizontalAlign>center</horizontalAlign>
    <verticalAlign>middle</verticalAlign>
    <color>black</color>
    <font>
        <name>Arial</name>
        <style>plain</style>
        <size>14</size>
    </font>
</item>
```



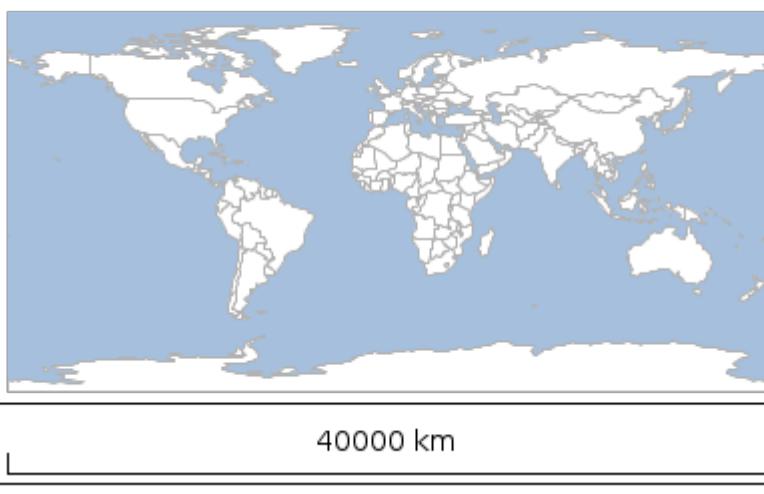
Scale :1:298177207

### Scale Bar

Draw scale bar.

JSON

```
{
  "x": 10,
  "y": 10,
  "width": 380,
  "height": 280,
  "type": "map",
  "name": "mainMap",
  "layers": [
    {"layertype": "layer", "file": "src/test/resources/data.gpkg", "layername": "ocean", "style": "src/test/resources/ocean.sld"},
    {"layertype": "layer", "file": "src/test/resources/data.gpkg", "layername": "countries", "style": "src/test/resources/countries.sld"}
  ]
}, {
  "x": 10,
  "y": 250,
  "width": 380,
  "height": 40,
  "type": "scalebar",
  "map": "mainMap",
  "strokeColor": "black",
  "strokeWidth": 1,
  "border": 5,
  "units": "METRIC",
  "fillColor": "white",
  "font": {
    "name": "Arial",
    "style": "plain",
    "size": 14
  }
}
```



XML

```

<item>
    <x>10</x>
    <y>10</y>
    <width>580</width>
    <height>240</height>
    <type>map</type>
    <name>mainMap</name>
    <layers>
        <layer>
            <layertype>layer</layertype>
            <file>src/test/resources/data.gpkg</file>
            <layername>ocean</layername>
            <style>src/test/resources/ocean.sld</style>
        </layer>
        <layer>
            <layertype>layer</layertype>
            <file>src/test/resources/data.gpkg</file>
            <layername>countries</layername>
            <style>src/test/resources/countries.sld</style>
        </layer>
        <layer>
            <layertype>layer</layertype>
            <file>src/test/resources/data.gpkg</file>
            <layername>states</layername>
            <style>src/test/resources/states.sld</style>
        </layer>
    </layers>
</item>
<item>
    <x>10</x>
    <y>250</y>
    <width>380</width>
    <height>40</height>
    <type>scalebar</type>
    <map>mainMap</map>
    <strokeColor>black</strokeColor>
    <strokeWidth>1</strokeWidth>
    <border>5</border>
    <units>US</units>
    <fillColor>white</fillColor>
    <font>
        <name>Arial</name>
        <style>plain</style>
        <size>14</size>
    </font>
</item>

```



20000 miles

## Line

Draw a line.

### JSON

```
{  
  "x": 10,  
  "y": 10,  
  "width": 180,  
  "height": 0,  
  "type": "line",  
  "strokeColor": "black",  
  "strokeWidth": 2  
}
```

### XML

```
<item>  
  <x>10</x>  
  <y>10</y>  
  <width>180</width>  
  <height>0</height>  
  <type>line</type>  
  <strokeColor>black</strokeColor>  
  <strokeWidth>2</strokeWidth>  
</item>
```

---

## Grid

Draw a grid to make it easier to place other items.

JSON

```
{  
  "x": 0,  
  "y": 0,  
  "width": 100,  
  "height": 100,  
  "type": "grid",  
  "size": 10,  
  "strokeColor": "black",  
  "strokeWidth": 0.5  
}
```



XML

```
<item>  
  <x>0</x>  
  <y>0</y>  
  <width>100</width>  
  <height>100</height>  
  <type>grid</type>  
  <size>10</size>  
  <strokeColor>black</strokeColor>  
  <strokeWidth>0.5</strokeWidth>  
</item>
```



## Paragraph

Draw paragraph.

## JSON

```
{  
  "x": 10,  
  "y": 10,  
  "width": 240,  
  "height": 140,  
  "type": "paragraph",  
  "text": "The Carto package contains classes for creating cartographic documents. All  
  items are added to the document with x and y coordinates whose origin is the upper  
  left and width and a height.",  
  "color": "black",  
  "font": {  
    "name": "Arial",  
    "style": "plain",  
    "size": 14  
  }  
}
```

The Carto package contains  
classes for creating cartographic  
documents. All items are added to  
the document with x and y  
coordinates whose origin is the  
upper left and width and a height.

## XML

```
<item>  
  <x>10</x>  
  <y>10</y>  
  <width>240</width>  
  <height>140</height>  
  <type>paragraph</type>  
  <text>The Carto package contains classes for creating cartographic documents. All  
  items are added to the document with x and y coordinates whose origin is the upper  
  left and width and a height.t</text>  
  <color>black</color>  
  <font>  
    <name>Arial</name>  
    <style>plain</style>  
    <size>14</size>  
  </font>  
</item>
```

The Carto package contains classes for creating cartographic documents. All items are added to the document with x and y coordinates whose origin is the upper left and width and a height.t

## Image

Draw an image.

JSON

```
{  
  "x": 10,  
  "y": 10,  
  "width": 512,  
  "height": 431,  
  "type": "image",  
  "path": "src/main/docs/static/images/geoc.png"  
}
```



XML

```
<item>
  <x>10</x>
  <y>10</y>
  <width>512</width>
  <height>431</height>
  <type>image</type>
  <path>src/main/docs/static/images/geoc.png</path>
</item>
```



## Table

Draw a table of data.

JSON

```
{  
  "x": 10,  
  "y": 10,  
  "width": 280,  
  "height": 80,  
  "type": "table",  
  "columns": ["ID", "Name", "Abbreviation"],  
  "rows": [  
    {"ID": 1, "Name": "Washington", "Abbreviation": "WA"},  
    {"ID": 2, "Name": "Oregon", "Abbreviation": "OR"},  
    {"ID": 3, "Name": "California", "Abbreviation": "CA"}  
  ]  
}
```

| ID | Name       | Abbreviation |
|----|------------|--------------|
| 1  | Washington | WA           |
| 2  | Oregon     | OR           |
| 3  | California | CA           |

XML

```

<item>
  <x>10</x>
  <y>10</y>
  <width>280</width>
  <height>80</height>
  <type>table</type>
  <columns>
    <column>ID</column>
    <column>Name</column>
    <column>Abbreviation</column>
  </columns>
  <rows>
    <row>
      <ID>1</ID>
      <Name>Washington</Name>
      <Abbreviation>WA</Abbreviation>
    </row>
    <row>
      <ID>2</ID>
      <Name>Oregon</Name>
      <Abbreviation>OR</Abbreviation>
    </row>
    <row>
      <ID>3</ID>
      <Name>California</Name>
      <Abbreviation>CA</Abbreviation>
    </row>
  </rows>
</item>

```

| ID | Name       | Abbreviation |
|----|------------|--------------|
| 1  | Washington | WA           |
| 2  | Oregon     | OR           |
| 3  | California | CA           |

## Filter Commands

### CQL to XML

Convert a CQL statement to an OCG XML Filter.

| Short Name | Long Name  | Description            |
|------------|------------|------------------------|
| -c         | --cql      | The CQL statement      |
|            | --help     | Print the help message |
|            | --web-help | Open help in a browser |

```
geoc filter cql2xml -c name=wa
```

```
<ogc:Filter xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc">
  <ogc:PropertyIsEqualTo>
    <ogc:PropertyName>name</ogc:PropertyName>
    <ogc:PropertyName>wa</ogc:PropertyName>
  </ogc:PropertyIsEqualTo>
</ogc:Filter>
```

# Geometry Commands

## Convert

Convert a geometry from one format to another.

| Short Name | Long Name        | Description  |
|------------|------------------|--|
| -i         | --input          | The input geometry   |
| -f         | --format         | The output format (wkt, geojson, gml2, gml3, kml, georss, gpx, csv, wkb) |
| -p         | --format-options | The output format options  |
| -t         | --type           | The output type (geometry, feature, layer)                               |
|            | --help           | Print the help message   |
|            | --web-help       | Open help in a browser   |

```
geoc geometry convert -i "POINT(-122.386371 47.581154)" -f geojson -t feature
```

```
{"type": "Feature", "geometry": {"type": "Point", "coordinates": [-122.3864, 47.5812]}, "properties": {}, "id": "1"}
```

## Decimal Degrees to Point

Convert a decimal degrees formatted string into a Point.

| Short Name | Long Name        | Description                     |
|------------|------------------|---------------------------------|
| -d         | --decimaldegrees | The decimal degrees             |
| -t         | --type           | The output type (xy, wkt, json) |

| <b>Short Name</b> | <b>Long Name</b> | <b>Description</b>     |
|-------------------|------------------|------------------------|
|                   | --help           | Print the help message |
|                   | --web-help       | Open help in a browser |

```
geoc geometry dd2pt -d "122d 31m 32.23s W, 47d 12m 43.28s N" -t wkt
```

```
POINT (-122.5256194444444 47.212022222222224)
```

## GeoHash Bounds

Calculate the geohashes for the given bounds.

| <b>Short Name</b> | <b>Long Name</b>  | <b>Description</b>   |
|-------------------|-------------------|--|
| -b                | --bounds          | The input geometry   |
| -t                | --type            | The encoding type (string or long). The default is string. |
| -n                | --number-of-chars | The number of characters. The default is 9.                |
| -d                | --bit-depth       | The bit depth. The default is 52.                          |
|                   | --help            | Print the help message                                     |
|                   | --web-help        | Open help in a browser                                     |

```
geoc geometry geohash bounds -b "120, 30, 120.0001, 30.0001" -t long -d 45
```

```
28147497671064
28147497671068
28147497671112
28147497671066
28147497671070
28147497671114
28147497671088
28147497671092
28147497671136
```

## GeoHash Decode

Decode a GeoHash to a Geometry.

| Short Name | Long Name  | Description                              |
|------------|------------|--|
| -i         | --input    | The input geohash                        |
| -t         | --type     | Whether the geohash is a point or bounds |
|            | --help     | Print the help message                   |
|            | --web-help | Open help in a browser                   |

```
geoc geometry geohash decode -i uf8vk6wjr -t point
```

```
POINT (35.0001668930054 60.0000715255737)
```

## GeoHash Encode

Encode a Geometry as a GeoHash.

| Short Name | Long Name         | Description  |
|------------|-------------------|--|
| -i         | --input           | The input geometry   |
| -t         | --type            | The encoding type (string or long). The default is string. |
| -n         | --number-of-chars | The number of characters. The default is 9.                |
| -d         | --bit-depth       | The bit depth. The default is 52.                          |
|            | --help            | Print the help message                                     |
|            | --web-help        | Open help in a browser                                     |

```
geoc geometry geohash encode -i "POINT(-122.386371 47.581154)"
```

```
c22yxjbuq
```

## GeoHash Neighbors

Get a geohash's neighbors.

| Short Name | Long Name         | Description                                 |
|------------|-------------------|---|
| -i         | --input           | The input geometry                          |
| -n         | --number-of-chars | The number of characters. The default is 9. |

| <b>Short Name</b> | <b>Long Name</b> | <b>Description</b>                |
|-------------------|------------------|-----------------------------------|
| -d                | --bit-depth      | The bit depth. The default is 52. |
|                   | --help           | Print the help message            |
|                   | --web-help       | Open help in a browser            |

```
geoc geometry geohash neighbors -i uf8vk6wj
```

NORTH,uf8vk6wjx  
 NORTHEAST,uf8vk6wm8  
 EAST,uf8vk6wm2  
 SOUTHEAST,uf8vk6wm0  
 SOUTH,uf8vk6wjp  
 SOUTHWEST,uf8vk6wjn  
 WEST,uf8vk6wjq  
 NORTHWEST,uf8vk6wjw

## Great Circle Arc

Create a great circle arc.

| <b>Short Name</b> | <b>Long Name</b> | <b>Description</b>     |
|-------------------|------------------|------------------------|
| -e                | --ellipsoid      | The ellipsoid          |
| -p                | --start-point    | The start point        |
| -t                | --end-point      | The end point          |
| -n                | --num-points     | The number of points   |
|                   | --help           | Print the help message |
|                   | --web-help       | Open help in a browser |

```
geoc geometry greatcirclearc -p POINT (-122 48) -t POINT (-0.102938 51.498749) -e
wgs84 -n 20
```

```
LINESTRING (-119.07040273132067 50.67129040608734, -115.79405787410982
53.25898813815459, -112.10566632488812 55.74416257443563, -107.93031121546862
58.102903619395605, -103.18586832746001 60.30516464523606, -97.78964326539094
62.313702219535685, -91.67188919322786 64.08357246715578, -84.79846274611634
65.56300075396796, -77.20148714844558 66.69673003845362, -69.00888413693454
67.4327000137296, -60.454039139748815 67.73150516117609, -51.847144661724116
67.57568999780139, -43.51024818547282 66.97446827309976, -35.70614774738105
65.96118559566465, -28.596592724101157 64.58499988892942, -22.24128289210202
62.90104269692094, -16.623473379491926 60.96269894447343, -11.681762264482387
58.81725900451406, -7.335682843452773 56.50439016948547, -3.501944007479139
54.05631263013969)
```



## Offset

Create a Geometry offset from the input Geometry.

| Short Name | Long Name           | Description                                     |
|------------|---------------------|---|
| -i         | --input             | The input geometry                              |
| -d         | --offset            | The offset distance                             |
| -s         | --quadrant-segments | The number of quadrant segments (defaults to 8) |
|            | --help              | Print the help message                          |
|            | --web-help          | Open help in a browser                          |

```
geoc geometry offset -i LINESTRING (-120.41362631285119 47.87883318858252,
-3.9909723099333974 39.24424611524387) -d 5 -s 8
```

```
LINestring (-120.0438126769743 52.86513822084032, -3.621158674056503  
44.23055114750167)
```



## orthodromicDistance

Calculate the orthodromic distance between two points..

| Short Name | Long Name     | Description            |
|------------|---------------|------------------------|
| -e         | --ellipsoid   | The ellipsoid          |
| -p         | --start-point | The start point        |
| -t         | --end-point   | The end point          |
|            | --help        | Print the help message |
|            | --web-help    | Open help in a browser |

```
geoc geometry orthodromicdistance -p POINT (-122 48) -t POINT (-0.102938 51.498749) -e  
wgs84
```

```
7674355.352400642
```

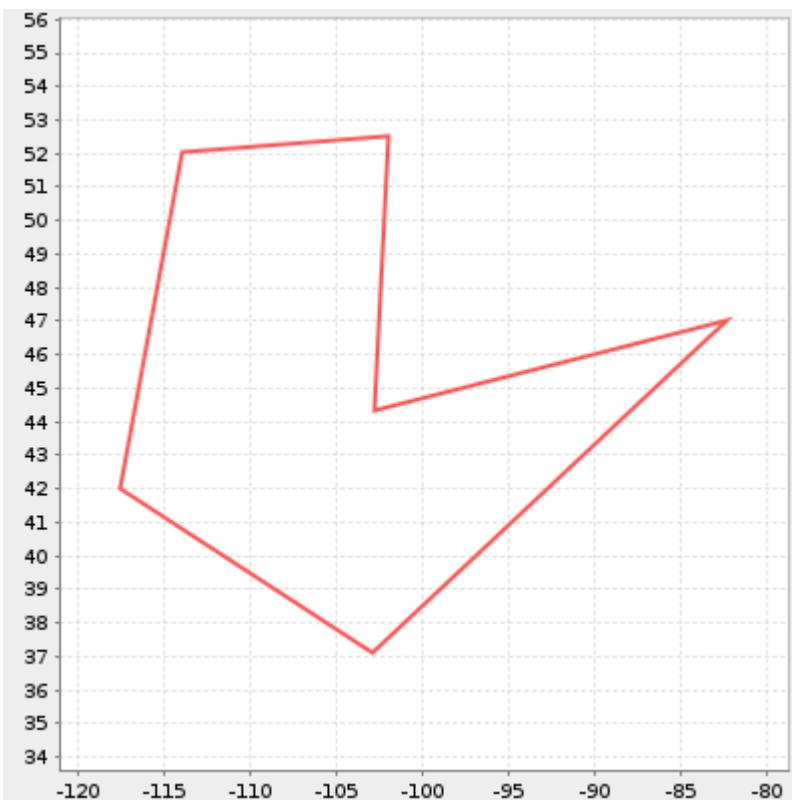
## Plot

Draw a geometry to a plot.

| Short Name | Long Name | Description        |
|------------|-----------|--------------------|
| -i         | --input   | The input geometry |
| -f         | --file    | The output file    |

| <b>Short Name</b> | <b>Long Name</b> | <b>Description</b>          |
|-------------------|------------------|-----------------------------|
| -w                | --width          | The image width             |
| -h                | --height         | The image height            |
| -l                | --legend         | Whether to show the legend  |
| -r                | --fill-coords    | Whether to fill coordinates |
| -p                | --fill-polys     | Whether to fill polygons    |
| -d                | --draw-coords    | Whether to draw coordinates |
|                   | --help           | Print the help message      |
|                   | --web-help       | Open help in a browser      |

```
geoc geometry plot -i "POLYGON ((-113.98365269610397 52.04260423559353,
-117.55190821991903 41.99216856357597, -102.82940482544078 37.1267755781612,
-82.26457660787091 47.05513909003821, -102.75935045963138 44.33220905070587,
-101.89775634863287 52.5472919646931, -113.98365269610397 52.04260423559353))" -f
target/geometry_plot.png
```



## Point to Decimal Degrees

Format a Point in Decimal Degrees.

| <b>Short Name</b> | <b>Long Name</b> | <b>Description</b> |
|-------------------|------------------|--------------------|
| -p                | --point          | The Point          |

| Short Name | Long Name  | Description                                    |
|------------|------------|--|
| -t         | --type     | The output type (dms, dms_char, ddm, ddm_char) |
|            | --help     | Print the help message                         |
|            | --web-help | Open help in a browser                         |

```
geoc geometry pt2dd -p "POINT (-122 48)" -t dms
```

```
-122° 0' 0.0000" W, 48° 0' 0.0000" N
```

# Map Commands

## Map Layers

Map layer is a simple string format that allows you to pass in information about a map layer.

It can contain the following properties:

- **layertype** = The type of layer (layer, raster, tile)
  - For layer layertype, you can use the same key value pairs used to specify a Workspace.
  - For raster layertype, you specify a source=file key value pair.
  - For tile layertype, you use the same key value pairs used to specify a Tile layer.
- **layername** = The name of the Layer/Raster/Tile
- **layerprojection** = The Projection
- **style** = The SLD, CSS, or other style

Examples:

- **Vector Layer**
  - **layertype=layer**      **dbtype=geopkg**      **database=/Users/user/Desktop/countries.gpkg**  
**layername=countries** **style=/Users/user/Desktop/countries.sld**
  - **layertype=layer**      **file=/Users/user/Desktop/geoc/polylines.csv**      **layername=polylines**  
**style=/Users/user/Desktop/geoc/polylines.sld**
  - **layertype=layer**                                    **file=/Users/user/Desktop/geoc/points.properties**  
**style=/Users/user/Desktop/geoc/points.sld**
  - **layertype=layer** **file=/Users/user/Projects/geoc/src/test/resources/polylines.shp**
  - **layertype=layer**      **directory=/Users/user/Projects/geoc/src/test/resources/points.properties**  
**layername=points**
- **Raster**

- **layertype=raster** **source=rasters/earth.tif**
- **Tile**
  - **layertype=tile** **file=world.mbtiles**
  - **layertype=tile** **type=geopackage** **file=states.gpkg**

## Draw

Draw a map.

| Short Name | Long Name          | Description            |
|------------|--------------------|------------------------|
| -l         | --layer            | The map layer          |
| -f         | --file             | The output image file  |
| -t         | --type             | The type of document   |
| -w         | --width            | The width              |
| -h         | --height           | The height             |
| -b         | --bounds           | The bounds             |
| -g         | --background-color | The background color   |
| -p         | --projection       | The projection         |
|            | --help             | Print the help message |
|            | --web-help         | Open help in a browser |

```
geoc map draw -l "layertype=layer file=src/test/resources/data.gpkg layername=ocean
style=src/test/resources/ocean.sld" -l "layertype=layer
file=src/test/resources/data.gpkg layername=countries
style=src/test/resources/countries.sld" -f target/map.png
```



## Map Cube

Draw a map cube.

| Short Name | Long Name      | Description                                 |
|------------|----------------|---|
| -l         | --layer        | The map layer                               |
| -f         | --file         | The output image file                       |
| -o         | --draw-outline | The flag to whether to draw outlines or not |
| -t         | --draw-tabs    | The flag to whether to draw tabs or not     |
| -s         | --tab-size     | The tab size                                |
| -i         | --title        | The title                                   |
| -c         | --source       | The data source or credits                  |
|            | --help         | Print the help message                      |
|            | --web-help     | Open help in a browser                      |

```
geoc map cube -l "layertype=layer file=src/test/resources/data.gpkg layername=ocean  
style=src/test/resources/ocean.sld" -l "layertype=layer  
file=src/test/resources/data.gpkg layername=countries  
style=src/test/resources/countries.sld" -o -f target/cube.png
```



Draw a blank map cube.

```
geoc map cube -o -f target/cube_blank.png
```



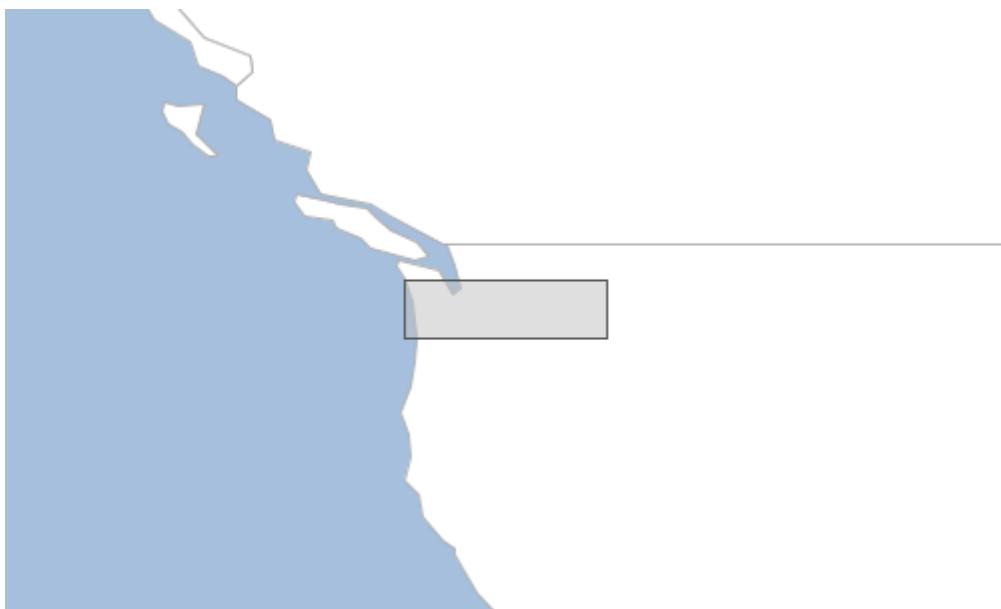
# Projection Commands

## Envelope

Get a Projection's envelope.

| Short Name | Long Name          | Description                                   |
|------------|--------------------|---|
| -e         | --epsg             | The EPSG Projection code                      |
| -g         | --geo-bounds       | The flag for whether to use geo bounds or not |
| -o         | --output-workspace | The output workspace                          |
| -r         | --output-layer     | The output layer                              |
|            | --help             | Print the help message                        |
|            | --web-help         | Open help in a browser                        |

```
geoc proj envelope -e EPSG:2927 -g -o target/envelope.shp
```



## WKT

Get the WKT of a Projection

| Short Name | Long Name     | Description                    |
|------------|---------------|--------------------------------|
| -e         | --epsg        | The EPSG Projection code       |
| -f         | --file        | The output File                |
| -c         | --citation    | The citations (epsg or esri)   |
| -i         | --indentation | The number of spaces to indent |
|            | --help        | Print the help message         |
|            | --web-help    | Open help in a browser         |

```
geoc proj wkt -e EPSG:2927
```

```

PROJCS["NAD83(HARN) / Washington South (ftUS)",
GEOGCS["NAD83(HARN)",
    DATUM["NAD83 (High Accuracy Reference Network)",
        SPHEROID["GRS 1980", 6378137.0, 298.25722101, AUTHORITY["EPSG","7019"]],
        TOWGS84[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
        AUTHORITY["EPSG","6152"]],
    PRIMEM["Greenwich", 0.0, AUTHORITY["EPSG","8901"]],
    UNIT["degree", 0.017453292519943295],
    AXIS["Geodetic longitude", EAST],
    AXIS["Geodetic latitude", NORTH],
    AUTHORITY["EPSG","4152"]],
PROJECTION["Lambert Conic Conformal (2SP)", AUTHORITY["EPSG","9802"]],
PARAMETER["Longitude of natural origin", -120.5],
PARAMETER["Latitude of false origin", 45.33333333333336],
PARAMETER["Latitude of 1st standard parallel", 47.33333333333336],
PARAMETER["False easting", 1640416.667],
PARAMETER["False northing", 0.0],
PARAMETER["Scale factor at natural origin", 1.0],
PARAMETER["Latitude of 2nd standard parallel", 45.83333333333336],
UNIT["ft_survey_us", 0.3048006096012192],
AXIS["Easting", EAST],
AXIS["Northing", NORTH],
AUTHORITY["EPSG","2927"]]

```

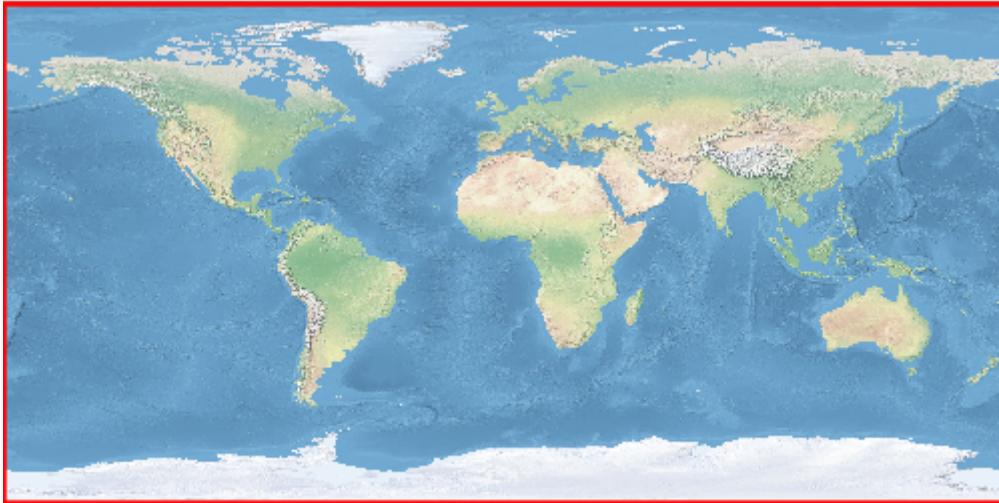
# Raster Commands

## Envelope

Get the Envelope of a Raster as a Vector Layer.

| Short Name | Long Name           | Description            |
|------------|---------------------|------------------------|
| -o         | --output-workspace  | The output workspace   |
| -r         | --output-layer      | The output layer       |
| -i         | --input-raster      | The input raster       |
| -l         | --input-raster-name | The input raster name  |
| -p         | --input-projection  | The input projection   |
|            | --help              | Print the help message |
|            | --web-help          | Open help in a browser |

```
geoc raster envelope -i src/test/resources/earth.tif -o target/earth_envelope.shp
```



## Info

Get information about a Raster.

| Short Name | Long Name           | Description            |
|------------|---------------------|------------------------|
| -i         | --input-raster      | The input raster       |
| -l         | --input-raster-name | The input raster name  |
| -p         | --input-projection  | The input projection   |
|            | --help              | Print the help message |
|            | --web-help          | Open help in a browser |

```
geoc raster info -i src/test/resources/earth.tif
```

Format: GeoTIFF  
Size: 800, 400  
Projection ID: EPSG:4326  
Projection WKT: GEOGCS["WGS 84",  
    DATUM["World Geodetic System 1984",  
        SPHEROID["WGS 84", 6378137.0, 298.257223563, AUTHORITY["EPSG", "7030"]],  
        AUTHORITY["EPSG", "6326"]],  
    PRIMEM["Greenwich", 0.0, AUTHORITY["EPSG", "8901"]],  
    UNIT["degree", 0.017453292519943295],  
    AXIS["Geodetic longitude", EAST],  
    AXIS["Geodetic latitude", NORTH],  
    AUTHORITY["EPSG", "4326"]]  
Extent: -179.99999999999997, -89.99999999998205, 179.99999999996405, 90.0  
Pixel Size: 0.4499999999999505, 0.4499999999999551  
Block Size: 800, 8  
Bands:  
    RED\_BAND  
        Min Value: 56.0 Max Value: 255.0  
    GREEN\_BAND  
        Min Value: 84.0 Max Value: 255.0  
    BLUE\_BAND  
        Min Value: 91.0 Max Value: 255.0

## Get Size

Get the Raster size (width,height).

| Short Name | Long Name           | Description            |
|------------|---------------------|------------------------|
| -i         | --input-raster      | The input raster       |
| -l         | --input-raster-name | The input raster name  |
| -p         | --input-projection  | The input projection   |
|            | --help              | Print the help message |
|            | --web-help          | Open help in a browser |

```
geoc raster size -i src/test/resources/earth.tif
```

```
800,400
```

## Style Commands

# Create

Create a simple style.

| Short Name | Long Name        | Description                   |
|------------|------------------|-------------------------------|
| -s         | --style-options  | A style options               |
| -t         | --type           | The output type (sld or ysld) |
| -w         | --writer-options | The StyleWriter options       |
| -o         | --output         | The output file               |
|            | --help           | Print the help message        |
|            | --web-help       | Open help in a browser        |

## Style Options:

- Fill properties
  - **fill** (color)
  - **fill-opacity** (0-1)
- Stroke properties
  - **stroke** (color)
  - **stroke-width** (double)
  - **stroke-opacity** (0-1)
- Shape properties
  - **shape** (color)
  - **shape-size** (double)
  - **shape-type** (circle, square, triangle, star, cross, or x)
- Label properties
  - **label** The field name (ID, NAME)
  - **label-size** (12)
  - **label-style** (normal, italic, oblique)
  - **label-weight** (normal, bold)
  - **label-family** (serif, Arial, Verdana)

```
geoc style create -s stroke=navy -s stroke-width=0.5 -t sld -o target/boundaries.sld
```

```

<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor xmlns=
"http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld" xmlns:gml=
"http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc" version="1.0.0">
  <sld:UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>Default Styler</sld:Name>
      <sld:FeatureTypeStyle>
        <sld:Name>name</sld:Name>
        <sld:Rule>
          <sld:LineSymbolizer>
            <sld:Stroke>
              <sld:CssParameter name="stroke">#000080</sld:CssParameter>
              <sld:CssParameter name="stroke-width">0.5</sld:CssParameter>
            </sld:Stroke>
          </sld:LineSymbolizer>
        </sld:Rule>
      </sld:FeatureTypeStyle>
    </sld:UserStyle>
  </sld:UserLayer>
</sld:StyledLayerDescriptor>

```

```

geoc style create -s fill=white -s stroke=black -s stroke-width=1.5 -t ysld -o
target/boundaries.ysld

```

```

name: Default Styler
feature-styles:
- name: name
  rules:
  - scale: [min, max]
    symbolizers:
    - polygon:
        fill-color: '#FFFFFF'
        fill-opacity: 0.6
    - line:
        stroke-color: '#000000'
        stroke-width: 1.5

```

## CSS to SLD

Convert CSS to SLD.

| <b>Short Name</b> | <b>Long Name</b> | <b>Description</b>      |
|-------------------|------------------|-------------------------|
| -i                | --input          | The input file or url   |
| -o                | --output         | The output file         |
| -w                | --writer-options | The StyleWriter options |
|                   | --help           | Print the help message  |
|                   | --web-help       | Open help in a browser  |

```
geoc style css2sld -i target/points.css -o target/points.sld
```

*points.css*

```
* {
  mark: symbol(circle);
  mark-size: 6px;
}

:mark {
  fill: red;
}
```

*points.sld*

```

<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor xmlns=
"http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld" xmlns:gml=
"http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc" version="1.0.0">
  <sld:UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>Default Styler</sld:Name>
      <sld:FeatureTypeStyle>
        <sld:Rule>
          <sld:PointSymbolizer>
            <sld:Graphic>
              <sld:Mark>
                <sld:WellKnownName>circle</sld:WellKnownName>
                <sld:Fill>
                  <sld:CssParameter name="fill">#ff0000</sld:CssParameter>
                </sld:Fill>
              </sld:Mark>
              <sld:Size>6</sld:Size>
            </sld:Graphic>
          </sld:PointSymbolizer>
        </sld:Rule>
        <sld:VendorOption name="ruleEvaluation">first</sld:VendorOption>
      </sld:FeatureTypeStyle>
    </sld:UserStyle>
  </sld:UserLayer>
</sld:StyledLayerDescriptor>

```

## SLD to Ysld

Convert SLD to Ysld

| Short Name | Long Name  | Description            |
|------------|------------|------------------------|
| -i         | --input    | The input file or url  |
| -o         | --output   | The output file        |
|            | --help     | Print the help message |
|            | --web-help | Open help in a browser |

```
geoc style sld2ysld -i target/points.sld -o target/points.ysld
```

*points.sld*

```

<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor xmlns=
"http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld" xmlns:gml=
"http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc" version="1.0.0">
  <sld:UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>Default Styler</sld:Name>
      <sld:FeatureTypeStyle>
        <sld:Rule>
          <sld:PointSymbolizer>
            <sld:Graphic>
              <sld:Mark>
                <sld:WellKnownName>circle</sld:WellKnownName>
                <sld:Fill>
                  <sld:CssParameter name="fill">#ff0000</sld:CssParameter>
                </sld:Fill>
              </sld:Mark>
              <sld:Size>6</sld:Size>
            </sld:Graphic>
          </sld:PointSymbolizer>
        </sld:Rule>
        <sld:VendorOption name="ruleEvaluation">first</sld:VendorOption>
      </sld:FeatureTypeStyle>
    </sld:UserStyle>
  </sld:UserLayer>
</sld:StyledLayerDescriptor>

```

*points.lysld*

```

name: Default Styler
feature-styles:
- name: name
  rules:
    - scale: [min, max]
      symbolizers:
        - point:
            size: 6
            symbols:
              - mark:
                  shape: circle
                  fill-color: '#FF0000'
  x-ruleEvaluation: first

```

# Ysld to SLD

Convert Ysld to SLD.

| Short Name | Long Name        | Description             |
|------------|------------------|-------------------------|
| -i         | --input          | The input file or url   |
| -o         | --output         | The output file         |
| -w         | --writer-options | The StyleWriter options |
|            | --help           | Print the help message  |
|            | --web-help       | Open help in a browser  |

```
geoc style ysld2sld -i target/points.ysld -o target/points.sld
```

*points.ysld*

```
name: Default Styler
feature-styles:
- name: name
  rules:
  - scale: [min, max]
    symbolizers:
    - point:
        size: 6
        symbols:
        - mark:
            shape: circle
            fill-color: '#FF0000'
  x-ruleEvaluation: first
```

*points.sld*

```

<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor xmlns=
"http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld" xmlns:gml=
"http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc" version="1.0.0">
  <sld:UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>Default Styler</sld:Name>
      <sld:FeatureTypeStyle>
        <sld:Name>name</sld:Name>
        <sld:Rule>
          <sld:PointSymbolizer>
            <sld:Graphic>
              <sld:Mark>
                <sld:WellKnownName>circle</sld:WellKnownName>
                <sld:Fill>
                  <sld:CssParameter name="fill">#FF0000</sld:CssParameter>
                </sld:Fill>
              </sld:Mark>
              <sld:Size>6</sld:Size>
            </sld:Graphic>
          </sld:PointSymbolizer>
        </sld:Rule>
        <sld:VendorOption name="ruleEvaluation">first</sld:VendorOption>
      </sld:FeatureTypeStyle>
    </sld:UserStyle>
  </sld:UserLayer>
</sld:StyledLayerDescriptor>

```

## Unique Values from Text

Create a Style from reading values in the unique values format.

| Short Name | Long Name        | Description                                    |
|------------|------------------|--|
| -f         | --field          | The field                                      |
| -g         | --geometry-type  | The geometry type (point, linestring, polygon) |
| -i         | --input          | The input file or url                          |
| -t         | --type           | The output type (sld or ysld)                  |
| -o         | --output         | The output file                                |
| -w         | --writer-options | The StyleWriter options                        |
|            | --help           | Print the help message                         |
|            | --web-help       | Open help in a browser                         |

```
geoc style uniquevaluesfromtext -i target/units.txt -f UNIT -g polygon -t sld -o target/units.sld
```

*units.txt*

```
AHa=#aa0c74
AHat=#b83b1f
AHcf=#964642
AHh=#78092e
AHpe=#78092e
AHt=#5f025a
AHt3=#e76161
Aa1=#fcfedcd
Aa2=#94474b
```

*units.sld*

```
<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor xmlns=
"http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld" xmlns:gml=
"http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc" version="1.0.0">
  <sld:UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>Default Styler</sld:Name>
      <sld:FeatureTypeStyle>
        <sld:Name>name</sld:Name>
        <sld:Rule>
          <ogc:Filter>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>UNIT</ogc:PropertyName>
              <ogc:Literal>AHa</ogc:Literal>
            </ogc:PropertyIsEqualTo>
          </ogc:Filter>
          <sld:PolygonSymbolizer>
            <sld:Fill>
              <sld:CssParameter name="fill">#aa0c74</sld:CssParameter>
            </sld:Fill>
          </sld:PolygonSymbolizer>
          <sld:LineSymbolizer>
            <sld:Stroke>
              <sld:CssParameter name="stroke">#760851</sld:CssParameter>
              <sld:CssParameter name="stroke-width">0.5</sld:CssParameter>
            </sld:Stroke>
          </sld:LineSymbolizer>
        </sld:Rule>
        <sld:Rule>
```

```

<ogc:Filter>
  <ogc:PropertyIsEqualTo>
    <ogc:PropertyName>UNIT</ogc:PropertyName>
    <ogc:Literal>AHat</ogc:Literal>
  </ogc:PropertyIsEqualTo>
</ogc:Filter>
<sld:PolygonSymbolizer>
  <sld:Fill>
    <sld:CssParameter name="fill">#b83b1f</sld:CssParameter>
  </sld:Fill>
</sld:PolygonSymbolizer>
<sld:LineSymbolizer>
  <sld:Stroke>
    <sld:CssParameter name="stroke">#802915</sld:CssParameter>
    <sld:CssParameter name="stroke-width">0.5</sld:CssParameter>
  </sld:Stroke>
</sld:LineSymbolizer>
</sld:Rule>
<sld:Rule>
  <ogc:Filter>
    <ogc:PropertyIsEqualTo>
      <ogc:PropertyName>UNIT</ogc:PropertyName>
      <ogc:Literal>AHcf</ogc:Literal>
    </ogc:PropertyIsEqualTo>
  </ogc:Filter>
  <sld:PolygonSymbolizer>
    <sld:Fill>
      <sld:CssParameter name="fill">#964642</sld:CssParameter>
    </sld:Fill>
  </sld:PolygonSymbolizer>
  <sld:LineSymbolizer>
    <sld:Stroke>
      <sld:CssParameter name="stroke">#69312e</sld:CssParameter>
      <sld:CssParameter name="stroke-width">0.5</sld:CssParameter>
    </sld:Stroke>
  </sld:LineSymbolizer>
</sld:Rule>
<sld:Rule>
  <ogc:Filter>
    <ogc:PropertyIsEqualTo>
      <ogc:PropertyName>UNIT</ogc:PropertyName>
      <ogc:Literal>AHh</ogc:Literal>
    </ogc:PropertyIsEqualTo>
  </ogc:Filter>
  <sld:PolygonSymbolizer>
    <sld:Fill>
      <sld:CssParameter name="fill">#78092e</sld:CssParameter>
    </sld:Fill>
  </sld:PolygonSymbolizer>
  <sld:LineSymbolizer>
    <sld:Stroke>

```

```

        <sld:CssParameter name="stroke">#540620</sld:CssParameter>
        <sld:CssParameter name="stroke-width">0.5</sld:CssParameter>
    </sld:Stroke>
</sld:LineSymbolizer>
</sld:Rule>
<sld:Rule>
<ogc:Filter>
<ogc:PropertyIsEqualTo>
    <ogc:PropertyName>UNIT</ogc:PropertyName>
    <ogc:Literal>AHpe</ogc:Literal>
</ogc:PropertyIsEqualTo>
</ogc:Filter>
<sld:PolygonSymbolizer>
<sld:Fill>
    <sld:CssParameter name="fill">#78092e</sld:CssParameter>
</sld:Fill>
</sld:PolygonSymbolizer>
<sld:LineSymbolizer>
<sld:Stroke>
    <sld:CssParameter name="stroke">#540620</sld:CssParameter>
    <sld:CssParameter name="stroke-width">0.5</sld:CssParameter>
</sld:Stroke>
</sld:LineSymbolizer>
</sld:Rule>
<sld:Rule>
<ogc:Filter>
<ogc:PropertyIsEqualTo>
    <ogc:PropertyName>UNIT</ogc:PropertyName>
    <ogc:Literal>AHT</ogc:Literal>
</ogc:PropertyIsEqualTo>
</ogc:Filter>
<sld:PolygonSymbolizer>
<sld:Fill>
    <sld:CssParameter name="fill">#5f025a</sld:CssParameter>
</sld:Fill>
</sld:PolygonSymbolizer>
<sld:LineSymbolizer>
<sld:Stroke>
    <sld:CssParameter name="stroke">#42013e</sld:CssParameter>
    <sld:CssParameter name="stroke-width">0.5</sld:CssParameter>
</sld:Stroke>
</sld:LineSymbolizer>
</sld:Rule>
<sld:Rule>
<ogc:Filter>
<ogc:PropertyIsEqualTo>
    <ogc:PropertyName>UNIT</ogc:PropertyName>
    <ogc:Literal>AHT3</ogc:Literal>
</ogc:PropertyIsEqualTo>
</ogc:Filter>
<sld:PolygonSymbolizer>

```

```

<sld:Fill>
  <sld.CssParameter name="fill">#e76161</sld.CssParameter>
</sld:Fill>
</sld:PolygonSymbolizer>
<sld:LineSymbolizer>
  <sld:Stroke>
    <sld.CssParameter name="stroke">#a14343</sld.CssParameter>
    <sld.CssParameter name="stroke-width">0.5</sld.CssParameter>
  </sld:Stroke>
</sld:LineSymbolizer>
</sld:Rule>
<sld:Rule>
  <ogc:Filter>
    <ogc:PropertyIsEqualTo>
      <ogc:PropertyName>UNIT</ogc:PropertyName>
      <ogc:Literal>Aa1</ogc:Literal>
    </ogc:PropertyIsEqualTo>
  </ogc:Filter>
<sld:PolygonSymbolizer>
  <sld:Fill>
    <sld.CssParameter name="fill">#fcedcd</sld.CssParameter>
  </sld:Fill>
</sld:PolygonSymbolizer>
<sld:LineSymbolizer>
  <sld:Stroke>
    <sld.CssParameter name="stroke">#b0a58f</sld.CssParameter>
    <sld.CssParameter name="stroke-width">0.5</sld.CssParameter>
  </sld:Stroke>
</sld:LineSymbolizer>
</sld:Rule>
<sld:Rule>
  <ogc:Filter>
    <ogc:PropertyIsEqualTo>
      <ogc:PropertyName>UNIT</ogc:PropertyName>
      <ogc:Literal>Aa2</ogc:Literal>
    </ogc:PropertyIsEqualTo>
  </ogc:Filter>
<sld:PolygonSymbolizer>
  <sld:Fill>
    <sld.CssParameter name="fill">#94474b</sld.CssParameter>
  </sld:Fill>
</sld:PolygonSymbolizer>
<sld:LineSymbolizer>
  <sld:Stroke>
    <sld.CssParameter name="stroke">#673134</sld.CssParameter>
    <sld.CssParameter name="stroke-width">0.5</sld.CssParameter>
  </sld:Stroke>
</sld:LineSymbolizer>
</sld:Rule>
</sld:FeatureTypeStyle>
</sld:UserStyle>

```

```
</sld:UserLayer>  
</sld:StyledLayerDescriptor>
```

# Tile Commands

## Tile Layers

All of the tile commands work with a tile layer.

Supported Tile Layers include:

- MBTiles
- GeoPackage
- TMS
- OSM
- UTFGrid
- Vector Tiles

**Tile layer configuration strings** are similar to layer and map layer configuration strings.

- **pyramid** = Several tile layers can take a pyramid attribute. You can use one of several well known pyramid names:
  - globalmercator
  - mercator
  - globalmercatorbottomleft
  - globalgeodetic
  - geodetic
  - file that contains pyramid metadata in csv, xml, or json format.
- **type** = The type of image layer.
  - mbtiles
  - geopackage
  - tms
  - osm

### mbtiles

- type=mbtiles file=states.mbtiles
- type=mbtiles file=states.mbtiles name=states description='The united states'
- states.mbtiles

### geopackage

- type=geopackage file=states.gpkg name=states pyramid=globalmercator
- states.gpkg

## tms

- type=tms file=/Users/you/tms format=jpeg
- type=tms file=/Users/you/tms format=png name=tms pyramid=geodetic

## osm

- type=osm url=http://a.tile.openstreetmap.org
- type=osm urls=http://a.tile.openstreetmap.org,http://b.tile.openstreetmap.org

## utfgrid

- type=utfgrid file=/Users/me/tiles/states

## vectortiles

- type=vectortiles name=states file=/Users/me/tiles/states format=mvt pyramid=GlobalMercator
- type=vectortiles name=states url=http://vectortiles.org format=pbf pyramid=GlobalGeodeti

# Delete

Delete tiles from a tile layer

| Short Name | Long Name    | Description                |
|------------|--------------|----------------------------|
| -l         | --tile-layer | The tile layer             |
| -i         | --tile       | The Tile Z/X/Y coordinates |
| -b         | --bounds     | The bounds                 |
| -z         | --zoom-level | The tile zoom level        |
| -x         | --minx       | The min x or col           |
| -y         | --miny       | The min y or row           |
| -c         | --maxx       | The max x or col           |
| -u         | --maxy       | The max y or row           |
| -w         | --width      | The raster width           |
| -h         | --height     | The raster height          |
|            | --help       | Print the help message     |
|            | --web-help   | Open help in a browser     |

```
geoc tile delete -l "type=mbtiles file=target/world.mbtiles" -z 2
```

# Generate

Generate tiles.

| Short Name | Long Name      | Description                            |
|------------|----------------|--|
| -l         | --tile-layer   | The tile layer                         |
| -f         | --field        | A field                                |
| -d         | --layer-fields | A List of sub fields for a layer       |
| -m         | --layer        | The map layer                          |
| -s         | --start-zoom   | The start zoom level                   |
| -e         | --end-zoom     | The end zoom level                     |
| -b         | --bounds       | The bounds                             |
| -t         | --metatile     | The metatile width,height              |
| -i         | --missing      | Whether to generate only missing tiles |
| -v         | --verbose      | The verbose flag                       |
|            | --help         | Print the help message                 |
|            | --web-help     | Open help in a browser                 |

MBTiles

```
geoc tile generate -l "type=mbtiles file=target/world.mbtiles" -m "layertype=layer  
file=src/test/resources/data.gpkg layername=ocean style=src/test/resources/ocean.sld"  
-m "layertype=layer file=src/test/resources/data.gpkg layername=countries  
style=src/test/resources/countries.sld" -s 0 -e 2 --verbose
```

Zoom Level 0

```
0). Tile(x:0, y:0, z:0)  
    Bounds(-2.003639514788131E7,-  
2.003747120513706E7,2.003639514788131E7,2.003747120513706E7,EPSC:3857)
```

Generating 1 tile took 0.076688084 seconds

Zoom Level 1

```
0). Tile(x:0, y:0, z:1)  
    Bounds(-2.003639514788131E7,-2.003747120513706E7,0.0,-3.725290298461914E-  
9,EPSC:3857)
```

```
1). Tile(x:1, y:0, z:1)  
    Bounds(0.0,-2.003747120513706E7,2.003639514788131E7,-3.725290298461914E-  
9,EPSC:3857)
```

```
2). Tile(x:0, y:1, z:1)  
    Bounds(-2.003639514788131E7,-3.725290298461914E-  
9,0.0,2.003747120513706E7,EPSC:3857)
```

```
3). Tile(x:1, y:1, z:1)  
    Bounds(0.0,-3.725290298461914E-
```

```
9,2.003639514788131E7,2.003747120513706E7,EPG:3857)
Generating 4 tiles took 0.284631997 seconds
Zoom Level 2
 0). Tile(x:0, y:0, z:2)
    Bounds(-2.003639514788131E7,-2.003747120513706E7,-1.0018197573940655E7,-
1.0018735602568535E7,EPG:3857)
  1). Tile(x:1, y:0, z:2)
    Bounds(-1.0018197573940655E7,-2.003747120513706E7,0.0,-
1.0018735602568535E7,EPG:3857)
  2). Tile(x:2, y:0, z:2)
    Bounds(0.0,-2.003747120513706E7,1.0018197573940655E7,-
1.0018735602568535E7,EPG:3857)
  3). Tile(x:3, y:0, z:2)
    Bounds(1.0018197573940653E7,-2.003747120513706E7,2.0036395147881307E7,-
1.0018735602568535E7,EPG:3857)
  4). Tile(x:0, y:1, z:2)
    Bounds(-2.003639514788131E7,-1.0018735602568535E7,-1.0018197573940655E7,-
3.725290298461914E-9,EPG:3857)
  5). Tile(x:1, y:1, z:2)
    Bounds(-1.0018197573940655E7,-1.0018735602568535E7,0.0,-3.725290298461914E-
9,EPG:3857)
  6). Tile(x:2, y:1, z:2)
    Bounds(0.0,-1.0018735602568535E7,1.0018197573940655E7,-3.725290298461914E-
9,EPG:3857)
  7). Tile(x:3, y:1, z:2)
    Bounds(1.0018197573940653E7,-1.0018735602568535E7,2.0036395147881307E7,-
3.725290298461914E-9,EPG:3857)
  8). Tile(x:0, y:2, z:2)
    Bounds(-2.003639514788131E7,-3.725290298461914E-9,-
1.0018197573940655E7,1.0018735602568528E7,EPG:3857)
  9). Tile(x:1, y:2, z:2)
    Bounds(-1.0018197573940655E7,-3.725290298461914E-
9,0.0,1.0018735602568528E7,EPG:3857)
  10). Tile(x:2, y:2, z:2)
    Bounds(0.0,-3.725290298461914E-
9,1.0018197573940655E7,1.0018735602568528E7,EPG:3857)
  11). Tile(x:3, y:2, z:2)
    Bounds(1.0018197573940653E7,-3.725290298461914E-
9,2.0036395147881307E7,1.0018735602568528E7,EPG:3857)
  12). Tile(x:0, y:3, z:2)
    Bounds(-2.003639514788131E7,1.001873560256853E7,-
1.0018197573940655E7,2.003747120513706E7,EPG:3857)
  13). Tile(x:1, y:3, z:2)
    Bounds(-
1.0018197573940655E7,1.001873560256853E7,0.0,2.003747120513706E7,EPG:3857)
  14). Tile(x:2, y:3, z:2)

Bounds(0.0,1.001873560256853E7,1.0018197573940655E7,2.003747120513706E7,EPG:3857)
  15). Tile(x:3, y:3, z:2)

Bounds(1.0018197573940653E7,1.001873560256853E7,2.0036395147881307E7,2.003747120513706
```

E7, EPSG:3857)

Generating 16 tiles took 0.97416854 seconds



## GeoPackage

```
geoc tile generate -l "type=geopackage file=target/world.gpkg name=world
pyramid=geodetic" -m "layertype=layer file=src/test/resources/data.gpkg
layername=ocean style=src/test/resources/ocean.sld" -m "layertype=layer
file=src/test/resources/data.gpkg layername=countries
style=src/test/resources/countries.sld" -s 0 -e 2 --verbose
```

### Zoom Level 0

```
0). Tile(x:0, y:0, z:0)
    Bounds(-179.99,-89.99,0.0,89.99,EPGS:4326)
```

```
1). Tile(x:1, y:0, z:0)
    Bounds(0.0,-89.99,179.99,89.99,EPGS:4326)
```

Generating 2 tiles took 0.076554984 seconds

### Zoom Level 1

```
0). Tile(x:0, y:0, z:1)
    Bounds(-179.99,0.0,-89.995,89.99,EPGS:4326)
```

```
1). Tile(x:1, y:0, z:1)
    Bounds(-89.995,0.0,0.0,89.99,EPGS:4326)
```

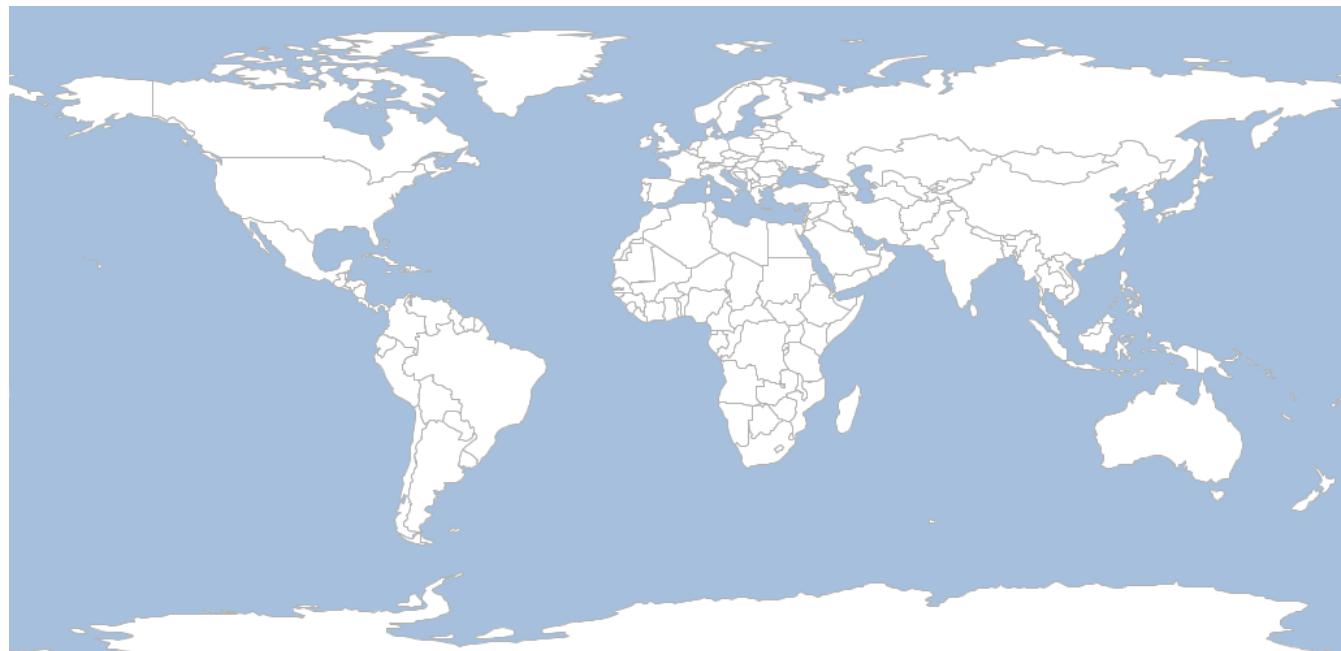
```

2). Tile(x:2, y:0, z:1)
    Bounds(0.0,0.0,89.995,89.99,EPSG:4326)
3). Tile(x:3, y:0, z:1)
    Bounds(89.995,0.0,179.99,89.99,EPSG:4326)
4). Tile(x:0, y:1, z:1)
    Bounds(-179.99,-89.99,-89.995,0.0,EPSG:4326)
5). Tile(x:1, y:1, z:1)
    Bounds(-89.995,-89.99,0.0,0.0,EPSG:4326)
6). Tile(x:2, y:1, z:1)
    Bounds(0.0,-89.99,89.995,0.0,EPSG:4326)
7). Tile(x:3, y:1, z:1)
    Bounds(89.995,-89.99,179.99,0.0,EPSG:4326)
Generating 8 tiles took 0.198743739 seconds
Zoom Level 2
0). Tile(x:0, y:0, z:2)
    Bounds(-179.99,44.995,-134.9925,89.99,EPSG:4326)
1). Tile(x:1, y:0, z:2)
    Bounds(-134.9925,44.995,-89.995,89.99,EPSG:4326)
2). Tile(x:2, y:0, z:2)
    Bounds(-89.995,44.995,-44.9975,89.99,EPSG:4326)
3). Tile(x:3, y:0, z:2)
    Bounds(-44.9975,44.995,0.0,89.99,EPSG:4326)
4). Tile(x:4, y:0, z:2)
    Bounds(0.0,44.995,44.9975,89.99,EPSG:4326)
5). Tile(x:5, y:0, z:2)
    Bounds(44.9975,44.995,89.995,89.99,EPSG:4326)
6). Tile(x:6, y:0, z:2)
    Bounds(89.995,44.995,134.9925,89.99,EPSG:4326)
7). Tile(x:7, y:0, z:2)
    Bounds(134.9925,44.995,179.99,89.99,EPSG:4326)
8). Tile(x:0, y:1, z:2)
    Bounds(-179.99,-7.105427357601002E-15,-134.9925,44.99499999999999,EPSG:4326)
9). Tile(x:1, y:1, z:2)
    Bounds(-134.9925,-7.105427357601002E-15,-89.995,44.99499999999999,EPSG:4326)
10). Tile(x:2, y:1, z:2)
    Bounds(-89.995,-7.105427357601002E-15,-44.9975,44.99499999999999,EPSG:4326)
11). Tile(x:3, y:1, z:2)
    Bounds(-44.9975,-7.105427357601002E-15,0.0,44.99499999999999,EPSG:4326)
12). Tile(x:4, y:1, z:2)
    Bounds(0.0,-7.105427357601002E-15,44.9975,44.99499999999999,EPSG:4326)
13). Tile(x:5, y:1, z:2)
    Bounds(44.9975,-7.105427357601002E-15,89.995,44.99499999999999,EPSG:4326)
14). Tile(x:6, y:1, z:2)
    Bounds(89.995,-7.105427357601002E-15,134.9925,44.99499999999999,EPSG:4326)
15). Tile(x:7, y:1, z:2)
    Bounds(134.9925,-7.105427357601002E-15,179.99,44.99499999999999,EPSG:4326)
16). Tile(x:0, y:2, z:2)
    Bounds(-179.99,-44.995,-134.9925,0.0,EPSG:4326)
17). Tile(x:1, y:2, z:2)
    Bounds(-134.9925,-44.995,-89.995,0.0,EPSG:4326)
18). Tile(x:2, y:2, z:2)

```

```
    Bounds(-89.995,-44.995,-44.9975,0.0,EPSG:4326)
19). Tile(x:3, y:2, z:2)
    Bounds(-44.9975,-44.995,0.0,0.0,EPSG:4326)
20). Tile(x:4, y:2, z:2)
    Bounds(0.0,-44.995,44.9975,0.0,EPSG:4326)
21). Tile(x:5, y:2, z:2)
    Bounds(44.9975,-44.995,89.995,0.0,EPSG:4326)
22). Tile(x:6, y:2, z:2)
    Bounds(89.995,-44.995,134.9925,0.0,EPSG:4326)
23). Tile(x:7, y:2, z:2)
    Bounds(134.9925,-44.995,179.99,0.0,EPSG:4326)
24). Tile(x:0, y:3, z:2)
    Bounds(-179.99,-89.99,-134.9925,-44.995,EPSG:4326)
25). Tile(x:1, y:3, z:2)
    Bounds(-134.9925,-89.99,-89.995,-44.995,EPSG:4326)
26). Tile(x:2, y:3, z:2)
    Bounds(-89.995,-89.99,-44.9975,-44.995,EPSG:4326)
27). Tile(x:3, y:3, z:2)
    Bounds(-44.9975,-89.99,0.0,-44.995,EPSG:4326)
28). Tile(x:4, y:3, z:2)
    Bounds(0.0,-89.99,44.9975,-44.995,EPSG:4326)
29). Tile(x:5, y:3, z:2)
    Bounds(44.9975,-89.99,89.995,-44.995,EPSG:4326)
30). Tile(x:6, y:3, z:2)
    Bounds(89.995,-89.99,134.9925,-44.995,EPSG:4326)
31). Tile(x:7, y:3, z:2)
    Bounds(134.9925,-89.99,179.99,-44.995,EPSG:4326)
```

Generating 32 tiles took 0.658979241 seconds



TMS

```
geoc tile generate -l "type=tms file=target/tiles" -m "layertype=layer  
file=src/test/resources/data.gpkg layername=ocean style=src/test/resources/ocean.sld"  
-m "layertype=layer file=src/test/resources/data.gpkg layername=countries  
style=src/test/resources/countries.sld" -s 0 -e 2 --verbose
```

```
Zoom Level 0  
0). Tile(x:0, y:0, z:0)  
    Bounds(-2.003639514788131E7, -  
2.0037471205137067E7, 2.003639514788131E7, 2.0037471205137067E7, EPSG:3857)  
Generating 1 tile took 0.091686638 seconds  
Zoom Level 1  
0). Tile(x:0, y:0, z:1)  
    Bounds(-2.003639514788131E7, -2.0037471205137067E7, 0.0, -3.725290298461914E-  
9, EPSG:3857)  
1). Tile(x:1, y:0, z:1)  
    Bounds(0.0, -2.0037471205137067E7, 2.003639514788131E7, -3.725290298461914E-  
9, EPSG:3857)  
2). Tile(x:0, y:1, z:1)  
    Bounds(-2.003639514788131E7, -3.725290298461914E-  
9, 0.0, 2.0037471205137067E7, EPSG:3857)  
3). Tile(x:1, y:1, z:1)  
    Bounds(0.0, -3.725290298461914E-  
9, 2.003639514788131E7, 2.0037471205137067E7, EPSG:3857)  
Generating 4 tiles took 0.25818566 seconds  
Zoom Level 2  
0). Tile(x:0, y:0, z:2)  
    Bounds(-2.003639514788131E7, -2.0037471205137067E7, -1.0018197573940655E7, -  
1.0018735602568535E7, EPSG:3857)  
1). Tile(x:1, y:0, z:2)  
    Bounds(-1.0018197573940655E7, -2.0037471205137067E7, 0.0, -  
1.0018735602568535E7, EPSG:3857)  
2). Tile(x:2, y:0, z:2)  
    Bounds(0.0, -2.0037471205137067E7, 1.0018197573940655E7, -  
1.0018735602568535E7, EPSG:3857)  
3). Tile(x:3, y:0, z:2)  
    Bounds(1.0018197573940653E7, -2.0037471205137067E7, 2.0036395147881307E7, -  
1.0018735602568535E7, EPSG:3857)  
4). Tile(x:0, y:1, z:2)  
    Bounds(-2.003639514788131E7, -1.0018735602568535E7, -1.0018197573940655E7, -  
3.725290298461914E-9, EPSG:3857)  
5). Tile(x:1, y:1, z:2)  
    Bounds(-1.0018197573940655E7, -1.0018735602568535E7, 0.0, -3.725290298461914E-  
9, EPSG:3857)  
6). Tile(x:2, y:1, z:2)  
    Bounds(0.0, -1.0018735602568535E7, 1.0018197573940655E7, -3.725290298461914E-  
9, EPSG:3857)  
7). Tile(x:3, y:1, z:2)  
    Bounds(1.0018197573940653E7, -1.0018735602568535E7, 2.0036395147881307E7, -  
3.725290298461914E-9, EPSG:3857)
```

```
8). Tile(x:0, y:2, z:2)
    Bounds(-2.003639514788131E7,-3.725290298461914E-9,-
1.0018197573940655E7,1.0018735602568528E7,EPNG:3857)
9). Tile(x:1, y:2, z:2)
    Bounds(-1.0018197573940655E7,-3.725290298461914E-
9,0.0,1.0018735602568528E7,EPNG:3857)
10). Tile(x:2, y:2, z:2)
    Bounds(0.0,-3.725290298461914E-
9,1.0018197573940655E7,1.0018735602568528E7,EPNG:3857)
11). Tile(x:3, y:2, z:2)
    Bounds(1.0018197573940653E7,-3.725290298461914E-
9,2.0036395147881307E7,1.0018735602568528E7,EPNG:3857)
12). Tile(x:0, y:3, z:2)
    Bounds(-2.003639514788131E7,1.001873560256853E7,-
1.0018197573940655E7,2.003747120513706E7,EPNG:3857)
13). Tile(x:1, y:3, z:2)
    Bounds(-
1.0018197573940655E7,1.001873560256853E7,0.0,2.003747120513706E7,EPNG:3857)
14). Tile(x:2, y:3, z:2)

Bounds(0.0,1.001873560256853E7,1.0018197573940655E7,2.003747120513706E7,EPNG:3857)
15). Tile(x:3, y:3, z:2)

Bounds(1.0018197573940653E7,1.001873560256853E7,2.0036395147881307E7,2.003747120513706
E7,EPNG:3857)
```

Generating 16 tiles took 0.860476737 seconds



## Vector Tiles (PBF)

```
geoc tile generate -l "type=vectortiles format=pbf file=target/vectortiles" -m  
"layertype=layer file=src/test/resources/data.gpkg layername=countries" -d  
countries=NAME,TYPE,LEVEL -s 0 -e 2 --verbose
```

### Zoom Level 0

```
0). Tile(x:0, y:0, z:0)  
    Bounds(-2.003639514788131E7, -  
2.0037471205137067E7, 2.003639514788131E7, 2.0037471205137067E7, EPSG:3857)
```

```
Generating 1 tile took 0.503889078 seconds
```

### Zoom Level 1

```
0). Tile(x:0, y:0, z:1)  
    Bounds(-2.003639514788131E7, -2.0037471205137067E7, 0.0, -3.725290298461914E-  
9, EPSG:3857)
```

```
1). Tile(x:1, y:0, z:1)  
    Bounds(0.0, -2.0037471205137067E7, 2.003639514788131E7, -3.725290298461914E-  
9, EPSG:3857)
```

```
2). Tile(x:0, y:1, z:1)  
    Bounds(-2.003639514788131E7, -3.725290298461914E-  
9, 0.0, 2.0037471205137067E7, EPSG:3857)
```

```
3). Tile(x:1, y:1, z:1)
```

```

    Bounds(0.0,-3.725290298461914E-
9,2.003639514788131E7,2.003747120513706E7,EPG:3857)
    Generating 4 tiles took 0.915530617 seconds
Zoom Level 2
    0). Tile(x:0, y:0, z:2)
        Bounds(-2.003639514788131E7,-2.003747120513706E7,-1.0018197573940655E7,-
1.0018735602568535E7,EPG:3857)
    1). Tile(x:1, y:0, z:2)
        Bounds(-1.0018197573940655E7,-2.003747120513706E7,0.0,-
1.0018735602568535E7,EPG:3857)
    2). Tile(x:2, y:0, z:2)
        Bounds(0.0,-2.003747120513706E7,1.0018197573940655E7,-
1.0018735602568535E7,EPG:3857)
    3). Tile(x:3, y:0, z:2)
        Bounds(1.0018197573940653E7,-2.003747120513706E7,2.0036395147881307E7,-
1.0018735602568535E7,EPG:3857)
    4). Tile(x:0, y:1, z:2)
        Bounds(-2.003639514788131E7,-1.0018735602568535E7,-1.0018197573940655E7,-
3.725290298461914E-9,EPG:3857)
    5). Tile(x:1, y:1, z:2)
        Bounds(-1.0018197573940655E7,-1.0018735602568535E7,0.0,-3.725290298461914E-
9,EPG:3857)
    6). Tile(x:2, y:1, z:2)
        Bounds(0.0,-1.0018735602568535E7,1.0018197573940655E7,-3.725290298461914E-
9,EPG:3857)
    7). Tile(x:3, y:1, z:2)
        Bounds(1.0018197573940653E7,-1.0018735602568535E7,2.0036395147881307E7,-
3.725290298461914E-9,EPG:3857)
    8). Tile(x:0, y:2, z:2)
        Bounds(-2.003639514788131E7,-3.725290298461914E-9,-
1.0018197573940655E7,1.0018735602568528E7,EPG:3857)
    9). Tile(x:1, y:2, z:2)
        Bounds(-1.0018197573940655E7,-3.725290298461914E-
9,0.0,1.0018735602568528E7,EPG:3857)
    10). Tile(x:2, y:2, z:2)
        Bounds(0.0,-3.725290298461914E-
9,1.0018197573940655E7,1.0018735602568528E7,EPG:3857)
    11). Tile(x:3, y:2, z:2)
        Bounds(1.0018197573940653E7,-3.725290298461914E-
9,2.0036395147881307E7,1.0018735602568528E7,EPG:3857)
    12). Tile(x:0, y:3, z:2)
        Bounds(-2.003639514788131E7,1.001873560256853E7,-
1.0018197573940655E7,2.003747120513706E7,EPG:3857)
    13). Tile(x:1, y:3, z:2)
        Bounds(-
1.0018197573940655E7,1.001873560256853E7,0.0,2.003747120513706E7,EPG:3857)
    14). Tile(x:2, y:3, z:2)

Bounds(0.0,1.001873560256853E7,1.0018197573940655E7,2.003747120513706E7,EPG:3857)
    15). Tile(x:3, y:3, z:2)

```

```
Bounds(1.0018197573940653E7,1.001873560256853E7,2.0036395147881307E7,2.003747120513706  
E7, EPSG:3857)
```

```
Generating 16 tiles took 0.854437388 seconds
```

## Tile Bounds

Get the Bounds of a tile.

| Short Name | Long Name    | Description            |
|------------|--------------|------------------------|
| -p         | --pyramid    | The tile pyramid       |
| -z         | --zoom-level | The tile zoom level    |
| -x         | --column     | The tile x or column   |
| -y         | --row        | The tile y or row      |
|            | --help       | Print the help message |
|            | --web-help   | Open help in a browser |

```
geoc tile get bounds -p mercator -z 3 -x 2 -y 1
```

```
POLYGON ((-10018197.573940655 -15028103.403852802, -10018197.573940655  
-10018735.602568537, -5009098.786970328 -10018735.602568537, -5009098.786970328  
-15028103.403852802, -10018197.573940655 -15028103.403852802))
```

## List Tiles

Get a list of tiles for a given geometry

| Short Name | Long Name    | Description            |
|------------|--------------|------------------------|
| -p         | --pyramid    | The tile pyramid       |
| -b         | --bounds     | The bounds             |
| -z         | --zoom-level | The tile zoom level    |
|            | --help       | Print the help message |
|            | --web-help   | Open help in a browser |

```
geoc tile list tiles -p mercator -z 10 -b  
2315277.538707974,4356146.199006655,2534193.2172859586,4470343.227121928
```

```
10/571/623
10/572/623
10/573/623
10/574/623
10/575/623
10/576/623
10/571/624
10/572/624
10/573/624
10/574/624
10/575/624
10/576/624
10/571/625
10/572/625
10/573/625
10/574/625
10/575/625
10/576/625
10/571/626
10/572/626
10/573/626
10/574/626
10/575/626
10/576/626
```

## Pyramid

Get a Pyramid from a TileLayer.

| Short Name | Long Name     | Description                       |
|------------|---------------|-----------------------------------|
| -l         | --tile-layer  | The tile layer                    |
| -o         | --output-type | The output type (text, xml, json) |
|            | --help        | Print the help message            |
|            | --web-help    | Open help in a browser            |

Text

```
geoc tile pyramid -l "type=geopackage file=src/test/resources/data.gpkg name=world" -o
text
```

```
EPSG:4326
-179.99,-89.99,179.99,89.99,EPGS:4326
TOP_LEFT
256,256
0,2,1,0.703125,0.703125
1,4,2,0.3515625,0.3515625
2,8,4,0.17578125,0.17578125
3,16,8,0.087890625,0.087890625
4,32,16,0.0439453125,0.0439453125
5,64,32,0.02197265625,0.02197265625
6,128,64,0.010986328125,0.010986328125
7,256,128,0.0054931640625,0.0054931640625
8,512,256,0.00274658203125,0.00274658203125
9,1024,512,0.001373291015625,0.001373291015625
10,2048,1024,6.866455078125E-4,6.866455078125E-4
11,4096,2048,3.4332275390625E-4,3.4332275390625E-4
12,8192,4096,1.71661376953125E-4,1.71661376953125E-4
13,16384,8192,8.58306884765625E-5,8.58306884765625E-5
14,32768,16384,4.291534423828125E-5,4.291534423828125E-5
15,65536,32768,2.1457672119140625E-5,2.1457672119140625E-5
16,131072,65536,1.0728836059570312E-5,1.0728836059570312E-5
17,262144,131072,5.364418029785156E-6,5.364418029785156E-6
18,524288,262144,2.682209014892578E-6,2.682209014892578E-6
19,1048576,524288,1.341104507446289E-6,1.341104507446289E-6
```

## JSON

```
geoc tile pyramid -l "type=geopackage file=src/test/resources/data.gpkg name=world" -o json
```

```
{
  "proj": "EPSG:4326",
  "bounds": {
    "minX": -179.99,
    "minY": -89.99,
    "maxX": 179.99,
    "maxY": 89.99
  },
  "origin": "TOP_LEFT",
  "tileSize": {
    "width": 256,
    "height": 256
  },
  "grids": [
    {
      "z": 0,
      "width": 2,
      "height": 1,
```

```
"xres": 0.703125,  
"yres": 0.703125  
},  
{  
    "z": 1,  
    "width": 4,  
    "height": 2,  
    "xres": 0.3515625,  
    "yres": 0.3515625  
},  
{  
    "z": 2,  
    "width": 8,  
    "height": 4,  
    "xres": 0.17578125,  
    "yres": 0.17578125  
},  
{  
    "z": 3,  
    "width": 16,  
    "height": 8,  
    "xres": 0.087890625,  
    "yres": 0.087890625  
},  
{  
    "z": 4,  
    "width": 32,  
    "height": 16,  
    "xres": 0.0439453125,  
    "yres": 0.0439453125  
},  
{  
    "z": 5,  
    "width": 64,  
    "height": 32,  
    "xres": 0.02197265625,  
    "yres": 0.02197265625  
},  
{  
    "z": 6,  
    "width": 128,  
    "height": 64,  
    "xres": 0.010986328125,  
    "yres": 0.010986328125  
},  
{  
    "z": 7,  
    "width": 256,  
    "height": 128,  
    "xres": 0.0054931640625,  
    "yres": 0.0054931640625
```

```
},
{
  "z": 8,
  "width": 512,
  "height": 256,
  "xres": 0.00274658203125,
  "yres": 0.00274658203125
},
{
  "z": 9,
  "width": 1024,
  "height": 512,
  "xres": 0.001373291015625,
  "yres": 0.001373291015625
},
{
  "z": 10,
  "width": 2048,
  "height": 1024,
  "xres": 6.866455078125E-4,
  "yres": 6.866455078125E-4
},
{
  "z": 11,
  "width": 4096,
  "height": 2048,
  "xres": 3.4332275390625E-4,
  "yres": 3.4332275390625E-4
},
{
  "z": 12,
  "width": 8192,
  "height": 4096,
  "xres": 1.71661376953125E-4,
  "yres": 1.71661376953125E-4
},
{
  "z": 13,
  "width": 16384,
  "height": 8192,
  "xres": 8.58306884765625E-5,
  "yres": 8.58306884765625E-5
},
{
  "z": 14,
  "width": 32768,
  "height": 16384,
  "xres": 4.291534423828125E-5,
  "yres": 4.291534423828125E-5
},
```

```

        "z": 15,
        "width": 65536,
        "height": 32768,
        "xres": 2.1457672119140625E-5,
        "yres": 2.1457672119140625E-5
    },
    {
        "z": 16,
        "width": 131072,
        "height": 65536,
        "xres": 1.0728836059570312E-5,
        "yres": 1.0728836059570312E-5
    },
    {
        "z": 17,
        "width": 262144,
        "height": 131072,
        "xres": 5.364418029785156E-6,
        "yres": 5.364418029785156E-6
    },
    {
        "z": 18,
        "width": 524288,
        "height": 262144,
        "xres": 2.682209014892578E-6,
        "yres": 2.682209014892578E-6
    },
    {
        "z": 19,
        "width": 1048576,
        "height": 524288,
        "xres": 1.341104507446289E-6,
        "yres": 1.341104507446289E-6
    }
]
}

```

XML

```
geoc tile pyramid -l "type=geopackage file=src/test/resources/data.gpkg name=world" -o
xml
```

```

<pyramid>
    <proj>EPSG:4326</proj>
    <bounds>
        <minX>-179.99</minX>
        <minY>-89.99</minY>
        <maxX>179.99</maxX>
        <maxY>89.99</maxY>

```

```
</bounds>
<origin>TOP_LEFT</origin>
<tileSize>
    <width>256</width>
    <height>256</height>
</tileSize>
<grids>
    <grid>
        <z>0</z>
        <width>2</width>
        <height>1</height>
        <xres>0.703125</xres>
        <yres>0.703125</yres>
    </grid>
    <grid>
        <z>1</z>
        <width>4</width>
        <height>2</height>
        <xres>0.3515625</xres>
        <yres>0.3515625</yres>
    </grid>
    <grid>
        <z>2</z>
        <width>8</width>
        <height>4</height>
        <xres>0.17578125</xres>
        <yres>0.17578125</yres>
    </grid>
    <grid>
        <z>3</z>
        <width>16</width>
        <height>8</height>
        <xres>0.087890625</xres>
        <yres>0.087890625</yres>
    </grid>
    <grid>
        <z>4</z>
        <width>32</width>
        <height>16</height>
        <xres>0.0439453125</xres>
        <yres>0.0439453125</yres>
    </grid>
    <grid>
        <z>5</z>
        <width>64</width>
        <height>32</height>
        <xres>0.02197265625</xres>
        <yres>0.02197265625</yres>
    </grid>
    <grid>
        <z>6</z>
```

```
<width>128</width>
<height>64</height>
<xres>0.010986328125</xres>
<yres>0.010986328125</yres>
</grid>
<grid>
<z>7</z>
<width>256</width>
<height>128</height>
<xres>0.0054931640625</xres>
<yres>0.0054931640625</yres>
</grid>
<grid>
<z>8</z>
<width>512</width>
<height>256</height>
<xres>0.00274658203125</xres>
<yres>0.00274658203125</yres>
</grid>
<grid>
<z>9</z>
<width>1024</width>
<height>512</height>
<xres>0.001373291015625</xres>
<yres>0.001373291015625</yres>
</grid>
<grid>
<z>10</z>
<width>2048</width>
<height>1024</height>
<xres>6.866455078125E-4</xres>
<yres>6.866455078125E-4</yres>
</grid>
<grid>
<z>11</z>
<width>4096</width>
<height>2048</height>
<xres>3.4332275390625E-4</xres>
<yres>3.4332275390625E-4</yres>
</grid>
<grid>
<z>12</z>
<width>8192</width>
<height>4096</height>
<xres>1.71661376953125E-4</xres>
<yres>1.71661376953125E-4</yres>
</grid>
<grid>
<z>13</z>
<width>16384</width>
<height>8192</height>
```

```
<xres>8.58306884765625E-5</xres>
<yres>8.58306884765625E-5</yres>
</grid>
<grid>
<z>14</z>
<width>32768</width>
<height>16384</height>
<xres>4.291534423828125E-5</xres>
<yres>4.291534423828125E-5</yres>
</grid>
<grid>
<z>15</z>
<width>65536</width>
<height>32768</height>
<xres>2.1457672119140625E-5</xres>
<yres>2.1457672119140625E-5</yres>
</grid>
<grid>
<z>16</z>
<width>131072</width>
<height>65536</height>
<xres>1.0728836059570312E-5</xres>
<yres>1.0728836059570312E-5</yres>
</grid>
<grid>
<z>17</z>
<width>262144</width>
<height>131072</height>
<xres>5.364418029785156E-6</xres>
<yres>5.364418029785156E-6</yres>
</grid>
<grid>
<z>18</z>
<width>524288</width>
<height>262144</height>
<xres>2.682209014892578E-6</xres>
<yres>2.682209014892578E-6</yres>
</grid>
<grid>
<z>19</z>
<width>1048576</width>
<height>524288</height>
<xres>1.341104507446289E-6</xres>
<yres>1.341104507446289E-6</yres>
</grid>
</grids>
</pyramid>
```

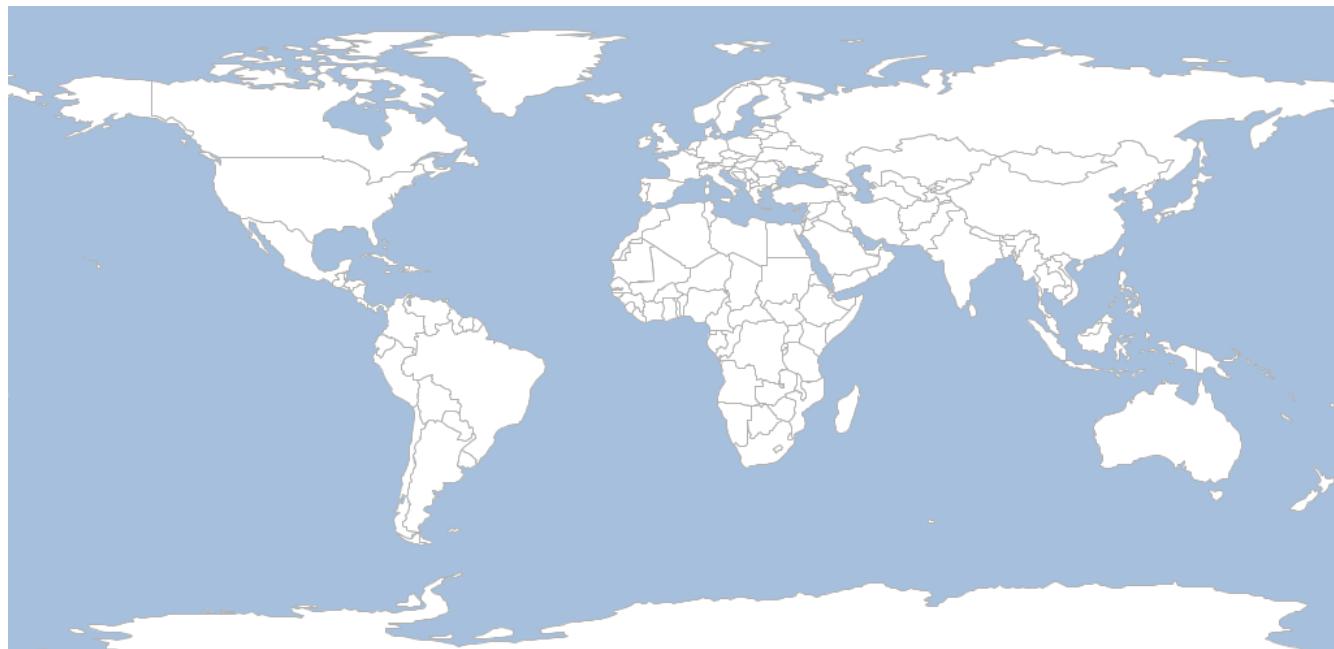
# Stitch Raster

Stitch image tiles together to create a Raster.

| Short Name | Long Name              | Description              |
|------------|------------------------|--------------------------|
| -l         | --tile-layer           | The tile layer           |
| -b         | --bounds               | The bounds               |
| -w         | --width                | The raster width         |
| -h         | --height               | The raster height        |
| -z         | --zoom-level           | The tile zoom level      |
| -x         | --minx                 | The min x or col         |
| -y         | --miny                 | The min y or row         |
| -c         | --maxx                 | The max x or col         |
| -u         | --maxy                 | The max y or row         |
| -o         | --output-raster        | The output raster        |
| -f         | --output-raster-format | The output raster format |
|            | --help                 | Print the help message   |
|            | --web-help             | Open help in a browser   |

Zoom Level

```
geoc tile stitch raster -l "type=geopackage file=src/test/resources/data.gpkg  
name=world" -o target/world_1.png -z 1
```



# Stitch Vector

Stitch vector tiles together to create a one or more Layers.

| Short Name | Long Name          | Description            |
|------------|--------------------|------------------------|
| -l         | --tile-layer       | The tile layer         |
| -b         | --bounds           | The bounds             |
| -w         | --width            | The raster width       |
| -h         | --height           | The raster height      |
| -z         | --zoom-level       | The tile zoom level    |
| -x         | --minx             | The min x or col       |
| -y         | --miny             | The min y or row       |
| -c         | --maxx             | The max x or col       |
| -u         | --maxy             | The max y or row       |
| -o         | --output-workspace | The output workspace   |
|            | --help             | Print the help message |
|            | --web-help         | Open help in a browser |

Zoom Level

```
geoc tile stitch vector -l "type=vectortiles format=pbf file=target/vectortiles" -o  
"type=geopackage file=target/world.gpkg name=world" -z 1
```

*Layers*

countries

*Schema*

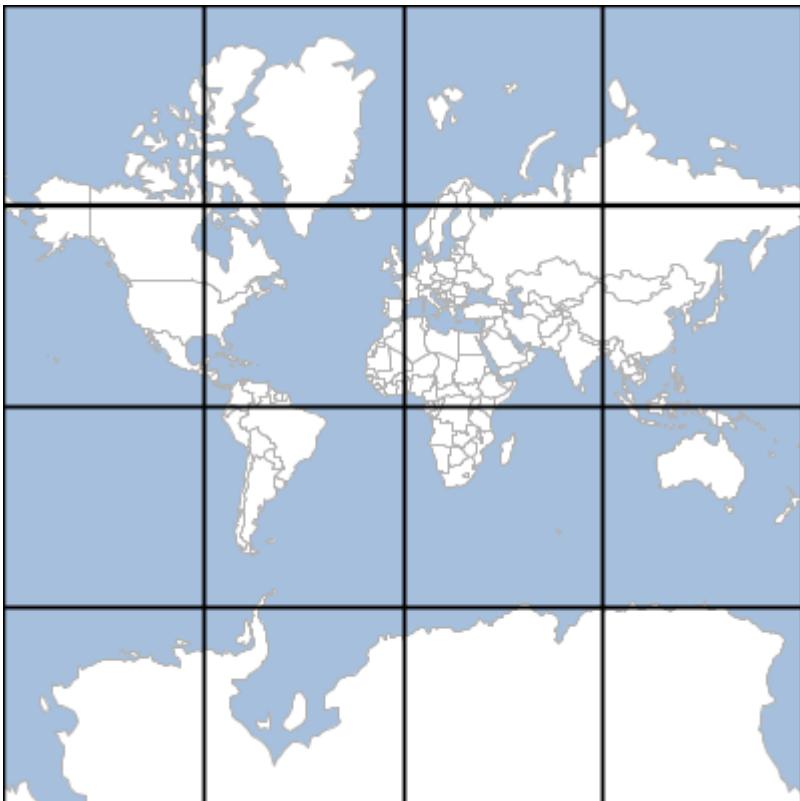
| Name     | Type    |
|----------|---------|
| geometry | Polygon |
| TYPE     | String  |
| LEVEL    | Long    |
| NAME     | String  |

# Vector Grid

Create a vector grid of a tile layers cells.

| <b>Short Name</b> | <b>Long Name</b>   | <b>Description</b>     |
|-------------------|--------------------|------------------------|
| -l                | --tile-layer       | The tile layer         |
| -b                | --bounds           | The bounds             |
| -z                | --zoom-level       | The tile zoom level    |
| -x                | --minx             | The min x or col       |
| -y                | --miny             | The min y or row       |
| -c                | --maxx             | The max x or col       |
| -u                | --maxy             | The max y or row       |
| -w                | --width            | The raster width       |
| -h                | --height           | The raster height      |
| -o                | --output-workspace | The output workspace   |
| -r                | --output-layer     | The output layer       |
|                   | --help             | Print the help message |
|                   | --web-help         | Open help in a browser |

```
geoc tile vector grid -l "type=geopackage file=src/main/resources/data.gpkg
name=world" -o target/world_grid_1.shp -z 2
```



## Vector Commands

# Add

Add a Feature to a Layer.

| Short Name | Long Name         | Description            |
|------------|-------------------|------------------------|
| -v         | --value           | A value 'field=value'  |
| -i         | --input-workspace | The input workspace    |
| -l         | --input-layer     | The input layer        |
|            | --help            | Print the help message |
|            | --web-help        | Open help in a browser |

```
geoc vector add -i target/locations.shp -v id=1 -v name=Seattle -v "the_geom=POINT (-122.334758 47.578364)"
```

| the_geom                      | name    | id |
|-------------------------------|---------|----|
| POINT (-122.334758 47.578364) | Seattle | 1  |



# Add Fields

Add one or more Fields to a Layer

| Short Name | Long Name          | Description                       |
|------------|--------------------|-----------------------------------|
| -f         | --field            | A Field in the format 'name=type' |
| -o         | --output-workspace | The output workspace              |
| -r         | --output-layer     | The output layer                  |
| -i         | --input-workspace  | The input workspace               |

| <b>Short Name</b> | <b>Long Name</b> | <b>Description</b>     |
|-------------------|------------------|------------------------|
| -l                | --input-layer    | The input layer        |
|                   | --help           | Print the help message |
|                   | --web-help       | Open help in a browser |

```
geoc vector addfields -i target/locations.shp -o target/locations_idname.shp -f id=int
-f name=string
```

### Schema

| <b>Name</b> | <b>Type</b> |
|-------------|-------------|
| the_geom    | Point       |
| name        | String      |
| id          | Integer     |

## Add Area Field

Add an area Field.

| <b>Short Name</b> | <b>Long Name</b>   | <b>Description</b>          |
|-------------------|--------------------|-----------------------------|
| -f                | --area-fieldname   | The name for the area Field |
| -o                | --output-workspace | The output workspace        |
| -r                | --output-layer     | The output layer            |
| -i                | --input-workspace  | The input workspace         |
| -l                | --input-layer      | The input layer             |
|                   | --help             | Print the help message      |
|                   | --web-help         | Open help in a browser      |

```
geoc vector addareafield -i src/test/resources/states.shp -o target/states_area.shp
```

### Schema

| <b>Name</b> | <b>Type</b>  |
|-------------|--------------|
| the_geom    | MultiPolygon |
| STATE_NAME  | String       |
| SUB_REGION  | String       |
| STATE_ABBR  | String       |
| AREA        | Double       |

## Values

| STATE_NAME           | SUB_REGION | STATE_ABBR | AREA               |
|----------------------|------------|------------|--------------------|
| Illinois             | E N Cen    | IL         | 15.396467068063995 |
| District of Columbia | S Atl      | DC         | 0.017769720828999  |
| Delaware             | S Atl      | DE         | 0.553317799081003  |
| West Virginia        | S Atl      | WV         | 6.493194953114009  |
| Maryland             | S Atl      | MD         | 2.625116892757991  |

## Add Length Field

Add an Length Field.

| Short Name | Long Name          | Description                   |
|------------|--------------------|-------------------------------|
| -f         | --length-fieldname | The name for the length Field |
| -o         | --output-workspace | The output workspace          |
| -r         | --output-layer     | The output layer              |
| -i         | --input-workspace  | The input workspace           |
| -l         | --input-layer      | The input layer               |
|            | --help             | Print the help message        |
|            | --web-help         | Open help in a browser        |

```
geoc vector addlengthfield -i src/test/resources/data.gpkg -l rivers -o target/rivers_length.shp -f length
```

## Schema

| Name     | Type            |
|----------|-----------------|
| the_geom | MultiLineString |
| name     | String          |
| label    | String          |
| length   | Double          |

## Values

| name        | label       | length            |
|-------------|-------------|-------------------|
| Brahmaputra | Brahmaputra | 25.21241966609205 |
| Mekong      | Mekong      | 34.97738061177052 |
| Ob          | Ob          | 48.39570358268261 |

| <b>name</b> | <b>label</b> | <b>length</b>     |
|-------------|--------------|-------------------|
| Peace       | Peace        | 44.84258394589285 |
| Donau       | Donau        | 26.67902946932429 |

## Add XY Fields

Add XY Fields.

| <b>Short Name</b> | <b>Long Name</b>   | <b>Description</b>                                      |
|-------------------|--------------------|---|
| -x                | --x-fieldname      | The name for the X Field                                |
| -y                | --y-fieldname      | The name for the Y Field                                |
| -a                | --algorithm        | The XY generation algorithm (centroid or interiorpoint) |
| -o                | --output-workspace | The output workspace                                    |
| -r                | --output-layer     | The output layer  |
| -i                | --input-workspace  | The input workspace                                     |
| -l                | --input-layer      | The input layer   |
|                   | --help             | Print the help message                                  |
|                   | --web-help         | Open help in a browser                                  |

```
geoc vector addxyfields -i src/test/resources/data.gpkg -l places -o
target/places_xy.shp -x x_coord -y y_coord -a centroid
```

### Schema

| <b>Name</b> | <b>Type</b> |
|-------------|-------------|
| the_geom    | Point       |
| NAME        | String      |
| x_coord     | Double      |
| y_coord     | Double      |

### Values

| <b>NAME</b>  | <b>x_coord</b> | <b>y_coord</b> |
|--------------|----------------|----------------|
| Vatican City | 12.4533865     | 41.9032822     |
| San Marino   | 12.4417702     | 43.9360958     |
| Vaduz        | 9.5166695      | 47.1337238     |
| Lobamba      | 31.1999971     | -26.4666675    |

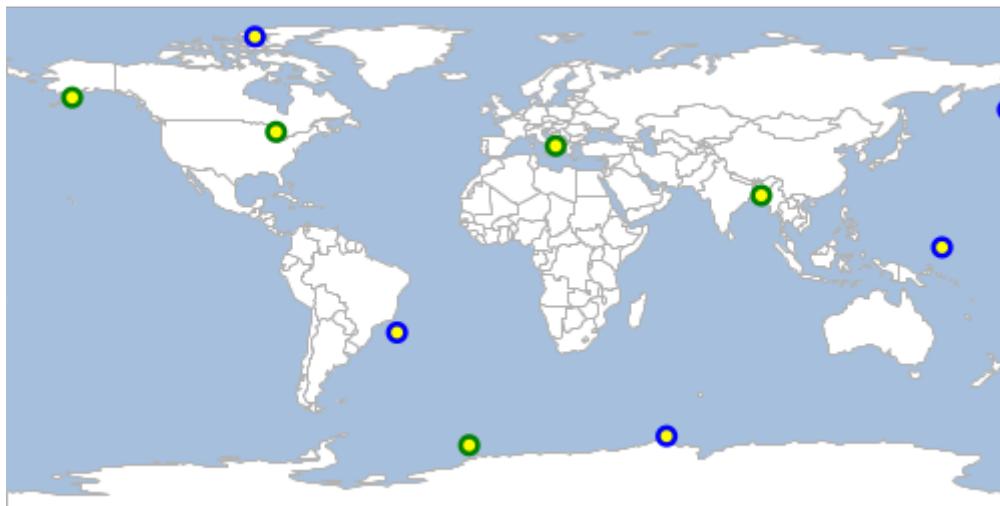
| NAME       | x_coord   | y_coord    |
|------------|-----------|------------|
| Luxembourg | 6.1300028 | 49.6116604 |

## Append

Add a Features from one layer to another Layer.

| Short Name | Long Name         | Description            |
|------------|-------------------|------------------------|
| -k         | --other-workspace | The other workspace    |
| -y         | --other-layer     | The other layer        |
| -i         | --input-workspace | The input workspace    |
| -l         | --input-layer     | The input layer        |
|            | --help            | Print the help message |
|            | --web-help        | Open help in a browser |

```
geoc vector append -i target/points1.shp -k target/points2.shp
```



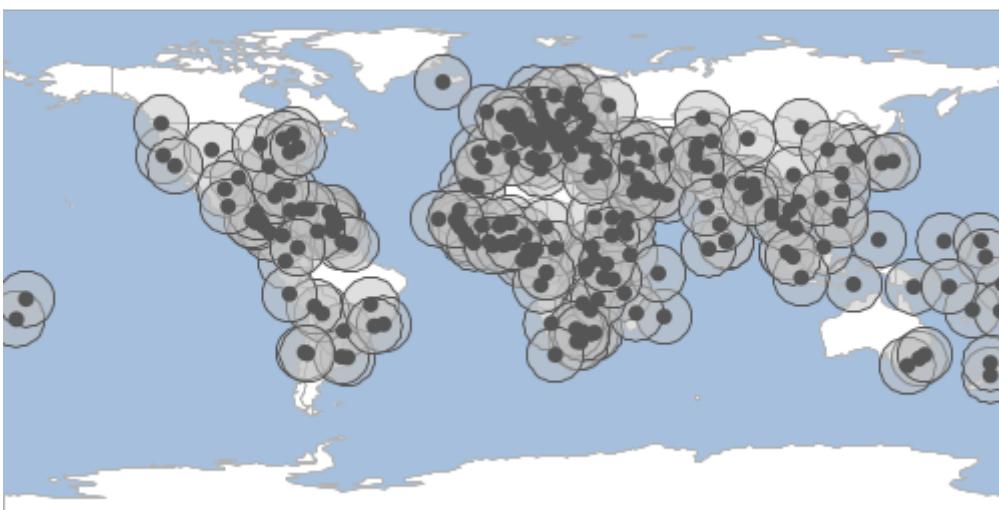
## Buffer

Buffer all of the features in a Layer.

| Short Name | Long Name          | Description                     |
|------------|--------------------|---------------------------------|
| -d         | --distance         | The buffer distance             |
| -q         | --quadrantsegments | The number of quadrant segments |

| <b>Short Name</b> | <b>Long Name</b>   | <b>Description</b>                           |
|-------------------|--------------------|--|
| -s                | --singl-sided      | Whether buffer should be single sided or not |
| -c                | --capstyle         | The cap style                                |
| -o                | --output-workspace | The output workspace                         |
| -r                | --output-layer     | The output layer                             |
| -i                | --input-workspace  | The input workspace                          |
| -l                | --input-layer      | The input layer                              |
|                   | --help             | Print the help message                       |
|                   | --web-help         | Open help in a browser                       |

```
geoc vector buffer -i src/test/resources/data.gpkg -l places -o
target/places_buffer.shp -d 10
```

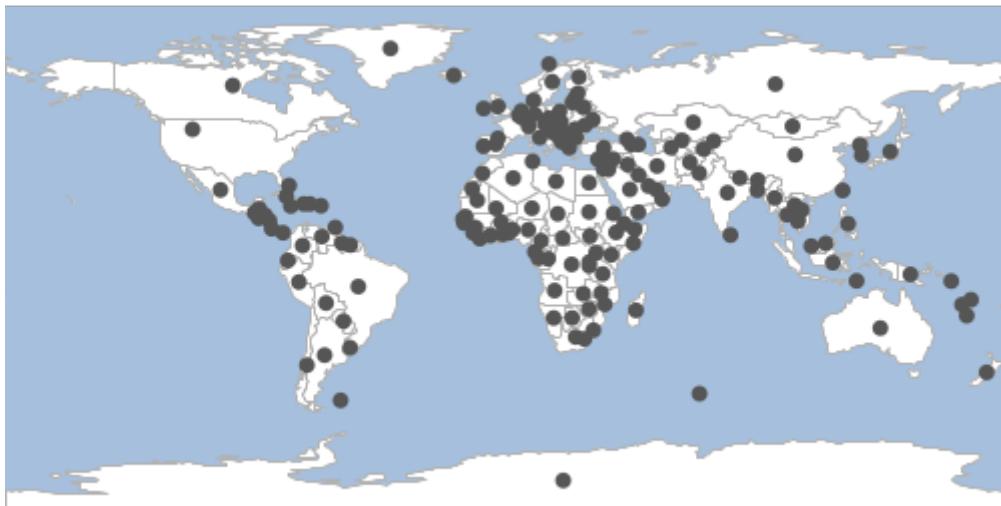


## Centroid

Calculate the centroid of all the features in a Layer.

| <b>Short Name</b> | <b>Long Name</b>   | <b>Description</b>     |
|-------------------|--------------------|------------------------|
| -o                | --output-workspace | The output workspace   |
| -r                | --output-layer     | The output layer       |
| -i                | --input-workspace  | The input workspace    |
| -l                | --input-layer      | The input layer        |
|                   | --help             | Print the help message |
|                   | --web-help         | Open help in a browser |

```
geoc vector centroid -i src/test/resources/data.gpkg -l countries -o target/countries_centroids.shp
```

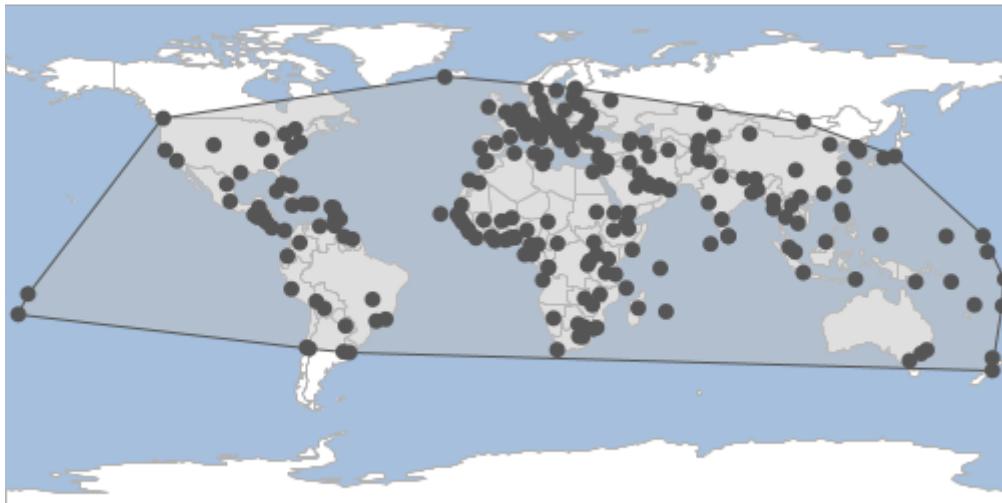


## Convexhull

Calculate the convexhull of all the features in a Layer.

| Short Name | Long Name          | Description            |
|------------|--------------------|------------------------|
| -o         | --output-workspace | The output workspace   |
| -r         | --output-layer     | The output layer       |
| -i         | --input-workspace  | The input workspace    |
| -l         | --input-layer      | The input layer        |
|            | --help             | Print the help message |
|            | --web-help         | Open help in a browser |

```
geoc vector convexhull -i src/test/resources/data.gpkg -l places -o target/convexhull.shp
```

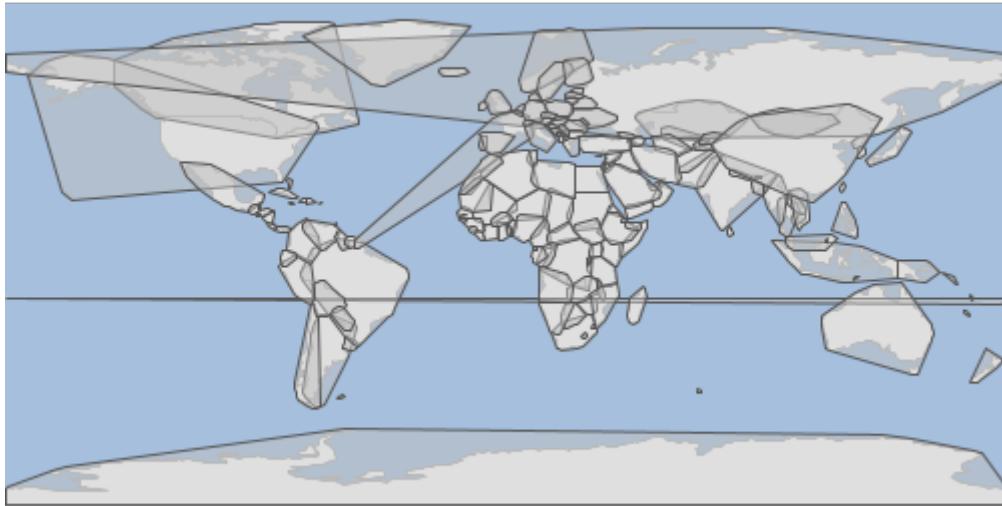


## Convexhulls

Calculate the convexhulls for each feature in a Layer.

| Short Name | Long Name          | Description            |
|------------|--------------------|------------------------|
| -o         | --output-workspace | The output workspace   |
| -r         | --output-layer     | The output layer       |
| -i         | --input-workspace  | The input workspace    |
| -l         | --input-layer      | The input layer        |
|            | --help             | Print the help message |
|            | --web-help         | Open help in a browser |

```
geoc vector convexhulls -i src/test/resources/data.gpkg -l countries -o target/convexhulls.shp
```



## Count

Count the Features in a Layer.

| Short Name | Long Name         | Description                           |
|------------|-------------------|---------------------------------------|
| -t         | --type            | Count features, geometries, or points |
| -i         | --input-workspace | The input workspace                   |
| -l         | --input-layer     | The input layer                       |
|            | --help            | Print the help message                |
|            | --web-help        | Open help in a browser                |

```
geoc vector count -i src/test/resources/data.gpkg -l places
```

243

## Create

Create a new Layer.

| Short Name | Long Name          | Description                       |
|------------|--------------------|-----------------------------------|
| -f         | --field            | A Field in the format 'name=type' |
| -o         | --output-workspace | The output workspace              |
| -r         | --output-layer     | The output layer                  |
|            | --help             | Print the help message            |

| <b>Short Name</b> | <b>Long Name</b> | <b>Description</b>     |
|-------------------|------------------|------------------------|
|                   | --web-help       | Open help in a browser |

```
geoc vector create -o target/locations.shp -f "the_geom=POINT EPSG:4326" -f id=integer  
-f name=string
```

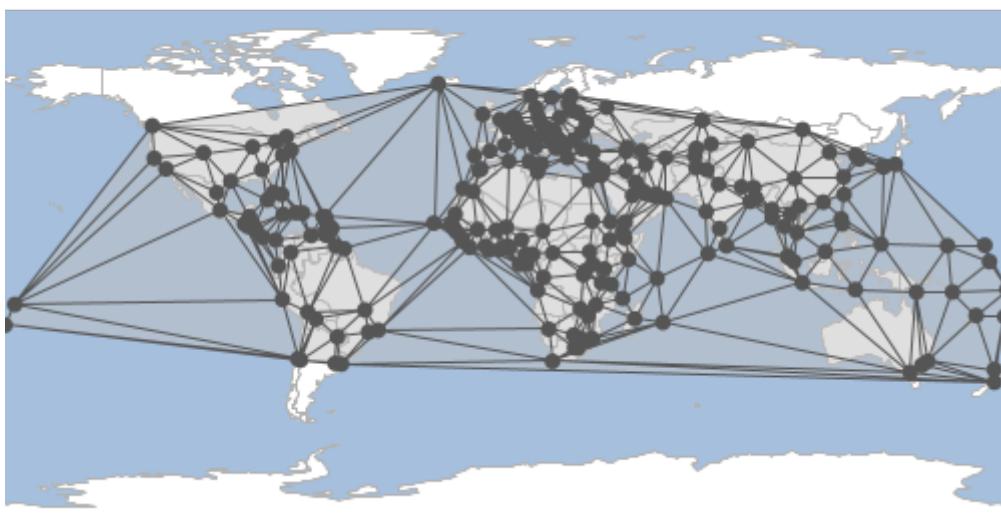
| <b>Name</b> | <b>Type</b> |
|-------------|-------------|
| the_geom    | Point       |
| name        | String      |
| id          | Integer     |

## Delaunay

Calculate a delaunay diagram of all the features in a Layer.

| <b>Short Name</b> | <b>Long Name</b>   | <b>Description</b>     |
|-------------------|--------------------|------------------------|
| -o                | --output-workspace | The output workspace   |
| -r                | --output-layer     | The output layer       |
| -i                | --input-workspace  | The input workspace    |
| -l                | --input-layer      | The input layer        |
|                   | --help             | Print the help message |
|                   | --web-help         | Open help in a browser |

```
geoc vector delaunay -i src/test/resources/data.gpkg -l places -o target/delaunay.shp
```

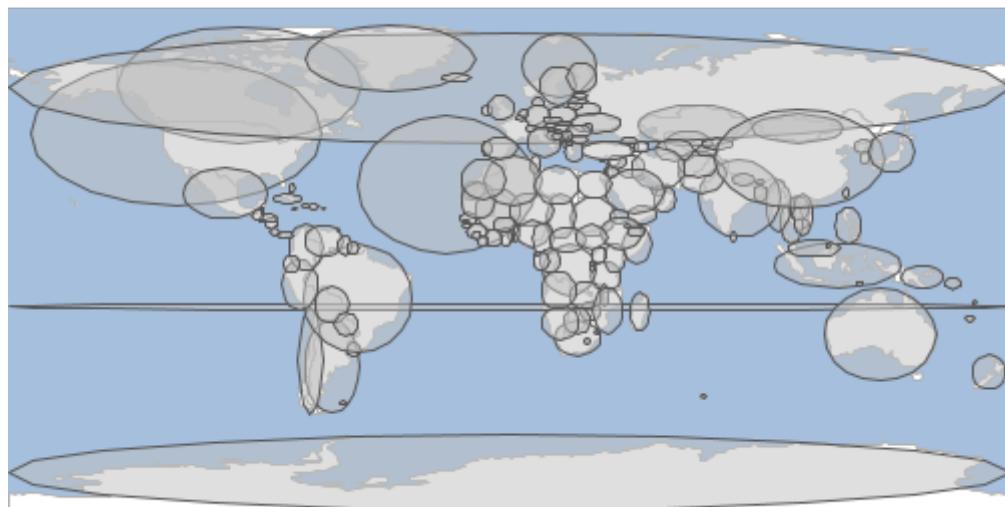


# Ellipse

Calculate the ellipse around each feature in a Layer.

| Short Name | Long Name          | Description  |
|------------|--------------------|--|
| -g         | --geometry         | The geometry expression  |
| -w         | --width            | The width of the bounds  |
| -h         | --height           | The height of the bounds   |
| -p         | --num-points       | The number of points   |
| -a         | --rotation         | The angle of rotation  |
| -u         | --unit             | The unit can either be degrees(d) or radians(r). The default is degrees. |
| -o         | --output-workspace | The output workspace   |
| -r         | --output-layer     | The output layer   |
| -i         | --input-workspace  | The input workspace  |
| -l         | --input-layer      | The input layer  |
|            | --help             | Print the help message   |
|            | --web-help         | Open help in a browser   |

```
geoc vector ellipse -i src/test/resources/data.gpkg -l countries -o target/ellipse.shp
```

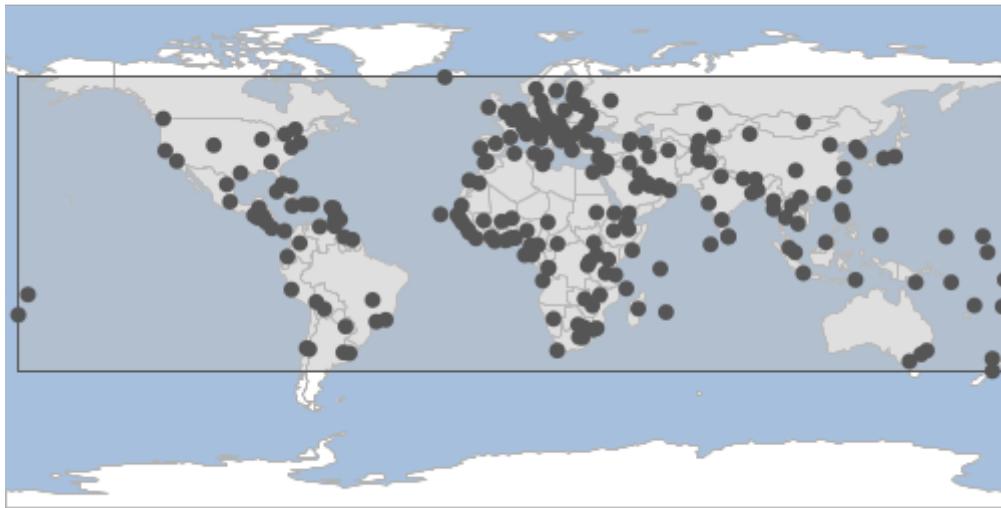


# Envelope

Calculate the envelope of all the features in a Layer.

| <b>Short Name</b> | <b>Long Name</b>   | <b>Description</b>     |
|-------------------|--------------------|------------------------|
| -o                | --output-workspace | The output workspace   |
| -r                | --output-layer     | The output layer       |
| -i                | --input-workspace  | The input workspace    |
| -l                | --input-layer      | The input layer        |
|                   | --help             | Print the help message |
|                   | --web-help         | Open help in a browser |

```
geoc vector envelope -i src/test/resources/data.gpkg -l places -o target/envelope.shp
```

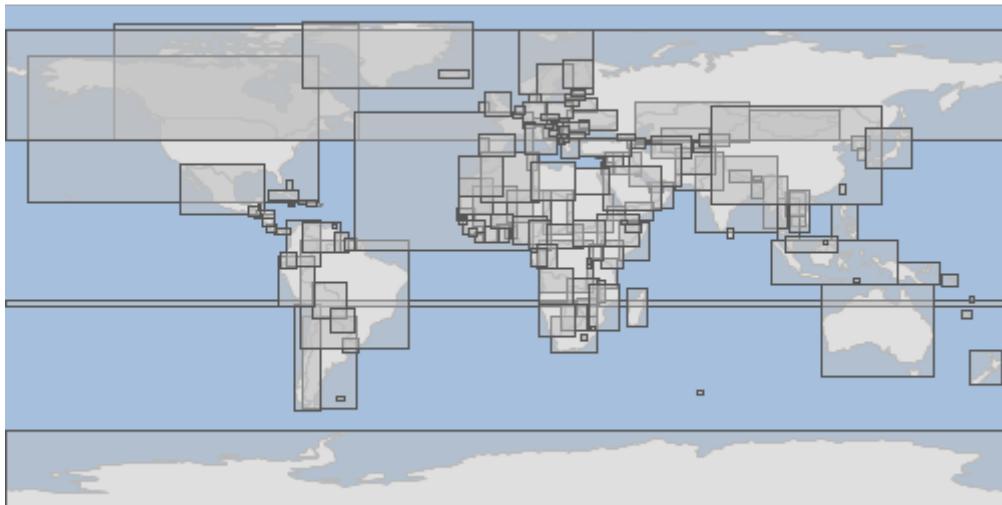


## Envelopes

Calculate the envelopes for each feature in a Layer.

| <b>Short Name</b> | <b>Long Name</b>   | <b>Description</b>     |
|-------------------|--------------------|------------------------|
| -o                | --output-workspace | The output workspace   |
| -r                | --output-layer     | The output layer       |
| -i                | --input-workspace  | The input workspace    |
| -l                | --input-layer      | The input layer        |
|                   | --help             | Print the help message |
|                   | --web-help         | Open help in a browser |

```
geoc vector envelopes -i src/test/resources/data.gpkg -l countries -o target/envelopes.shp
```



## From

Create a Layer from a string of KML, CSV, GML, GEORSS, GEOBUF, GPX or GeoJSON.

| Short Name | Long Name          | Description                                |
|------------|--------------------|--|
| -t         | --text             | The text                                   |
| -f         | --format           | The string format (CSV, GeoJSON, KML, GML) |
| -g         | --geometry-type    | The geometry type                          |
| -p         | --format-options   | A format options 'key=value'               |
| -o         | --output-workspace | The output workspace                       |
| -r         | --output-layer     | The output layer                           |
|            | --help             | Print the help message                     |
|            | --web-help         | Open help in a browser                     |

*GeoJSON*

*points.json*

```
{
  "type": "FeatureCollection",
  "features": [
    {"type": "Feature", "geometry": {"type": "Point", "coordinates": [-66.5103, -76.7195]}, "properties": {"id": 0}, "id": "randompoints.1"}, {"type": "Feature", "geometry": {"type": "Point", "coordinates": [-91.1419, 13.2189]}, "properties": {"id": 1}, "id": "randompoints.2"}, {"type": "Feature", "geometry": {"type": "Point", "coordinates": [67.5116, 15.6594]}, "properties": {"id": 2}, "id": "randompoints.3"}, {"type": "Feature", "geometry": {"type": "Point", "coordinates": [-164.9723, 14.9566]}, "properties": {"id": 3}, "id": "randompoints.4"}, {"type": "Feature", "geometry": {"type": "Point", "coordinates": [-169.3699, 9.3991]}, "properties": {"id": 4}, "id": "randompoints.5"}]
}
```

```
cat points.json | geoc vector from -f csv
```



CSV

*points.csv*

```
"the_geom:Point:EPSG:4326","id:Integer"
"POINT (-19.679267028642215 16.57439718741766)","0"
"POINT (2.9906441501594827 -58.23060660548015)","1"
"POINT (6.488652504697882 -37.82957086866762)","2"
"POINT (131.3732722678286 -22.286026554889062)","3"
"POINT (-31.12164174173816 40.174418870404764)","4"
```

```
cat points.csv | geoc vector from -f csv
```

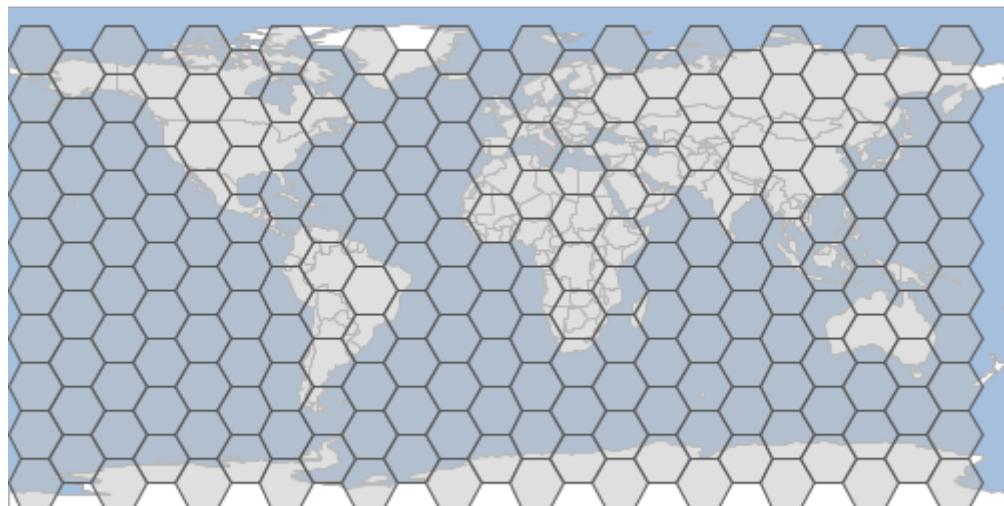


# Graticule - Hexagon

Create hexagon graticules.

| Short Name | Long Name          | Description                       |
|------------|--------------------|-----------------------------------|
| -g         | --geometry         | The geometry                      |
| -l         | --length           | The length                        |
| -s         | --spacing          | The spacing (defaults to -1)      |
| -t         | --orientation      | The orientation (flat or angled). |
| -o         | --output-workspace | The output workspace              |
| -r         | --output-layer     | The output layer                  |
|            | --help             | Print the help message            |
|            | --web-help         | Open help in a browser            |

```
geoc vector graticule hexagon -g -180,-90,180,90 -l 10 -o target/hexagons.shp
```



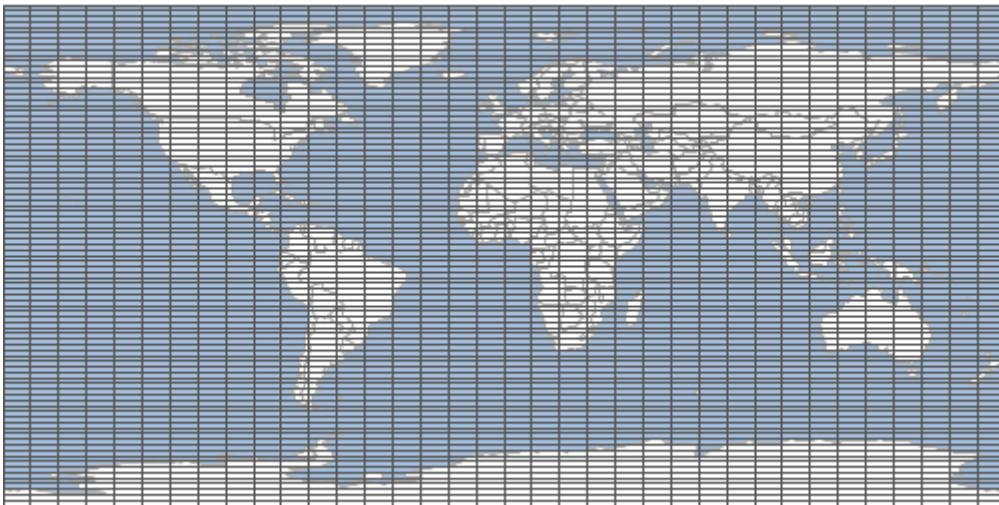
# Graticule - Line

Create line graticules.

| Short Name | Long Name  | Description                  |
|------------|------------|------------------------------|
| -g         | --geometry | The geometry                 |
| -s         | --spacing  | The spacing (defaults to -1) |

| <b>Short Name</b> | <b>Long Name</b>   | <b>Description</b>   |
|-------------------|--------------------|--|
| -l                | --line-definition  | Each line definition has comma delimited orientation (vertical or horizontal), level, and spacing) |
| -o                | --output-workspace | The output workspace   |
| -r                | --output-layer     | The output layer   |
|                   | --help             | Print the help message   |
|                   | --web-help         | Open help in a browser   |

```
geoc vector graticule line -g -180,-90,180,90 -l vertical,2,10 -l horizontal,1,2 -o
target/lines.shp
```

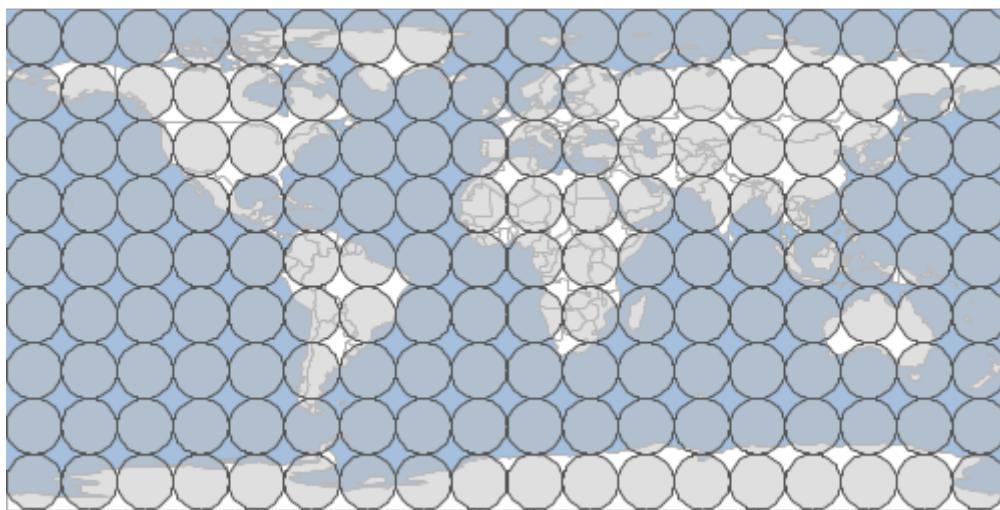


## Graticule - Oval

Create oval graticules.

| <b>Short Name</b> | <b>Long Name</b>   | <b>Description</b>     |
|-------------------|--------------------|------------------------|
| -g                | --geometry         | The geometry           |
| -l                | --length           | The length             |
| -o                | --output-workspace | The output workspace   |
| -r                | --output-layer     | The output layer       |
|                   | --help             | Print the help message |
|                   | --web-help         | Open help in a browser |

```
geoc vector graticule oval -g -180,-90,180,90 -l 20 -o target/ovals.shp
```

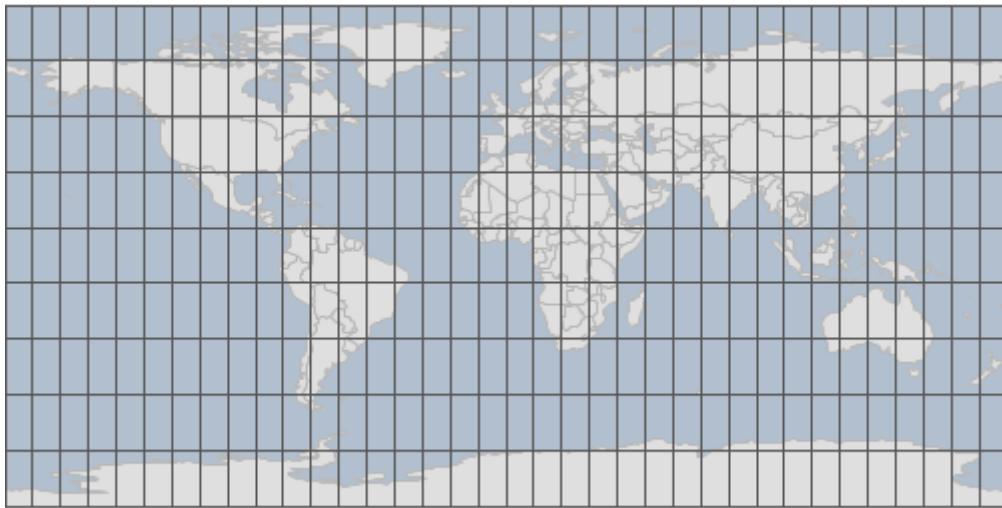


## Graticule - Rectangle

Create rectangle graticules.

| Short Name | Long Name          | Description                  |
|------------|--------------------|------------------------------|
| -g         | --geometry         | The geometry                 |
| -w         | --width            | The width                    |
| -h         | --height           | The height                   |
| -s         | --spacing          | The spacing (defaults to -1) |
| -o         | --output-workspace | The output workspace         |
| -r         | --output-layer     | The output layer             |
|            | --help             | Print the help message       |
|            | --web-help         | Open help in a browser       |

```
geoc vector graticule rectangle -g -180,-90,180,90 -w 10 -h 20 -o  
target/rectangles.shp
```

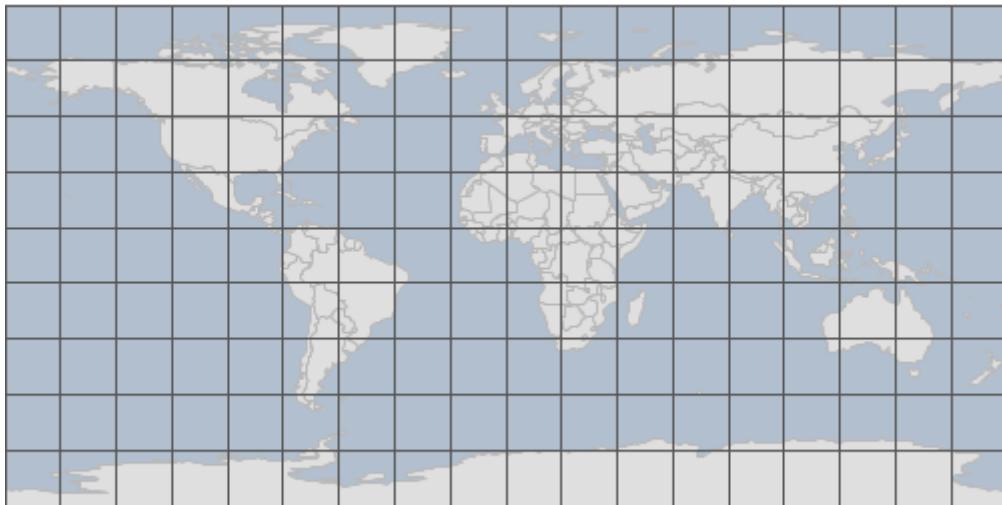


## Graticule - Square

Create square graticules.

| Short Name | Long Name          | Description                  |
|------------|--------------------|------------------------------|
| -g         | --geometry         | The geometry                 |
| -l         | --length           | The length                   |
| -s         | --spacing          | The spacing (defaults to -1) |
| -o         | --output-workspace | The output workspace         |
| -r         | --output-layer     | The output layer             |
|            | --help             | Print the help message       |
|            | --web-help         | Open help in a browser       |

```
geoc vector graticule square -g -180,-90,180,90 -l 20 -o target/squares.shp
```



## Info

Get information about a Layer.

| Short Name | Long Name         | Description            |
|------------|-------------------|------------------------|
| -i         | --input-workspace | The input workspace    |
| -l         | --input-layer     | The input layer        |
|            | --help            | Print the help message |
|            | --web-help        | Open help in a browser |

```
geoc vector info -i src/test/resources/data.gpkg -l countries
```

```

Name: countries
Geometry: MultiPolygon
Extent: -180.0, -90.0, 180.00000000000006, 83.64513000000001
Projection ID: EPSG:4326
Projection WKT: GEOGCS["WGS 84",
    DATUM["World Geodetic System 1984",
        SPHEROID["WGS 84", 6378137.0, 298.257223563, AUTHORITY["EPSG","7030"]],
        AUTHORITY["EPSG","6326"]],
    PRIMEM["Greenwich", 0.0, AUTHORITY["EPSG","8901"]],
    UNIT["degree", 0.017453292519943295],
    AXIS["Geodetic longitude", EAST],
    AXIS["Geodetic latitude", NORTH],
    AUTHORITY["EPSG","4326"]]

Feature Count: 177
Fields:
the_geom: MultiPolygon
featurecla: String
scalerank: Integer
LABELRANK: Integer
SOVEREIGNT: String
SOV_A3: String
ADM0_DIF: Integer
...

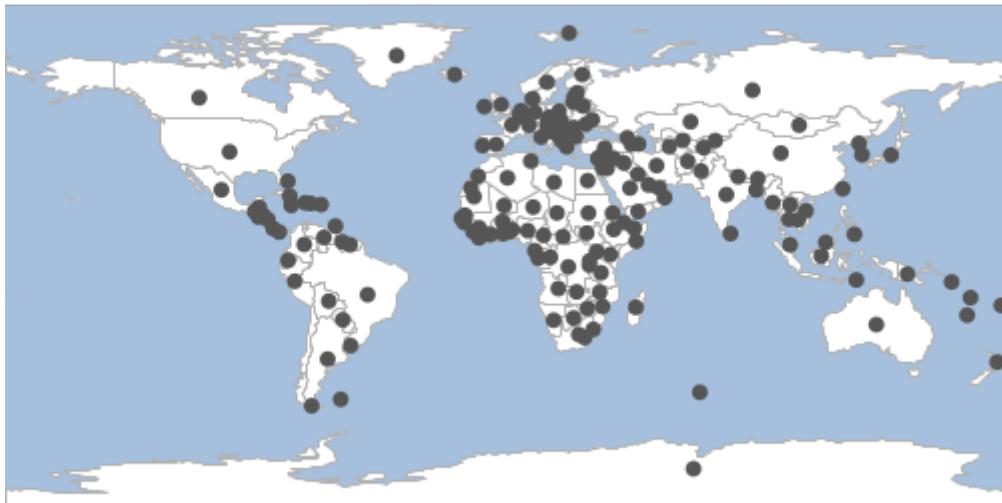
```

## Interior Point

Calculate the interior point of all the features in a Layer.

| Short Name | Long Name          | Description            |
|------------|--------------------|------------------------|
| -o         | --output-workspace | The output workspace   |
| -r         | --output-layer     | The output layer       |
| -i         | --input-workspace  | The input workspace    |
| -l         | --input-layer      | The input layer        |
|            | --help             | Print the help message |
|            | --web-help         | Open help in a browser |

```
geoc vector interiorPoint -i src/test/resources/data.gpkg -l countries -o
target/countries_interiorpoints.shp
```



## Layer List

List the Layers in a Workspace.

| Short Name | Long Name         | Description            |
|------------|-------------------|------------------------|
| -i         | --input-workspace | The input workspace    |
|            | --help            | Print the help message |
|            | --web-help        | Open help in a browser |

```
geoc vector list layers -i src/test/resources/data.gpkg
```

```
countries
graticules
ocean
places
rivers
states
```

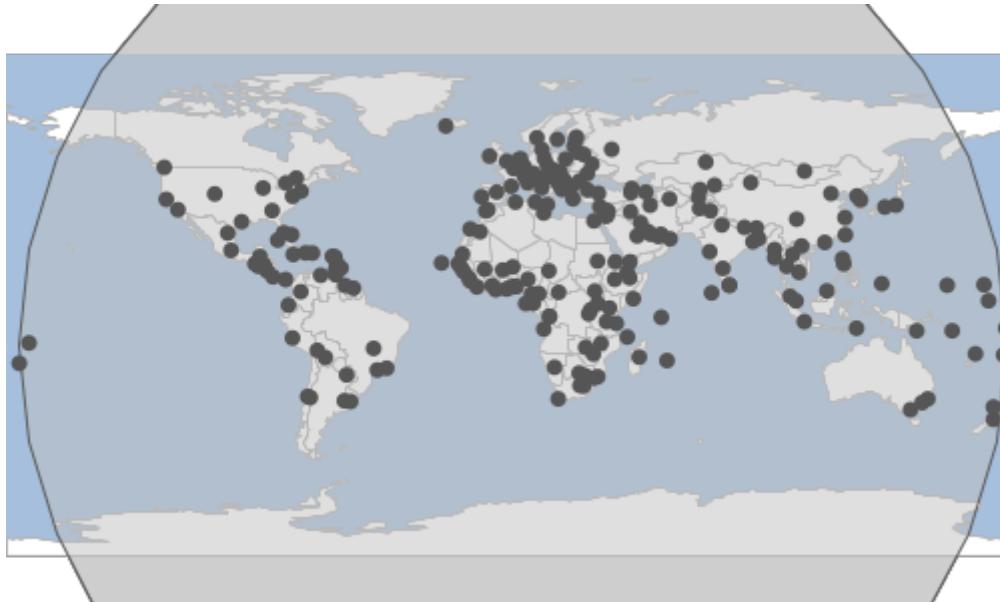
## Minimum Bounding Circle

Calculate the minimum bounding circle of all the features in a Layer.

| Short Name | Long Name          | Description          |
|------------|--------------------|----------------------|
| -o         | --output-workspace | The output workspace |
| -r         | --output-layer     | The output layer     |
| -i         | --input-workspace  | The input workspace  |
| -l         | --input-layer      | The input layer      |

| <b>Short Name</b> | <b>Long Name</b> | <b>Description</b>     |
|-------------------|------------------|------------------------|
|                   | --help           | Print the help message |
|                   | --web-help       | Open help in a browser |

```
geoc vector mincircle -i src/test/resources/data.gpkg -l places -o
target/mincircle.shp
```

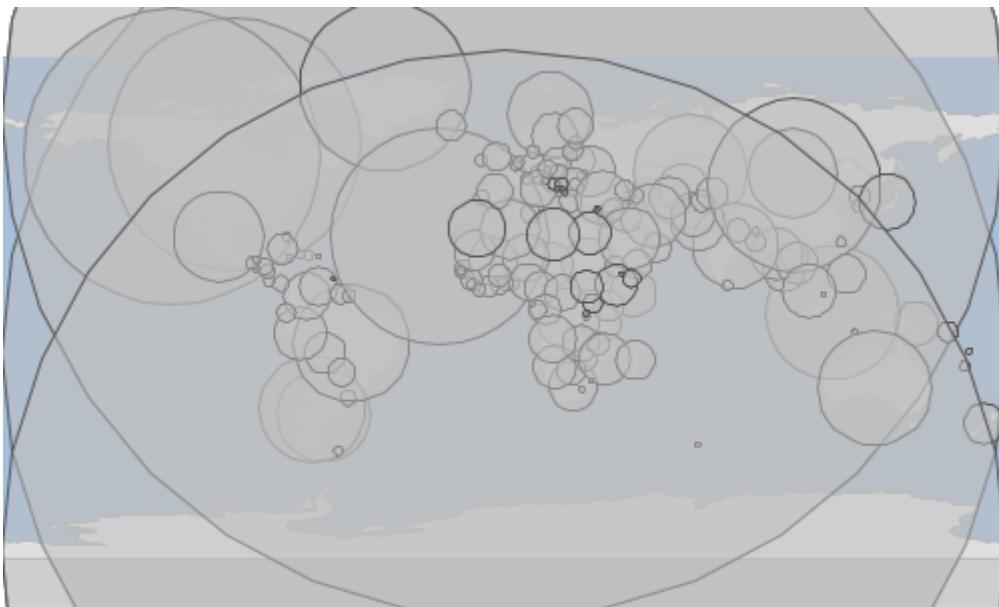


## Minimum Bounding Circles

Calculate the minimum bounding circle for each feature in a Layer.

| <b>Short Name</b> | <b>Long Name</b>   | <b>Description</b>     |
|-------------------|--------------------|------------------------|
| -o                | --output-workspace | The output workspace   |
| -r                | --output-layer     | The output layer       |
| -i                | --input-workspace  | The input workspace    |
| -l                | --input-layer      | The input layer        |
|                   | --help             | Print the help message |
|                   | --web-help         | Open help in a browser |

```
geoc vector mincircles -i src/test/resources/data.gpkg -l countries -o
target/mincircles.shp
```

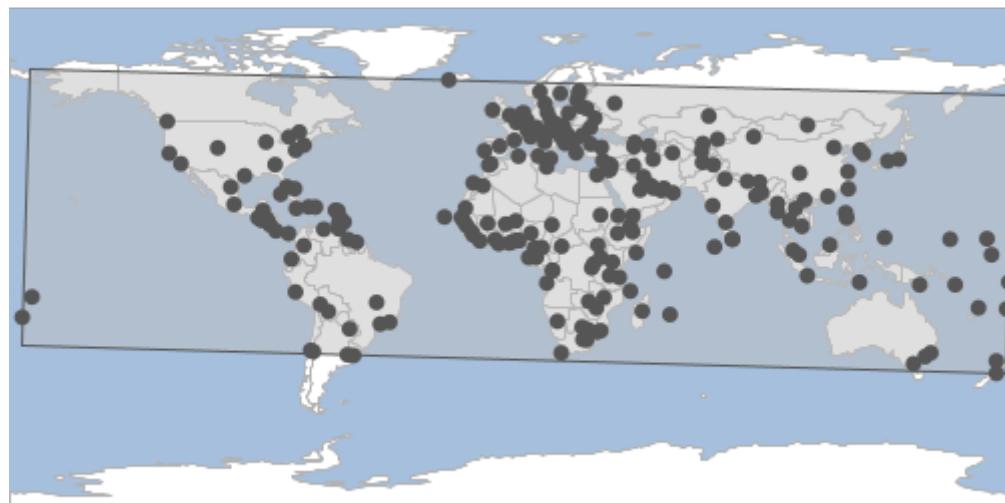


## Minimum Bounding Rectangle

Calculate the minimum bounding rectangle of all the features in a Layer.

| Short Name | Long Name          | Description            |
|------------|--------------------|------------------------|
| -o         | --output-workspace | The output workspace   |
| -r         | --output-layer     | The output layer       |
| -i         | --input-workspace  | The input workspace    |
| -l         | --input-layer      | The input layer        |
|            | --help             | Print the help message |
|            | --web-help         | Open help in a browser |

```
geoc vector minrect -i src/test/resources/data.gpkg -l places -o target/minrect.shp
```

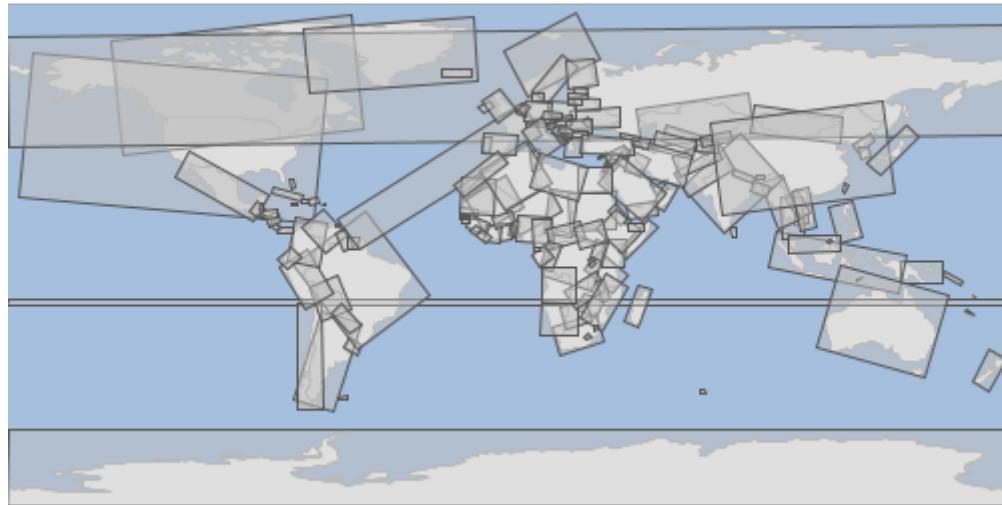


# Minimum Bounding rects

Calculate the minimum bounding rectangle for each feature in a Layer.

| Short Name | Long Name          | Description            |
|------------|--------------------|------------------------|
| -o         | --output-workspace | The output workspace   |
| -r         | --output-layer     | The output layer       |
| -i         | --input-workspace  | The input workspace    |
| -l         | --input-layer      | The input layer        |
|            | --help             | Print the help message |
|            | --web-help         | Open help in a browser |

```
geoc vector minrects -i src/test/resources/data.gpkg -l countries -o target/minrects.shp
```



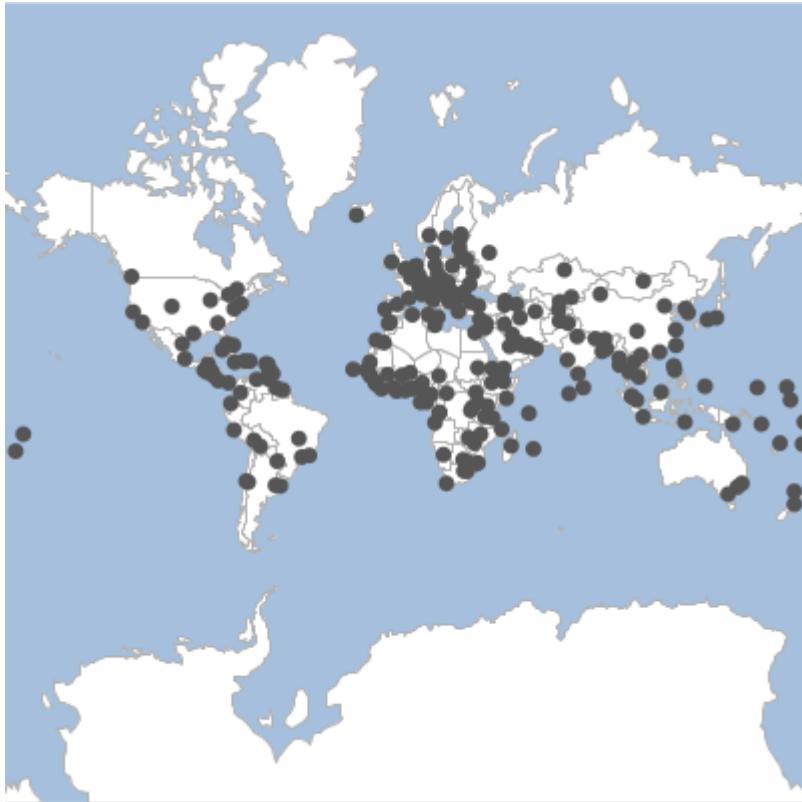
# Project

Project the input Layer to another Projection and save it as the output Layer.

| Short Name | Long Name           | Description           |
|------------|---------------------|-----------------------|
| -s         | --source-projection | The source projection |
| -t         | --target-projection | The target projection |
| -o         | --output-workspace  | The output workspace  |
| -r         | --output-layer      | The output layer      |
| -i         | --input-workspace   | The input workspace   |
| -l         | --input-layer       | The input layer       |

| <b>Short Name</b> | <b>Long Name</b> | <b>Description</b>     |
|-------------------|------------------|------------------------|
|                   | --help           | Print the help message |
|                   | --web-help       | Open help in a browser |

```
geoc vector project -i src/test/resources/data.gpkg -l places -o target/mercator.gpkg
-r places -s EPSG:4326 -t EPSG:3857
```



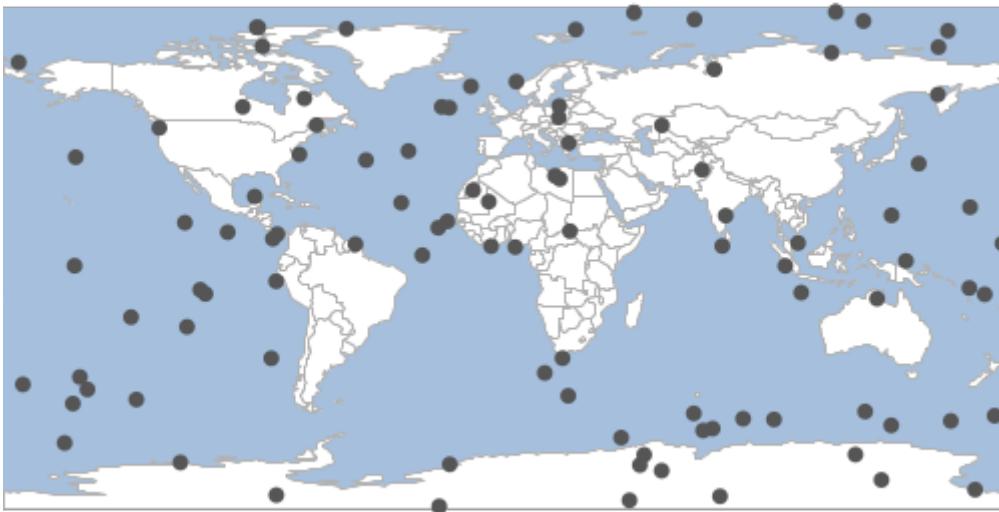
## Random Points

Generate random points.

| <b>Short Name</b> | <b>Long Name</b>        | <b>Description</b>  |
|-------------------|-------------------------|---|
| -n                | --number                | The number of points  |
| -p                | --projection            | The projection  |
| -g                | --geometry              | The geometry  |
| -d                | --grid                  | Whether to create random points in grid                     |
| -c                | --constrained-to-circle | Whether the points should be constrained to a circle or not |
| -f                | --gutter-fraction       | The size of the gutter between cells                        |
| -e                | --geom-fieldname        | The geometry field name                                     |

| <b>Short Name</b> | <b>Long Name</b>   | <b>Description</b>     |
|-------------------|--------------------|------------------------|
| -u                | --id-fieldname     | The id field name      |
| -o                | --output-workspace | The output workspace   |
| -r                | --output-layer     | The output layer       |
|                   | --help             | Print the help message |
|                   | --web-help         | Open help in a browser |

```
geoc vector randompoints -n 100 -g -180,-90,180,90 -o target/randompoints.shp
```



## To

Write a Layer to a String format (CSV, GeoJSON, KML, GML, GEORSS, GPX).

| <b>Short Name</b> | <b>Long Name</b>  | <b>Description</b>                                      |
|-------------------|-------------------|---|
| -f                | --format          | The string format (CSV, GeoJSON, KML, GML, GEORSS, GPX) |
| -p                | --format-options  | A format options 'key=value'                            |
| -i                | --input-workspace | The input workspace                                     |
| -l                | --input-layer     | The input layer   |
|                   | --help            | Print the help message                                  |
|                   | --web-help        | Open help in a browser                                  |

*GeoJSON*

```
geoc vector to -i target/randompoints.shp -f geojson
```

```
{"type": "FeatureCollection", "features": [{"type": "Feature", "geometry": {"type": "Point", "coordinates": [146.3526, -79.4573]}, "properties": {"id": 0}, "id": "randompoints.1"}, {"type": "Feature", "geometry": {"type": "Point", "coordinates": [72.6441, -1.5896]}, "properties": {"id": 1}, "id": "randompoints.2"}, {"type": "Feature", "geometry": {"type": "Point", "coordinates": [157.5863, 87.7214]}, "properties": {"id": 2}, "id": "randompoints.3"}, {"type": "Feature", "geometry": {"type": "Point", "coordinates": [-19.3248, -66.3298]}, "properties": {"id": 3}, "id": "randompoints.4"}, {"type": "Feature", "geometry": {"type": "Point", "coordinates": [-12.8938, -9.8001]}, "properties": {"id": 4}, "id": "randompoints.5"}]}
```

CSV

```
geoc vector to -i target/randompoints.shp -f csv
```

```
"the_geom:Point:EPSG:4326", "id:Integer"
"POINT (23.157983446157573 68.21328238838453)", "0"
"POINT (-168.29558547318035 70.50528988575832)", "1"
"POINT (-35.23281107180023 -8.50283678663682)", "2"
"POINT (-75.56335250523455 -0.5862584159271762)", "3"
"POINT (-149.96701997941562 48.39950979116222)", "4"
```

## Schema

Get a Layer's Schema.

| Short Name | Long Name         | Description                        |
|------------|-------------------|------------------------------------|
| -p         | --pretty-print    | Whether to pretty print the output |
| -i         | --input-workspace | The input workspace                |
| -l         | --input-layer     | The input layer                    |
|            | --help            | Print the help message             |
|            | --web-help        | Open help in a browser             |

```
geoc vector schema -i src/test/resources/data.gpkg -l countries -p
```

| -----             | -----        | ----- |
|-------------------|--------------|-------|
| name              | type         |       |
| ----- ----- ----- |              |       |
| the_geom          | MultiPolygon |       |
| featurecla        | String       |       |
| scalerank         | Integer      |       |
| LABELRANK         | Integer      |       |
| SOVEREIGNT        | String       |       |

|            |         |
|------------|---------|
| SOV_A3     | String  |
| ADM0_DIF   | Integer |
| LEVEL      | Integer |
| TYPE       | String  |
| ADMIN      | String  |
| ADM0_A3    | String  |
| GEOU_DIF   | Integer |
| GEOUNIT    | String  |
| GU_A3      | String  |
| SU_DIF     | Integer |
| SUBUNIT    | String  |
| SU_A3      | String  |
| BRK_DIFF   | Integer |
| NAME       | String  |
| NAME_LONG  | String  |
| BRK_A3     | String  |
| BRK_NAME   | String  |
| ABBREV     | String  |
| POSTAL     | String  |
| FORMAL_EN  | String  |
| FORMAL_FR  | String  |
| NAME_CIAWF | String  |
| NOTE_ADMIN | String  |
| NOTE_BRK   | String  |
| NAME_SORT  | String  |
| NAME_ALT   | String  |
| MAPCOLOR7  | Integer |
| MAPCOLOR8  | Integer |
| MAPCOLOR9  | Integer |
| MAPCOLOR13 | Integer |
| POP_EST    | Double  |
| POP_RANK   | Integer |
| POP_YEAR   | Integer |
| GDP_MD     | Integer |
| GDP_YEAR   | Integer |
| ECONOMY    | String  |
| INCOME_GRP | String  |
| FIPS_10    | String  |
| ISO_A2     | String  |
| ISO_A2_EH  | String  |
| ISO_A3     | String  |
| ISO_A3_EH  | String  |
| ISO_N3     | String  |
| ISO_N3_EH  | String  |
| UN_A3      | String  |
| WB_A2      | String  |
| WB_A3      | String  |
| WOE_ID     | Integer |
| WOE_ID_EH  | Integer |
| WOE_NOTE   | String  |
| ADM0_A3_IS | String  |

|            |         |
|------------|---------|
| ADM0_A3_US | String  |
| ADM0_A3_FR | String  |
| ADM0_A3_RU | String  |
| ADM0_A3_ES | String  |
| ADM0_A3_CN | String  |
| ADM0_A3_TW | String  |
| ADM0_A3_IN | String  |
| ADM0_A3_NP | String  |
| ADM0_A3_PK | String  |
| ADM0_A3_DE | String  |
| ADM0_A3_GB | String  |
| ADM0_A3_BR | String  |
| ADM0_A3_IL | String  |
| ADM0_A3_PS | String  |
| ADM0_A3_SA | String  |
| ADM0_A3_EG | String  |
| ADM0_A3_MA | String  |
| ADM0_A3_PT | String  |
| ADM0_A3_AR | String  |
| ADM0_A3_JP | String  |
| ADM0_A3_KO | String  |
| ADM0_A3_VN | String  |
| ADM0_A3_TR | String  |
| ADM0_A3_ID | String  |
| ADM0_A3_PL | String  |
| ADM0_A3_GR | String  |
| ADM0_A3_IT | String  |
| ADM0_A3_NL | String  |
| ADM0_A3_SE | String  |
| ADM0_A3_BD | String  |
| ADM0_A3_UA | String  |
| ADM0_A3_UN | Integer |
| ADM0_A3_WB | Integer |
| CONTINENT  | String  |
| REGION_UN  | String  |
| SUBREGION  | String  |
| REGION_WB  | String  |
| NAME_LEN   | Integer |
| LONG_LEN   | Integer |
| ABBREV_LEN | Integer |
| TINY       | Integer |
| HOMEPART   | Integer |
| MIN_ZOOM   | Double  |
| MIN_LABEL  | Double  |
| MAX_LABEL  | Double  |
| NE_ID      | Long    |
| WIKIDATAID | String  |
| NAME_AR    | String  |
| NAME_BN    | String  |
| NAME_DE    | String  |
| NAME_EN    | String  |

|            |        |
|------------|--------|
| NAME_ES    | String |
| NAME_FA    | String |
| NAME_FR    | String |
| NAME_EL    | String |
| NAME_HE    | String |
| NAME_HI    | String |
| NAME_HU    | String |
| NAME_ID    | String |
| NAME_IT    | String |
| NAME_JA    | String |
| NAME_KO    | String |
| NAME_NL    | String |
| NAME_PL    | String |
| NAME_PT    | String |
| NAME_RU    | String |
| NAME_SV    | String |
| NAME_TR    | String |
| NAME_UK    | String |
| NAME_UR    | String |
| NAME_VI    | String |
| NAME_ZH    | String |
| NAME_ZHT   | String |
| FCLASS_ISO | String |
| FCLASS_US  | String |
| FCLASS_FR  | String |
| FCLASS_RU  | String |
| FCLASS_ES  | String |
| FCLASS_CN  | String |
| FCLASS_TW  | String |
| FCLASS_IN  | String |
| FCLASS_NP  | String |
| FCLASS_PK  | String |
| FCLASS_DE  | String |
| FCLASS_GB  | String |
| FCLASS_BR  | String |
| FCLASS_IL  | String |
| FCLASS_PS  | String |
| FCLASS_SA  | String |
| FCLASS_EG  | String |
| FCLASS_MA  | String |
| FCLASS_PT  | String |
| FCLASS_AR  | String |
| FCLASS_JP  | String |
| FCLASS_KO  | String |
| FCLASS_VN  | String |
| FCLASS_TR  | String |
| FCLASS_ID  | String |
| FCLASS_PL  | String |
| FCLASS_GR  | String |
| FCLASS_IT  | String |
| FCLASS_NL  | String |

|           |        |
|-----------|--------|
| FCLASS_SE | String |
| FCLASS_BD | String |
| FCLASS_UA | String |

## Voronoi

Calculate a voronoi diagram of all the features in a Layer.

| Short Name | Long Name          | Description            |
|------------|--------------------|------------------------|
| -o         | --output-workspace | The output workspace   |
| -r         | --output-layer     | The output layer       |
| -i         | --input-workspace  | The input workspace    |
| -l         | --input-layer      | The input layer        |
|            | --help             | Print the help message |
|            | --web-help         | Open help in a browser |

```
geoc vector voronoi -i src/test/resources/data.gpkg -l places -o target/voronoi.shp
```

